ECCC

On Identity Testing of Tensors, Low-rank Recovery and Compressed Sensing

Michael A. Forbes[*]   Amir Shpilka[†]

November 2, 2011

Abstract

We study the problem of obtaining efficient, deterministic, *black-box polynomial identity testing algorithms* for depth-3 set-multilinear circuits (over arbitrary fields). This class of circuits has an efficient, deterministic, white-box polynomial identity testing algorithm (due to Raz and Shpilka [RS05]), but has no known such black-box algorithm. We recast this problem as a question of finding a low-dimensional subspace $\mathcal{H}$, spanned by rank 1 tensors, such that any non-zero tensor in the dual space $\ker(\mathcal{H})$ has high rank. We obtain explicit constructions of essentially optimal-size hitting sets for tensors of degree 2 (matrices), and obtain quasi-polynomial sized hitting sets for arbitrary tensors (but this second hitting set is less explicit).

We also show connections to the task of performing *low-rank recovery* of matrices, which is studied in the field of compressed sensing. Low-rank recovery asks (say, over $\mathbb{R}$) to recover a matrix M from few measurements, under the promise that M is rank $\leq r$. In this work, we restrict our attention to recovering matrices that are exactly rank $\leq r$ using deterministic, non-adaptive, linear measurements, that are free from noise. Over $\mathbb{R}$, we provide a set (of size $4nr$) of such measurements, from which M can be recovered in $\mathcal{O}(rn^2 + r^3n)$ field operations, and the number of measurements is essentially optimal. Further, the measurements can be taken to be all rank-1 matrices, or all sparse matrices. To the best of our knowledge no explicit constructions with those properties were known prior to this work.

We also give a more formal connection between low-rank recovery and the task of *sparse (vector) recovery*: any sparse-recovery algorithm that exactly recovers vectors of length n and sparsity $2r$, using m non-adaptive measurements, yields a low-rank recovery scheme for exactly recovering $n \times n$ matrices of rank $\leq r$, making $2nm$ non-adaptive measurements. Furthermore, if the sparse-recovery algorithm runs in time τ, then the low-rank recovery algorithm runs in time $\mathcal{O}(rn^2 + n\tau)$. We obtain this reduction using linear-algebraic techniques, and not using convex optimization, which is more commonly seen in compressed sensing algorithms.

Finally, we also make a connection to *rank-metric codes*, as studied in coding theory. These are codes with codewords consisting of matrices (or tensors) where the distance of matrices A and B is $\mathrm{rank}(A - B)$, as opposed to the usual hamming metric. We obtain essentially optimal-rate codes over matrices, and provide an efficient decoding algorithm. We obtain codes over tensors as well, with poorer rate, but still with efficient decoding.

Contents

1   Introduction

We start with a motivating example. Let $\mathbf{x}$ and $\mathbf{y}$ be vectors of n variables each. Let M be an $n \times n$ matrix (over some field, say $\mathbb{R}$), and define the quadratic form

$$f_M(\mathbf{x}, \mathbf{y}) \stackrel{\text{def}}{=} \mathbf{x}^\dagger M \mathbf{y} \ .$$

Suppose now that we are given an oracle to f_M, that can evaluate f_M on inputs $(\mathbf{x}, \mathbf{y})$ that we supply. The type of question we consider is: how many (deterministically chosen) evaluations of f_M must we make in order to determine whether A is non-zero?

It is not hard to show that n^2 evaluations to f_M are necessary and sufficient to determine whether A is non-zero. The question becomes more interesting when we are promised that $\text{rank}(M) \leq r$. That is, given that $\text{rank}(M) \leq r$, can we (deterministically) determine whether $M = 0$ using $\ll n^2$ evaluations of f_M? It is not hard to show that there (non-explicitly) *exist* $\approx 2nr$ evaluations to determine whether $M = 0$, and one of the new results in this paper is to give an *explicit* construction of $2nr$ such evaluations (over $\mathbb{R}$).

We also consider various generalizations of this problem. The first generalization is to move from matrices (which are in a sense 2 dimensional) to the more general notion of *tensors* (which are in a sense d-dimensional). That is, a tensor is a map $T : [n]^d \to \mathbb{F}$ and like a matrix we can define a polynomial

$$f_T(x_{1,1}, \ldots, x_{1,n}, \ldots, x_{d,1}, \ldots, x_{d,n}) \stackrel{\text{def}}{=} \sum_{i_1, \ldots, i_d \in [n]} T(i_1, \ldots, i_d) \prod_{j=1}^{d} x_{j, i_j} \ .$$

As with matrices, tensors have a notion of rank (defined later), and we can ask: given that $\text{rank}(T) \leq r$ how many (deterministically chosen) evaluations of f_T are needed to determine whether $T = 0$. As $T = 0$ iff $f_T = 0$, we see that this problem is an instance of *polynomial identity testing*, which asks: given oracle access to a polynomial f that is somehow "simple", how many (deterministically chosen) queries to f are needed to determine whether $f = 0$?

The above questions ask whether a certain matrix or tensor is zero. However, we can also ask for more, and seek to reconstruct this matrix/tensor fully. That is, how many (deterministically chosen) evaluations to f_M are needed to determine M? This question can be seen to be related to compressed sensing and sparse recovery, where the goal is to reconstruct a "simple" object from "few" measurements. In this case, "simple" refers to the matrix being low-rank, as opposed to a vector being sparse. As above, it is not hard to show that there *exist* $\approx 4nr$ evaluations that determine M, and this paper gives an *explicit* construction of $4nr$ such evaluations, as well as an efficient algorithm to reconstruct M from these evaluations.

We will now place this work in a broader context by providing background on polynomial identity testing, compressed sensing and low-rank recovery, and the theory of rank-metric codes.

1.1   Polynomial Identity Testing

Polynomial identity testing (PIT) is the problem of deciding whether a polynomial (specified by an arithmetic circuit) computes the identically zero polynomial. The obvious deterministic algorithm that completely expands the polynomial unfortunately takes exponential time. This is in contrast to the fact that there are several (quite simple) randomized algorithms that solve this problem quite efficiently. Further, some of these randomized algorithms treat the polynomial as a *black-box*, so that they only use the arithmetic circuit to evaluate the polynomial on chosen points, as opposed

to a *white-box* algorithm which can examine the internal structure of the circuit. Even in the white-box model, no efficient deterministic algorithms are known for general circuits.

Understanding the deterministic complexity of PIT has come to be an important problem in theoretical computer science. Starting with the work of Kabanets and Impagliazzo [KI04], it has been shown that the existence of efficient deterministic (white-box) algorithms for PIT has a tight connection with the existence of explicit functions with large circuit complexity. As proving lower bounds on circuit complexity is one of the major goals of theoretical computer science, this has led to much research into PIT.

Stronger connections are known when the deterministic algorithms are black-box. For, any such algorithm corresponds to a *hitting set*, which is a set of evaluation points such that any small arithmetic circuit computing a non-zero polynomial must evaluate to non-zero on at least one point in the set. Heintz and Schnorr [HS80], as well as Agrawal [Agr05], showed that any deterministic black-box PIT algorithm very easily yields explicit polynomials that have large arithmetic circuit complexity. Moreover, Agrawal and Vinay [AV08] showed that a deterministic construction of a polynomial size hitting set for arithmetic circuits of depth-4 gives rise to a quasi-polynomial sized hitting set for general arithmetic circuits. Thus, the black-box deterministic complexity of PIT becomes interesting even for constant-depth circuits. However, currently no polynomial size hitting sets are known for general depth-3 circuits. Much of recent work on black-box deterministic PIT has identified certain subclasses of circuits for which small hitting sets can be constructed, and this work fits into that paradigm. See [SY10] for a survey of recent results on PIT.

One subclass of depth-3 circuits is the model of *set-multilinear* depth-3 circuits, first introduced by Nisan and Wigderson [NW96]. Raz and Shpilka [RS05] gave a polynomial-time white-box PIT algorithm for non-commutative arithmetic formulas, which contains set-multilinear depth-3 circuits as a subclass. However, no polynomial-time black-box deterministic PIT algorithm is known for set-multilinear depth-3 circuits. The best known black-box PIT results for the class of set-multilinear circuits, with top fan-in $\leq r$ and degree d, are hitting sets of size $\min(n^d, \mathsf{poly}((nd)^r))$, where the first part of bound comes from a simple argument (presented in Lemma 3.11), and the second part of the bound ignores that we have set-multilinear polynomials, and simply uses the best known hitting sets for so-called $\Sigma\Pi\Sigma(k)$ circuits as established by Saxena and Seshadhri [SS11]. For non-constant d and r, these bounds are super-polynomial. Improving the size of these hitting sets is the primary motivation for this work.

To connect PIT for set-multilinear depth-3 circuits with the above questions on matrices and tensors, we now note that any such circuit of top fan-in $\leq r$, degree d, on dn variables (and thus size $\leq dnr$), computes a polynomial f_T, where T is an $[n]^d$ tensor of rank $\leq r$. Conversely, any such f_T can be computed by such a circuit. Thus, constructing better hitting sets for this class of circuits is exactly the question of finding smaller sets of (deterministically chosen) evaluations to f_T to determine whether $T = 0$.

1.2   Low-Rank Recovery and Compressed Sensing

Low-rank Recovery (LRR) asks (for matrices) to recover an $n \times n$ matrix M from few *measurements* of M. Here, a measurement is some inner product $\langle M, H \rangle$, where H is an $n \times n$ matrix and the inner product $\langle \cdot, \cdot \rangle$ is the natural inner product on n^2 long vectors. This can be seen as the natural generalization of the *sparse recovery* problem, which asks to recover sparse vectors from few linear measurements. For, over matrices, our notion of sparsity is simply that of being low-rank.

Sparse recovery and compressed sensing are active areas of research, see for example [CSw]. Much of this area focuses on constructing distributions of measurements such that the unknown sparse vector can be recovered efficiently, with high probability. Also, it is often assumed that the

sequence of measurements will not depend on any of the measurement results, and this is known as *non-adaptive sparse recovery*. We note that Indyk, Price and Woodruff [IPW11] showed that *adaptive sparse recovery* can outperform non-adaptive measurements in certain regimes. Much of the existing work also focuses on efficiency concerns, and various algorithms coming from convex programming have been used. As such, these algorithms tend to be stable under noise, and can recover approximations to the sparse vector (and can even do so only if the original vector was approximately sparse). One of the initial achievements in this field is an efficient algorithm for recovery of a k-sparse[1] approximation of n-entry vector in $\mathcal{O}(k \log(n/k))$ measurements [CRT05].

Analogous questions for low-rank recovery have also been explored (for example, see [lrr] and references there in). Initial work (such as [CT09, CP09]) asked the question of low-rank *matrix completion*, where entries of a low-rank matrix M are revealed individually (as opposed measuring linear combinations of matrix entries). It was shown in these works that for an $n \times n$ rank $\leq r$ matrix that $\mathcal{O}(nr\mathsf{polylog}n)$ noisy samples suffice for *nuclear-norm minimization* to complete the matrix efficiently. Further works (such as [ENP11]) prove that a randomly chosen set of measurements (with appropriate parameters) gives enough information for low-rank recovery, other works (such as [CP11, RFP10]) giving explicit conditions on the measurements that guarantee that the nuclear norm minimization algorithm works, and finally other works seek alternative algorithms for certain ensembles of measurements (such as[2] [KOH11]). As in the sparse recovery case, most of these work seek stable algorithms that can deal with noisy measurements as well as matrices that are only approximately low-rank. Finally, we note that some applications (such as quantum state tomography) have additional requirements for their measurements (for example, they should be easy to prepare as quantum states) and some work has gone into this as well [GLF+10, Gro09].

We now make a crucial observation which shows that black-box PIT for the quadratic form f_M is actually very closely related to low-rank recovery of M. That is, note that $f_M(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\dagger M \mathbf{y} = \langle M, \mathbf{x}^\dagger \mathbf{y} \rangle$. That is, an evaluation of f_M corresponds to a measurement of M, and in particular this measurement is realized as a rank-1 matrix. Thus, we see that any low-rank-recovery algorithm that only uses rank-1 measurement can also determine if M is non-zero, and thus also performs PIT for quadratic forms. Conversely, suppose we have a black-box PIT algorithm for rank $\leq 2r$ quadratic forms. Note then that for any M, N with rank $\leq r$, $M - N$ has rank $\leq 2r$. Thus, if $M \neq N$ then f_{M-N} will evaluate to non-zero on some point in the hitting set. As $f_{M-N} = f_M - f_N$, it follows that a hitting set for rank $\leq 2r$ matrices will distinguish M and N. In particular, this shows that information-theoretically any hitting set for rank $\leq 2r$ matrices is also an LRR set. Thus, in addition to constructing hitting sets for the quadratic forms f_M, this paper will also use those hitting sets as LRR sets, and also give efficient LRR algorithms for these constructions.

1.3   Rank-Metric Codes

Most existing work on LRR has focused on random measurements, whereas the interesting aspect of PIT is to develop deterministic evaluations of polynomials. As the main motivation for this paper is to develop new PIT algorithms, we will seek deterministic LRR schemes. Further, we will want results that are field independent, and so this work will focus on noiseless measurements (and matrices that are exactly of rank $\leq r$). In such a setting, LRR constructions are very related to *rank-metric codes*. These codes (related to *array codes*), are error-correcting codes where the messages are matrices (or tensors) and the normal notion of distance (the Hamming metric) is replaced by the rank metric (that is, the distance of matrices M and N is $\mathrm{rank}(M - N)$). Over matrices, these

---

[1] A vector is k-sparse if it has at most k non-zero entries.

[2] Interestingly, [KOH11] use what they call *subspace expanders* a notion that was studied before in a different context in theoretical computer science and mathematics under the name of *dimension expanders* [LZ08, DS08].

codes were originally introduced independently by Gabidulin, Delsarte and Roth [GK72, Gab85b, Gab85a, Del78, Rot91]. They showed, using ideas from BCH codes, how to get optimal (that is, meeting an analogue of the Singleton bound) rank-metric codes over matrices, as well as how to decode these codes efficiently. A later result by Meshulam [Mes95] constructed rank-metric codes where every codeword is a Hankel matrix. Roth [Rot91] also showed how to construct rank-metric codes from *any* hamming-metric code, but did not provide a decoding algorithm. Later, Roth [Rot96] considered rank-metric codes over tensors and gave decoding algorithms for a constant number of errors. Roth also discussed analogues to the Gilbert-Varshamov and Singleton bounds in this regime. This alternate metric is motivated by *crisscross errors* in data storage scenarios, where corruption can occur in bursts along a row or column of a matrix (and are thus rank-1 errors).

We now explain how rank-metric codes are related to LRR. Suppose we have a set of matrices $\mathcal{H}$ which form a set of (non-adaptive, deterministically chosen) LRR measurements that can recover rank $\leq r$ matrices. Define the code $\mathcal{C}$ as the set of matrices orthogonal to each matrix in $\mathcal{H}$. Thus, $\mathcal{C}$ is a linear code. Further, given some $M \in \mathcal{C}$ and E such that $\text{rank}(E) \leq r$, it follows that $\mathcal{H}(M + E) = \mathcal{H}E$ (where we abuse notation and treat M and E as n^2-long vectors, and $\mathcal{H}$ as an $|\mathcal{H}| \times n^2$ matrix). That $\mathcal{H}$ is an LRR set means that E can be recovered from the measurements $\mathcal{H}E$. Thus the code $\mathcal{C}$ can correct r errors (and has minimum distance $\geq 2r + 1$, by a standard coding theory argument, as encapsulated in Lemma 8.4). Similarly, given a rank-metric code $\mathcal{C}$ that can correct up to rank $\leq r$ errors, the parity checks of this code define an LRR scheme. Thus, a small LRR set is equivalent to a rank-metric code with good rate.

The previous subsection showed the tight connection between LRR and PIT. Via the above paragraph, we see that hitting sets for quadratic forms are equivalent to rank-metric codes, when the parity check constraints are restricted to be rank 1 matrices.

1.4   Reconstruction of Arithmetic Circuits

Even more general than the PIT and LRR problems, we can consider the problem of reconstruction of general arithmetic circuits only given oracle access to the evaluation of that circuit. This is the arithmetic analog of the problem of learning a function using membership queries. For more background on reconstruction of arithmetic circuits we refer the reader to [SY10]. Just as with the PIT and LRR connection, PIT for a specific circuit class gives information-theoretic reconstruction for that circuit class. As we consider the PIT question for tensors, we can also consider the reconstruction problem.

The general reconstruction problem for tensors of degree d and rank r was considered before in the literature [BBV96, BBB+00, KS06] where learning algorithms were given for any value of r. However, those algorithms are inherently randomized. Also of note is that the algorithms of [BBB+00, KS06] output a *multiplicity automata*, which in the context of arithmetic circuits can be thought of as an *arithmetic branching program*. In contrast, the most natural form of the reconstruction question would be to output a degree d tensor.

1.5   Our Results

In this subsection we informally summarize our results. We again stress that our results handle matrices of exactly rank $\leq r$, and we consider non-adaptive, deterministic measurements. The culminating result of this work is the connection showing that low-rank recovery reduces to performing sparse-recovery, and that we can use dual Reed-Solomon codes to instantiate the sparse-recovery oracle to achieve a low-rank recovery set that only requires rank-1 (or even sparse) measurements. We find the fact that we can transform an algorithm for a combinatorial property (recovering sparse

signals) to an algorithm for an algebraic property (recovering low-rank matrices) quite interesting.

Hitting Sets for Matrices and Tensors We begin with constructions of hitting sets for matrices, so as to get black box PIT for quadratic forms. By improving a construction of rank-preserving matrices from Gabizon-Raz [GR08], we are able to show the following result, which we can then leverage to construct hitting sets.

Theorem (Theorem 5.1). *Let $n \geq r \geq 1$. Let $\mathbb{F}$ be a "large" field, and let $g \in \mathbb{F}$ have "large" multiplicative order. Let M be an $n \times n$ matrix of rank $\leq r$ over $\mathbb{F}$. Let $\hat{f}_M(x, y) = \mathbf{x}^\dagger M \mathbf{y}$ be the bivariate polynomial defined by the vectors $\mathbf{x} \in \mathbb{F}^n$ and $\mathbf{y} \in \mathbb{F}^n$ such that[3] $(\mathbf{x})_i = x^i$ and $(\mathbf{y})_i = y^i$.*
  Then M is non-zero iff one of the univariate polynomials $\hat{f}_M(x, x), \hat{f}_M(x, gx), \ldots, \hat{f}_M(x, g^{r-1}x)$ is non-zero.

Intuitively this says that we can test if the quadratic form f_M is zero by testing whether each of r univariate polynomials are zero. As these univariate polynomials are of degree $< 2n$, it follows that we can interpolate them fully using $2n$ evaluations. As such a univariate polynomial is zero iff all of these evaluations are zero, this yields a $2nr$ sized hitting set. While this only works for "large" fields, we can combine this with results on simulation of large fields (see Section 6.3) to derive results over any field with some loss. This is encapsulated in the next results for black-box PIT, where the log factors are unnecessary over large fields.

Theorem (Corollaries 6.13 and 6.17). *Let $n \geq r \geq 1$. Let $\mathbb{F}$ be any field, then there is a $\mathsf{poly}(n)$-explicit[4] hitting set for $n \times n$ matrices of rank $\leq r$, of size $\mathcal{O}(nr \lg^2 n)$.*

Theorem (Corollary 6.18). *Let $n, r \geq 1$ and $d \geq 2$. Let $\mathbb{F}$ be any field, then there is a $\mathsf{poly}((nd)^d, r^{\lg d})$-explicit hitting set for $[n]^d$ tensors of rank $\leq r$, of size $\mathcal{O}(dnr^{\lg d} \cdot (d \lg(nd))^d)$.*

If $\mathbb{F}$ is large enough then the $\mathcal{O}((d \lg(nd))^d)$ term is unnecessary. In such a situation, this is a quasi-polynomial sized hitting set, improving on the $\min(n^d, \mathsf{poly}((nd)^r))$ sized hitting set achievable by invoking the best known results for $\Sigma\Pi\Sigma(k)$ circuits [SS11]. However, this hitting set is not as explicit as the construction of [SS11] since it takes at least n^d time to compute, as opposed to $\mathsf{poly}(n, d, r)$. Nevertheless, although it takes $\mathsf{poly}((nd)^d, r^{\lg d})$ time to construct the set, the fact that it is of quasi-polynomial size is quite interesting and novel. Indeed, in general it is not clear at all how to construct a quasi-polynomial sized hitting set for general circuits (or just for depth-3 circuits), when one is allowed even an $\exp(nd)$ construction time (where n is the number of variables, and d is the degree of the output polynomial). We note that this result improves on the two obvious hitting sets seen in Lemmas 3.11 and 3.13. The first gives n^d tensors in the hitting set and is $\mathsf{polylog}(n, d, r)$-explicit while the second gives a set of size $\approx dnr$ while not being explicit at all. The above result non-trivially interpolates between these two results. Finally, we mention that in Remark 6.9 we explain how one can achieve (roughly) a $\mathsf{poly}(r(dn)^{\sqrt{d}})$-constructible hitting set of the same size. As this is a somewhat mild improvement (this is still not the explicitness that we were looking for) we only briefly sketch the argument.

Low-Rank Recovery As mentioned in the previous section, black-box PIT results imply LRR constructions in an information theoretic sense. Thus, the above hitting sets imply LRR constructions but the algorithm for recovery is not implied by the above result. To yield algorithmic

---

[3]In this paper, vectors and matrices are indexed from zero, so $\mathbf{x} = (1, x, x^2, \ldots, x^{n-1})^\dagger$.

[4]A $n \times n$ matrix is t-explicit if each entry can be (deterministically) computed in t steps, where field operations are considered unit cost.

results, we actually establish a stronger claim. That is, we first show that the above hitting sets embed a natural sparse-recovery set arising from the dual Reed-Solomon code. Then we develop an algorithm that shows that *any* sparse-recovery set gives rise to a low-rank-recovery set, and that recovery can be performed efficiently given an oracle for sparse recovery. This connection (in the context that any error-correcting code in the hamming metric yields an error-correcting code in the rank-metric) was independently made by Roth [Rot91] (see Theorem 3), who did not give a recovery procedure for the resulting LRR scheme. The next theorem, which is the main result of the paper, shows this connection is also efficient with respect to recovery.

Theorem (Theorem 7.19). *Let $n \geq r \geq 1$. Let $\mathcal{V}$ be a set of (non-adaptive) measurements for $2r$-sparse-recovery for n-long vectors. Then there is a $\mathsf{poly}(n)$-explicit set $\mathcal{H}$, which is a (non-adaptive) rank $\leq r$ low-rank-recovery set for $n \times n$ matrices, with a recovery algorithm running in time $\mathcal{O}(rn^2 + n\tau)$, where τ is the amount of time needed to do sparse-recovery from $\mathcal{V}$. Further, $|\mathcal{H}| = 2n|\mathcal{V}|$, and each matrix in $\mathcal{H}$ is n-sparse.*

This result shows that sparse-recovery and low-rank recovery (at least in the exact case) are very closely connected. Interestingly, this shows that sparse-recovery (which can be regarded as a combinatorial property) and low-rank recovery (which can be regarded as an algebraic property) are tightly connected. Many fruitful connections have taken this form, such as in spectral graph theory, and perhaps the connection presented here will yield yet further results.

Also, the algorithm used in the above result is purely linear-algebraic, in contrast to the convex optimization approaches that many compressed sensing works use. However, we do not know if the above result is stable to noise, and regard this issue as an important question left open by this work.

When the above result is combined with our hitting set results, we achieve the following LRR scheme for matrices (and an LRR scheme for tensors, with parameters similar to Corollary 6.18 mentioned above, and Corollary 8.6 mentioned below, is derived in Corollary 8.2).

Theorem (Corollary 7.26). *Let $n \geq r \geq 1$. Over any field $\mathbb{F}$, there is an $\mathsf{poly}(n)$-explicit set $\mathcal{H}$, of $\mathcal{O}(rn \lg^2 n)$ size, such that measurements against $\mathcal{H}$ allow recovery of $n \times n$ matrices of rank $\leq r$ in time $\mathsf{poly}(n)$. Further, the matrices in $\mathcal{H}$ can be chosen to be all rank 1, or all n-sparse.*

We note again that over large fields these logarithmic factors are seen to be unneeded.

Some prior work [GK72, Gab85b, Gab85a, Del78, Rot91] on LRR focused on finite fields, and as such based their results on BCH codes. The above result is based on (dual) Reed-Solomon codes, and as such works over any field (when combined with results allowing simulation of large fields by small fields). Other prior work [RFP10] on exact LRR permitted randomized measurements, while we achieve deterministic measurements.

Further, we are able to do LRR with measurements that are either all n-sparse, or all rank-1. As Roth [Rot91] independently observed, the n-sparse LRR measurements can arise from any (hamming-metric) error-correcting code (but he did not provide decoding). Tan, Balzano and Draper [TBD11] showed that random $(n \lg n)$-sparse measurements provide essentially the same low-rank recovery properties as random measurements. Thus, our results essentially achieve this deterministically.

We further observe that a specific code (the dual Reed-Solomon code) allows a change of basis for the measurements, and in this new basis the measurements are all rank 1. Recht et al. [RFP10] asked whether low-rank recovery was possible when the measurements were rank 1 (or "factored"), as such measurements could be more practical as they are simpler to generate and store in memory. Thus, our construction answers this question in the positive direction, at least for exact LRR.

Rank-Metric Codes Appealing to the connection between LRR and rank-metric codes, we achieve the following constructions of rank-metric codes.

Theorem (Corollary 8.5). *Let $\mathbb{F}$ be any field, $n \geq 1$ and $1 \leq r \leq n/2$. Then there are $\mathsf{poly}(n)$-explicit rank-metric codes with $\mathsf{poly}(n)$-time decoding for up to r errors, with parameters $[[n]^2, (n - 2r)^2 \cdot \mathcal{O}(\lg^2 n), 2r + 1]_{\mathbb{F}}$, and the parity checks on this code can be chosen to be all rank-1 matrices, or all n-sparse matrices.*

Earlier work on rank-metric codes over finite fields [GK72, Gab85b, Gab85a, Del78, Rot91] achieved $[[n]^2, n(n - 2r), 2r + 1]_{\mathbb{F}_q}$ rank-metric codes, with efficient decoding algorithms. These are optimal (meeting the analogue of the Singleton bound for rank-metric codes). However, these constructions only work over finite fields. While our code achieves a worse rate, its construction works over any field, and over infinite fields the $\mathcal{O}(\lg^2 n)$ term is unneeded. Further, Roth [Rot91] observed that the resulting $[[n]^2, (n - 2r)^2, 2r + 1]$ code is optimal (see discussion of his Theorem 3) over algebraically closed fields (which are infinite).

We are also able to give rank-metric codes over tensors, which can correct errors up to rank $\approx n^{d/\lg d}$ (out of a maximum n^{d-1}), while still achieving constant rate. The rank-metric code arising from the naive low-rank recovery of Lemma 3.11 never achieves constant rate, and prior work by Roth [Rot96] only gave decoding against a constant number of errors.

Theorem (Corollary 8.6). *Let $\mathbb{F}$ be any field, $n, r \geq 1$ and $d \geq 2$. Then there are $\mathsf{poly}((nd)^d, r^{\lg d})$-explicit rank-metric codes with $\mathsf{poly}((nd)^d, r^{\lg d})$-time decoding for up to r errors, with parameters $[[n]^d, n^d - \mathcal{O}(d^2 n r^{\lg d} \lg(dn)), 2r + 1]_{\mathbb{F}}$.*

We note here that our decoding algorithm will return the *entire* tensor, which is of size n^d. Trivially, any algorithm returning the entire tensor must take at least n^d time. In this case, the level of explicitness of the code we achieve is reasonable. However, a more desirable result would be for the algorithm to return a rank $\leq r$ representation of the tensor, and thus the n^d lower bound would not apply so that one could hope for faster decoding algorithms. Unfortunately, even for $d = 3$ an efficient algorithm to do so would imply $\mathsf{P} = \mathsf{NP}$. That is, if an algorithm (even one which is not a rank-metric decoding or low-rank recovery algorithm) could produce a rank $\leq r$ decomposition for any rank $\leq r$ tensor, then one could compute tensor-rank by as it is the minimum r such that the resulting rank $\leq r$ decomposition actually computes the desired tensor (this can be checked in $\mathsf{poly}(n^d)$ time). However, Håstad [Hås90] showed that tensor-rank (over finite fields) is NP-hard for any fixed $d \geq 3$. It follows that for any (fixed) $d \geq 3$, if one could recover (even in $\mathsf{poly}(n^d)$-time) a rank $\leq r$ tensor into its rank $\leq r$ decomposition, then $\mathsf{P} = \mathsf{NP}$. Thus, we only discuss recovery of a tensor by reproducing its entire list of entries, as opposed to its more concise representation.

Finally, we remark that in [Rot96] Roth discussed the question of decoding rank-metric codes of degree $d = 3$, gave decoding algorithms for errors of rank 1 and 2, and wrote that "Since computing tensor rank is an intractable problem, it is unlikely that we will have an efficient decoding algorithm ... otherwise, we could use the decoder to compute the rank of any tensor. Hence, if there is any efficient decoding algorithm, then we expect such an algorithm to recover the error tensor without necessarily obtaining its rank. Such an algorithm, that can handle any prescribed number of errors, is not yet known." Thus, our work gives the first such algorithm for tensors of degree $d > 2$.

1.6 Proof Overview

In this section we give proof outlines of the results mentioned so far.

Hitting Sets for Matrices The main idea for our hitting set construction is to reduce the question of hitting (non-zero) $n \times n$ matrices to a question of hitting (non-zero) $r \times r$ matrices. Once this reduction is performed, we can then run the naive hitting set of Lemma 3.11, which queries all r^2 entries. This can loosely be seen in analogy with the kernelization process in fixed-parameter tractability, where a problem depending on the input size, n, and some parameter, k, can be solved by first reducing to an instance of size $f(k)$, and then brute-forcing this instance.

To perform this kernelization, we first note that any $n \times n$ matrix M of rank exactly r can be written as $M = PQ^\dagger$, where P and Q are $n \times r$ matrices of rank exactly r. To reduce M to an $r \times r$ matrix, it thus suffices to reduce P and Q each to $r \times r$ matrices, denoted P' and Q'. As this reduction must preserve the fact that M is non-zero, we need that $P'Q' \neq 0$. We enforce this requirement by insisting that P' and Q' are also rank exactly r, so that $M' = P'Q'$ is also non-zero.

To achieve this rank-preservation, we turn to a lemma of Gabizon-Raz [GR08] (we note that this lemma has been used before for black-box PIT [KS08, SS11]). They gave an explicit family of $\mathcal{O}(nr^2)$-many $r \times n$-matrices $\{A_\ell\}_\ell$, such that for any P and Q of rank exactly r, at least one matrix A_ℓ from the family is such that $\mathrm{rank}(A_\ell P) = \mathrm{rank}(A_\ell Q) = r$. Translating this result into our problem, it follows that one of the $r \times r$ matrices $A_\ell M A_\ell^\dagger$ is full-rank. The (i,j)-th entry of $A_\ell M A_\ell^\dagger$ is $\langle M, (A_\ell)_i (A_\ell)_j^\dagger \rangle$, where $(A_\ell)_i$ is the i-th row of A_ℓ. It follows that querying each entry in these $r \times r$ matrices corresponds to a rank 1 measurement of M, and thus make up a hitting set. As there were $\mathcal{O}(nr^2)$ choices of ℓ and r^2 choices of (i,j), this gives a $\mathcal{O}(nr^4)$-sized hitting set.

To achieve a smaller hitting set, we use the following sequence of ideas. First, we observe that in the above, we can always assume $i = 0$. Loosely, this is because $A_\ell M A_\ell^\dagger$ is always full-rank, or zero. Thus, only the first row of $A_\ell M A_\ell^\dagger$ needs to be queried to determine this. Second, we improve upon the Gabizon-Raz lemma, and provide an explicit family of rank-preserving matrices with size $\mathcal{O}(nr)$. This follows from modifying their construction so the degree of a certain determinant is smaller. To ensure that the determinant is a non-zero polynomial, we show that it has a unique monomial that achieves maximal degree, and that the term achieving maximal degree has a non-zero coefficient as a Vandermonde determinant (formed from powers of an element g, which has large multiplicative order) is non-zero. Finally, we observe that the hitting set constraints can be viewed as a constraints regarding polynomial interpolation. This view shows that some of the constraints are linearly-dependent, and thus can be removed. Each of the above observations saves a factor of r in the size of the hitting set, and thus produces an $\mathcal{O}(nr)$-sized hitting set.

Low-Rank Recovery Having constructed hitting sets, Lemma 3.10 implies that the same construction yields low-rank-recovery sets. As this lemma does not provide a recovery algorithm, we provide one. To do so, we must first change the basis of our hitting set. That is, the hitting set $\mathcal{B}$ yields a set of constraints on a matrix M, and we are free to choose another basis for these constraints, which we call $\mathcal{D}$. The virtue of this new basis is that each constraint is non-zero only on some k-diagonal (the entries (i,j) such that $i + j = k$). It turns out that these constraints are the parity checks of a dual Reed-Solomon code with distance $\Theta(r)$. This code can be decoded efficiently using what is known as Prony's method [dP95], which was developed in 1795. We give an exposition in Section 7.1, where we show how to syndrome-decode this code up to half its minimum distance, counting erasures as half-errors. Thus, given a $\Theta(r)$-sparse vector (which can be thought of as errors from the vector $\mathbf{0}$) these parity checks impose constraints from which the sparse vector can be recovered. Put another way, our low-rank-recovery set naturally embeds a sparse-recovery set along each k-diagonal.

Thus, in designing a recovery algorithm for our low-rank recovery set, we do more and show how to recover from any set of measurements which embed a sparse-recovery set along each k-diagonal.

In terms of error-correcting codes, this shows that any hamming-metric code yields a rank-metric code over matrices, and that decoding the rank-metric code efficiently reduces to decoding the hamming-metric code.

To perform recovery, we introduce the notion of a matrix being in $(< k)$-upper-echelon form. Loosely, this says that $M^{(<k)}$, the entries (i,j) of the matrix with $i + j < k$, are in row-reduced echelon form. We then show that for any matrix M in $(< k)$-upper-echelon form, the k-diagonal is $2\operatorname{rank}(M)$-sparse. As an example, suppose $M^{(<k)}$ was entirely zero. It follows then that M is in $(< k)$-upper-echelon form. Further, the rows that have non-zero entries on the k-diagonal of M are then linearly-independent, as they form a triangular system. It follows that the k-diagonal can only have $\operatorname{rank}(M)$ non-zero entries. The more general case is slightly more complicated technically, but not conceptually. Thus, this echelon-form translates the notion of low-rank into the notion of sparsity.

The algorithm then follows naturally. We induct on k, first putting $M^{(<k)}$ into $(< k)$-upper-echelon form (using row-reduction), and then invoking a sparse-recovery oracle on the k-diagonal of M to recover it. This then yields $M^{(\leq k)}$, and we increment k. However, as described so far, the use of the sparse-recovery oracle is adaptive. We show that the row-reduction procedure can be understood such that the adaptive use of the sparse-recovery oracle can be simulated using non-adaptive calls to the oracle. More specifically, we will apply the measurements of the sparse-recovery oracle on each k-diagonal of M (which may not be sparse), and show how to compute the measurements of the adaptive algorithm (where the k-diagonals are sparse) from the measurements made. Putting these steps together, this shows that exact non-adaptive low-rank-recovery reduces to exact non-adaptive sparse-recovery. Instantiating this claim with our hitting sets from above gives a concrete low-rank-recovery set, with accompanied recovery algorithm.

Hitting Sets and Low-Rank Recovery for Tensors The results for matrices naturally generalize to tensors in the sense that an $[\![n]\!]^{2d}$ tensor can be viewed as an $[\![n^d]\!]^2$ matrix. However, we can do better. Specifically, the hitting set results were done via *variable reduction*, as encapsulated by Theorem 5.1, which shows that a rank $\leq r$ bivariate polynomial $f_M(x,y) = (1,x,x^2,\ldots,x^{n-1})M(1,y,y^2,\ldots,y^{n-1})^\dagger$ is zero iff a set of r univariate polynomials are all zero. Further, the degrees of these polynomials is only twice the original degree. As each univariate polynomial can be interpolated using $\mathcal{O}(n)$ measurements, this yields $\mathcal{O}(nr)$ measurements total. This motivates the more general idea of treating a degree d tensor as a d-variate polynomial, and showing that we can test whether this polynomial is zero by testing if a collection of d'-variate polynomials are zero, for $d' < d$. Recursing on this procedure then reduces the d-variate case to the univariate case, and the univariate case is brute-force interpolated.

The recursion scheme we develop for this is to show that a d-variate polynomial is zero iff r $d/2$-variate polynomials are zero, and this naturally leads to an $\mathcal{O}(dnr^{\lg d})$-sized hitting set. To prove its correctness, we show that the bivariate case (corresponding to matrices) applied to two groups of variables allows us to reduce to a single group of variables (with an increase in the number of polynomials to test). Finally, since we saw how to do low-rank recovery for matrices, and the tensor-case essentially only uses the matrix case, we can also turn this hitting set procedure into a low-rank recovery algorithm.

Simulation of Large Fields by Small Fields Most all of the results mentioned require a field of size $\approx \mathsf{poly}(n^d)$. When getting results over small fields, we show that, with some loss, we can simulate such large fields inside the hitting sets. We break-up each tensor H in the original hitting set into new tensors H_i such that for any $\mathbb{F}$-tensor T, $\langle T, H \rangle$ can be reconstructed from the set of

values $\{\langle T, \tilde{H}_i \rangle\}_i$. To do so, we use the well-known representation of a extension field $\mathbb{K}$ of $\mathbb{F}$ as a field of matrices over $\mathbb{F}$. As the entries of a rank-1 tensor are multiplications of d elements of $\mathbb{K}$, we can expand these multiplications out as iterated matrix multiplications, which yields $(\dim_{\mathbb{F}} \mathbb{K})^{d+1}$ terms to consider, each of which corresponds to some $\tilde{H}_i$.

Rank-Metric Codes The above techniques give the existence of low-rank-recovery sets (and corresponding algorithms) for tensors, over any field. Via the connections presented in Section 1.3, this readily yields rank-metric codes with corresponding parameters.

2 Notation

We now fix some notation. For a positive integer n we denote $[n] \overset{\mathrm{def}}{=} \{1, \ldots, n\}$ and $[\![n]\!] \overset{\mathrm{def}}{=} \{0, \ldots, n-1\}$. We use $\binom{S}{k}$ to denote the set of all subsets of S of size k. Given a set S of integers, we denote $n - S \overset{\mathrm{def}}{=} \{n - s : s \in S\}$. All logarithms will be base 2. Given a polynomial $f \in \mathbb{F}[x_1, \ldots, x_m]$, $\deg(f)$ will denote the *total degree* of f, and $\deg_{x_i}(f)$ will denote the *individual degree* of f in the variable x_i. That is, the polynomial xy has total degree 2 and individual degree 1 in the variable x and individual degree 0 in the variable z. Given a monomial $\mathbf{x}^\alpha$, $\mathfrak{C}_{\mathbf{x}^\alpha}(f)$ will denote the coefficient of $\mathbf{x}^\alpha$ in the polynomial f.

Vectors, matrices, and tensors will all begin indexing from 0, instead of from 1. The number n will typically refer to the number of rows of a matrix, and m the number of columns. I_n will denote the $n \times n$ identity matrix. Denote $E_{i,j}$ to be the $n \times n$ square matrix with its (i, j)-th entry being 1, and all other entries being zero. A vector is k-sparse if it has at most k non-zero entries. Given a matrix A, $A^\dagger$ will denotes its transpose. Given a vector $\mathbf{x} \in \mathbb{F}^n$, $|\mathbf{x}| \overset{\mathrm{def}}{=} n$.

A list of n values in $\mathbb{F}$ is *t-explicit* if each entry can be computed in t steps, where we allow operations in $\mathbb{F}$ to be done at unit cost.

Frequently throughout this paper we will divide a matrix into its diagonals, which we define as the entries (i, j) where $i + j$ is constant. The following notation will make this discussion more convenient.

Notation 2.1. *Let M be an $n \times m$ matrix. The k-**diagonal of** M is the set of entries $\{M_{i,j}\}_{i+j=k}$. The $(\leq k)$-**diagonals of** M is the set of entries $\{M_{i,j}\}_{i+j \leq k}$. The $(< k)$-**diagonals of** M is the set of entries $\{M_{i,j}\}_{i+j < k}$*
$M^{(k)}$, $M^{(\leq k)}$ and $M^{(<k)}$ will denote the k-diagonal, $(\leq k)$-diagonals and $(< k)$-diagonals of M, respectively.

This notation will be frequently abused, in that a diagonal will refer to a set of positions in a matrix in addition to referring to the values in those positions. However, *the main diagonal* of a matrix will refer to the entries $\{(i, i)\}_i$ of that matrix.

3 Preliminaries

In this section we formally define tensors as well as the PIT and LRR problems. We first discuss tensors, and their notion of rank. Rank-metric codes will be defined and discussed in Section 8. Recall that we index starting at 0, so we will use the product space $[\![n]\!]^d$ instead of $[n]^d$ for the domains of tensors.

Definition 3.1. *A **tensor** over a field $\mathbb{F}$ is a function $T : \prod_{j=1}^d [\![n_j]\!] \to \mathbb{F}$. It is said to have degree d and size $(n_1, \ldots, n_d)$. If all of the n_j are equal to n, then T is said to have size $[\![n]\!]^d$.*

Given two tensor T_1, T_2 of size $\prod_{j=1}^d [\![n_j]\!]$, $\langle T_1, T_2\rangle \overset{\text{def}}{=} \sum_{i_j \in [\![n_j]\!]} T_1(i_1,\ldots,i_d)T_2(i_1,\ldots,i_d)$.

Note that the above inner product is the natural inner product when regarding a $\prod_{j=1}^d [\![n_j]\!]$ tensor as a vector of dimension $\prod_{j=1}^d n_j$. We now define the notion of rank. Loosely, a tensor is rank 1 if it can be "factored" along each dimension, and a tensor is rank $\leq r$ if it can be expressed as the sum of $\leq r$ rank 1 tensors.

Definition 3.2. *A tensor $T : \prod_{j=1}^d [\![n_j]\!] \to \mathbb{F}$ is **rank-one** if for $j \in [d]$ there are vectors $\mathbf{v}_j \in \mathbb{F}^{n_j} \setminus \{\mathbf{0}\}$ such that $T = \otimes_{j=1}^d \mathbf{v}_j$. That is, for all $i_j \in [n_j]$, $T(i_1,\ldots,i_d) = \prod_{j=1}^d \mathbf{v}_j(i_j)$ where $\mathbf{v}_j(i_j)$ denotes the i_j-th coordinate of $\mathbf{v}_j$.*

*The **rank** of a tensor $T : \prod_{j=1}^d [\![n_j]\!] \to \mathbb{F}$, is defined as the minimum number of terms in a summation of rank-1 tensors expressing T, that is,*

$$\text{rank}_{\mathbb{F}}(T) = \min \left\{ r : T = \sum_{\ell=1}^r \otimes_{j=1}^d \mathbf{v}_{j,\ell}, \ \mathbf{v}_{j,\ell} \in \mathbb{F}^{n_j} \right\} .$$

As one might hope, when $d = 2$ the above definitions reduce to the definition of a matrix, and matrix-rank, respectively. Further, the inner-product is then their Frobenius inner product. That is, $\langle M_1, M_2 \rangle = \text{Trace}(M_1 M_2^\dagger)$.

We now define the polynomial of a tensor.

Definition 3.3. *Let $T : \prod_{j=1}^d [\![n_j]\!] \to \mathbb{F}$ be a tensor, and let $\mathbf{x}_1, \ldots, \mathbf{x}_d$ be vectors of variables, so $\mathbf{x}_j = (x_{j,0}, \ldots, x_{j,n_j-1})$ for all $j \in [d]$. Then define*

$$f_T(\mathbf{x}_1, \ldots, \mathbf{x}_d) \overset{\text{def}}{=} \sum_{i_j \in [\![n_j]\!]} T(i_1, \ldots, i_d) \prod_{j=1}^D x_{j,i_j} = \langle T, \mathbf{x}_1 \otimes \cdots \otimes \mathbf{x}_d \rangle ,$$

and define the d-variate polynomial

$$\hat{f}_T(x_1, \ldots, x_d) \overset{\text{def}}{=} \sum_{i_j \in [\![n_j]\!]} T(i_1, \ldots, i_d) \prod_{j=1}^D x_j^{i_j} = f_T(\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_d) ,$$

where $(\hat{\mathbf{x}}_j)_i \overset{\text{def}}{=} x_j^i$.

Note that the second equality in the first equation of the above definition follows from the definition of the inner product over tensors. As a matrix M is also a tensor, we will also use this notation when considering the polynomial $f_M(\mathbf{x}, \mathbf{y}) \overset{\text{def}}{=} \mathbf{x}^\dagger M \mathbf{y}$, as the above definition readily generalizes the notion of a quadratic form. Note that $\hat{f}_T$ allows us to consider any d-variate polynomial to be a tensor, and the *rank* of such a polynomial will simply be the rank of the corresponding tensor.

We now show the connection of these polynomials f_T to set-multilinear depth-3 circuits. We do not seek to define all of the relevant terms in this notion, and instead refer the reader to the recent survey [SY10], and will simply define the subclass we are interested in.

Definition 3.4. *For $j \in [d]$, let $\mathbf{x}_j = (x_{j,0}, \ldots, x_{j,n-1})$ be vectors of variables. A degree d, set-multilinear, $\Sigma\Pi\Sigma$ circuit with top fan-in r, is a polynomial of the following form*

$$C(\mathbf{x}_1, \ldots, \mathbf{x}_d) = \sum_{\ell=1}^r \prod_{j=1}^d \langle \mathbf{v}_{j,\ell}, \mathbf{x}_j \rangle$$

where each $\mathbf{v}_{j,\ell} \in \mathbb{F}^n$.

We now see the following connection between these circuits and tensors.

Lemma 3.5. *The polynomials computed by degree d set-multilinear $\Sigma\Pi\Sigma$ circuits, with top fan-in $\leq r$, on dn variables, are exactly the polynomials f_T, for tensors $T : [\![n]\!]^d \to \mathbb{F}$ with rank $\leq r$.*

Proof. $\Longleftarrow$: Suppose T is of rank $\leq r$, so $T = \sum_{\ell=1}^r \otimes_{j=1}^d \mathbf{v}_{j,\ell}$ for $\mathbf{v}_{j,\ell} \in \mathbb{F}^n$. Then $f_T = \langle T, \mathbf{x}_1 \otimes \cdots \otimes \mathbf{x}_d \rangle = \sum_{\ell=1}^r \langle \otimes_{j=1}^d \mathbf{v}_{j,\ell}, \mathbf{x}_1 \otimes \cdots \otimes \mathbf{x}_d \rangle = \sum_{\ell=1}^r \prod_{j=1}^d \langle \mathbf{v}_{j,\ell}, \mathbf{x}_j \rangle$, and this final polynomial is computed as a set-multilinear $\Sigma\Pi\Sigma$ circuit.

$\Longrightarrow$: This argument is simply the reverse of the above. $\qquad\square$

We also get the following result for the polynomial $\hat{f}_T$.

Lemma 3.6. *For $T : [\![n]\!]^d \to \mathbb{F}$ with rank $\leq r$, $\hat{f}_T(x_1, \ldots, x_d) = \sum_{\ell=1}^r \prod_{j=1}^d p_{j,\ell}(x_j)$, where $\deg p_{j,\ell} < n$.*

Proof. As T is rank $\leq r$, $T = \sum_{\ell=1}^r \otimes_{j=1}^d \mathbf{v}_{j,\ell}$ for $\mathbf{v}_{j,\ell} \in \mathbb{F}^n$. Then $\hat{f}_T = f_T(\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_d) = \sum_{\ell=1}^r \prod_{j=1}^d \langle \mathbf{v}_{j,\ell}, \hat{\mathbf{x}}_j \rangle$. Taking $p_{j,\ell}(x_j) \stackrel{\text{def}}{=} \langle \mathbf{v}_{j,\ell}, \hat{\mathbf{x}}_j \rangle$ yields the result. $\qquad\square$

Recall that, as discussed in the introduction, set-multilinear $\Sigma\Pi\Sigma$ circuits have a white-box polynomial-time PIT algorithm due to Raz and Shpilka [RS05] but no known polynomial-sized black-box PIT algorithm. By the above connection, this is the same as creating hitting sets for tensors, which we will now define.

Definition 3.7. *Let $\mathbb{K}$ be an extension of $\mathbb{F}$. A **hitting set** $\mathcal{H}$ **for** $\prod_{j=1}^d [\![n_j]\!]$ **tensors of rank** $\leq r$ **over** $\mathbb{F}$ is a set of points $\mathcal{H} \subseteq \prod_{j=1}^d (\mathbb{K}^{n_j})$ such that for any $T : \prod_{j=1}^d [\![n_j]\!] \to \mathbb{F}$ of rank $\leq r$, T is a non-zero iff there exists $(\mathbf{a}_1, \ldots, \mathbf{a}_d) \in \mathcal{H}$ such that $f_T(\mathbf{a}_1, \ldots, \mathbf{a}_d) \neq 0$.*

However, we saw in Definition 3.3 that evaluating f_T is equivalent to taking an inner product of T with a rank-1 tensor. This leads to the following equivalent definition.

Definition 3.8 (Reformulation of Definition 3.7)**.** *Let $\mathbb{K}$ be an extension of $\mathbb{F}$. A **hitting set** $\mathcal{H}$ **for** $\prod_{j=1}^d [\![n_j]\!]$ **tensors of rank** $\leq r$ **over** $\mathbb{F}$ is a set of rank-1 tensors $\mathcal{H} \subseteq \mathbb{K}^{\prod_{j=1}^d [\![n_j]\!]}$ such that for any $T : \prod_{j=1}^d [\![n_j]\!] \to \mathbb{F}$ of rank $\leq r$, T is a non-zero iff there exists $H \in \mathcal{H}$ such that $\langle T, H \rangle \neq 0$.*

*If $\mathcal{H}$ instead is not constrained to consist of rank-1 tensors, then we say $\mathcal{H}$ is an **improper hitting set**.*

As is common in PIT literature, we allow the use of the extension field $\mathbb{K}$, and in our case $|\mathbb{K}| \leq \mathsf{poly}(|\mathbb{F}|)$ will be sufficient. However, the results of Section 6.3 will show how to remove the need for $\mathbb{K}$ from our results (with some loss).

We now define our notion of a low-rank recovery set, extending Definition 3.8. Note that we drop here the restriction that the tensors must be rank 1.

Definition 3.9. *A set of tensors $\mathcal{R} \subseteq \mathbb{K}^{\prod_{j=1}^d [\![n_j]\!]}$ is an r-**low-rank-recovery set** if for every tensor $T : \prod_{j=1}^d [\![n_j]\!] \to \mathbb{F}$ with rank $\leq r$, T is uniquely determined by $\mathbf{y}$, where $\mathbf{y} \in \mathbb{K}^{\mathcal{R}}$ is defined by $y_R \stackrel{\text{def}}{=} \langle T, R \rangle$, for $R \in \mathcal{R}$.*

*An algorithm performs **recovery** from $\mathcal{R}$ if, for each such T, it recovers T given $\mathbf{y}$.*

We now show that, despite low-rank recovery being a stronger notion than a hitting set, hitting sets imply low-rank recovery with some loss in parameters, as seen by the following lemma.

Lemma 3.10. *If $\mathcal{H}$ is a (proper or improper) hitting-set for $\prod_{j=1}^{d} [\![n_j]\!]$ tensors of rank $\leq 2r$, then $\mathcal{H}$ is an r-low-rank-recovery set for $\prod_{j=1}^{d} [\![n_j]\!]$ tensors also.*

Proof. Let $A, B \in \mathbb{F}^{\prod_{j=1}^{d} [\![n_j]\!]}$ be two tensors of rank $\leq r$ such that their inner products with the tensors in $\mathcal{H}$ are the same. By linearity of the inner product, it follows then that the tensor $A - B$ has rank $\leq 2r$ and has zero inner product with each tensor in $\mathcal{H}$. As $\mathcal{H}$ is a hitting set, it follows that $A - B = 0$, and thus $A = B$. Therefore, tensors of rank $\leq r$ are determined by their inner products with $\mathcal{H}$ and thus $\mathcal{H}$ is an r-low-rank-recovery set. $\qquad\square$

We now discuss some trivial LRR results. The first result is the obvious low-rank recovery construction, which is extremely explicit but requires many measurements.

Lemma 3.11. *For $n \geq 1$, $d \geq 2$, there is a $\mathsf{polylog}(n, d, r)$-explicit r-low-rank-recovery set for $[\![n]\!]^d$ tensors, of size n^d. Further, recovery of T is possible in $\mathsf{poly}(n^d)$ time.*

Proof. For $\mathbf{i} = (i_1, \ldots, i_d) \in [\![n]\!]$, let the rank 1 tensor $R_{\mathbf{i}} : [\![n]\!]^d \to \mathbb{F}$ be the rank 1 tensor, which is the indicator function for the set $\{(i_1, \ldots, i_d)\}$. Thus, $\langle T, R_{\mathbf{i}} \rangle = T(i_1, \ldots, i_d)$. It follows that $T = 0$ iff each such inner product is zero, and further that recovery of T is possible (in $\mathsf{poly}(n^d)$ time). The explicitness of the recovery set is also clear. $\qquad\square$

We now will show that, via the probabilistic method, one can show that much smaller low-rank recovery sets exist. To do so, we first cite the following form of the Schwartz-Zippel Lemma.

Lemma 3.12 (Schwartz-Zippel Lemma [Sch80, Zip79]). *Let $f \in \mathbb{F}[x_1, \ldots, x_m]$ be a non-zero polynomial of total degree $\leq d$, and $S \subseteq \mathbb{F}$. Then $\mathrm{Pr}_{\mathbf{x} \in S^m}[f(\mathbf{x}) = 0] \leq d/|S|$.*

We now give a (standard) probabilistic method proof that small hitting sets exist (over finite fields). We present this not as a tight result, but as an example of what parameters one can hope to achieve.

Lemma 3.13. *Let $\mathbb{F}_q$ be the field on q elements. Let $n \geq 1$ and $q > d \geq 2$. Then there is a hitting set for $[\![n]\!]^d$ tensors of rank $\leq r$, of size $\leq dnr/\log_q(q/d) + 1 \approx dnr$. Further, there is an r-low-rank recovery set of size $\leq 2dnr/\log_q(q/d) + 2$.*

Proof. For any non-zero tensor $T : [\![n]\!]^d \to F$, f_T has degree d, and thus by the Schwartz-Zippel Lemma, for a random $\mathbf{a} \in \mathbb{F}_q^n$, $f_T(\mathbf{a}) = 0$ with probability at most d/q. There are at most q^{dnr} such non-zero tenors. By a union bound, it follows that k random points are not a hitting set for rank $\leq r$ tensors with probability at most $q^{dnr}(d/q)^k$, which is < 1 if $k > dnr/\log_q(q/d)$. The low-rank-recovery set follows from Lemma 3.10. $\qquad\square$

We now briefly remark on the tightness of the above result. The general case of tensors is not well understood, as it is not well-understood how many tensors there are of a given rank. For matrices, the situation is much more clear. In particular, Roth [Rot91] showed (using the language of rank-metric codes) that over finite fields the best (improper) hitting set for $n \times n$ matrices of rank $\leq r$ is of size nr, and over algebraically closed fields the best (improper) hitting set is of size $(2n - r)r$. As we will aim to be field independent, the second bound is more relevant, and we indeed match this bound (as seem in Theorem 5.10) with a proper hitting set.

Clearly, the above lemma is non-explicit. However, it yields a much smaller hitting set than the n^d result given in Lemma 3.11. Note that previous work (even for $d = 2$) on LRR and rank-metric codes did not focus on requiring that the measurements are rank-1 tensors, and thus cannot be

used for PIT. Given this lack of knowledge, this paper seeks to construct proper hitting sets, and low-rank-recovery sets, that are both explicit *and* small.

We remark that any explicit hitting set naturally leads to tensor rank lower bounds[5]. The following lemma, which can be seen as a special case of the more general results of Heintz-Schnorr [HS80] and Agrawal [Agr05], shows this connection more concretely.

Lemma 3.14. *Let $\mathcal{H}$ be a hitting set for $[\![n]\!]^d$ tensors of rank $\leq r$, such that $|\mathcal{H}| < n^d$. Then there is a $\mathsf{poly}(n^d, |\mathcal{H}|)$-explicit tensor of rank $> r$.*

Proof. Consider the constraints imposed on a tensor T by the system of equations $\langle T, \mathcal{H} \rangle = \mathbf{0}$. There are $|\mathcal{H}|$ constraints and n^d variables. It follows that there is a non-zero T solving this system. By the definition of a hitting set, it follows that $\mathrm{rank}(T) \not\leq r$. That T is explicit follows from Gaussian Elimination. $\qquad\square$

For $d = 2$, the above is less interesting, as matrix rank is well understood and we know many matrices of high rank. For $d \geq 3$, tensor rank is far less understood. For $d = 3$, the best known lower bounds for the rank of explicit tensors, over arbitrary fields, due to Alexeev, Forbes, and Tsimerman [AFT11], are $3n - \mathcal{O}(\lg n)$ (over $\mathbb{F}_2$, a lower bound of $3.52n$ is known, essentially due to Brown and Dobkin [BD80]). More generally, for any fixed d, no explicit tensors are known with tensor rank $\omega(n^{\lfloor d/2 \rfloor})$. The above lemma shows that constructing hitting sets is at least as hard as getting a lower bound on any specific tensor. In particular, constructing a hitting set for $[\![n]\!]^d$ tensors of rank $\leq r$ of size $\mathcal{O}(dnr^k)$ with $k < 2$ would yield new tensor rank lower bounds for odd d, in particular $d = 3$. Such lower bounds would imply new circuit lower bounds, using the results of Strassen [Str73] and Raz [Raz10]. Our results give a hitting set with $k \approx \lg d$, and we leave open whether further improvements are possible.

We will mention the definitions and preliminaries of rank-metric codes in Section 8.

3.1 Paper Outline

We briefly outline the rest of the paper. In Section 4 we give our improved construction of rank-preserving matrices, which were first constructed by Gabizon-Raz [GR08]. In Section 5 we then use this construction to give our reduction from bivariate identity testing to univariate identity testing (Section 5.1), which then readily yields our hitting set for matrices (Section 5.2). In Section 5.3 we show an equivalent hitting set, which is more useful for low-rank-recovery.

Section 6 extends the above results to tensors, where Section 6.1 reduces d-variate identity testing to univariate identity testing, and Section 6.2 uses this reduction to construction hitting sets for tensors. Finally, Section 6.3 shows how to extend these results to any field.

Low-rank recovery of matrices is discussed in Section 7. It is split into two parts. Section 7.1 shows how to decode dual Reed-Solomon codes, which we use as a sparse-recovery oracle. Section 7.2 shows how to, given any such sparse-recovery oracle, perform low-rank-recovery of matrices. Instantiating the oracle with dual Reed-Solomon codes gives our low-rank-recovery construction.

Section 8 shows how to extend our LRR algorithms to tensors, and how to use these results to construct rank-metric codes. Finally, Section 9 discusses some problems left open by this work.

---

[5]This connection, along with the connection to rank-metric codes mentioned earlier, can be put in a more broad setting: hitting sets (and thus lower-bounds) for circuits from some class $\mathcal{C}$ are in a sense equivalent to $\mathcal{C}$-metric linear codes. That is, codes where $\mathrm{dist}(x, y)$ is defined as the size of the smallest circuit whose truth table is the string $x - y$. We do not pursue this idea further in this work.

4 Improved Construction of Rank-preserving Matrices

In this section we will give an improved version of the Gabizon-Raz lemma [GR08] on the construction of rank-preserving matrices. The goal is to transform an r-dimensional subspace living in an n-dimensional ambient space, to an r-dimensional subspace living in an r-dimensional ambient space. We will later show (see Theorem 5.1) how to use such a transformation to reduce the problem of PIT for $n \times m$ matrices of rank $\leq r$ to the problem of PIT for $r \times r$ matrices of rank $\leq r$.

We first present the Gabizon-Raz lemma ([GR08], Lemma 6.1), stated in the language of this paper.

Lemma (Gabizon-Raz ([GR08], Lemma 6.1)). *Let $1 \leq r \leq n$. Let $M \in \mathbb{F}^{n \times r}$ be of rank r. Define $A_\alpha \in \mathbb{F}^{r \times n}$ by $(A_\alpha)_{i,j} = \alpha^{ij}$. Then there are $\leq nr^2$ values $\alpha \in \mathbb{F}$ such that $\mathrm{rank}(A_\alpha M) < r$.*

Our version of this lemma gives a set of matrices parameterized by α where there are only nr values of α that lead to $\mathrm{rank}(A_\alpha M) < r$. This extra factor of r allows us to achieve an $\mathcal{O}((n+m)r)$-sized hitting set for matrices instead of a $\mathcal{O}((n+m)r^2)$-sized hitting set. We comment more on the necessity of this improvement in Remark 5.3. We now state our version of this lemma. Our proof is very similar to that of Gabizon-Raz.

Theorem 4.1. *Let $1 \leq r \leq n$. Let $M \in \mathbb{F}^{n \times r}$ be of rank r. Let $\mathbb{K}$ be a field extending $\mathbb{F}$, and let $g \in \mathbb{K}$ be an element of order $\geq n$. Define $A_\alpha \in \mathbb{K}^{r \times n}$ by $(A_\alpha)_{i,j} = (g^i \alpha)^j$. Then there are $\leq nr - \binom{r+1}{2} < nr$ values $\alpha \in \mathbb{K}$ such that $\mathrm{rank}(A_\alpha M) < r$.*

Proof. We will now treat α as a variable, and thus refer to A_α simply as A. The matrix AM is an $r \times r$ matrix, and thus the claim will follow from showing that $\det(AM)$ is a non-zero polynomial in α of degree $\leq nr - \binom{r+1}{2}$. As $r \geq 1$, $nr - \binom{r+1}{2} < nr$.

To analyze this determinant, we invoke the Cauchy-Binet formula.

Lemma (Cauchy-Binet Formula, Lemma A.1). *Let $m \geq n \geq 1$. Let $A \in \mathbb{F}^{n \times m}$, $B \in \mathbb{F}^{m \times n}$. For $S \subseteq [\![m]\!]$, let A_S be the $n \times |S|$ matrix formed from A by taking the columns with indices in S. Let B_S be defined analogously, but with rows. Then*

$$\det(AB) = \sum_{S \in \binom{[\![m]\!]}{n}} \det(A_S) \det(B_S)$$

so that

$$\det(AM) = \sum_{S \in \binom{[\![n]\!]}{r}} \det(A_S) \det(M_S)$$

For $S = \{k_1, \ldots, k_r\}$,

$$\det(A_S) = \begin{vmatrix} (\alpha)^{k_1} & \cdots & (\alpha)^{k_r} \\ (g\alpha)^{k_1} & \cdots & (g\alpha)^{k_r} \\ \vdots & \ddots & \vdots \\ (g^{r-1}\alpha)^{k_1} & \cdots & (g^{r-1}\alpha)^{k_r} \end{vmatrix} = \begin{vmatrix} 1 & \cdots & 1 \\ g^{k_1} & \cdots & g^{k_r} \\ \vdots & \ddots & \vdots \\ (g^{k_1})^{r-1} & \cdots & (g^{k_r})^{r-1} \end{vmatrix} \cdot \alpha^{\sum_{\ell=1}^r k_\ell}$$

$$= \alpha^{\sum_{\ell=1}^r k_\ell} \prod_{1 \leq i < j \leq r} (g^{k_j} - g^{k_i})$$

By assumption the order of g is $\geq n$, so the elements $(g^k)_{0 \leq k < n}$ are distinct, implying that the above Vandermonde determinant is non-zero.

Further, we observe that $\deg_\alpha \det(A_S) = \sum_{k \in S} k$. As $S \in \binom{[\![n]\!]}{r}$, it follows that $\sum_{k \in S} k \leq \sum_{k=n-r}^{n-1} k = nr - \binom{r+1}{2}$, and thus $\deg_\alpha \det(AM) \leq nr - \binom{r+1}{2}$ also.

We now show $\det(AM)$ is not identically zero, as a polynomial in α. We show this by showing that there is no cancellation of terms at the highest degree of $\det(AM)$. That is, there is a unique set $S \in \binom{[\![n]\!]}{r}$ maximizing $\sum_{k \in S} k$ subject to $\det(M_S) \neq 0$. This is proven by the following lemma.

Lemma 4.2. *Let $m \geq n \geq 1$. Let M be a $n \times m$ matrix of rank n. For $S \subseteq [\![m]\!]$, denote M_S as the $n \times |S|$ matrix formed by taking the columns in M (in order) whose indices are in S. Denote $w(S) \overset{\text{def}}{=} \sum_{s \in S} s$. Then there is a unique set $S \in \binom{[\![m]\!]}{n}$ that maximizes $w(S)$ subject to $\det(M_S) \neq 0$.*

Proof. The proof uses the ideas of the Steinitz Exchange Lemma. That is, recall the following facts in linear algebra. If sets S_1, S_2 are both sets of linearly independent vectors, and $|S_1| > |S_2|$, then there is some $\mathbf{v} \in S_1 \setminus S_2$ such that $S_2 \cup \{\mathbf{v}\}$ is linearly independent. Thus, if S_1, S_2 are both sets of linearly independent vectors and $|S_1| = |S_2|$ then for any $\mathbf{w} \in S_2 \setminus S_1$ there is a vector $\mathbf{v} \in S_1 \setminus S_2$ such that $(S_2 \setminus \{\mathbf{w}\}) \cup \{\mathbf{v}\}$ is linearly independent.

Now suppose (for contradiction) that there are two different sets $S_1, S_2 \subseteq [\![m]\!]$ that maximize $w(S)$ over the sets such that $\det(M_S) \neq 0$, so that $|S_1| = |S_2| = n$. Pick the smallest index k in the (non-empty) symmetric difference $(S_2 \setminus S_1) \cup (S_1 \setminus S_2)$. Without loss of generality suppose $k \in S_2 \setminus S_1$. It follows that there is an index $l \in S_1 \setminus S_2$ such that the columns in $S_3 \overset{\text{def}}{=} (S_2 \setminus \{k\}) \cup \{l\}$ are linearly independent (by the Steinitz Exchange Lemma), and thus $\det(M_{S_3}) \neq 0$ as $|S_3| = n$ by construction.

By choice of k and construction of l, $k \neq l$ and thus $k < l$. Thus, $w(S_3) = w(S_2) + l - k > w(S_2)$. However, this contradicts that S_2 was a maximizer to $w(S)$ subject to $\det(M_S) \neq 0$. Thus, the assumption of non-unique maximizers is false; there must be a unique maximizer. $\qquad\square$

Thus $\det(AM)$ is a non-zero polynomial of degree $\leq nr - \binom{r+1}{2}$ in α, so there are at most that many values such that $\det(AM) = 0$. $\qquad\square$

We remark that Lemma 4.2 can be seen as a special case of a more general result about matroids, which states that if each element in the ground set has a unique (positive) weight, then there is a unique independent set with maximal weight. However, as we index matrix columns starting at 0 this general fact does not immediately apply. Rather, we implicitly use that all bases in vector matroids have the same number of vectors. In such a case, the weight function can be shifted by an additive constant without affecting the property of having a unique maximizer.

We now extend the above result to the case when the rank of the $n \times r$ matrix may be less than r. This will be useful when studying hitting sets for rank $\leq r$ matrices, for then we do not know the true rank of the unknown matrix, and only have the bound of "$\leq r$".

Corollary 4.3. *Let $1 \leq s \leq r \leq n$. Let $M \in \mathbb{F}^{n \times r'}$ be of rank s, for $r' \geq s$. Let $\mathbb{K}$ be a field extending $\mathbb{F}$, and let $g \in \mathbb{K}$ be an element of order $\geq n$. Define $A_\alpha \in \mathbb{K}^{r \times n}$ by $(A_\alpha)_{i,j} = (g^i \alpha)^j$. Then there are $\leq nr - \binom{r+1}{2} < nr$ values $\alpha \in \mathbb{K}$ such that the first s rows of $A_\alpha M$ have rank $< s$.*

Proof. Consider $M' \in \mathbb{F}^{n \times s}$ to be a matrix formed from s basis columns of M. It follows, from Theorem 4.1, that there are at most $ns - \binom{s+1}{2}$ values of α such that the $s \times n$ matrix A'_α has $\mathrm{rank}(A'_\alpha M') < s$. As $\mathrm{rank}(A M') = \mathrm{rank}(A M)$ holds for any A, there are at most $ns - \binom{s+1}{2}$ many values of α such that $\mathrm{rank}(A'_\alpha M) < s$. Also, as $ns - \binom{s+1}{2} \leq nr - \binom{r+1}{2}$ for $s \leq r \leq n$, it also holds that there are $\leq nr - \binom{r+1}{2}$ values of α such that $\mathrm{rank}(A'_\alpha M) < s$. Finally the claim follows by observing that, by construction, $A'_\alpha M$ is exactly the first s rows of $A_\alpha M$. $\qquad\square$

5 Identity Testing for Matrices

The previous section showed how we can map an r-dimensional subspace of an n-dimensional ambient space to an r-dimensional subspace of an r-dimensional ambient space. In this section, we will use this map to reduce the PIT problem for rank $\leq r$ matrices of size $n \times m$ to the PIT problem from rank $\leq r$ matrices of size $r \times r$. This will be done by applying the dimension reduction twice, once to the rows and once to the columns. Further, the $r \times r$ version can be solved in r^2 evaluations, using the naive approach of Lemma 3.11 in querying each entry in the matrix. When phrased this way, one can show that this gives a $\Theta((n+m)\mathsf{poly}(r))$-sized hitting set. This reduction idea is analogous to the kernelization technique used in fixed-parameter tractability, but we do not develop this connection further. While this idea demonstrates the feasibility of the rough bound cited above, we actually achieve a $\Theta((n+m)r)$-sized hitting set via tighter analysis.

5.1 Variable Reduction

Before giving the hitting set construction and its analysis, we first present the main theorem used in the analysis. While its statement seems unrelated to the intuition presented above, the proof will exploit this intuition. When interpreting the result, recall that we index entries in matrices (and vectors) starting at 0, as well as recalling the definition of $\hat{f}_T$ from a tensor T.

Theorem 5.1. *Let $m \geq n \geq r \geq 1$. Let $\mathbb{K}$ be an extension of $\mathbb{F}$ such that $g \in \mathbb{K}$ has order $\geq m$. Let M be an $n \times m$ matrix of rank $\leq r$ over $\mathbb{F}$. Then M is non-zero (over $\mathbb{F}$) iff one of the univariate polynomials $\hat{f}_M(x,x), \hat{f}_M(x,gx), \dots, \hat{f}_M(x,g^{r-1}x)$ is non-zero (over $\mathbb{K}$).*

Proof. ($\Longleftarrow$) : If M is zero then so must all $\hat{f}_M(x,g^i x)$ be as well. Taking the contrapositive yields this direction.

($\Longrightarrow$) : Say $\mathrm{rank}(M) = s$. By assumption $0 < s \leq r$. Recall that putting M into reduced row-echelon form yields a decomposition $M = PQ^{\dagger}$, such that $P \in \mathbb{F}^{n \times s}$ and $Q \in \mathbb{F}^{m \times s}$ such that $\mathrm{rank}(P) = \mathrm{rank}(Q) = s$. We remark that it is crucial for our proof that we have "$\mathrm{rank}(P) = \mathrm{rank}(Q) = s$" here. Invoking the bound "$\mathrm{rank}(P), \mathrm{rank}(Q) \leq s$", which one gets directly via the definition of rank of M, is insufficient.

We now exploit the kernelization idea mentioned above. Consider the matrices $A_{\alpha} \in \mathbb{K}^{r \times n}$ and $B_{\alpha} \in \mathbb{K}^{r \times m}$ defined by $(A_{\alpha})_{i,j} = (g^i \alpha)^j$ and $(B_{\alpha})_{i,j} = (g^i \alpha)^j$. Now consider $A_{\alpha} P$ and $B_{\alpha} Q$, which have sizes $r \times s$ each. Write them in block notation as $\begin{pmatrix} P'_{\alpha} \\ P''_{\alpha} \end{pmatrix}$ and $\begin{pmatrix} Q'_{\alpha} \\ Q''_{\alpha} \end{pmatrix}$ such that P'_{α} and Q'_{α} are both $s \times s$ matrices.

By our refinement of the Gabizon-Raz lemma [GR08], our Corollary 4.3, it follows that there are $< nr$ values of α such that $\mathrm{rank}(P'_{\alpha}) < s$ and $< mr$ values of α such that $\mathrm{rank}(Q'_{\alpha}) < s$. By the union bound, there are $< (n+m)r$ values such that $\mathrm{rank}(P'_{\alpha}) < s$ or $\mathrm{rank}(Q'_{\alpha}) < s$. Let $\mathbb{H}$ be an extension field of $\mathbb{K}$, such that $|\mathbb{H}| \geq (n+m)r$. It follows that there is some $\alpha \in \mathbb{H}$ such that $\mathrm{rank}(P'_{\alpha}) = s$ and $\mathrm{rank}(Q'_{\alpha}) = s$. Fix this as the value of α, and we now drop α from our notation.

Via block multiplication we see that

$$AMB^{\dagger} = AP(BQ)^{\dagger} = \begin{pmatrix} P' \\ P'' \end{pmatrix} \begin{pmatrix} Q' & Q'' \end{pmatrix} = \begin{pmatrix} P'Q' & P'Q'' \\ P''Q' & P''Q'' \end{pmatrix}$$

As $\mathrm{rank}(P') = s$ and $\mathrm{rank}(Q') = s$, it follows that $\mathrm{rank}(P'Q') = s$. We remark that it is here where the naive bound "$\mathrm{rank}(P), \mathrm{rank}(Q) \leq s$" is insufficient, and we crucially use that "$\mathrm{rank}(P) = \mathrm{rank}(Q) = s$".

As rank$(P'Q') = s$, and $P'Q'$ is an $s \times s$ matrix, it follows that some entry in its first row (which has index 0, by our notation) is non-zero. As $P'Q'$ is a principal minor of $AMB^\dagger$, it follows that some entry in the first row of $AMB^\dagger$ is non-zero. Denote row i of A as A_i, and row j of B as B_j. As the first row of $AMB^\dagger$ is $A_0MB^\dagger$, it follows then there is some $0 \leq \ell \leq r-1$ such that $A_0MB_\ell^\dagger \neq 0$. Expanding this evaluation out, we see that

$$A_0MB_\ell^\dagger = \langle M, A_0B_\ell^\dagger \rangle = \sum_{i=0,j=0}^{n-1,m-1} M_{i,j} \cdot (A_0)_i (B_\ell)_j$$

$$= \sum_{i=0,j=0}^{n-1,m-1} M_{i,j} A_{0,i} B_{\ell,j}$$

$$= \sum_{i=0,j=0}^{n-1,m-1} M_{i,j} (g^0 \alpha)^i (g^\ell \alpha)^j$$

$$= \hat{f}_M(\alpha, g^\ell \alpha)$$

Thus, we see that $\hat{f}_M(x, g^\ell x)$ has a non-zero point over the field $\mathbb{H}$. It follows that it is a non-zero polynomial over $\mathbb{H}$. As it has coefficients over $\mathbb{K}$, $\hat{f}_M(x, g^\ell x)$ is non-zero over $\mathbb{K}$ as well. $\qquad\square$

Remark 5.2. We now remark on how to implement the kernelization idea, mentioned in the introduction to this section, in a more straight-forward sense. One can see that rank$(P'Q') = s$ shows that $AMB^\dagger \neq 0$. As $AMB^\dagger$ is of size $r \times r$, we can then run the naive r^2-size hitting set of Lemma 3.11 for $r \times r$-sized matrices, which checks each individual entry. Noting that the (i,j)-th entry of $AMB^\dagger$ is equal to $\langle M, A_iB_j^\dagger \rangle$ we see that we can implement this naive hitting set as a hitting set for $n \times m$ matrices.

Thus, for each α there are r^2 rank-1 matrices to test, and we need at most $(n+m)r$ choices of α (where here we assume $\mathbb{K}$ is at least this big). It follows that there exists an explicit hitting set of size $(n+m)r^3$.

Remark 5.3. We briefly discuss the necessity of our version of the Gabizon-Raz lemma for the above proof. The above proof does not invoke our version of the lemma in the fullest, in the sense that the nr bound on the number of "bad" α was only used in the sense that it was a finite bound. Thus, given that our version of the lemma "only" improves the nr^2 bound of Gabizon-Raz to nr, it may be unclear why our version is needed here.

The crucial use of our version of the lemma is keeping the degree low. That is, if one invoked the original Gabizon-Raz lemma, one would result in "M is non-zero iff one of the univariate polynomials $\hat{f}_M(x, x), \hat{f}_M(x, x^2), \ldots, \hat{f}_M(x, x^r)$ is non-zero". While this is correct, it will lead to a larger hitting set as one needs to interpolate r polynomials, each of degree $\approx rn$, which will give a $\Theta(nr^2)$-sized set instead of the $\Theta(nr)$-sized set we are able to achieve.

We also state an equivalent version of this result, which will be useful for higher-degree tensors.

Corollary 5.4. *Let $m \geq n \geq r \geq 1$. Over the field $\mathbb{F}$, consider the bivariate polynomial $f(x,y) = \sum_{i=1}^r p_i(x)q_i(y)$ such that $\deg(p_i) < n$ and $\deg(q_i) < m$ for all i. Let $\mathbb{K}$ be an extension of $\mathbb{F}$ such that $g \in \mathbb{K}$ has order $\geq m$.*

Then f is non-zero (over $\mathbb{F}$) iff one of the univariate polynomials $f(x,x), f(x,gx), \ldots, f(x,g^{r-1}x)$ is non-zero (over $\mathbb{K}$).

5.2 The Hitting Set for Matrices

In this subsection we use the theorem of the last subsection to construct hitting sets for matrices. First, recall our notion of a hitting set for matrices, as given in Definition 3.8. Now recall that Theorem 5.1 shows that for any M of rank $\leq r$, M is non-zero iff one of the polynomials in $\{\hat{f}_M(x, g^\ell x)\}_{0 \leq \ell < r}$ is non-zero. In the preliminaries it was seen that evaluating one of these polynomials at a point α is equivalent to taking an inner product $\langle M, A \rangle$ with a rank-1 matrix A. This leads naturally to the following idea: interpolate each of the r polynomials in $\{\hat{f}_M(x, g^\ell x)\}_{0 \leq \ell < r}$. As each polynomial is of degree $\leq n + m - 2$, this will lead to $(n + m - 1)r$ inner products. Then M is non-zero iff one of these inner products is non-zero. This is the exact idea, which we now make formal.

Construction 5.5. *Let $m \geq n \geq r \geq 1$. Let $\mathbb{K}$ be an extension of $\mathbb{F}$ such that $g \in \mathbb{K}$ is of order $\geq m$ and $\alpha_0, \ldots, \alpha_{n+m-2} \in \mathbb{K}$ are distinct. Let $B_{k,\ell} \in \mathbb{K}^{n \times m}$ to be the rank-1 matrix defined by $(B_{k,\ell})_{i,j} = \alpha_k^i (g^\ell \alpha_k)^j$, and let $\mathcal{B}_{r,n,m} \stackrel{\text{def}}{=} \{B_{k,\ell}\}_{0 \leq \ell < r, 0 \leq k \leq n+m-2}$.*

We now give the analysis for this hitting set.

Theorem 5.6. *Let $m \geq n \geq r \geq 1$. Then $\mathcal{B}_{r,n,m}$, as defined in Construction 5.5, has the following properties:*

1. *$\mathcal{B}_{r,n,m}$ is a hitting set for $n \times m$ matrices of rank $\leq r$ over $\mathbb{F}$.*

2. *$|\mathcal{B}_{r,n,m}| = (n + m - 1)r$*

3. *$\mathcal{B}_{r,n,m}$ can be computed in $\mathsf{poly}(m)$ operations, where operations (including a successor function in some enumeration of $\mathbb{K}$) over $\mathbb{K}$ are counted at unit cost.*

Proof. $\underline{|\mathcal{B}_{r,n,m}| = (n + m - 1)r}$: This is by definition.

$\underline{\mathcal{B}_{r,n,m} \text{ can be computed in } \mathsf{poly}(m) \text{ operations}}$: We assume here an enumeration of elements in $\mathbb{K}$ such that the successor in this enumeration can be computed at unit cost. We also will assume testing whether an element is zero, as well as arithmetic operations in the field, are done at unit cost.

First observe that there are at most m solutions to $x^m - 1$ over $\mathbb{K}$, so if we enumerate $m + 1$ elements of $\mathbb{K}$, then we can find a $g \in \mathbb{K}$ with order $\geq m$. This is in $\mathsf{poly}(m)$ operations. Similarly, the enumeration will give us $n + m - 1$ distinct elements which yield the desired α_k. Then, computing each $B_{k,l}$ can be done in $\mathsf{poly}(m)$ steps, and there are $\mathsf{poly}(m)$ of them. Thus, all of $\mathcal{B}_{r,n,m}$ can be computed in this many operations.

$\underline{\mathcal{B}_{r,n,m} \text{ is a hitting set}}$: $\mathcal{B}_{r,n,m}$ is a set of rank-1 matrices by construction, so it remains to prove that it hits each low-rank matrix. Let M be $n \times m$ matrix of rank $\leq r$ in $\mathbb{F}$. By Theorem 5.1, we see that M is non-zero iff one of the polynomials $\{\hat{f}_M(x, g^\ell x)\}_{0 \leq \ell < r}$ is non-zero. Thus, $\hat{f}_M(\alpha_k, g^\ell \alpha_k) = \sum_{0 \leq i < n, 0 \leq j < m} M_{i,j} \alpha_k^i (g^\ell \alpha_k)^j = \langle M, B_{k,\ell} \rangle$. As each $\hat{f}_M(x, g^\ell x)$ is of degree $\leq n + m - 2$ and we evaluate each polynomial at $n + m - 1$ points, each $\hat{f}_M(x, g^\ell x)$ is fully determined by these evaluations via the polynomial interpolation map. Specifically, if $\hat{f}_M(x, g^\ell x)$ is non-zero then it must have a non-zero evaluation for some α_k. As some $\hat{f}_M(x, g^\ell x)$ is non-zero by Theorem 5.1, it follows that $\langle M, B_{k,\ell} \rangle \neq 0$ for some $0 \leq \ell < r$ and $0 \leq k \leq n + m - 2$. $\qquad \square$

One deficiency with this construction is that for large r it is suboptimal by a factor of 2. That is, in the regime where $n = m$ and $r = n - 1$ this construction gives a hitting set of size $(2n - 1)(n - 1) \approx 2n^2$. However, the naive hitting set yields an n^2-sized setting. In the next subsection we show that this is an artifact of the analysis. That is, by pruning unneeded matrices

from the hitting set, one can show that our construction always (for $r < n$) does better than the naive construction. This result proven in Theorem 5.10.

5.3 An Alternate Construction

In the previous subsections we saw that a low-rank matrix M is non-zero iff one of the polynomials $\{\hat{f}_M(x, g^\ell x)\}$ was non-zero. To construct a hitting set, we then interpolated each $\hat{f}_M$ at enough points to determine which, if any, were non-zero. However, we are interpolating many "related" polynomials all on the same points, so it is natural to wonder if there are some redundancies in this process.

To phrase things differently, observe that testing a matrix M against a hitting set $\mathcal{H}$ is really asking of $M \in \ker \mathcal{H}$. The promise that M is low-rank ensures that $M \in \ker \mathcal{H}$ iff M is zero. The number of tests done is $|\mathcal{H}|$, but the number of *actual* tests is $\operatorname{rank}(\mathcal{H})$, where we consider $\mathcal{H}$ as vectors in the vector space $\mathbb{K}^{nm}$. That is, some of the matrices in $\mathcal{H}$ may be linearly dependent, and these are redundancies that can be pruned.

The aim of this section is to present hitting sets (and improper hitting sets) that have linearly independent test matrices. The initial motivation is to observe that the point of the evaluations of the $\{\hat{f}_M(x, g^\ell x)\}$ was to interpolate the coefficients. Thus, instead of doing these evaluations, we can express the coefficients of the $\{\hat{f}_M(x, g^\ell x)\}$ directly as linear combinations of the entries in M. This will lead to the following improper hitting set.

Construction 5.7. *Let $m \geq n \geq r \geq 1$. Let $\mathbb{K}$ be an extension of $\mathbb{F}$ such that $g \in \mathbb{K}$ is of order $\geq m$. Let $D_{k,\ell} \in \mathbb{K}^{n \times m}$ be the matrix defined be*

$$(D_{k,\ell})_{i,j} = \begin{cases} g^{\ell j} & if\ i + j = k \\ 0 & else \end{cases}$$

Define $\mathcal{D}_{r,n,m} \overset{\text{def}}{=} \{D_{k,\ell}\}_{\substack{0 \leq k \leq n+m-2 \\ 0 \leq \ell < r}}$, and $\mathcal{D}'_r \overset{\text{def}}{=} \{D_{k,\ell}\}_{\substack{0 \leq k \leq n+m-2 \\ 0 \leq \ell < \min(r,k+1,(n+m)-(k+1))}}$.

We now analyze this construction.

Theorem 5.8. *Let $m \geq n \geq r \geq 1$. Then $\mathcal{D}_{r,n,m}$, as defined in Construction 5.7, has the following properties:*

1. *$\mathcal{D}_{r,n,m}$ is an improper hitting set for $n \times m$ matrices of rank $\leq r$ over $\mathbb{F}$.*

2. *$\operatorname{Span}(\mathcal{D}_{r,n,m}) = \operatorname{Span}(\mathcal{B}_{r,n,m})$ (as vectors in $\mathbb{K}^{nm}$)*

3. *$|\mathcal{D}_{r,n,m}| = (n + m - 1)r$*

4. *Each matrix in $\mathcal{D}_{r,n,m}$ is n-sparse.*

5. *$\mathcal{D}_{r,n,m}$ can be computed in $\operatorname{poly}(m)$ operations, where operations (including a successor function in some enumeration of $\mathbb{K}$) over $\mathbb{K}$ are counted at unit cost.*

and $\mathcal{D}'_{r,n,m}$, as defined in Construction 5.7, has the following properties:

1. *$\mathcal{D}'_{r,n,m}$ is an improper hitting set for $n \times m$ matrices of rank $\leq r$ over $\mathbb{F}$.*

2. *$\mathcal{D}'_{r,n,m}$ is linearly independent (as vectors in $\mathbb{K}^{nm}$) and $\operatorname{Span}(\mathcal{D}_{r,n,m}) = \operatorname{Span}(\mathcal{D}'_{r,n,m})$*

3. *$|\mathcal{D}'_{r,n,m}| = (n + m - r)r$*

4. Each matrix in $\mathcal{D}'_{r,n,m}$ is n-sparse.

5. $\mathcal{D}'_{r,n,m}$ can be computed in $\mathsf{poly}(m)$ operations, where operations (including a successor function in some enumeration of $\mathbb{K}$) over $\mathbb{K}$ are counted at unit cost.

Proof. $|\mathcal{D}_{r,n,m}| = (n + m - 1)r$: This is by definition.

Sparsity of $\mathcal{D}_{r,n,m}$: Each matrix in the hitting set has support in some k-diagonal, and each diagonal has at most n non-zero entries.

$\mathcal{D}_{r,n,m}$ can be computed in $\mathsf{poly}(m)$ operations: The details are very similar to the proof that $\mathcal{B}_{r,n,m}$ can be computed in $\mathsf{poly}(m)$ operations, as seen in Theorem 5.6, so we omit the specifics.

$\mathcal{D}_{r,n,m}$ is an improper hitting set: Let M be $n \times m$ matrix of rank $\leq r$ in $\mathbb{F}$. By Theorem 5.1, we see that M is non-zero iff one of the polynomials $\{\hat{f}_M(x, g^\ell x)\}_{0 \leq \ell < r}$ is non-zero. Recall the notation that $\mathfrak{C}_{x^k}(f)$ denotes the coefficient of f on x^k. Thus, $\mathfrak{C}_{x^k}(\hat{f}_M(x, g^\ell x)) = \sum_{i+j=k} M_{i,j} g^{\ell j} = \langle M, D_{k,\ell} \rangle$. Thus, it follows that some $\hat{f}_M(x, g^\ell x)$ is non-zero iff one the inner products $\langle M, D_{k,\ell} \rangle$ is non-zero. Invoking Theorem 5.1 completes this claim.

This can also be seen from the fact $\mathrm{Span}(\mathcal{D}_{r,n,m}) = \mathrm{Span}(\mathcal{B}_{r,n,m})$. Thus, a for a matrix M, $M \in \ker \mathcal{D}_{r,n,m} \iff M \in \ker \mathcal{B}_{r,n,m}$.

$\mathrm{Span}(\mathcal{D}_{r,n,m}) \supseteq \mathrm{Span}(\mathcal{B}_{r,n,m})$: For any M (not just those of rank $\leq r$) we have that $\langle M, D_{k,\ell} \rangle = \mathfrak{C}_{x^k}(\hat{f}_M(x, g^\ell x))$ and $\langle M, B_{k,\ell} \rangle = \hat{f}_M(\alpha_k, g^\ell \alpha_k)$ and thus $\langle M, B_{k,\ell} \rangle = \sum_{k'=0}^{n+m-2} \alpha_k^{k'} \langle M, D_{k',\ell} \rangle$. By taking M for each element in some basis, it follows that $B_{k,\ell} = \sum_{k'=0}^{n+m-2} \alpha_k^{k'} D_{k',\ell}$

$\mathrm{Span}(\mathcal{D}_{r,n,m}) \subseteq \mathrm{Span}(\mathcal{B}_{r,n,m})$: Similar to the above case, we get that for any M,

$$\langle M, D_{k,\ell} \rangle = \sum_{k'=0}^{n+m-2} \mathfrak{C}_{x^{k'}} \left(\prod_{k'' \neq k} \frac{x - \alpha_{k'}}{\alpha_{k''} - \alpha_{k'}} \right) \langle M, B_{k',\ell} \rangle$$

via Lagrange interpolation. As the coefficients of this linear dependence are independent of M (they only depend on the α_k), by taking M for each element of some basis it follows that the same linear dependence for $D_{k,\ell}$ and $\{B_{k',\ell}\}_{k'}$ exists, giving the claim.

$\mathcal{D}'_{r,n,m}$ can be computed in $\mathsf{poly}(m)$ operations: As with $\mathcal{D}_{r,n,m}$, these details are omitted.

Sparsity of $\mathcal{D}'_{r,n,m}$: Each matrix in the hitting set has support in some k-diagonal, and each diagonal has at most n non-zero entries.

$\mathcal{D}'_{r,n,m}$ is an improper hitting set: This follows from showing that $\mathcal{D}_{r,n,m} \subseteq \mathrm{Span}(\mathcal{D}'_{r,n,m})$, as this implies that for a matrix M, $M \in \ker \mathcal{D}_{r,n,m} \iff M \in \ker \mathcal{D}'_{r,n,m}$. Thus, as $\mathcal{D}_{r,n,m}$ is an improper hitting set so is $\mathcal{D}'_{r,n,m}$.

$\mathrm{Span}(\mathcal{D}_{r,n,m}) \supseteq \mathrm{Span}(\mathcal{D}'_{r,n,m})$: This is clear, as $\mathcal{D}_{r,n,m} \supseteq \mathcal{D}'_{r,n,m}$.

$\mathrm{Span}(\mathcal{D}_{r,n,m}) \subseteq \mathrm{Span}(\mathcal{D}'_{r,n,m})$: Begin by observing that $D_{k,\ell}$ is non-zero only on the k-diagonal and the k-diagonal has $\min(k + 1, n, (n + m) - (k + 1))$ entries. Further, the k-diagonals of the matrices $\{D_{k,\ell}\}_{0 \leq \ell < r}$ form the rows of a $r \times \min(k+1, n, (n+m)-(k+1))$ Vandermonde matrix. This Vandermonde matrix is formed by taking powers of $\leq n$ consecutive powers of g, which by the order of g are distinct. It follows that the first $\min(r, \min(k+1, n, (n+m)-(k+1)))$ of the $\{D_{k,\ell}\}_{0 \leq \ell < r}$ form a basis for the rest. As $r \leq n$, $\min(r, \min(k+1, n, (n+m)-(k+1))) = \min(r, k+1, (n+m)-(k+1))$, so $\{D_{k,\ell}\}_{0 \leq \ell < \min(r,k+1,(n+m)-(k+1))}$ are a basis for $\{D_{k,\ell}\}_{0 \leq \ell < r}$ (recall that we start indexing from zero). Ranging over all k shows that the claim holds.

$\mathcal{D}'_{r,n,m}$ is linearly independent: Notice that $D_{k,\ell}$ and $D_{k',\ell'}$ have disjoint support if $k \neq k'$. The previous paragraph shows $\{D_{k,\ell}\}_{0 \leq \ell < \min(r,k+1,(n+m)-(k+1))}$ are linearly independent for each k, and the fact about disjoint support for differing k shows that taking the union over k does not introduce any linearly dependencies.

$\underline{|\mathcal{D}'_{r,n,m}| = (n+m-r)r}$: For $r \le k+1 \le (n+m)-r$ we see that $r = \min(r, k+1, (n+m)-(k+1))$, so $\mathcal{D}'_{r,n,m}$ offers no savings in this regime. For $0 \le k < r$, $\mathcal{D}'_{r,n,m}$ takes $r - (k+1)$ fewer matrices then $\mathcal{D}_{r,n,m}$. For $n+m > k+1 \ge (n+m)-r$, $\mathcal{D}'_{r,n,m}$ takes $r - ((n+m)-(k+1))$ fewer matrices then $\mathcal{D}_{r,n,m}$. It follows that $|\mathcal{D}'_{r,n,m}| = |\mathcal{D}_{r,n,m}| - r(r-1) = (n+m-1)r - r(r-1) = (n+m-r)r$. $\qquad\square$

We sketch another proof of this result in Remark 7.24.

The above results also imply that $\text{rank}(\mathcal{B}_{r,n,m}) = (n+m-r)r$, which is better than the analysis given in Theorem 5.6. This immediately gives that there are explicit $(n+m-r)r$-sized (proper) hitting sets for $n \times m$ matrices of rank $\le r$, as we can (in $\mathsf{poly}(m)$ steps) find a basis for $\mathcal{B}_{r,n,m}$. This basis will consist of rank-1 matrices, and also be the desired hitting set. However, in the interest of being more explicit, we present the following construction.

Construction 5.9. *Let $m \ge n \ge r \ge 1$. Let $\mathbb{K}$ be an extension of $\mathbb{F}$ such that $g \in \mathbb{K}$ is of order $\ge m$ and $\alpha_0, \ldots, \alpha_{n+m-2} \in \mathbb{K}$ are distinct. Let $B'_{k,\ell} \in \mathbb{K}^{n \times m}$ to be the rank-1 matrix defined by $(B'_{k,\ell})_{i,j} = \alpha_k^i (g^\ell \alpha_k)^j$, and let $\mathcal{B}'_{r,n,m} \overset{\text{def}}{=} \{B_{k,\ell}\}_{0 \le \ell < r, 0 \le k \le (n+m-2)-2\ell}$.*

We now give the analysis for this hitting set.

Theorem 5.10. *Let $m \ge n \ge r \ge 1$. Then $\mathcal{B}'_{r,n,m}$, as defined in Construction 5.9, has the following properties:*

1. *$\text{Span}\,\mathcal{B}'_{r,n,m} = \text{Span}\,\mathcal{B}_{r,n,m}$, where $\mathcal{B}_{r,n,m}$ is defined in Construction 5.5.*

2. *$\mathcal{B}'_{r,n,m}$ is a hitting set for $n \times m$ matrices of rank $\le r$ over $\mathbb{F}$.*

3. *$|\mathcal{B}'_{r,n,m}| = (n+m-r)r$*

4. *$\mathcal{B}'_{r,n,m}$ is linearly independent (as vectors in $\mathbb{K}^{nm}$)*

5. *$\mathcal{B}'_{r,n,m}$ can be computed in $\mathsf{poly}(m)$ operations, where operations (including a successor function in some enumeration of $\mathbb{K}$) over $\mathbb{K}$ are counted at unit cost.*

Proof. $\underline{|\mathcal{B}_{r,n,m}| = (n+m-r)r}$: The size is equal to $\sum_{\ell=0}^{r-1}((n+m-1)-2\ell) = (n+m-1)r - 2\binom{r}{2} = (n+m-r)r$.

$\underline{\mathcal{B}'_{r,n,m} \text{ can be computed in } \mathsf{poly}(m) \text{ operations}}$: The details are very similar to the proof that $\mathcal{B}_{r,n,m}$ can be computed in $\mathsf{poly}(m)$ operations, as seen in Theorem 5.6, so we omit the specifics.

$\underline{\mathcal{B}'_{r,n,m} \text{ is an hitting set}}$: This follows from showing that $\mathcal{B}_{r,n,m} \subseteq \text{Span}(\mathcal{B}'_{r,n,m})$, as this implies that for a matrix M, $M \in \ker \mathcal{B}_{r,n,m} \iff M \in \ker \mathcal{B}'_{r,n,m}$. Thus, as $\mathcal{B}_{r,n,m}$ is an hitting set so is $\mathcal{B}'_{r,n,m}$.

$\underline{\text{Span}\,\mathcal{B}'_{r,n,m} \subseteq \text{Span}\,\mathcal{B}_{r,n,m}}$: This is clear as $\mathcal{B}'_{r,n,m} \subseteq \mathcal{B}_{r,n,m}$.

$\underline{\text{Span}\,\mathcal{B}'_{r,n,m} \supseteq \text{Span}\,\mathcal{B}_{r,n,m}}$: We will actually show $\mathcal{D}_{r,n,m} \subseteq \text{Span}\,\mathcal{B}'_{r,n,m}$, which by Theorem 5.8 is sufficient. Let M be any matrix (even of rank $> r$). We will show that the inner-products $\langle M, \mathcal{B}'_{r,n,m} \rangle$ determine the inner-products $\langle M, \mathcal{D}_{r,n,m} \rangle$. Then we show that this implies the claim.

Recall that the inner-product of a matrix $D \in \mathcal{D}_{r,n,m}$ is simply a coefficient $\mathfrak{C}_{x^k}(\hat{f}_M(x, g^i x))$ for some $0 \le k \le n+m-2$ and $0 \le i < r$. So to prove the claim we will speak of these coefficients determining other such coefficients.

Now observe that for any $k \in \{0, \ldots, r-1\}$, the coefficients $\mathfrak{C}_{x^k}(\hat{f}_M(x, x))$, $\mathfrak{C}_{x^k}(\hat{f}_M(x, gx))$, $\ldots, \mathfrak{C}_{x^k}(\hat{f}_M(x, g^{r-1}x))$ are linear combinations of the $k+1 \le r$ elements in $\{M_{i,j}\}_{i+j=k}$. Just as in the analysis of $\mathcal{D}'_{r,n,m}$ in Theorem 5.8, the first $k+1$ of these linear combinations are rows of a Vandermonde matrix over distinct numbers, and thus these linear combinations span all

22

vectors. Thus, it follows that the coefficients $\{\mathfrak{C}_{x^k}\hat{f}_M(x, g^i x)\}_{0 \le i < k+1}$ determine the coefficients $\{\mathfrak{C}_{x^k}\hat{f}_M(x, g^i x)\}_{0 \le i < r}$.

Similarly, for any $k \in \{(n+m) - (r+1), \ldots, (n+m) - 2\}$ the coefficients $\{\mathfrak{C}_{x^k}\hat{f}_M(x, g^i x)\}_{0 \le i < (n+m)-(k+1)}$ determine the coefficients $\{\mathfrak{C}_{x^k}\hat{f}_M(x, g^i x)\}_{0 \le i < r}$. We now use these facts in the following claim.

Claim 5.11. *The coefficients of $\hat{f}_M(x, g^{k+1} x)$ are determined by the coefficients of $\hat{f}_M(x, x), \hat{f}_M(x, gx), \ldots, \hat{f}_M(x, g^k x)$ and the evaluations of $\hat{f}_M(x, g^{k+1} x)$ to any $(n+m-1)-2(k+1)$ distinct points.*

Proof. By the above reasoning, the coefficients $\mathfrak{C}_{x^{k'}}(\hat{f}_M(x, g^{k+1} x))$ with $k' \in \{0, \ldots, k\} \cup \{(n+m-2) - k, \ldots, (n+m) - 2\}$ are already determined by the coefficients given.

Now, consider the polynomial

$$h(x) \overset{\text{def}}{=} \frac{\hat{f}_M(x, g^{k+1} x) - \sum_{k'=0}^{k} \mathfrak{C}_{x^{k'}}(\hat{f}_M(x, g^{k+1} x)) x^{k'} - \sum_{k'=(n+m-2)-k}^{n+m-2} \mathfrak{C}_{x^{k'}}(\hat{f}_M(x, g^{k+1} x)) x^{k'}}{x^{k+1}}$$

By construction, h of degree $\le (n+m-2) - 2(k+1)$, and evaluation of h is possible given oracle access to $\hat{f}_M(x, g^{k+1} x)$ as the relevant coefficients referenced are already determined.

Thus, it follows that h is determined by interpolation at any $(n+m-1) - 2(k+1)$ distinct points. Once h is determined, the above equation determines the as yet undetermined coefficients of $\hat{f}_M(x, g^{k+1} x)$. $\qquad\square$

Thus, to determine all of the coefficients of the polynomials $\{\hat{f}_M(x, g^\ell x)\}_{0 \le \ell < r}$ we first interpolate $\hat{f}_M(x, x)$ at $n+m-1$ distinct points. The above claim then shows how to interpolate $\hat{f}_M(x, gx)$ using $(n+m-1)-2$ evaluations to $\hat{f}_M(x, gx)$, given access to the coefficients of $\hat{f}_M(x, x)$. Inducting on the above claim shows we can interpolate all of the coefficients in $\{\hat{f}_M(x, g^\ell x)\}_{0 \le \ell < r}$ from the evaluations $\{\hat{f}_M(\alpha_k, g^\ell \alpha_k)\}_{0 \le \ell < r, 0 \le k \le (n+m-2)-2\ell}$. Rephrasing this, we see that the inner-products $\langle M, \mathcal{D}_{r,n,m} \rangle$ are determined by the inner-products $\langle M, \mathcal{B}'_{r,n,m} \rangle$.

Now consider a matrix $B \notin \text{Span } \mathcal{B}'_{r,n,m}$. It follows that the dual space of $\mathcal{B}'_{r,n,m}$ is strictly larger than the dual space of $\mathcal{B}'_{r,n,m} \cup \{B\}$, so that there is a non-zero matrix M_0 such that $\langle M_0, \mathcal{B}'_{r,n,m} \rangle = \mathbf{0}$ but $\langle M_0, B \rangle \ne 0$. But as $\langle 0_{n \times m}, \mathcal{B}'_{r,n,m} \rangle = \mathbf{0}$ and $\langle 0_{n \times m}, B \rangle = 0$, it follows that the inner-product $\langle M_0, \mathcal{B}'_{r,n,m} \rangle$ does not determine the inner-product $\langle M_0, B \rangle$. As $\langle M, \mathcal{B}'_{r,n,m} \rangle$ determines $\langle M, \mathcal{D}_{r,n,m} \rangle$, it must be that $\mathcal{D}_{r,n,m} \subseteq \text{Span } \mathcal{B}'_{r,n,m}$.

$\underline{\mathcal{B}'_{r,n,m} \text{ is linearly independent:}}$ As $\text{Span}(\mathcal{B}'_{r,n,m}) = \text{Span}(\mathcal{D}'_{r,n,m})$, $|\mathcal{B}'_{r,n,m}| = |\mathcal{D}'_{r,n,m}|$, and $\mathcal{D}'_{r,n,m}$ is linearly independent, it follows that $\mathcal{B}'_{r,n,m}$ is also. $\qquad\square$

Thus, we achieve an explicit hitting set of size $(n+m-r)r$. For $r = n$ we see that this equals nm, matching the naive bound. For $r \le n-1$, $(n+m-r)r$ is increasing with r, so $(n+m-r)r \le (n+m-(n-1))(n-1) = (m+1)(n-1) = nm+n-m-1 < nm$. Thus, we see that our hitting set is always smaller than the naive hitting set, for $r < n$.

6 Identity Testing for Tensors

In this section we show how to construct hitting sets for $[\![n]\!]^d$ tensors of arbitrary degree d. We will only discuss tensors of shape $[\![n]\!]^d$ for simplicity. The proof technique will be to use the results for $d = 2$ as a black-box as a way to induct on d. That is, Corollary 5.4 shows that one can test identity of degree $< n$, rank $\le r$ bivariate polynomials by testing the identity of r univariate polynomials, each of degree $< 2n$. This effectively reduces the $d = 2$ case to the $d = 1$ case, while increasing the

number of polynomials to test by a factor of r. As degree $< 2n$ univariate polynomials can be fully interpolated cheaply, this shows that this is a viable base case for recursion.

Intuitively, it seems like this variable reduction process should be able to be continued so that a rank $\leq r$ d-variate polynomial can be identity tested by testing identity of $\approx r^d$ univariate polynomials each of degree $\approx dn$. This is indeed possible. However, we are able to do better here by using a reduction process that reduces a d-variate polynomial to a $d/2$-variate polynomial while only increasing the number of polynomials to test by a factor of r. Thus, a d-variate polynomial can identity tested by testing $\approx r^{\lg d}$ univariate polynomials, each of degree $< dn$. Unfortunately, this set of polynomials will require $\approx (dn)^d$ time to construct.

The section will be split into two parts. The first will state the variable reduction theorem that was mentioned above. The second part will detail the hitting set arising from this theorem.

6.1 Variable Reduction

As with the $d = 2$ case, will need a variable reduction result in order to construct our hitting set. We detail this result in this subsection. We first illustrate some lemmas about variable reduction.

Lemma 6.1. *Let $f(x_1, \ldots, x_d)$ be a d-variate polynomial. Let $\pi : [d] \to [d]$ be a permutation. Then, $f(x_1, \ldots, x_d) = 0$ iff $f(x_{\sigma(1)}, \ldots, x_{\sigma(d)}) = 0$.*

Proof. Consider the map $\mathbb{N}^d \to \mathbb{N}^d$ defined by $(i_1, \ldots, i_d) \mapsto (i_{\sigma(1)}, \ldots, i_{\sigma(d)})$. This is exactly the action on the degrees of monomials over the variables $x_1, \ldots, x_d$ when performing the substitution $x_i \mapsto x_{\sigma(i)}$. Note that this map is bijective.

Thus, when mapping $f(x_1, \ldots, x_d)$ to $f(x_{\sigma(1)}, \ldots, x_{\sigma(d)})$ we see that there can be no cancellations, as distinct monomials are mapped to distinct monomials. Thus, the two polynomials have the same number of non-zero coefficients. In particular, they are either both zero or non-zero. □

The above lemma is most useful in conjunction with the next lemma, which shows a simple d-variate to $(d-1)$-variate reduction.

Lemma 6.2. *Let $f(x, y, z_1, \ldots, z_d)$ be a $(d+2)$-variate polynomial such that $\deg_x(f) < n$. Then for any $m \geq n$, $f(x, y, z_1, \ldots, z_d) = 0$ iff $f(x, x^m, z_1, \ldots, z_d) = 0$.*

Proof. Consider the map $\mathbb{N}^{d+2} \to \mathbb{N}^{d+1}$ defined by $(i_1, i_2, i_3, \ldots, i_{d+2}) \mapsto (i_1 + mi_2, i_3, \ldots, i_{d+2})$. This is exactly the action on the degrees of monomials over the variables $x, y, z_1, \ldots, z_d$ when performing the substitution $y \mapsto x^m$.

Notice that this map is injective when restricted to $[\![n]\!] \times \mathbb{N}^{d+1}$, as $n \leq m$. That is, if $i + mj = i' + mj'$ with $(i, j), (i', j') \in [\![n]\!] \times \mathbb{Z}$ then $i \equiv i' \mod m$ which means $i = i'$, and thus $j = j'$ as well.

Thus, when mapping $f(x, y, z_1, \ldots, z_d)$ to $f(x, x^m, z_1, \ldots, z_d)$ we see that there can be no cancellations, as distinct monomials are mapped to distinct monomials. Thus, the two polynomials have the same number of non-zero coefficients. In particular, they are either both zero or non-zero. □

The above lemmas show that we can "reshape" our polynomials, in the sense that we have fewer variables but larger individual degrees. To perform our d-variate variable reduction, we will reshape our polynomial into a bivariate polynomial, such that the individual degrees are now $\approx n^{d/2}$. We can then apply our bivariate variable reduction to get a univariate polynomial of degree $\approx n^{d/2}$. One can then reverse the reshaping, to yield a $d/2$-variate polynomial, with individual degrees $\approx n$. One then recurses appropriately.

In order to understand the recursion pattern sketched above, we will introduce the following function.

Definition 6.3. *Let $n \geq 1$, $b \geq 0$. Let $0 \leq k < 2^d$. Define*

$$L_{n,b}(k, i_1, \ldots, i_d) = \sum_{\substack{1 \leq j \leq d \\ \lfloor k/2^{j-1} \rfloor \equiv 1 \bmod 2}} i_j (n2^b)^{\lfloor k/2^j \rfloor}$$

We now observe that it obeys the following properties.

Proposition 6.4. *Let $n \geq 1$, $b \geq 0$, with $0 \leq k < 2^d$. Then*

1.

$$L_{n,b}(k, i_1, \ldots, i_d) = \begin{cases} 0 & \text{if } k = 0 \\ i_1(n2^b)^{\lfloor k/2 \rfloor} + L_{n,b}(\lfloor k/2 \rfloor, i_2, \ldots, i_d) & k \equiv 1 \bmod 2 \\ L_{n,b}(\lfloor k/2 \rfloor, i_2, \ldots, i_d) & \text{else} \end{cases}$$

2. *For $b \geq 1$, $L_{2n,b-1}(k, i_1, \ldots, i_d) = L_{n,b}(k, i_1, \ldots, i_d)$*

3. *$L_{n,b}(k, i_1, \ldots, i_d) \leq (n2^b)^{\lfloor k/2 \rfloor} \sum_{j \in [d]} i_j$*

4. *$L_{n,b}(k, i_1, \ldots, i_d)$ can be computed in time $\mathsf{poly}(|n|, b, d, k, |i_1|, \ldots, |i_d|)$, where $|\cdot|$ is the length, in bits, of a number.*

Proof. (1): We first note that $\lfloor \lfloor k/2^j \rfloor / 2^{j'} \rfloor = \lfloor k/2^{j+j'} \rfloor$, which is most easily seen by observing that these operations bit truncate (on the right) the binary representation of k. If $k = 0$ then in both formulas $L_{n,b}(k, i_1, \ldots, i_d) = 0$. If $k \equiv 1 \bmod 2$, then

$$L_{n,b}(k, i_1, \ldots, i_d) = i_1(n2^b)^{\lfloor k/2 \rfloor} + \sum_{\substack{2 \leq j \leq d \\ \lfloor k/2^{j-1} \rfloor \equiv 1 \bmod 2}} i_j (n2^b)^{\lfloor k/2^j \rfloor}$$

$$= i_1(n2^b)^{\lfloor k/2 \rfloor} + \sum_{\substack{1 \leq j \leq d-1 \\ \lfloor k/2^{j-2} \rfloor \equiv 1 \bmod 2}} i_{j+1} (n2^b)^{\lfloor k/2^{j-1} \rfloor}$$

$$= i_1(n2^b)^{\lfloor k/2 \rfloor} + \sum_{\substack{1 \leq j \leq d-1 \\ \lfloor \lfloor k/2 \rfloor / 2^{j-1} \rfloor \equiv 1 \bmod 2}} i_{j+1} (n2^b)^{\lfloor \lfloor k/2 \rfloor / 2^j \rfloor}$$

$$= i_1(n2^b)^{\lfloor k/2 \rfloor} + L_{n,b}(\lfloor k/2 \rfloor, i_2, \ldots, i_d)$$

which is exactly the above recursion. The case $k \equiv 0 \bmod 2$ is analogous.

(2): The definition of $L_{n,b}$ only depends on $n2^b$. Thus, as $2n \cdot 2^{b-1} = n \cdot 2^b$, this is immediate.

(3): This is immediate.

(4): The natural way of computing the formula $L_{n,b}(k, i_1, \ldots, i_d)$ is done in the given time bound. $\square$

We will now prove our multi-variate variable reduction theorem. We prove here the case when the number of variables is a power of 2, for simplicity. The general case, with some loss, will follow as a corollary. The following notation will make the presentation simpler.

Notation 6.5. *Let $f(\langle h_1(j), \ldots, h_k(j) \rangle_{j=1}^r)$ denote*

$$f(h_1(1), \ldots, h_k(1), h_1(2), \ldots, h_k(2), \ldots, h_1(r), \ldots, h_k(r))$$

We will use this notation heavily in the following proof.

Theorem 6.6. *Let $n \geq 1$, $d \geq 1$ and $b \geq d - 1$. Let $\mathbb{K}$ be an extension of $\mathbb{F}$ such that $g \in \mathbb{K}$ has order $\geq (n2^b)^{2^{d-1}}$. Let $T : [\![n]\!]^{2^d} \to \mathbb{F}$ be a tensor of rank $\leq r$. Let $\hat{f}_T(x_0, \ldots, x_{2^d-1}) = \sum_{\ell=1}^{r} \prod_{i=0}^{2^d-1} p_{i,\ell}(x_i)$, where $\deg p_{i,\ell} < n$.*

Then $\hat{f}_T$ is non-zero (over $\mathbb{F}$) iff one of the univariate polynomials in the set

$$\{\hat{f}_T(g^{L_{n,b}(0,i_1,\ldots,i_d)}x, g^{L_{n,b}(1,i_1,\ldots,i_d)}x, \ldots, g^{L_{n,b}(2^d-1,i_1,\ldots,i_d)}x)\}_{0 \leq i_1,\ldots,i_d < r}$$

is non-zero (over $\mathbb{K}$).

Proof. The proof will be by induction. For simplicity we write f for $\hat{f}_T$.

<u>$d = 1$:</u> Note that $L_{n,b}(0, i_1) = 0$ and $L_{n,b}(1, i_1) = i_1$, so this case follows from Corollary 5.4.

<u>$d > 1$:</u> We will first reshape f into a bivariate polynomial, and appeal to the $d = 1$ case. We will then un-reshape this polynomial into a 2^{d-1}-variate polynomial, and then appeal to induction.

By induction on Lemma 6.2 (and appealing to Lemma 6.1 to see that Lemma 6.2 applies to any two variables, not just the first) we see that

$$f(\langle x_j \rangle_{j=0}^{2^d-1}) = 0 \text{ iff } f(\langle x_0^{(n2^b)^j}, x_1^{(n2^b)^j} \rangle_{j=0}^{2^{d-1}-1}) = 0 \tag{1}$$

(where so far we only need that $b \geq 1$).

We split the rest of the proof into two claims. The first claim shows how we can, using the bivariate case, test identity of the right-hand-side of Equation (1) by testing identity of a set of r polynomials, each of 2^{d-1} variables. The second claim shows how testing identity of these new polynomials can be reduced to testing identity of univariate polynomials, where we use the induction hypothesis.

Claim 6.7.

$$f(\langle x_0^{(n2^b)^j}, x_1^{(n2^b)^j} \rangle_{j=0}^{2^{d-1}-1}) = 0$$

iff

$$\{f(\langle x_j, g^{i_1(n2^b)^j}x_j \rangle_{j=0}^{2^{d-1}-1})\}_{0 \leq i_1 < r} = 0$$

Proof. First observe that

$$f'(x_0, x_1) \overset{\text{def}}{=} f(\langle x_0^{(n2^b)^j}, x_1^{(n2^b)^j} \rangle_{j=0}^{2^{d-1}-1})$$

$$= f(x_0, x_1, x_0^{n2^b}, x_1^{n2^b}, x_0^{(n2^b)^2}, x_1^{(n2^b)^2}, \ldots, x_0^{(n2^b)^{2^{d-1}-1}}, x_1^{(n2^b)^{2^{d-1}-1}})$$

$$= \sum_{\ell=1}^{r} \left(\prod_{j=0}^{2^{d-1}-1} p_{2j,\ell}(x_0^{(n2^b)^j}) \right) \left(\prod_{j=0}^{2^{d-1}-1} p_{2j+1,\ell}(x_1^{(n2^b)^j}) \right)$$

so we can apply Corollary 5.4 to see that $f'(x_0, x_1) = 0$ iff $\{f'(x_0, g^{i_1}x_0)\}_{0 \leq i_1 < r} = 0$, which, when expanded, is equivalent to

$$\{f(\langle x_0^{(n2^b)^j}, g^{i_1(n2^b)^j} x_0^{(n2^b)^j} \rangle_{j=0}^{2^{d-1}-1})\}_{0 \leq i_1 < r} = 0$$

using that the order of g is $\geq (n2^b)^{2^{d-1}} > \deg_{x_0} f', \deg_{x_1} f'$. Using that $2^b \geq 2^{d-1} \geq 2$, we can undue the variable substitutions $x_j \mapsto x_0^{(n2^b)^j}$. That is, applying Lemma 6.2 in reverse, we see that the above set of polynomials is zero iff

$$\{f(\langle x_j, g^{i_1(n2^b)^j}x_j \rangle_{j=0}^{2^{d-1}-1})\}_{0 \leq i_1 < r} = 0$$

which is exactly the claim. $\qquad\square$

Claim 6.8.
$$f(\langle x_j, g^{i_1(n2^b)^j} x_j\rangle_{j=0}^{2^{d-1}-1}) = 0$$

iff
$$\{f(\langle g^{L_{n,b}(j,i_1,\ldots,i_d)} x\rangle_{j=0}^{2^d-1})\}_{0\le i_2,\ldots,i_d < r} = 0$$

Proof. First observe that

$$f'(x_0, x_1, \ldots, x_{2^{d-1}-1}) \overset{\text{def}}{=} f(\langle x_j, g^{i_1(n2^b)^j} x_j\rangle_{j=0}^{2^{d-1}-1})$$
$$= \sum_{\ell=1}^{r} \prod_{j=0}^{2^{d-1}-1} p_{2j,\ell}(x_{2j}) \cdot p_{2j+1,\ell}(g^{i_1(n2^b)^j} x_{2j})$$

so f' is 2^{d-1}-variate, having individual degrees $< 2n$. Thus, applying induction to the theorem for the 2^{d-1}-variate case (and using $b-1$ instead of b, noticing that $b-1 \ge (d-1)-1$ also holds), we get that $f'(x_0, x_1, \ldots, x_{2^{d-1}-1}) = 0$ iff

$$\{f'(\langle g^{L_{2n,b-1}(j,i_2,\ldots,i_d)} x\rangle_{j=0}^{2^{d-1}-1})\}_{0\le i_2,\ldots,i_d < r} = 0$$

or in terms of f,

$$\{f(\langle g^{L_{2n,b-1}(j,i_2,\ldots,i_d)} x, g^{i_1(n2^b)^j + L_{2n,b-1}(j,i_2,\ldots,i_d)} x\rangle_{j=0}^{2^{d-1}-1})\}_{0\le i_2,\ldots,i_d < r} = 0$$

where we have used that the order of $g \ge (n2^b)^{2^{d-1}} \ge (2n \cdot 2^{b-1})^{2^{(d-1)-1}}$. Invoking Proposition 6.4.(2) and Proposition 6.4.(1) we see that the above polynomials being zero is equivalent to

$$\{f(\langle g^{L_{n,b}(2j,i_1,\ldots,i_d)} x, g^{L_{n,b}(2j+1,i_1,\ldots,i_d)} x\rangle_{j=0}^{2^{d-1}-1})\}_{0\le i_2,\ldots,i_d < r} = 0$$

and reindexing, this is equivalent to

$$\{f(\langle g^{L_{n,b}(j,i_1,\ldots,i_d)} x\rangle_{j=0}^{2^d-1})\}_{0\le i_2,\ldots,i_d < r} = 0$$

which is the claim. $\qquad\square$

Chaining together Equation 1 and the above two claims, yields the theorem. $\qquad\square$

Remark 6.9. Let $D = 2^d$. In the above proof we use a recursion scheme that reduces to the problem when $D \to 2$ and $D \to D/2$. This gives rise to the recursion $T(D) \le T(2) + T(D/2)$, where $T(D)$ is the minimum number such that a D-variate rank $\le r$ polynomial can be identity tested using $r^{T(D)}$ univariate polynomials. There is also the recursion $S(D) \le r(Dn)^{D/2} + S(D/2)$, where $S(D)$ is the maximum degree of g seen in this reduction to the univariate case.

One can do slightly better than this scheme by using the "square root trick", where we break up the D-variate case into two copies of the $\sqrt{D}$-variate case. This yields the recursions $T(D) \le 2T(\sqrt{D})$ and $S(D) \le r(Dn)^{\sqrt{D}} \cdot S(\sqrt{D}) + S(\sqrt{D})$. This yields the same solution to T, but has now that $S(D) = \mathcal{O}(r(Dn)^{\mathcal{O}(\sqrt{D})})$ instead of $r(Dn)^{D/2}$. While this is an improvement, it is somewhat mild.

Similarly, one can give other recursion schemes that minimize S (so it is $\mathsf{poly}(n, D, R)$), but at the cost of making $T(D) \approx D$.

6.2 The Hitting Set for Tensors

In this subsection we use the variable reduction theorem of the last subsection to construct hitting sets for tensors. First, recall our notion of a hitting set for tensors from Section 3, as well as the definitions of the polynomial f_T and $\hat{f}_T$ associated with T. As $\mathfrak{C}_{x_1^{i_1} \cdots x_d^{i_d}}(\hat{f}_T) = T(i_1, \ldots, i_d)$ we see that $T = 0$ iff $\hat{f}_T = 0$. Theorem 6.6 shows that $\hat{f}_T = 0$ iff a set of univariate polynomials are all zero. Thus, to test if T is zero we can interpolate each of these polynomials. As these polynomials are defined via $\hat{f}_T$, these interpolations can be realized as inner-products with T. This will yield our hitting set, which we now make formal.

Construction 6.10. *Let $n, r \geq 1$ and $d \geq 2$. Let $\mathbb{K}$ be an extension of $\mathbb{F}$ such that $g \in \mathbb{K}$ is of order $\geq (2dn)^d$ and $\alpha_1, \ldots, \alpha_{dn} \in \mathbb{K}$ are distinct. Let $B_{k,\ell_1,\ldots,\ell_{\lceil \lg d \rceil}} : [\![n]\!]^d \to \mathbb{K}$ to be the rank-1 tensor defined by*

$$B_{k,\ell_1,\ldots,\ell_{\lceil \lg d \rceil}}(i_1, \ldots, i_d) \overset{\text{def}}{=} \prod_{j=1}^{d} (g^{L_{n,\lceil \lg d \rceil}(j,\ell_1,\ldots,\ell_{\lceil \lg d \rceil})} \alpha_k)^{i_j}$$

and let $\mathcal{B}_{d,n,r} \overset{\text{def}}{=} \{B_{k,\ell_1,\ldots,\ell_{\lceil \lg d \rceil}}\}_{0 \leq \ell_1,\ldots,\ell_{\lceil \lg d \rceil} < r, 1 \leq k \leq dn}$.

We now give the analysis for this hitting set.

Theorem 6.11. *Let $n, r \geq 1$ and $d \geq 2$. Then $\mathcal{B}_{d,n,r}$, as defined in Construction 6.10, has the following properties:*

1. *$\mathcal{B}_{d,n,r}$ is a hitting set for $[\![n]\!]^d$ tensors of rank $\leq r$ over $\mathbb{F}$.*

2. *$|\mathcal{B}_{d,n,r}| = dnr^{\lceil \lg d \rceil}$*

3. *$\mathcal{B}_{d,n,r}$ can be computed in $\mathsf{poly}((2dn)^d, r^{\lceil \lg d \rceil})$ operations, where operations (including a successor function in some enumeration of $\mathbb{K}$) over $\mathbb{K}$ are counted at unit cost.*

Proof. $\underline{|\mathcal{B}_{d,n,r}| = dnr^{\lceil \lg d \rceil}}$: This is by definition.

$\underline{\mathcal{B}_{d,n,r} \text{ can be computed in } \mathsf{poly}((2dn)^d, r^{\lceil \lg d \rceil}) \text{ operations}}$: We assume here an enumeration of elements in $\mathbb{K}$ such that the successor in this enumeration can be computed at unit cost. We also will assume testing whether an element is zero, as well as the field elements, are done at unit cost.

First observe that there are at most $(2dn)^d$ solutions to $x^{(2dn)^d} - 1$ over $\mathbb{K}$, so if we enumerate $(2dn)^d + 1$ elements of $\mathbb{K}$, they we can find a $g \in \mathbb{K}$ with order $\geq (2dn)^d$. This is in $\mathsf{poly}((2dn)^d)$ operations. Similarly, the enumeration will give us dn distinct elements which yield the desired α_k.

By Proposition 6.4, $L_{n,\lceil \lg d \rceil}(j, \ell_1, \ldots, \ell_{\lceil \lg d \rceil})$ can be computed in $\mathsf{poly}(d, n, r)$ steps, and this number is $\leq (2dn)^d$, so computing $g^{L_{n,\lceil \lg d \rceil}(j,\ell_1,\ldots,\ell_{\lceil \lg d \rceil})}$ will take at most $\mathsf{poly}((2dn)^d, r)$ operations. Computing the powers of α_k will take $\mathsf{poly}(d, r)$ time. Thus, each $B_{k,\ell_1,\ldots,\ell_{\lceil \lg d \rceil}}$ can be done in $\mathsf{poly}((2dn)^d, r^{\lceil \lg d \rceil})$ steps. As there are $\mathsf{poly}(dnr^{\lceil \lg d \rceil})$ of them, all of $\mathcal{B}_{r,n,m}$ can be computed in $\mathsf{poly}((2dn)^d, r^{\lceil \lg d \rceil})$ operations.

$\underline{\mathcal{B}_{d,n,r} \text{ is a hitting set}}$: By construction $\mathcal{B}_{d,n,r}$ is a set of rank-1 tensors, so it remains to show that it hits each low-rank tensor. Consider any $T : [\![n]\!]^d \to \mathbb{F}$ of rank $\leq r$. We now apply Theorem 6.6 to $\hat{f}_T$, where we consider $\hat{f}_T$ as a $2^{\lceil \lg d \rceil}$-variate polynomial of rank $\leq r$ (by padding $\hat{f}_T$ with dummy variables), individual degrees $< n$, and taking $b = \lceil \lg d \rceil$. This shows that $\hat{f}_T = 0$ iff

$$\{\hat{f}_T(g^{L_{n,\lceil \lg d \rceil}(0,\ell_1,\ldots,\ell_{\lceil \lg d \rceil})}x, g^{L_{n,\lceil \lg d \rceil}(1,\ell_1,\ldots,\ell_{\lceil \lg d \rceil})}x, \ldots, g^{L_{n,\lceil \lg d \rceil}(d-1,\ell_1,\ldots,\ell_{\lceil \lg d \rceil})}x)\}_{0 \leq \ell_1,\ldots,\ell_{\lceil \lg d \rceil} < r} = 0$$

(over $\mathbb{K}$). Each of the above univariate polynomials has degree $\leq d(n-1)$, so interpolating them at $dn \geq d(n-1) + 1$ points will completely determine them. In particular, the above polynomials are zero iff all the evaluations at any dn are zero.

Now we observe, just as in the matrix case, that evaluating the $(\ell_1, \ldots, \ell_{\lceil \lg d \rceil})$-th polynomial in the above set at the point α_k is exactly the same as the inner product $\langle T, B_{k,\ell_1,\ldots,\ell_{\lceil \lg d \rceil}} \rangle$. Thus, $T = 0$ iff $\hat{f}_T = 0$ iff all of these inner-products is zero. This exactly means that $\mathcal{B}_{d,n,r}$ is a hitting set. $\qquad \square$

We remark that this hitting set is of quasi-polynomial size as a rank $\leq r$ tensor $T : [\![n]\!]^d \to F$ can be represented using dnr field elements. However, its construction time is exponential in d. We leave it as an open question as to whether the construction time can be made to match (up to polynomial factors) the size of the hitting set.

6.3 Identity Testing for Tensors over Small Fields

Thus far we have assumed the existence of an element $g \in \mathbb{K}$ of large order. In doing so, all of our hitting sets are tensors over the field $\mathbb{K}$ instead of the base field $\mathbb{F}$. While this is a common assumption when the polynomials of interest are of high degree, the polynomials arising from $[\![n]\!]^d$ tensors on dn variables are of degree $\leq d$, so hitting sets still exist for when $\mathbb{F}$ is $\mathcal{O}(d)$ sized (as seen in Lemma 3.13). In this section, we explore this question and show how to transform hitting sets over $\mathbb{K}$ to hitting sets over $\mathbb{F}$, with some loss. Combining this with the above results, we construct explicit hitting sets over any $\mathbb{F}$.

We first detail a field simulation result that produces improper hitting sets.

Proposition 6.12. *Let $\mathbb{K}$ be an extension of $\mathbb{F}$, with $k = \dim_{\mathbb{F}} \mathbb{K}$. For $\ell \in [\![k]\!]$, let $\varphi_\ell : \mathbb{K} \to \mathbb{F}^k$ denote the k projection maps to the standard basis coordinates of $\mathbb{K}$.*

Let $\mathcal{H} \subseteq \mathbb{K}^{[\![n]\!]^d}$ be an improper hitting-set for $[\![n]\!]^d$ tensors of rank $\leq r$. For $H \in \mathcal{H}$ define $\tilde{H}_\ell$ by

$$(\tilde{H}_\ell)_{i_1,\ldots,i_d} = \varphi_\ell(H_{i_1,\ldots,i_d})$$

and define

$$\tilde{\mathcal{H}} = \{\tilde{H}_\ell\}_{H \in \mathcal{H}, \ell \in [\![k]\!]}$$

Then

1. *If all tensors in $\mathcal{H}$ are s-sparse, then so are all tensors in $\tilde{\mathcal{H}}$.*

2. *$|\tilde{\mathcal{H}}| = k \cdot |\mathcal{H}|$.*

3. *$\tilde{\mathcal{H}}$ is an improper hitting set for $[\![n]\!]^d$ tensors of rank $\leq r$.*

Proof. (1): If $H_{i_1,\ldots,i_d} = 0$ then it follows that $(\tilde{H}_\ell)_{i_1,\ldots,i_d} = 0$ for all ℓ.

(2): This is by construction.

(3): Let $\alpha_0, \ldots, \alpha_{k-1}$ be the standard basis for $\mathbb{K}$ as a $\mathbb{F}$-vector-space. Then it follows that $H = \sum_{\ell \in [\![k]\!]} H_\ell \, \alpha_\ell$.

Consider some tensor $T : [\![n]\!]^d \to \mathbb{F}$ of rank $\leq r$. Then we know that there is some $H \in \mathcal{H}$ such that $\langle T, H \rangle \neq 0$. It follows that there must be some ℓ with $\langle T, H_\ell \rangle \neq 0$. $\qquad \square$

We now apply this to our hitting set results.

Corollary 6.13. *Let $m \geq n \geq r \geq 1$. Over any field $\mathbb{F}$, there is an $\mathsf{poly}(m)$-explicit improper hitting set for $n \times m$ matrices of rank $\leq r$, of size $\mathcal{O}(rm \lg m)$. Further, each matrix in the hitting set is $\mathcal{O}(n)$-sparse.*

Proof. If $\mathbb{F}$ has an element of order $\geq m$, then Theorem 5.8 suffices.

If not, let $\mathbb{K}$ be an extension field of $\mathbb{F}$ such that $\dim_\mathbb{F} \mathbb{K} = \Theta(\lg m)$, and thus there is an element of order $\geq m$ in $\mathbb{K}$. Such an extension can be explicitly described by an irreducible polynomial over $\mathbb{F}$ of degree $\Theta(\lg m)$, which can found in $\mathsf{poly}(m)$ time, in which time we can also find g. Using Theorem 5.8 to get an n-sparse (improper) hitting-set over $\mathbb{K}$ for these $\mathbb{F}$-matrices, and applying Proposition 6.12 yields the result. $\square$

Corollary 6.14. *Let $n, r \geq 1$, $d \geq 2$. Over any field $\mathbb{F}$, there is an $\mathsf{poly}((2nd)^d, r^{\mathcal{O}(\lg d)})$-explicit improper hitting set for $[\![n]\!]^d$-tensors of rank $\leq r$, of size $\mathcal{O}(dnr^{\mathcal{O}(\lg d)} \cdot (d \lg 2dn))$.*

Proof. If $\mathbb{F}$ has an element of order $\geq (2nd)^d$, then Theorem 6.11 suffices.

If not, let $\mathbb{K}$ be an extension field of $\mathbb{F}$ such that $\dim_\mathbb{F} \mathbb{K} = \Theta(d \lg(2nd))$, and thus there is an element of order $\geq (2nd)^d$ in $\mathbb{K}$. Such an extension can be explicitly described by an irreducible polynomial over $\mathbb{F}$ of degree $\Theta(d \lg 2nd)$, which can found in $\mathsf{poly}((2nd)^d)$ time, in which time we can also find g. Using Theorem 6.11 to get a hitting-set over $\mathbb{K}$ for these $\mathbb{F}$-matrices, and applying Proposition 6.12 yields the result. $\square$

The above results only yield improper hitting sets. We now show how to preserve the rank-1 property of the original hitting set, and thus get proper hitting sets over small fields. To do, we first recall a standard fact in algebra showing that $\mathbb{K}$ is isomorphic to a subring of $\mathbb{F}$-matrices.

Lemma 6.15. *Let $\mathbb{K}$ be an extension of $\mathbb{F}$, and let $k = \dim_\mathbb{F} \mathbb{K} < \infty$ so that $\mathbb{K} = \mathbb{F}^k$ as vector spaces. For any $\alpha \in \mathbb{K}$ define the linear map $\mu_\alpha : \mathbb{F}^k \to \mathbb{F}^k$ given by the multiplication map $x \mapsto \alpha x$. Let $M_\alpha \in \mathbb{F}^{k \times k}$ be the associated matrix. Then the map $M_{(\cdot)} : \mathbb{K} \to \mathbb{F}^{k \times k}$ is an isomorphism as $\mathbb{F}$-algebras.*

Proof. The map is clearly well-defined. To see the additive homomorphism, note that as $(\alpha + \beta)\gamma = \alpha\gamma + \beta\gamma$ for any $\alpha, \beta, \gamma \in \mathbb{K}$, it follows that $M_{\alpha+\beta} \cdot \gamma = M_\alpha \gamma + M_\beta \gamma$ for any $\gamma \in \mathbb{F}^k = \mathbb{K}$ (where we abuse notation by writing γ to denote an element in $\mathbb{K}$ as well as its representation as a vector in $\mathbb{F}^k$). Taking γ for each vector in some basis shows that $M_{\alpha+\beta} = M_\alpha + M_\beta$.

Similarly, to see the multiplicative homomorphism note that for any $\alpha, \beta, \gamma \in \mathbb{K}$ we have that $(\alpha\beta)\gamma = \alpha(\beta\gamma)$. Thus it must be that $M_\alpha M_\beta \gamma = \alpha\beta\gamma = M_{\alpha\beta}\gamma$. Again, taking γ over each vector in a basis determines a linear operator. Thus it must be that $M_\alpha M_\beta = M_{\alpha\beta}$.

Noting that for $\alpha \in \mathbb{F}$ we have that $M_\alpha = \alpha I_k$ we then gain $\mathbb{F}$-linearity of the map.

If $\alpha \neq 0$ then $M_\alpha \cdot M_{\alpha^{-1}} = M_1 = I_k$, so M_α is invertible. Thus, if $M_\alpha = M_\beta$ then $M_{\alpha-\beta} = 0_k$, which implies that $\alpha - \beta = 0$ (as else $M_{\alpha-\beta}$ would be invertible) and thus $\alpha = \beta$. This implies the map is injective.

As a map is surjective onto its image by definition, this establishes the $\mathbb{F}$-algebra homomorphism. $\square$

We now show how to use this alternate representation of $\mathbb{K}$ as a way to simulate hitting sets defined over $\mathbb{K}$ by hitting sets defined over $\mathbb{F}$.

Proposition 6.16. *Let $\mathbb{K}$ be an extension of $\mathbb{F}$, with $k = \dim_\mathbb{F} \mathbb{K}$. Let $\mathcal{H} \subseteq \mathbb{K}^{[\![n]\!]^d}$ be a hitting-set for $[\![n]\!]^d$ tensors of rank $\leq r$. For $H = \otimes_{j=1}^d \mathbf{v}_j \in \mathbb{K}^{[\![n]\!]^d}$ define $\tilde{\mathbf{v}}_{j,\ell_0,\ldots,\ell_d} \in \mathbb{F}^n$ by*

$$(\tilde{\mathbf{v}}_{j,\ell_0,\ldots,\ell_d})_i = (M_{(\mathbf{v}_j)_i})_{\ell_{j-1},\ell_j}$$

where $M_{(\cdot)} : \mathbb{K} \to \mathbb{F}^{k \times k}$ is the isomorphism of Lemma 6.15 and define

$$\tilde{H}_{\ell_0,\ldots,\ell_d} = \bigotimes_{j=1}^d \tilde{\mathbf{v}}_{j,\ell_0,\ldots,\ell_d}$$

30

and define

$$\tilde{\mathcal{H}} = \{\tilde{H}_{\ell_0,\ldots,\ell_{d-1},0}\}_{H \in \mathcal{H}, 0 \leq \ell_0,\ldots,\ell_{d-1} < k}$$

Then

1. *$\tilde{\mathcal{H}}$ is a set of rank-1 $\mathbb{F}$-tensors of shape $[\![n]\!]^d$.*

2. *$|\tilde{\mathcal{H}}| = k^d \cdot |\mathcal{H}|$.*

3. *$\tilde{\mathcal{H}}$ is a hitting set for $[\![n]\!]^d$ tensors of rank $\leq r$.*

Proof. (1): This is by construction.

(2): This is by construction.

(3): Consider some tensor $T : [\![n]\!]^d \to \mathbb{F}$ of rank $\leq r$. Then we know that there is some $H \in \mathcal{H}$ with $H = \otimes_{j=1}^d \mathbf{v}_j$, such that $\langle T, H \rangle \neq 0$. Then we see that (we now abuse notation, by writing μ now to denote the map $M_{(\cdot)}$)

$$\mu(\langle T, H \rangle)_{\ell_0,\ell_d} = \mu\left(\sum_{i_1,\ldots,i_d \in [\![n]\!]} T(i_1,\ldots,i_d) \prod_{j=1}^d (\mathbf{v}_j)_{i_j}\right)_{\ell_0,\ell_d}$$

$$= \sum_{i_1,\ldots,i_d \in [\![n]\!]} T(i_1,\ldots,i_d) \left(\prod_{j=1}^d \mu\left((\mathbf{v}_j)_{i_j}\right)\right)_{\ell_0,\ell_d}$$

fully expanding the matrix multiplication of d matrices, each $k \times k$,

$$= \sum_{i_1,\ldots,i_d \in [\![n]\!]} T(i_1,\ldots,i_d) \sum_{\ell_1,\ell_1,\ldots,\ell_{d-1} \in [\![k]\!]} \prod_{j=1}^d \mu\left((\mathbf{v}_j)_{i_j}\right)_{\ell_{j-1},\ell_j}$$

$$= \sum_{\ell_1,\ell_1,\ldots,\ell_{d-1} \in [\![k]\!]} \sum_{i_1,\ldots,i_d \in [\![n]\!]} T(i_1,\ldots,i_d) \prod_{j=1}^d \mu\left((\mathbf{v}_j)_{i_j}\right)_{\ell_{j-1},\ell_j}$$

$$= \sum_{\ell_1,\ell_1,\ldots,\ell_{d-1} \in [\![k]\!]} \langle T, \tilde{H}_{\ell_0,\ldots,\ell_d} \rangle$$

So it follows that if $\mu(\langle T, H \rangle)_{\ell_0,\ell_d} \neq 0$ then there is some $\ell_1,\ldots,\ell_{d-1} \in [\![k]\!]$ such that $\langle T, \tilde{H}_{\ell_0,\ldots,\ell_d} \rangle \neq 0$.

Let γ_0 denote the element in $\mathbb{K}$ corresponding to $\mathbf{e}_0 \in \mathbb{F}^k$ (the standard basis vector with a 1 in the zero position). Note that $\gamma_0 \neq 0$. Then it follows that for any $\alpha \in \mathbb{K}$ that $M_\alpha \mathbf{e}_0 = M_\alpha \gamma_0 = \alpha\gamma_0$ (where we abuse notation by writing $\alpha\gamma_0$ to denote an element in $\mathbb{K}$ as well as the vector representing $\alpha\gamma_0$ in $\mathbb{F}^k$). Thus, α is fully recoverable from $M_\alpha \mathbf{e}_0$, and in particular, $\alpha = 0$ iff $M_\alpha \mathbf{e}_0 = 0$.

Thus, to test if $\langle T, H \rangle = 0$ (over $\mathbb{K}$) it is enough to test if $\mu(\langle T, H \rangle)_{\ell_0,0} = 0$ (over $\mathbb{F}$) for all $\ell_0 \in [\![k]\!]$. Combining this with the above we see that $\langle T, \mathcal{H} \rangle = \mathbf{0}$ (over $\mathbb{K}$) iff $\langle T, \tilde{\mathcal{H}} \rangle = \mathbf{0}$. $\qquad\square$

We now use the above result to get hitting sets for matrices and tensors over any field.

Corollary 6.17. *Let $m \geq n \geq r \geq 1$. Over any field $\mathbb{F}$, there is an $\mathsf{poly}(m)$-explicit hitting set for $n \times m$ matrices of rank $\leq r$, of size $\mathcal{O}(rm \lg^2 m)$.*

Proof. If $\mathbb{F}$ has an element of order $\geq m$, then Theorem 5.6 suffices.

If not, let $\mathbb{K}$ be an extension field of $\mathbb{F}$ such that $\dim_{\mathbb{F}} \mathbb{K} = \Theta(\lg m)$, and thus there is an element of order $\geq m$ in $\mathbb{K}$. Such an extension can be explicitly described by an irreducible polynomial over $\mathbb{F}$ of degree $\Theta(\lg m)$, which can found in $\mathsf{poly}(m)$ time, in which time we can also find g. Using Theorem 5.6 to get a hitting-set over $\mathbb{K}$ for these $\mathbb{F}$-matrices, and applying Proposition 6.16 yields the result. $\qquad\square$

Corollary 6.18. *Let $n, r \geq 1$, $d \geq 2$. Over any field $\mathbb{F}$, there is an $\mathsf{poly}((2nd)^d, r^{\mathcal{O}(\lg d)})$-explicit hitting set for $[\![n]\!]^d$-tensors of rank $\leq r$, of size $\mathcal{O}(dnr^{\mathcal{O}(\lg d)}(d \lg 2dn)^d)$.*

Proof. If $\mathbb{F}$ has an element of order $\geq (2nd)^d$, then Theorem 6.11 suffices.

If not, let $\mathbb{K}$ be an extension field of $\mathbb{F}$ such that $\dim_{\mathbb{F}} \mathbb{K} = \Theta(d \lg(2nd))$, and thus there is an element of order $\geq (2nd)^d$ in $\mathbb{K}$. Such an extension can be explicitly described by an irreducible polynomial over $\mathbb{F}$ of degree $\Theta(d \lg 2nd)$, which can found in $\mathsf{poly}((2nd)^d)$ time, in which time we can also find g. Using Theorem 6.11 to get a hitting-set over $\mathbb{K}$ for these $\mathbb{F}$-matrices, and applying Proposition 6.16 yields the result. $\qquad\square$

7 Explicit Low Rank Recovery of Matrices

Thus far we have discussed identity testing for matrices (and tensors). There the main concern is to (deterministically) determine whether the matrix is identically zero. However, we may also ask for more, in that we may want to (deterministically) reconstruct the entire matrix. Throughout this section we will only discuss deterministic measurements which are linear (so are inner products with the unknown matrix or vector), non-adaptive (so the measurements are independent of the unknown matrix or vector) and noiseless. The focus on deterministic measurements differs from prior work, which typically focuses on showing that certain distributions of measurements allow recovery with high probability. That the measurements are restricted to be linear is a common assumption in compressed sensing. Non-adaptiveness is also a common assumption, but it is important to note that recent work [IPW11] shows that adaptivity in (noisy) sparse-recovery can be more powerful than non-adaptivity. Finally, we assume our matrices are *exactly* rank $\leq r$, not just close to some matrix that is rank $\leq r$, and we assume that our measurements are noiseless. This is not quite practical for compressed sensing, but some previous work also makes this assumption [GK72, Gab85b, Gab85a, Del78, Rot91, Rot96, RFP10]. Further, the noiseless case is more natural for our applications to rank-metric codes, and allows the results to be field independent.

We begin by noting that low-rank recovery (recall Definition 3.9, which we consider in this section only for matrices) generalizes the notion of sparse-recovery, which is the defined formally as the following.

Definition 7.1. *A set of vectors $\mathcal{V} \subseteq \mathbb{K}^n$ is an s-**sparse-recovery set** if for every vector $\mathbf{x} \in \mathbb{F}^n$ with at most s non-zero entries, $\mathbf{x}$ is uniquely determined by $\mathbf{y}$, where $\mathbf{y} \in \mathbb{K}^{\mathcal{V}}$ is defined by $y_{\mathbf{v}} \overset{\text{def}}{=} \langle \mathbf{x}, \mathbf{v} \rangle$, for $\mathbf{v} \in \mathcal{V}$.*

*An algorithm performs **recovery** from $\mathcal{R}$ if, for each such $\mathbf{x}$, it recovers that $\mathbf{x}$ given $\mathbf{y}$.*

That LRR generalizes the sparse-recovery is formalized in the following claim.

Lemma 7.2. *Given an r-low-rank recovery set $\mathcal{R}$ for $n \times n$ matrices, there is a set $\mathcal{V} \subseteq \mathbb{F}^n$, efficiently constructible from $\mathcal{R}$, with $|\mathcal{V}| = |\mathcal{R}|$, such that $\mathcal{V}$ is an r-sparse-recovery set.*

Proof. Given an r-sparse vector $\mathbf{x} \in \mathbb{F}^n$ construct the diagonal matrix $\Lambda \in \mathbb{F}^{n \times n}$ with $\mathbf{x}$ on its diagonal. Thus, Λ is rank $\leq r$. Thus, if we can perform r-low-rank-recovery we can also do r-sparse recovery. Each such measurement of Λ can be seen to also be a linear measurement of $\mathbf{x}$, so this yields $\mathcal{V}$. $\qquad\square$

The purpose of this section is to show that the two problems (when concerned with non-adaptive, exact measurements) are essentially equivalent. That is, one can (efficiently) perform low-rank-recovery given *any* construction of a sparse-recovery set.

To motivate the reduction from low-rank-recovery to sparse-recovery, we will show that our above hitting set results already imply low-rank-recovery results, and that these hitting sets can be seen as being constructed from a well-known sparse-recovery construction. We begin by recalling Lemma 3.10 (standard) fact that *any* hitting set family yields a low-rank-recovery family, so in particular our results do so. Combining the above with our constructions of hitting sets, we derive the following corollary.

Corollary 7.3. *The sets $\mathcal{B}_{2r,n,m}$, $\mathcal{D}_{2r,n,m}$, $\mathcal{D}'_{2r,n,m}$, and $\mathcal{B}'_{2r,n,m}$ (from Construction 5.5, Construction 5.7 and Construction 5.9) are r-low-rank-recovery sets.*

However, the above results are non-constructive. That is, they show that recovery is information-theoretically possible from this set of matrices, but do not give any insight how to perform this recovery efficiently. The purpose of this section is to show that we can strengthen Corollary 7.3 such that the recovery can be efficiently performed.

To motivate our recovery algorithm, let us first discuss the r-low-rank-recovery set $\mathcal{D}_{2r,n,m}$. For an $n \times m$ matrix M, consider the constraints that the system $\langle M, \mathcal{D}_{2r,n,m} \rangle = \mathbf{0}$ imposes on M. By construction of $\mathcal{D}_{2r,n,m}$, we see that each k-diagonal of M has $2r$ constraints imposed on it. If we write the k-diagonal of M as $\mathbf{x}$, we can express the constraints on $\mathbf{x}$ as $A\mathbf{x} = 0$, where A is of size $2r \times |\mathbf{x}|$, where $|\mathbf{x}|$ denotes the size of the k-diagonal. Further, A has the format (when $2r \leq k+1 \leq n$)

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & g & g^2 & \cdots & g^{|x|-1} \\ 1 & g^2 & g^4 & \cdots & g^{2(|x|-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & g^{2r-1} & g^{2(r-1)} & \cdots & g^{(2r-1)(|x|-1)} \end{pmatrix} \qquad (2)$$

which is important because of the following claim.

Lemma 7.4. *Let $\mathbf{x}$ be an r-sparse $\mathbb{F}$-vector. Let g be of order $\geq |\mathbf{x}|$ in some extension $\mathbb{K}$ of $\mathbb{F}$, and let A be an $2r \times |\mathbf{x}|$ sized matrix of the form in Equation (2). Then $\mathbf{x}$ is determined by $A\mathbf{x}$.*

Proof. Suppose $\mathbf{x}$ and $\mathbf{y}$ are two r-sparse vectors such that $A\mathbf{x} = A\mathbf{y}$. By linearity we then have that $A(\mathbf{x} - \mathbf{y}) = 0$, so that A has a linear dependence on $\leq 2r$ of the columns.

However, as the order of g is $\geq |\mathbf{x}|$, each $2r \times 2r$ minor of A is a Vandermonde matrix on distinct entries, and so is full-rank. In particular, any linear dependence on $\leq 2r$ of the rows must be zero. So $\mathbf{x} - \mathbf{y} = 0$, so $\mathbf{x} = \mathbf{y}$. Thus, $\mathbf{x}$ is determined by $A\mathbf{x}$. $\qquad\square$

Note that the row-space of the above matrix is a Reed-Solomon code, and so the above lemma shows the standard fact that the dual Reed-Solomon code has good distance. In particular, we can do error correction for up to r errors. This is exactly the question of r-sparse recovery (when we are correcting errors from the $\mathbf{0}$ codeword).

This lemma shows that at each k-diagonal, $\mathcal{D}_{2r,n,m}$ embeds an r-sparse-recovery set. Thus, it seems plausible that a low-rank-recovery algorithm for $\mathcal{D}_{2r,n,m}$ might only use this fact in its construction, and thus show low-rank-recovery can be done whenever each of the k-diagonals are measured according to an r-sparse-recovery set. Indeed, this is what is shown by Theorem 7.19.

The reduction from low-rank-recovery to sparse-recovery is detailed in the following two subsections. The first subsection details a slightly stronger notion of sparse-recovery, which we call *advice-sparse-recovery*. This notion requires sparse-recovery when supplied with some advice on the support of the unknown vector. This is the correct notion of sparse-recovery when attempting to do low-rank-recovery, but the standard notion is sufficient with some loss in parameters. We describe a well-known algorithm, known as Prony's method, for efficiently performing the recovery illustrated in Lemma 7.4, and show that this method can be modified to also achieve advice-sparse-recovery.

The second subsection gives the reduction from low-rank-recovery to sparse-recovery. Combining this with our modifications to Prony's method, we conclude that the low-rank-recovery shown in Corollary 7.3 can also be performed efficiently.

7.1 Prony's Method and Syndrome Decoding of Dual Reed-Solomon Codes

In this section we detail an algorithm for efficiently performing the sparse-recovery demonstrated in Corollary 7.4. While our discovery of the algorithm was independent of prior work, it was original detailed by Prony [dP95] in 1795 and is well-known in the signal-processing community (see [PCM88] and references there-in). It can also be seen as syndrome decoding of the dual to the Reed-Solomon code. What we detail here is not exactly the original method, as we seek an *advice-sparse-recovery set*, which is a slightly stronger condition which will be useful in our low-rank-recovery algorithm. In coding theory terminology, we are seeking to syndrome decode the dual Reed-Solomon code in the presence of erasures. We now define this stronger notion.

Definition 7.5. *A set of vectors $\mathcal{V} \subseteq \mathbb{F}^n$ is an s-**advice-sparse-recovery set** if for every $S \in \binom{[\![n]\!]}{\leq 2s}$, and vector $\mathbf{x} \in \mathbb{F}^n$ with $\leq s - |S|/2$ non-zero entries outside of S, $\mathbf{x}$ is uniquely determined by S and $\mathbf{y}$, where $\mathbf{y} \in \mathbb{F}^{\mathcal{V}}$ is defined by $y_{\mathbf{v}} \overset{\text{def}}{=} \langle \mathbf{x}, \mathbf{v} \rangle$, for $\mathbf{v} \in \mathcal{V}$.*

*An algorithm performs **recovery** from $\mathcal{V}$ if, for each such $\mathbf{x}$, it recovers that $\mathbf{x}$ given S and $\mathbf{y}$.*

Note that the vector $\mathbf{y}$ can also be defined as $\mathbf{y} = V\mathbf{x}$, where $V \in \mathbb{F}^{\mathcal{V} \times n}$ is the matrix whose rows are those vectors in $\mathcal{V}$.

The motivation for this new definition is to capture situations where $\mathbf{x}$ is known to have sparse support overall, and further some of its support is already known and given by the set S. The results below show that exploiting this knowledge allows $|\mathcal{V}|$ to be smaller. To see why this might be intuitively plausible, one can count degrees of freedom. In an s-sparse vector $\mathbf{x}$, there are intuitively $2s$ degrees of freedom: it takes s degrees to determine $\text{Supp}(\mathbf{x})$, and it takes s degrees to determine $(x_i)_{i \in \text{Supp}(\mathbf{x})}$.

In the above definition of a s-advice-sparse-recovery set, the unknown vector $\mathbf{x}$ can have a support of size $2s$ (when $|S| = 2s$). If one ignores the set S, there would be $4s$ degrees of freedom, by the above argument, leading one to expect a lower bound of "$|\mathcal{V}| \geq 4s$". However, if one exploits this knowledge, then there are only $s - |S|/2$ degrees of freedom to determine $\text{Supp}(\mathbf{x})$, and $|S| + (s - |S|/2)$ degrees of freedom to determine $(x_i)_{i \in \text{Supp}(\mathbf{x})}$, which gives a total of $2s$ degrees of freedom.

Thus we see that using the information given in S can reduce the degrees of freedom in $\mathbf{x}$, and below we match this intuition by recovering $\mathbf{x}$ from $2s$ measurements. This intuition is the same intuition in coding theory that an erasure is a "half error", but specialized to syndrome decoding.

In the next subsection, we will see that r-low-rank-recovery reduces to the problem of r-advice-sparse-recovery. When $S = \emptyset$ then r-advice-sparse-recovery is exactly the notion of an r-sparse-recovery. However, we will need S to have size up to $2r$. Note that regardless of the size of S, $\mathbf{x}$ will be $2r$-sparse. Thus the following lemma is immediate.

Lemma 7.6. *Let $\mathcal{V}$ be a $2s$-sparse-recovery set. Then $\mathcal{V}$ is also a s-advice-sparse-recovery set.*

To our knowledge, the existing work on Prony's method gives an algorithm for perform sparse-recovery. However, in our reduction advice-sparse-recovery is more natural. The above lemma shows that these notions are equivalent, up to a loss in parameters. However, to get better constructions we detail how to modify Prony's method to achieve advice-sparse-recovery without a loss in parameters.

Algorithm 1 Prony's method with an advice set

1: **procedure** PRONYSMETHOD(n,s,S,y,$\{g_0,\ldots,g_{n-1}\}$)
2:     **if** $|S|$ odd **then**
3:         Enlarge S by 1 position
4:     **end if**
5:     $t \overset{\text{def}}{=} |S|/2$
6:     Construct $A \in \mathbb{F}^{(s+t)\times(s+t+1)}$, $A_{i,j} \overset{\text{def}}{=} \begin{cases} g_{k_j}^i & \text{if } i < |S| \\ y_{i+j-|S|} & \text{else} \end{cases}$ $\triangleright$ for $S = \{k_0,\ldots,k_{|S|-1}\}$
7:     Convert A to row-reduced echelon form
8:     Let $r \in [\![s+t]\!]$ be the largest number so the $r \times r$ leading principal minor of A is full rank.
9:     Let $\mathbf{c} \in \mathbb{F}^{r+1}$ be a non-zero vector in the nullspace of leading $r \times (r+1)$ minor of A.
10:     Define $p(x) \overset{\text{def}}{=} \sum_{i=0}^r c_i x^i$
11:     $T \overset{\text{def}}{=} \{k | p(g_k) = 0\}$ $\triangleright$ T will be $\text{Supp}(\mathbf{x})$
12:     $D \in \mathbb{F}^{2s \times T}$, $D_{i,k} \overset{\text{def}}{=} g_k^i$, for $k \in T$
13:     Solve $D\mathbf{z} = \mathbf{y}$ for $\mathbf{z}$ (using Gaussian Elimination)
14:     Define $\mathbf{x} \in \mathbb{F}^n$, as $x_k = \begin{cases} z_k & \text{if } k \in T \\ 0 & \text{else} \end{cases}$
15:     **return x**
16: **end procedure**

Theorem 7.7. *Let $\mathbb{F}$ be a field, and let $g_0,\ldots,g_{n-1} \in \mathbb{F}$ be distinct. Let $\mathbf{v}_i \in \mathbb{F}^n$ be the vector with entries $(\mathbf{v}_i)_j \overset{\text{def}}{=} g_j^i$. Then the set $\mathcal{V} = \{\mathbf{v}_i\}_{i=0}^{2s-1}$ is an s-advice-sparse-recovery set. Further, PRONYSMETHOD($n, s, S, V\mathbf{x}, \{g_0,\ldots,g_{n-1}\}$) (Algorithm 1) recovers $\mathbf{x}$ in $\mathcal{O}(s^3 + sn)$ operations (where operations over $\mathbb{F}$ are counted at unit cost), where $V \in \mathbb{F}^{2s \times n}$ is the matrix with the vectors in $\mathcal{V}$ as its rows.*

In particular, if $g \in \mathbb{F}$ has order at least n, we can take $g_j = g^j$.

Proof. As above, define $V \in \mathbb{F}^{2s \times n}$ to be the matrix whose rows are those vectors $\mathbf{v}_i$. That is, $V_{i,j} = g_j^i$. As the g_j are distinct, it follows that every $2s \times 2s$ minor of V is an invertible Vandermonde matrix. It follows that each subset of $\leq 2s$ columns of V are linearly independent.

Define $\mathbf{g}_j \in \mathbb{F}^{2s}$ by $(\mathbf{g}_j)_i \overset{\text{def}}{=} g_j^i$. It follows that the $\mathbf{g}_j$ are the columns of V. For a vector $\mathbf{a} \in \mathbb{F}^m$, define $\mathbf{a}^{[\ell,k]} \in \mathbb{F}^{k-\ell+1}$ to be the vector with entries $a_\ell,\ldots,a_k$.

$\mathcal{V}$ is a s-advice-sparse-recovery set: Consider a set $S \in \binom{[\![n]\!]}{\leq 2s}$ and vectors $\mathbf{x}, \mathbf{w} \in \mathbb{F}^n$ where each have at most $s - |S|/2$ non-zero entries outside of S. Suppose that $V\mathbf{x} = V\mathbf{y}$. By linearity, this

35

yields the vector $\mathbf{x} - \mathbf{w}$ such that $V(\mathbf{x} - \mathbf{w}) = 0$ and $\mathbf{x} - \mathbf{w}$ has at most $2(s - |S|/2)$ non-zero entries outside of S. In total, $\mathbf{x} - \mathbf{w}$ has at most $|S| + 2(s - |S|/2) = 2s$ non-zero entries. However, as mentioned above, each subset of $\leq 2s$ columns of V are linearly independent. As $\mathbf{0} = V(\mathbf{x} - \mathbf{w})$ is a linear combination of $\leq 2s$ columns of V, it follows that $\mathbf{x} - \mathbf{w} = 0$. Thus, any such $\mathbf{x}$ is uniquely determined by S and $V\mathbf{x}$.

Algorithm 1 performs recovery: Consider a set $S \in \binom{[\![n]\!]}{\leq 2s}$, with $S = \{k_0, \ldots, k_{|S|-1}\}$. For any vector $\mathbf{x}$ the condition that $|\operatorname{Supp}(\mathbf{x}) \setminus S| \leq s - |S|/2$ implies that $|\operatorname{Supp}(\mathbf{x}) \setminus S| \leq s - \lceil |S|/2 \rceil$ by integrality. It follows that we may assume the set S has even size, as we can always enlarge it by one position without changing the above constraints on the support of $\mathbf{x}$. (If $S = [\![n]\!]$ prior to this enlargement, we simulate $n + 1$ long vectors). Now define t so $|S| = 2t$.

Consider vector $\mathbf{x} \in \mathbb{F}^n$ with at most $\nu \leq s - |S|/2 = s - t$ non-zero entries outside of S. By construction of $\mathbf{y}$ (recall $\mathbf{y} = V\mathbf{x}$),

$$\mathbf{y} = \sum_{k \in S} x_k \mathbf{g}_k + \sum_{k \in \operatorname{Supp}(\mathbf{x}) \setminus S} x_k \mathbf{g}_k \tag{3}$$

The aim of this analysis will be to show that we can determine $\operatorname{Supp}(\mathbf{x})$ and then leverage this to solve the above equation for $\mathbf{x}$.

We now establish some theory to analyze the algorithm. The above equation can be refined to see that

$$\mathbf{y}^{[\ell,\ell']} = \sum_{k \in S} x_k \mathbf{g}_k^{[\ell,\ell']} + \sum_{k \in \operatorname{Supp}(\mathbf{x}) \setminus S} x_k \mathbf{g}_k^{[\ell,\ell']} = \sum_{k \in S} x_k g_k^\ell \mathbf{g}_k^{[0,\ell'-\ell]} + \sum_{k \in \operatorname{Supp}(\mathbf{x}) \setminus S} x_k g_k^\ell \mathbf{g}_k^{[0,\ell'-\ell]} \tag{4}$$

We note here that the rows of A involving $\mathbf{y}$ can be written as $\mathbf{y}^{[0,s+t]}, \ldots, \mathbf{y}^{[s-t-1,2s-1]}$. As $\mathbf{y}$ has $2s$ entries, each of these vectors is well-defined, and each entry in $\mathbf{y}$ is used in A.

We now establish some claims about A using that $\nu = |\operatorname{Supp}(\mathbf{x}) \setminus S|$.

Claim 7.8. *The* $(|S| + \nu + 1) \times (|S| + \nu + 1)$ *leading principal minor of* A *is singular.*

Proof. Denote this leading minor by M. The rows of M are of the form $\mathbf{g}_{k_j}^{[0,|S|+\nu]}$ for $j < |S|$, and $\mathbf{y}^{[\ell,|S|+\nu+\ell]}$ for $0 \leq \ell < \nu$. Trivially, for each $j < |S|$, $\mathbf{g}_{k_j}^{[0,|S|+\nu]} \in \operatorname{Span}\{\mathbf{g}_k^{[0,|S|+\nu]}\}_{k \in \operatorname{Supp}(\mathbf{x}) \cup S}$. Further, Equation 4 shows that $\mathbf{y}^{[\ell,|S|+\nu+\ell]} \in \operatorname{Span}\{\mathbf{g}_k^{[0,|S|+\nu]}\}_{k \in \operatorname{Supp}(\mathbf{x}) \cup S}$. Thus, the $|S|+\nu+1$ rows of M each lie in a $\leq (|S| + \nu)$-dimensional subspace, implying that M is singular. $\square$

Claim 7.9. *The* $(|S| + \nu) \times (|S| + \nu)$ *leading principal minor of* A *is invertible.*

Proof. Denote this leading minor by M. We will show that $M = BC$, for $B, C \in \mathbb{F}^{(|S|+\nu) \times (|S|+\nu)}$ both invertible, which implies the claim.

Let the rows of C be the vectors $\mathbf{g}_k^{[0,|S|+\nu-1]}$, for each $k \in \operatorname{Supp}(\mathbf{x}) \cup S$. We will index the rows by the g_k, and assume that the first $|S|$ such g_k are those with $k \in S$. This is a Vandermonde matrix, and as such is invertible.

Let B be defined by

$$B_{i,g_k} = \begin{cases} 1 & \text{if } i = k < |S| \\ 0 & \text{if } i \neq k, i < |S| \\ x_k g_k^{i-|S|} & \text{else} \end{cases}$$

It follows from Equation 4 that $M = BC$. Note that B has the form

$$\begin{bmatrix} I_{|S|} & 0 \\ EX_1 & FX_2 \end{bmatrix}$$

where $X_1 \in \mathbb{F}^{|S| \times |S|}$ is the diagonal matrix with diagonal entries x_k, for $k \in S$ (ordered to match C), $X_2 \in \mathbb{F}^{\nu \times \nu}$ is the diagonal matrix with diagonal entries x_k, for $k \in \text{Supp}(\mathbf{x}) \setminus S$ (ordered to match C), $E \in \mathbb{F}^{\nu \times |S|}$ is the Vandermonde matrix with entries $E_{i,g_k} \overset{\text{def}}{=} g_k^i$, for $k \in S$, and $F \in \mathbb{F}^{\nu \times \nu}$ is the (invertible) Vandermonde matrix with entries $F_{i,g_k} \overset{\text{def}}{=} g_k^i$ for $k \in \text{Supp}(\mathbf{x}) \setminus S$.

Note that X_1 might entirely be zero, but X_2 must be invertible by assumption that $\mathbf{x}$ has exactly ν non-zero entries outside of S. As F is invertible, it follows that FX_2 is invertible, and thus B is also invertible.

Thus, $M = BC$ with B and C both invertible matrices. The claim follows. $\qquad\square$

As the first $|S|$ rows of A are rows of a Vandermonde matrix, it follows that the first $|S|$ leading principal minors are all invertible. This, along with the above two claims, thus show that $|S| + \nu$ is the minimum r such the $(r+1) \times (r+1)$ leading principal minor of A is singular. It follows that in Algorithm 1 the r value chosen in Step 8 is in fact $|S| + \nu$.

We now show that the $\mathbf{c}$ chosen by the algorithm also has significance.

Claim 7.10. *Let* $p(x) \overset{\text{def}}{=} \prod_{k \in \text{Supp}(\mathbf{x}) \cup S}(x - g_k) = \sum_{i=0}^{|S|+\nu} c_i x^i$. *Then the vector* $\mathbf{c} \in \mathbb{F}^{|S|+\nu+1}$ *defined by those coefficients* c_i *is in the nullspace of the* $(|S| + \nu) \times (|S| + \nu + 1)$ *leading minor of A.*

Proof. Denote this leading minor by M.

Note that for any g_k with $k \in \text{Supp}(\mathbf{x}) \cup S$ has that $\langle \mathbf{g}_k^{[0,|S|+\nu]}, \mathbf{c} \rangle = 0$, as this simply says that $p(g_k) = 0$. Thus, we see that $\mathbf{c}$ is orthogonal to the first $|S|$ rows of M.

Now observe that Equation 4 shows that the last ν rows of M are all in the span of the vectors $\mathbf{g}_k^{[0,|S|+\nu]}$ for $k \in \text{Supp}(\mathbf{x}) \cup S$. As $\mathbf{c}$ is orthogonal to each of these vectors by construction, we see that it must also be orthogonal to the last ν rows of M.

Thus, $\mathbf{c}$ is orthogonal to each row of M, and thus is in its nullspace. $\qquad\square$

The algorithm chooses *some* $\mathbf{c}$ that is in the nullspace of the $(|S|+\nu) \times (|S|+\nu+1)$ leading minor of A. However, as the $(|S| + \nu) \times (|S| + \nu)$ leading principal minor of A is invertible, it follows that the $(|S| + \nu) \times (|S| + \nu + 1)$ leading minor of A has a nullspace of dimension 1. Thus, the $\mathbf{c}$ chosen by the algorithm must be a (non-zero) multiple of the coefficient vector of $\prod_{k \in \text{Supp}(\mathbf{x}) \cup S}(x - g_k)$. It follows that the set T is equal to $\text{Supp}(x) \cup S$.

Thus, Equation 3 gives a linear system for $\mathbf{y}$ with $\leq 2s$ variables, and $2s$ equations, where $\mathbf{x}$ (restricted to $\text{Supp}(x) \cup S$) is a solution. The system is full-rank, so $\mathbf{x}$ is the only solution. Further, $\mathbf{x}$ can be recovered via Gaussian Elimination, and this is exactly what Algorithm 1 does. Thus, correctness is also established in this case.

Algorithm 1 runs in $\mathcal{O}(s^3 + sn)$ operations: Constructing the matrix A takes $\mathcal{O}(s^2)$ operations, as that is the size of the matrix and each entry can be computed in $\mathcal{O}(1)$ operations (the g_k^i are computed with i increasing). Converting A to reduced-row echelon form takes $\mathcal{O}(s^3)$ operations. Determining the number r in Step 8 also takes $\mathcal{O}(s)$ operations, as $r = \max\{i | A_{i,i} \neq 0\}$. Determining the vector $\mathbf{c}$ takes $\mathcal{O}(s)$ because the $r \times (r+1)$ minor is row-reduced echelon form. That is, for $1 \leq i \leq r$, $c_i = -A_{i,r+1}$ and $c_{r+1} = 1$. Constructing p and T takes $\mathcal{O}(sn)$ time, as we just test if $p(g_k) = 0$ for each k, and p is of degree $\mathcal{O}(s)$. D is a Vandermonde matrix with at most $\mathcal{O}(s^2)$ entries, and so constructing D takes $\mathcal{O}(s^2)$ steps. Solving for $\mathbf{z}$ takes $\mathcal{O}(s^3)$ steps, and determining the final $\mathbf{x}$ takes $\mathcal{O}(n)$ steps. $\qquad\square$

This theorem provides us with an s-advice-sparse-recovery set, using $2s$ measurements. We will now leverage this in the next subsection to get a full algorithm for low-rank-recovery.

7.2   Low Rank Recovery

In this subsection we describe how the problem of (exact, non-adaptive) r-low-rank-recovery deterministically reduces to the problem of (exact, non-adaptive) r-advice-sparse-recovery. We will first define a normal form for a matrix which we call $(< k)$-*upper-echelon form*, which (recalling the notation of Section 2) is roughly defined as saying that a matrix M has $M^{(<k)}$ in reduced row-echelon form. We then show that for any matrix M in this form, the diagonal $M^{(k)}$ is sparse. Thus, using sparse-recovery we can then recover this diagonal. This process is then continued by using row-reduction to put M in $(\le k)$-upper-echelon form, and then recovering $M^{(k+1)}$ and so on.

The above process uses the sparse-recovery oracle in an adaptive way. The algorithm we detail below will actually use the sparse-recovery oracle non-adaptively. The measurements made to the matrix M will be the sparse-recovery oracle applied to each k-diagonal. While these diagonals are not themselves sparse, we show that the row-reduction of M (that makes M into upper-echelon form) acts such that we can simulate the adaptive measurements from the non-adaptive measurements by computing the suitable corrections.

We now begin by describing some structural properties of matrices, which we will apply to understand upper-echelon form.

Definition 7.11. *Let M be an $n \times m$ matrix. The entry (i,j) is a **leading non-zero entry**, if $M_{i,j} \ne 0$ and $M_{i,j'} = 0$ for $j' < j$.*

Denote $\mathrm{LNE}(M)$ to be the set of all such leading non-zero entries. If S is a subset of entries in M, denote $\mathrm{LNE}(S) \stackrel{\mathrm{def}}{=} \mathrm{LNE}(M) \cap S$.

Denote $\mathrm{LNE}_R(S)$ to be set containing the rows of the coordinates in $\mathrm{LNE}(S)$, and denote $\mathrm{LNE}_C(S)$ to be the multi-set containing the columns of the coordinates in $\mathrm{LNE}(S)$.

It is clear that each row can have at most one leading non-zero entry, and possibly none. A column could be associated with several leading non-zero entries.

Definition 7.12. *An $n \times m$ matrix M is in $(< k)$-**upper-echelon form** if, for each $(i,j) \in \mathrm{LNE}(M^{(<k)})$, $M_{i',j} = 0$ for all $i < i' < k - j$.*

Note that a matrix is $(< k)$-upper-echelon if it is $(< k')$-upper-echelon and $k' \ge k$, and that every matrix is vacuously in (≤ 0)-upper-echelon form.

We now recall the following standard linear-algebraic fact about triangular systems, phrased in the language of leading non-zero entries.

Lemma 7.13. *Let M be an $n \times m$ matrix with all non-zero rows, such that $\mathrm{LNE}_C(M)$ has no repetitions. Then the rows of M are linearly independent.*

Proof. Denote the column of the leading non-zero entry of row i by j_i. Each row must have such a value as each row is non-zero. As linear independence is invariant under permutation, we assume without loss of generality that the rows are ordered such that the j_i are strictly increasing with i. This is possible as the j_i are assumed to be distinct. Write these rows as vectors $\mathbf{v}^{(i)}$. Now consider any non-trivial linear combination $\sum_i c_i \mathbf{v}^{(i)}$. Pick i_0 to be the least number such that $c_{i_0} \ne 0$. As the j_i are strictly increasing, it follows that the j_{i_0}-th entry of $\mathbf{v}^{(i)}$ is zero for $i > i_0$. Thus, we now expand out the i_0-th index of the above summation

$$\left(\sum_i c_i \mathbf{v}^{(i)}\right)_{j_{i_0}} = \sum_{i < i_0} c_i \cdot \mathbf{v}^{(i)}_{j_{i_0}} + c_{i_0} \mathbf{v}^{(i_0)}_{j_{i_0}} + \sum_{i_0 < i} c_i \cdot \mathbf{v}^{(i)}_{j_{i_0}} = \sum_{i < i_0} 0 \cdot \mathbf{v}^{(i)}_{j_{i_0}} + c_{i_0} \mathbf{v}^{(i_0)}_{j_{i_0}} + \sum_{i_0 < i} c_i \cdot 0 = c_{i_0} \mathbf{v}^{(i_0)}_{j_{i_0}} \ne 0$$

Thus we see that this linear combination is non-zero, and as this was any non-trivial linear combination it follows these rows are linearly independent. □

We now show that matrices in upper-echelon form cannot have many leading non-zero entries.

Lemma 7.14. *Let M be an $n \times m$ matrix of rank $\leq r$. If M is $(< k)$-upper-echelon, then $|\operatorname{LNE}(M^{(<k)})| \leq r$. Further, $\operatorname{LNE}_C(M^{(<k)})$ has no repetitions.*

Proof. Given $(i, j) \in \operatorname{LNE}(M^{(<k)})$, $(< k)$-upper-echelon form implies that $M_{i',j} = 0$ for any i' with $i < i' < k - j$. It follows that given two distinct entries $(i, j), (i', j) \in M^{(<k)}$ at most one can be a leading non-zero entry. Thus we see that $\operatorname{LNE}_C(M^{(<k)})$ has no repetitions.

Lemma 7.13 then implies that the rows in $\operatorname{LNE}_R(M^{(<k)})$ are linearly independent. Thus, $|\operatorname{LNE}(M^{(<k)})| \leq \operatorname{rank}(M) \leq r$. $\qquad\square$

The next lemma is the key insight of the algorithm. It shows that, for any matrix in $(< k)$-upper-echelon form, the k-diagonal must be sparse. Further, the sparseness is bounded by twice the rank of the matrix (the lemma presents a more refined statement).

Lemma 7.15. *Let M be an $n \times m$ matrix with rank $\leq r$, such that M is in $(< k)$-upper-echelon form with $0 \leq k \leq n + m - 2$. Let $s \overset{\mathrm{def}}{=} |\operatorname{LNE}(M^{(<k)})|$, $I \overset{\mathrm{def}}{=} \operatorname{LNE}_R(M^{(<k)})$, $J \overset{\mathrm{def}}{=} \operatorname{LNE}_C(M^{(<k)})$.*

Then $M^{(k)}$ has $\leq r - s$ non-zero entries with columns outside $S \overset{\mathrm{def}}{=} (k - I) \cup J$, and thus $M^{(k)}$ is $(r + s)$-sparse.

Proof. Note that by Lemma 7.14 we have that $s \leq r$, so that $r - s \geq 0$ and $r + s \leq 2r$.

Let I' be the rows that contain non-zero entries in $M^{(k)}$, whose columns lie outside S. We will show that the rows in $I \cup I'$ are linearly independent. This will complete the claim as $|I'| \leq \operatorname{rank}(M) - |I| \leq r - s$, and observing that $|S| \leq 2s$.

Now consider the columns of the leading non-zero entries of the rows in I'. Any row $i \in I$ intersects $M^{(k)}$ at column $k - i \in S$. This means that row i cannot contain a non-zero entry in $M^{(k)}$ with column outside of S, so I and I' are disjoint.

Any row i with a non-zero entry in $M^{(<k)}$ must have a leading non-zero entry in $M^{(<k)}$, and thus any such i is contained in I. Thus, as I and I' are disjoint, it follows that any row $i' \in I'$ only has zero entries within $M^{(<k)}$. As such a row i' has a non-zero entry on $M^{(k)}$, it follows that the leading non-zero entry of a row $i' \in I'$ is $(i', k - i')$. This implies that the columns of the leading non-zero entries of the rows in I' are distinct (and outside of S by construction).

The rows in I have leading non-zero entries in $J \subseteq S$ and by Lemma 7.14, J has no repetitions. Thus, it follows that the rows $I \cup I'$ all have distinct columns for their leading non-zero entries, which, by Lemma 7.13, implies that these rows are linearly independent. Invoking the rank bound, as mentioned above, completes the proof. $\qquad\square$

This lemma motivates the following idea for low-rank reconstruction. Iteratively, convert (using row-reduction) the matrix into $(< k)$-upper-echelon form and then reconstruct, using any sparse-recovery method, the k-th diagonal. This is exactly the algorithm we will present. However, to establish correctness, we need to first understand how to convert a matrix into $(< k)$-upper-echelon form, even in situations when $M^{(\geq k)}$ is unknown.

To do this, we will use row-reduction, as implemented by left-multiplication by lower-triangular matrices. The following lemma shows that such multiplication can be computed on the partial matrices $M^{(<k)}$.

Lemma 7.16. *Let M be an $n \times m$ matrix, and L be an $n \times n$ lower-triangular matrix. Then $(LM)^{(<k)}$ is computable in $\mathcal{O}(\min(n, k) \min(m, k) k)$ arithmetic operations from L and $M^{(<k)}$.*

Proof. An entry $(LM)_{i,j}$, for $i + j < k$, is equal to $\sum_{l=0}^{n} L_{i,l} M_{l,j}$, which equals $\sum_{l=1}^{i} L_{i,l} M_{l,j}$ as L is lower-triangular. Further, this sum is computable from L and the $(< k)$-diagonals of M as $l + j \le i + j < k$. The time bound is the obvious bound on computing each of $\mathcal{O}(\min(n, k) \min(m, k))$ sums of $\le k$ terms. $\qquad \square$

We now establish a useful property on composing left-multiplication of special types of lower-triangular matrices.

Lemma 7.17. *Let L, L' be $n \times n$ invertible, lower-triangular matrices, with all 1's along the main diagonal. Then LL' is an invertible, lower-triangular matrix, with all 1's along the main diagonal,*
Further, if both $L - I_n$ and $L' - I_n$ only have non-zero entries in a subset J of the columns, then $LL' - I_n$ also has this property.

Proof. That facts that LL' is an invertible, lower-triangular matrix and has all 1's along the main diagonal, are each straightforward.

We now prove the desired property of $LL' - I_n$. Consider some entry (i, j) in LL', with $j \notin J$ and $i > j$. It is then that

$$
(LL')_{i,j} = \sum_{k \in [\![n]\!]} L_{i,k} L'_{k,j} = \sum_{i \ge k \ge j} L_{i,k} L'_{k,j} = L_{i,i} L'_{i,j} + \sum_{i > k > j} L_{i,k} L'_{k,j} + L_{i,j} L'_{j,j}
$$
$$
= 1 \cdot L'_{i,j} + \sum_{i > k > j} L_{i,k} L'_{k,j} + L_{i,j} \cdot 1
$$

Observe that as $i > j$ and $j \notin J$, $L'_{i,j} = L'_{k,j} = L_{i,j} = 0$ (for any $k > j$). Thus, the above sum is zero. Hence, the desired entries (i, j) with $i > j$ and $j \notin J$ are zero, proving the claim. $\qquad \square$

We now use these lemmas to analyze Algorithm 2, which gives a way to transform a matrix in $(< k)$-upper-echelon into one which is $(\le k)$-upper-echelon, and does so efficiently.

Algorithm 2 Transform a $(< k)$-upper-echelon matrix into $(\le k)$-upper-echelon form

---

1: **procedure** MakeUpperEchelon(M,n,m,k)
2: $\quad$ $L \leftarrow I_n$
3: $\quad$ **for all** $(i, j) \in \text{LNE}(M^{(<k)})$ **do**
4: $\quad\quad$ $L \leftarrow (I_n - \frac{M_{k-j,j}}{M_{i,j}} E_{k-j,i}) \cdot L$ $\qquad \triangleright M_{i,j} \ne 0$ as (i, j) is leading non-zero entry in row i
5: $\quad$ **end for**
6: $\quad$ **return** L
7: **end procedure**

Claim 7.18. *Let M be an $n \times m$ matrix of rank $\le r$, such that M is in $(< k)$-upper-echelon form, for $0 \le k \le n + m - 2$. Then the procedure MakeUpperEchelon(M, n, m, k) (Algorithm 2) runs in $\mathcal{O}(rn)$ time and returns an invertible $n \times n$ lower-triangular matrix L computed only from $M^{(\le k)}$, such that LM is $(\le k)$-upper-echelon and $(LM)^{(<k)} = M^{(<k)}$.*
Also, L is the product of $\le r$ elementary matrices and each main diagonal entry is equal to 1.
Further, $L - I_n$ only has non-zero entries with columns in $\text{LNE}_R(M^{(<k)})$.

Proof. $\underline{(LM)^{(<k)} = M^{(<k)}}$: We argue that the identity $(LM)^{(<k)} = M^{(<k)}$ is invariant. As $L = I_n$ initially, the identity holds at the beginning of the algorithm. We now proceed by induction.

In each run of Line 4, we add a multiple of row i to row $k - j$ in LM, where $(i, j) \in \text{LNE}(M^{(<k)})$ and thus $i + j < k$. Thus, row i in M has the first $j - 1$ entries being zero. By induction on the

identity $(LM)^{(<k)} = M^{(<k)}$, the first $j-1$ entries in row i of LM are also zero when Line 4 is run. It follows that the only action of this update to $(LM)^{(\leq k)}$ is to set $(LM)_{k-j,j} = 0$. Thus, $(LM)^{(<k)}$ is unchanged, so $(LM)^{(<k)} = M^{(<k)}$ still holds.

$\underline{LM \text{ has } (\leq k)\text{-upper-echelon form}}$: As $(LM)^{(<k)} = M^{(<k)}$ throughout the algorithm, and M is in $(< k)$-upper-echelon form, it follows that LM is in $(< k)$-upper-echelon form at termination. To show LM is in $(\leq k)$-upper-echelon form upon termination, it suffices to show that $(LM)_{k-j,j} = 0$ for all $j \in \mathrm{LNE}_C(M^{(<k)})$. As running Line 4 has exactly this effect (and these updates are disjoint and idempotent, thus do not conflict), and this line is run for all $(i,j) \in \mathrm{LNE}(M^{(<k)})$, it follows that LM is in $(\leq k)$-upper-echelon form on termination.

$\underline{L \text{ computable from the } (\leq k)\text{-diagonals of } M}$: This is straightforward, as each query to M is within the $(\leq k)$-diagonals.

$\underline{L \text{ is the product of } \leq r \text{ elementary matrices}}$: Each update to L by Line 4 left-multiplies L by an elementary matrix. By Lemma 7.14, $|\mathrm{LNE}(M^{(<k)})| \leq r$, so the loop of the algorithm is run at most r times.

$\underline{\text{Structure of } L}$: By construction, L is the product of matrices of the form $I_n + cE_{k-j,i}$, where $i + j < k$ and $(i,j) \in \mathrm{LNE}(M^{(<k)})$. Regardless of the value of c, such a matrix is invertible, lower-triangular, with main diagonal entries all 1, and all non-zero entries of $(I_n + cE_{k-j,i}) - I_n$ have columns in $\mathrm{LNE}_R(M^{(<k)})$. By Lemma 7.17 it follows that L also has these properties.

$\underline{\text{Complexity}}$: Left-multiplication by an elementary matrix can be done in $\mathcal{O}(n)$ steps, and by the above analysis, there are $\leq r$ such multiplications. Further, by storing the leading non-zero entries in each row, the pairs (i,j) can be determined in $\mathcal{O}(n)$ time. Thus the time is $\mathcal{O}(rn)$ overall. $\qquad \square$

We now present the low-rank recovery algorithm, and its analysis.

Algorithm 3 Reconstruct a matrix from inner-products $\{\langle M, R \rangle\}_{R \in \mathcal{R}_k, 0 \leq k \leq n+m-2}$

1: **procedure** LOWRANKRECOVERY$(n,m,\{\langle M, R \rangle\}_{R \in \mathcal{R}_k, 0 \leq k \leq n+m-2})$
2: $\quad L \leftarrow I_n$
3: $\quad N \leftarrow 0^{n \times m}$
4: $\quad P \leftarrow 0^{n \times m}$
5: $\quad$ **for** $0 \leq k \leq n+m-2$ **do**
6: $\quad\quad A \leftarrow 0^{n \times m}$
7: $\quad\quad A^{(k)} \leftarrow ((L - I_n)N)^{(k)}$
8: $\quad\quad S \leftarrow \big(k - \mathrm{LNE}_R((P^{(<k)}))\big) \cup \mathrm{LNE}_C(P^{(<k)})$
9: $\quad\quad P^{(k)} \leftarrow \mathrm{SR}_k(\{\langle M, R \rangle + \langle A, R \rangle\}_{R \in \mathcal{R}_k}, S)$
10: $\quad\quad N^{(k)} \leftarrow P^{(k)} - A^{(k)}$
11: $\quad\quad L_k \leftarrow \mathrm{MAKEUPPERECHELON}(P, n, m, k)$
12: $\quad\quad P^{(k)} \leftarrow (L_k P)^{(k)}$ $\qquad\qquad\qquad\qquad\qquad\qquad$ ▷ Update $\mathrm{LNE}(P^{(\leq k)})$
13: $\quad\quad L \leftarrow L_k L$
14: $\quad$ **end for**
15: **end procedure**

Theorem 7.19. *Let $m \geq n \geq r \geq 1$. For $0 \leq k \leq n+m-2$, let $\mathcal{R}_k$ be sets of $n \times m$ matrices such that*

1. *For $k' \neq k$, $R^{(k')} = 0$ for $R \in \mathcal{R}_k$*

2. *$\{R^{(k)}\}_{R \in \mathcal{R}_k}$ forms a $\min(r, k+1, (n+m)-(k+1))$-advice-sparse-recovery set.*

Then $\mathcal{R} = \bigcup_k \mathcal{R}_k$ *is an r-low-rank-recovery set.*

If, for each k, the set $\{R^{(k)}\}_{R \in \mathcal{R}_k}$ *has an* $\min(r, k+1, (n+m) - (k+1))$-advice-sparse-recovery *algorithm* SR_k *running in time* t_k, *then Algorithm 3 performs r-low-rank-recovery for* $\mathcal{R}$ *in time* $\mathcal{O}\left(rnm + \sum_{k=2}^{n+m} (t_k + n|\mathcal{R}_k|)\right)$.

Proof. We will first show that $\mathcal{R}$ is an r-low-rank-recovery set by showing that Algorithm 3 performs recovery, assuming oracle access to r-advice-sparse-recovery oracles SR_k. We will then analyze the run-time.

Claim 7.20. *The following invariants hold at Line 14, at the end of the loop.*

1. $N^{(\leq k)} = M^{(\leq k)}$

2. $P^{(\leq k)} = (LM)^{(\leq k)}$

3. P *is in* $(\leq k)$-upper-echelon form

4. L *is lower-triangular, invertible , main diagonal is all 1's, and* $L - I_n$ *only has non-zero entries with columns in* $\mathrm{LNE}_R(P^{(<k)})$

Proof. The proof will be by induction.

<u>$k = 0$:</u> The loop begins with $L = I_n$, $N = 0_n$, $P = 0_n$. It follows that $A = 0_n$ in this run of the loop, and that $S = \emptyset$. Thus, $P^{(0)}$ is set to $\mathrm{SR}_0(\{\langle M, R \rangle\}_{R \in \mathcal{R}_0}, \emptyset)$. As $r \geq 1$, we get that $\mathcal{R}_0^{(0)}$ is a 1-advice-sparse-recovery set and as $M^{(0)}$ has at most 1 element, it follows that SR_0 recovers it correctly and thus $P^{(0)} = M^{(0)}$ after Line 9. As $A = 0_n$ it follows that $N^{(\leq 0)} = M^{(\leq 0)}$ also, satisfying Invariant 1.

Now observe that the procedure MAKEUPPERECHELON, when run on $k = 0$, will always return I_n. Thus, L_k, and L, are both I_n at the end of the loop, satisfying Invariant 4. Invariant 3 is vacuously true as any matrix is in 1-upper-echelon form. Finally, using that $L = L_k = I_n$, we see that P is unchanged after Line 9 and so $P^{(\leq 0)} = (LM)^{(\leq 0)}$, satisfying Invariant 2.

<u>$k > 0$:</u> Using that the invariants held at $k-1$, we now establish them at k. As $P^{(<k)} = (LM)^{(<k)}$ and P is in $(< k)$-upper-echelon form, it follows that LM is in $(< k)$-upper-echelon form. By Lemma 7.15, it follows $(LM)^{(k)}$ has at most $r - s/2$ non-zero entries with columns outside of $S = (k - \mathrm{LNE}_R((LM)^{(<k)})) \cup \mathrm{LNE}_C((LM)^{(<k)})$, where $s = |\mathrm{LNE}((LM)^{(<k)})|$ and $|S| \leq 2s$. However, using again that $P^{(<k)} = (LM)^{(<k)}$ it follows that $(LM)^{(k)}$ has at most $r - |S|/2$ non-zero entries with columns outside of S, where S is as constructed in Line 8. As $(LM)^{(k)}$ has $\min(k+1, (n+m) - (k+1), n)$ non-zero entries total, and $\mathcal{R}_k$ is an $\min(r, k+1, (n+m) - (k+1))$-advice-sparse-recovery set, it follows (as $r \leq n$) that $\mathrm{SR}_k(\{\langle LM, R \rangle\}_{R \in \mathcal{R}_k}, S)$ successfully recovers $(LM)^{(k)}$. That is, if $r \neq \min(r, k+1, (n+m) - (k+1))$ then we have enough measurements to fully recover $(LM)^{(k)}$ regardless of its sparsity and the value of S(and the oracle will perform this recovery), and if $r = \min(r, k+1, (n+m) - (k+1))$ then we use the advice-sparse-recovery oracle.

We now use the following claim to show how the $\{\langle LM, R \rangle\}$ can be computed.

Claim 7.21. *At the beginning of the loop in Line 5,* $(LM)^{(k)} = M^{(k)} + ((L - I_n)N)^{(k)}$

Proof. As $LM = M + (L - I_n)M$, it is enough to show that $((L - I_n)M)^{(k)} = ((L - I_n)N)^{(k)}$.

By induction on the above invariants, L is lower-triangular, with all 1's along the main diagonal, and $N^{(<k)} = M^{(<k)}$. Thus, $(L - I_n)_{i,\ell} = 0$ for $i \leq \ell$, and $M_{l,j} = N_{\ell,j}$ for $\ell < k - j$. For any $j \leq k$,

$$((L - I_n)M)_{k-j,j} = \sum_{\ell \in [\![n]\!]} (L - I_n)_{k-j,\ell} M_{\ell,j} = \sum_{\ell < k-j} (L - I_n)_{k-j,\ell} M_{\ell,j} = \sum_{\ell < k-j} (L - I_n)_{k-j,\ell} N_{\ell,j}$$
$$= \sum_{\ell \in [\![n]\!]} (L - I_n)_{k-j,\ell} N_{\ell,j} = ((L - I_n)N)_{k-j,j}$$

Thus $((L - I_n)M)^{(k)} = ((L - I_n)N)^{(k)}$, giving the claim. $\qquad\square$

The above claim shows that at Line 9 we have that $\langle LM, R \rangle = \langle M, R \rangle + \langle A, R \rangle$, for all $R \in \mathcal{R}_k$, using that $R^{(k')} = 0$ for $k' \neq k$. This shows that Line 9 correctly implements advice-sparse-recovery of $(LM)^{(k)}$, and thus sets $P^{(k)}$ to this value. It follows that at the end of this line that $P^{(\leq k)} = (LM)^{(\leq k)}$.

$\underline{\text{Invariant 1}}$: Using the identity proved in the above claim, and the just proven fact that $P^{(\leq k)} = (LM)^{(\leq k)}$ at the end of Line 9, it follows that at the end of Line 10 that $N^{(k)} = M^{(k)}$, and thus $N^{(\leq k)} = M^{(\leq k)}$. As N is not changed further, this establishes Invariant 1.

$\underline{\text{Invariant 3}}$: We now examine Lines 11–13. As P has only changed in its k-diagonal, it is still in $(< k)$-upper-echelon form. Thus, Line 11 returns L_k such that $L_k P$ is in $(\leq k)$-upper-echelon form, by Claim 7.18. Further $(L_k P)^{(\leq k)}$ only differs from $P^{(\leq k)}$ along the k-diagonal, so it follows that after the update in Line 12 that P is in $(\leq k)$-upper-echelon form. As P is not further modified, this establishes Invariant 3.

$\underline{\text{Invariant 2}}$: Further, as we take $L \leftarrow L_k L$ in Line 13 and previously had that $P^{(\leq k)} = (LM)^{(\leq k)}$, it follows that at the end of Line 13 we have that $P^{(\leq k)} = (LM)^{(\leq k)}$ still, as both P and LM have been multiplied by L_k. This establishes Invariant 2.

$\underline{\text{Invariant 4}}$: In Line 11, Claim 7.18 shows that L_k is a lower-triangular and invertible matrix, with main diagonal entries all 1's, and $L_k - I_n$ only has non-zero entries in columns $\text{LNE}_R(P^{(<k)})$. As $P^{(<k)}$ is not modified further, this remains true at the end of the loop at Line 14. By induction, at Line 5 we have that L is lower-triangular, invertible, with main diagonal entries all 1's, and $L - I_n$ only has non-zero entries in columns $\text{LNE}_R(P^{(<(k-1))})$. As $P^{(<(k-1))}$ remains unchanged throughout this iteration of the loop, this is also true at the beginning of Line 13. By Lemma 7.17, it follows that after Line 13 L still has the properties of being lower-triangular, invertible, main diagonal entries being 1's, and $L - I_n$ only has non-zero entries in $\text{LNE}_R(P^{(<k)})$. This establishes Invariant 4

Thus, each of the invariants are established for this value of k given that they hold for $k - 1$, so the invariants hold for all k by induction. $\qquad\square$

The above claim shows that at the end of the algorithm, $N^{(\leq k)} = M^{(\leq k)}$ for $k = n + m - 2$. But this implies $N = M$, and thus M is reconstructed successfully.

$\underline{\text{Run-time Analysis}}$: We now bound the run-time of Algorithm 3. The steps outside the for-loop take $\mathcal{O}(nm)$, so it suffices to bound each step of the loop. We will show that each step of the loop takes $\mathcal{O}(rn + t_k + n|\mathcal{R}_k|)$ steps. As there are $n + m$ such iterations of the loop, the quoted bound follows.

We begin by noting that the algorithm will not recompute $\text{LNE}(P^{(<k)})$ at each stage. Instead, this will be maintained throughout the algorithm. As each row of P can have at most one leading non-zero entry, this is easily stored and indexed. Further, as $P^{(<k)} = (LM)^{(<k)}$ and the rank bound on M shows, via Lemma 7.14, that $|\text{LNE}((LM)^{(<k)})| \leq r$, it follows that if the set $\text{LNE}(P^{(<k)})$ is maintained as a linked list, that traversing it entirely takes $\mathcal{O}(r)$ time.

Note that we do not need to modify $\text{LNE}(P^{(<k)})$ when running MakeUpperEchelon, and can defer modification to after Line 12. At that point $P^{(\leq k)}$ has been determined, and can be used to compute $\text{LNE}(P^{(<(k+1))}) = \text{LNE}(P^{(\leq k)})$ in $\mathcal{O}(n)$ time. Thus, $\text{LNE}(P^{(<k)})$ can be maintained within the quoted time bounds, and accessed as a $\mathcal{O}(r)$ sized linked list.

We now analyze the lines of the loop. As written, Line 6 takes $\Theta(nm)$ time, which is above the quoted run-time bounds. However, one can observe that A is only ever accessed at the values $A^{(k)}$, when noting that $R \in \mathcal{R}_k$ is only non-zero on its k-diagonal. Thus, Line 6 is actually superfluous and can be omitted.

Line 7 takes $\mathcal{O}(rn)$ steps. For, the above invariants show that $L - I_n$ only has non-zero entries in the columns $\text{LNE}_R(P^{(<k)})$, and as discussed above this set has at most r elements. Thus, each of the $\leq n$) elements of $A^{(k)}$ is the sum of $\leq r$ elements of N. Thus $A^{(k)}$ can be computed in $\mathcal{O}(rn)$ steps.

Line 8 takes $\mathcal{O}(r)$ steps, as $\text{LNE}_R(P^{(<k)})$ is pre-computed.

Line 9 takes $\mathcal{O}(t_k + n|\mathcal{R}_k|)$ steps. For, each inner product $\langle A, R \rangle$ takes $\mathcal{O}(n)$ steps (as each matrix is only non-zero on the k-diagonal, which has at most n entries), and there are $|\mathcal{R}_k|$ such inner-products. Running SR_k takes t_k steps, by definition.

Line 10 takes $\mathcal{O}(n)$ steps, as the k-diagonal has at most this many entries.

Line 11 takes $\mathcal{O}(rn)$ steps by Claim 7.18.

Lines 12 takes $\mathcal{O}(rn)$ steps, for as used above, $L_k - I_n$ has only non-zero entries with columns in $\text{LNE}_R(P^{(<k)})$, so each entry in $(L_k P)^{(k)}$ is the sum of at most $r + 1$ products of entries in L_k and P, and these products are determined by $\text{LNE}_R(P^{(<k)})$. As there are at most n such entries, the bound follows.

Line 13 takes $\mathcal{O}(rn)$ steps. This is because L_k, by Claim 7.18, is the product of $\leq r$ elementary matrices, and left-multiplication by an elementary matrix takes $\mathcal{O}(n)$ steps. As MakeUpperEchelon computes L_k as a product of elementary matrices, the computation of $L_k L$ can also use this decomposition and thus is compute in $\mathcal{O}(rn)$ steps.

Thus, the entire loop runs in $\mathcal{O}(rn + t_k + n|\mathcal{R}_k|)$ steps, and there are at most $n + m$ iterations of the loop, giving the bound. □

We now apply this reduction to our hitting set $\mathcal{D}'_{2r,n,m}$, which embeds the sparse-recovery measurements corresponding to the dual Reed-Solomon code.

Corollary 7.22. *Let $1 \leq r \leq n/2$, $m \geq n \geq 1$. Then $\mathcal{D}'_{2r,n,m}$ (from Construction 5.7) has*

1. $|\mathcal{D}'_{2r,n,m}| = 2(n + m - 2r)r$

2. *Each matrix in $\mathcal{D}'_{2r,n,m}$ is n-sparse.*

3. $\mathcal{D}'_{2r,n,m}$ *is a r-low-rank-recovery set*

4. *Algorithm 3, combined with Algorithm 1, performs low-rank-recovery for $\mathcal{D}'_{2r,n,m}$ in time $\mathcal{O}(rnm + (n + m)r^3)$*

Proof. (1): This is by construction.

(2): Each matrix in $\mathcal{D}'_{2r,n,m}$ has its support contained in some k-diagonal, and each k-diagonal has at most n elements.

(3): We will first show that the measurements that $\mathcal{D}'_{2r,n,m}$ performs on each k-diagonal comprise a $\min(2r, k + 1, (n + m) - (k + 1))$-advice-sparse-recovery set.

First consider the case when $k + 1 < 2r \leq n$. Then $\min(2r, k + 1, (n + m) - (k + 1)) = k + 1$, and $\mathcal{D}'_{2r,n,m}$ places $k + 1$ constraints on this k-diagonal $M^{(k)}$, which has $k + 1$ entries. The constraint

44

matrix V is of size $(k+1) \times (k+1)$ with $V_{\ell,j} = g^{\ell j}$. As g has order $\geq n$, the elements $1, g, \ldots, g^k$ are distinct. So these constraints form an invertible Vandermonde system and so $M^{(k)}$ (regardless of the rank of M) can be completely recovered from these measurements. In particular, V forms a $(k+1)$-advice-sparse-recovery set. As the Vandermonde system can be inverted in $\mathcal{O}(k^3) = \mathcal{O}(r^3)$ time, we see that $(k+1)$-advice-sparse-recovery can be performed in this time.

Similarly, now consider the case when $(n+m) - (k+1) < 2r \leq n$ (so it follows that $m \leq k$). Then $\min(2r, k+1, (n+m) - (k+1)) = (n+m) - (k+1)$, and $\mathcal{D}'_{2r,n,m}$ places $(n+m) - (k+1)$ constraints on this k-diagonal $M^{(k)}$, which has $(n+m) - (k+1)$ entries. The constraint matrix V is of size $((n+m) - (k+1)) \times ((n+m) - (k+1))$ with $V_{\ell,j} = g^{\ell(k-(m-1)+j)}$. As g has order $\geq n$, the elements $g^{k-(m-1)}, g^{k-(m-1)+1}, \ldots, g^{n-1}$ are distinct. So these constraints form an invertible Vandermonde system and so $M^{(k)}$ (regardless of the rank of M) can be completely recovered from these measurements. In particular, V forms a $((n+m) - (k+1))$-advice-sparse-recovery set. As the Vandermonde system can be inverted in $\mathcal{O}(((n+m) - (k+1))^3) = \mathcal{O}(r^3)$ time, we see that $((n+m) - (k+1))$-advice-sparse-recovery can be performed in this time.

Now consider the general case when $2r \leq k+1, (n+m) - (k+1)$. Then $\min(2r, k+1, (n+m) - (k+1)) = 2r$, and $\mathcal{D}'_{2r,n,m}$ places $2r$ constraints on this k-diagonal $M^{(k)}$, which has $\min(k+1, n, (n+m) - (k+1))$ entries. The constraint matrix V is of size $2r \times \min(k+1, n, (n+m) - (k+1))$ with $V_{\ell,j} = g^{\ell(\max(0,k-(m-1))+j)}$. As g has order $\geq n$, the elements

$$g^{\max(0,k-(m-1))}, g^{\max(0,k-(m-1))+1}, \ldots, g^{\max(0,k-(m-1))+\min(k+1,n,(n+m)-(k+1))-1}$$

are distinct. Thus, it follows from Theorem 7.7 that V is a r-advice-sparse-recovery set, and that recovery can be done in $\mathcal{O}(r^3 + n)$ steps.

Thus, by Theorem 7.19, it follows that $\mathcal{D}'_{2r,n,m}$ is a r-low-rank-recovery set.

(4): By the analysis done for (3), we see that Theorem 7.19 shows that Algorithm 3 (along with the r-advice-sparse-recovery performed by Algorithm 1) yields a $\mathcal{O}(rnm + (n+m)r^3)$-time recovery algorithm for $\mathcal{D}'_{2r,n,m}$. $\qquad \square$

Remark 7.23. We briefly note that for $r > n/2$ we have that $|\mathcal{D}'_{2r,n,m}| \geq nm$ (one cannot use the formula "$|\mathcal{D}'_{2r,n,m}| = 2(n+m-2r)r$" here, but the bound $|\mathcal{D}'_{2r,n,m}| \leq |\mathcal{D}_{2r,n,m}| = 2(n+m-1)r$ is still valid). Thus, for $r > n/2$ there is no gain from using $\mathcal{D}'_{2r,n,m}$ over the obvious nm low-rank-recovery set that queries each entry in the matrix.

Remark 7.24. One can also use Algorithm 3 to reprove Theorem 5.8, that is, to reprove that $\mathcal{D}_{r,n,m}$ is a hitting set (note that we use r and not $2r$ here). To do so, note that Lemma 7.15 shows that for a rank $\leq r$ matrix M, if $M^{(<k)} = 0$ then $M^{(k)}$ is r-sparse.

Thus, if $\langle M, \mathcal{D}_{r,n,m} \rangle = \mathbf{0}$ then this implies that for each k, $\langle M^{(k)}, \mathcal{R}_k \rangle = \mathbf{0}$, where $\mathcal{R}_k$ is the r-sparse-recovery set formed from the dual Reed-Solomon code. So if $M^{(k)}$ is r-sparse then by the properties of $\mathcal{R}_k$ it must be that $M^{(k)} = \mathbf{0}$.

Combining the two observations above, we see that $M^{(<k)} = 0 \implies M^{(k)} = \mathbf{0}$, and thus $M^{(<k)} = 0 \implies M^{(\leq k)} = \mathbf{0}$. Inducting on k shows that $M = 0_{n \times m}$. Thus, if $M \neq 0$ and M is rank $\leq r$ then $\langle M, \mathcal{D}_{r,n,m} \rangle \neq \mathbf{0}$, showing that $\mathcal{D}_{r,n,m}$ is a hitting set.

Given that $\mathcal{D}'_{2r,n,m}$ admits efficient low-rank-recovery, we can recall the above results that show that these measurements are equivalent to the $\mathcal{B}'_{2r,n,m}$ measurements. Thus, we also get that this second set admits efficient low-rank-recovery.

Corollary 7.25. *Let* $1 \leq r \leq n/2$, $m \geq n \geq 1$. *Then* $\mathcal{B}'_{2r,n,m}$ *(from Construction 5.9) has*

1. $|\mathcal{B}'_{2r,n,m}| = 2(n+m-2r)r$

2. Each matrix in $\mathcal{B}'_{2r,n,m}$ is rank 1.

3. $\mathcal{B}'_{2r,n,m}$ is a r-low-rank-recovery set

4. Algorithm 3, combined with Algorithm 1, performs low-rank-recovery for $\mathcal{B}'_{2r,n,m}$ in time $\mathcal{O}(rm^2 + mr^3)$

Proof. (1): This is by construction.

(2): This is also by construction.

(3): By Theorem 5.10 and Theorem 5.8 we see have that $\operatorname{Span} \mathcal{D}'_{r,n,m} = \operatorname{Span} \mathcal{B}'_{r,n,m}$. In particular, the measurements $\langle M, \mathcal{D}'_{r,n,m} \rangle$ can be reconstructed from the measurements $\langle M, \operatorname{Span} \mathcal{B}'_{r,n,m} \rangle$. As the above corollary shows that $\mathcal{D}'_{r,n,m}$ is r-low-rank-recovery set, it follows that $\mathcal{B}'_{r,n,m}$ is also.

(4): The analysis given in Theorem 5.10 gives an algorithm for reconstructing the measurements $\langle M, \mathcal{D}'_{r,n,m} \rangle$ from the measurements $\langle M, \operatorname{Span} \mathcal{B}'_{r,n,m} \rangle$, and does so interpolating r polynomials of degree $\leq n+m$. As evaluations of these polynomials takes $\mathcal{O}(r)$ steps, and polynomial interpolation takes $\mathcal{O}(m^2)$ steps for polynomials of this degree, we see that we can complete this interpolation in $\mathcal{O}(rm^2 + r^2m) = \mathcal{O}(rm^2)$ steps. Once the measurements $\langle M, \mathcal{D}'_{r,n,m} \rangle$ are computed, we can appeal to the above corollary. $\qquad\square$

The above results only work over fields when we have an element g of large order. However, the results of Subsection 6.3 show that we can simulate these results over small fields. Indeed, this is also the case here.

Corollary 7.26. *Let $m \geq n \geq r \geq 1$. Over any field $\mathbb{F}$, there is an $\mathsf{poly}(m)$-explicit r-low-rank-recovery set for $n \times m$ matrices, which has size $\mathcal{O}(rm \lg m)$ and is such that each recovery matrix is $\mathcal{O}(n)$-sparse. There is also an $\mathsf{poly}(m)$-explicit r-low-rank-recovery set for $n \times m$ matrices, which has size $\mathcal{O}(rm \lg^2 m)$ and is such that each recovery matrix is rank 1. Further, recovery from either of these low-rank-recovery sets can be performed in $\mathsf{poly}(m)$ time.*

Proof. We begin by noting that both Proposition 6.12 and Proposition 6.16 preserve the property of being a low-rank-recovery set, not just that of being a hitting set. That is, each of these propositions take a $\mathbb{K}$-matrix H in the original low-rank-recovery set and construct some family of $\mathbb{F}$-matrices $\{\tilde{H}_{\ell,\ell'}\}_{\ell,\ell'}$ such that for any matrix M, $\langle M, H \rangle$ can be efficiently recovered from the sums $\{\sum_\ell \alpha_\ell \langle M, \tilde{H}_\ell \rangle\}_{\ell'}$, for some coefficients $\alpha_\ell \in \mathbb{K}$. Thus the measurements $\langle M, \mathcal{H} \rangle$ are efficiently recoverable from the measurements $\langle M, \mathcal{H} \rangle$.

Finally, appealing to the constructions of low-rank-recovery sets as given in Corollary 7.22 (to which Proposition 6.12 is applied) and Corollary 7.25 (to which Proposition 6.16 is applied) completes the claim. $\qquad\square$

8 Rank-Metric Tensor codes

We now discuss low-rank-recovery of tensors, for any d, and apply our results to the construction of rank-metric codes. We begin with showing that the matrix low-rank-recovery algorithm can be extended to the $d > 2$ case.

Theorem 8.1. *Let $n, r \geq 1$ and $d \geq 2$. Then $\mathcal{B}_{d,n,2r}$, as defined in Construction 6.10, has*

1. $|\mathcal{B}_{d,n,2r}| \leq \mathcal{O}(dn(2r)^{\mathcal{O}(\lg d)})$

2. $\mathcal{B}_{d,n,2r}$ *is an r-low-rank-recovery set, and recovery can be performed in time $\mathsf{poly}((2dn)^d, (2r)^{\mathcal{O}(\lg d)})$*

Proof. (1): This is by construction.

(2): The hitting set allows us to interpolate the polynomials stated in the hypothesis of Theorem 6.6. Once we have the coefficients of this polynomial, we can undo the reductions used in the proof of Theorem 6.6. That is, that proof uses Lemmas 6.1 and 6.2 to reshape polynomials by merging their variables. This is clearly efficiently reversible. More crucially, the proof uses the bivariate variable reduction of Theorem 5.1 for rank $\leq r$ matrices, but when we take $2r$ distinct powers of g. However, Corollary 7.22 shows that one can recover $\hat{f}_M(x, y)$ from the polynomials $\{\hat{f}_M(x, g^i x)\}_{i \in [\![2r]\!]}$ in $\mathsf{poly}(\deg_x(\hat{f}_M), \deg_y(\hat{f}_M), r)$ steps. As the degrees involved in Theorem 6.6 are only up to $(2dn)^d$, this is within the stated time bounds. Thus, we can also reverse the bivariate variable reduction steps used in Theorem 6.6. Combining these steps shows that we can fully recover the entire polynomial $\hat{f}_T(x_1, \ldots, x_d)$, which gives the tensor T. $\qquad\square$

We next observe that, just as with Corollary 7.26, we can perform this low-rank-recovery over small fields, when incurring a loss.

Corollary 8.2. *Let $n, r \geq 1$ and $d \geq 2$. Over any field $\mathbb{F}$, there is an $\mathsf{poly}((2nd)^d, r^{\mathcal{O}(\lg d)})$-explicit r-low-rank-recovery set for $[\![n]\!]^d$ tensors, which has size $\mathcal{O}(dn(2r)^{\mathcal{O}(\lg d)} \cdot (d \lg 2dn)^d)$ and is such that each recovery tensor is rank 1. Further, there is an $\mathsf{poly}((2nd)^d, r^{\mathcal{O}(\lg d)})$-explicit r-low-rank-recovery set for $[\![n]\!]^d$ tensors, which has size $\mathcal{O}(dn(2r)^{\mathcal{O}(\lg d)} \cdot d \lg 2dn)$. Further, recovery from either of these low-rank-recovery sets can be performed in $\mathsf{poly}((2nd)^d, r^{\mathcal{O}(\lg d)})$ time.*

Proof. Like Corollary 7.26, we apply Propositions 6.16 and 6.12 to a low-rank-recovery set, where here we use the above set from Theorem 8.1. As Propositions 6.16 and 6.12, as well as Theorem 8.1, are efficiently implementable, so are the resulting low-rank-recovery sets. $\qquad\square$

We now apply these results to create error correcting codes over the rank-metric, which we now define. We will restrict our attention to linear codes in this work.

Definition 8.3. *A $[\![n]\!]^d, k, r]\!]_{\mathbb{F}}$ **rank-metric code** $\mathcal{C}$ is a k-dimensional subspace of $\mathbb{F}^{[\![n]\!]^d}$ (the space of $[\![n]\!]^d$ tensors) such that for all $T_1 \neq T_2 \in \mathcal{C}$, $\mathrm{rank}(T_1 - T_2) \geq r$. Denote r as the **distance** of the code.*

*An algorithm Dec **corrects** e **errors** against $\mathcal{C}$ if for any $T \in \mathcal{C}$ and $E \in \mathbb{F}^{[\![n]\!]^d}$ with $\mathrm{rank}(E) \leq e$ it is such that $\mathsf{Dec}(T + E) = T$.*

Thus this is the natural definition for error-correcting codes when we use the rank-metric (notice that rank-distance is in fact a metric) as the notion of distance. As we are interested in linear codes $T_1 - T_2 \in \mathcal{C}$ also, so an equivalent definition to the above would say that $r \leq \mathrm{rank}(T)$ for all $0 \neq T \in \mathcal{C}$. Just as with the Hamming-metric, if we have a distance $2r + 1$ code $\mathcal{C}$ then it is information theoretically possible to decode up to r errors. The converse is shown below.

Lemma 8.4. *Let $\mathcal{C}$ be a $[\![n]\!]^d, k, r']\!]_{\mathbb{F}}$ rank-metric code that can correct up to r errors. Then $r' \geq 2r + 1$.*

Proof. Suppose not for contradiction. Then there are two tensors $T_1 \neq T_2 \in \mathcal{C}$ such that $\mathrm{rank}(T_2 - T_1) \leq 2r$. But then $T_2 - T_1 = S_1 + \cdots + S_{2r}$, where these S_i are all rank-1 (or rank-0) tensors. Then it follows that $T_1 + S_1 + \cdots + S_r$ is r-close to both T_1 and T_2, which is impossible as the correctness of the decoding procedure indicates that there should be a unique tensor that $T_1 + S_1 + \cdots + S_r$ is r-close to. $\qquad\square$

Corollary 8.5. *Let $\mathbb{F}$ be a field, $m \geq n \geq r \geq 1$. Then there are $\mathsf{poly}(m)$-explicit rank-metric codes with $\mathsf{poly}(m)$-time decoding for up to r errors, with parameters:*

1. $[\![n]\!] \times [\![m]\!], nm - 2(n + m - 2r)r, 2r + 1]_{\mathbb{F}}$, *if $|\mathbb{F}| > m$, and the parity checks on this code can be either all rank-1 matrices, or all $\mathcal{O}(n)$-sparse matrices.*

2. $[\![n]\!] \times [\![m]\!], nm - 2(n + m - 2r)r \cdot \mathcal{O}(\lg m), 2r + 1]_{\mathbb{F}}$, *any $\mathbb{F}$, and the parity checks on this code are all $\mathcal{O}(n)$-sparse matrices.*

3. $[\![n]\!] \times [\![m]\!], nm - 2(n + m - 2r)r \cdot \mathcal{O}(\lg^2 m), 2r + 1]_{\mathbb{F}}$, *any $\mathbb{F}$, and the parity checks on this code are all rank-1 matrices.*

Proof. We first generically show how to define an $[nm, nm - |\mathcal{H}|, 2r + 1]_{\mathbb{F}}$ rank-metric code $\mathcal{C}$ from an r-low-rank-recovery set $\mathcal{H}$ and how to use the low-rank-recovery algorithm for $\mathcal{H}$ to decode $\mathcal{C}$ up to r errors. The corollary is then immediate by using the results of Corollaries 7.25, 7.22, 7.26, and invoking the efficiency of their low-rank-recovery.

Define $\mathcal{C}$ to be the matrices in the nullspace of $\mathcal{H}$. That is, $\mathcal{C} = \{M : \langle M, \mathcal{C} \rangle = 0\}$. It is clear that $\mathcal{C}$ is a subspace (and assuming that the matrices in $\mathcal{H}$ are linearly independent, which is true for the low-rank-recovery sets $\mathcal{D}'_{2r,n,m}$ and $\mathcal{B}'_{2r,n,m}$) and has dimension $nm - |\mathcal{H}|$.

Now consider some $T \in \mathcal{C}$ and matrix E with $\text{rank}(E) \leq r$. Abusing notation, consider T and E as nm-long vectors, and $\mathcal{H}$ as a $|\mathcal{H}| \times nm$ matrix. It follows that $\mathcal{H}(T + E) = \mathcal{H}E$ as $T \in \mathcal{C}$. As $\mathcal{H}$ is an r-low-rank-recovery set, it follows that we can recover E from $\mathcal{H}E$, and thus can recover T, performing successful decoding of up to r errors. By Lemma 8.4 we see that the minimum distance of this code is $\geq 2r + 1$. $\qquad\square$

We now separately state the result for tensors, which is proved exactly as the above corollary, but using the relevant low-rank-recovery results for tensors.

Corollary 8.6. *Let $\mathbb{F}$ be a field, $n, r \geq 1$ and $d \geq 2$. Then there are $\mathsf{poly}((2nd)^d, (2r)^{\mathcal{O}(\lg d)})$-explicit rank-metric codes with $\mathsf{poly}((2nd)^d, (2r)^{\mathcal{O}(\lg d)})$-time decoding for up to r errors, with parameters:*

1. $[\![n]\!]^d, n^d - dn(2r)^{\lceil \lg d \rceil}, 2r + 1]_{\mathbb{F}}$, *if $|\mathbb{F}| > (2nd)^d$, and the parity checks on this code are all rank-1 tensors,*

2. $[\![n]\!]^d, n^d - dnr^{\lceil \lg d \rceil} \cdot \mathcal{O}(d \lg(2dn)), 2r + 1]_{\mathbb{F}}$, *any $\mathbb{F}$,*

3. $[\![n]\!]^d, n^d - dnr^{\lceil \lg d \rceil} \cdot \mathcal{O}((d \lg(2dn))^d), 2r + 1]_{\mathbb{F}}$, *any $\mathbb{F}$, and the parity checks on this code are all rank-1 tensors.*

9 Discussion

We briefly discuss some directions for further research.

Reducing Noisy Low-Rank Recovery to Noisy Sparse Recovery We showed in Theorem 7.19 that low-rank-recovery of matrices can be done using any sparse-recovery oracle. This reduction was for non-adaptive measurements, and was done in the presence of no noise. As much of the compressed sensing community is interested in the noisy case (so M is only close to rank $\leq r$) the main open question of this work is whether the reduction extends to the noisy case.

Smaller Hitting Sets While the observations of Roth [Rot91] show that our hitting set for matrices is optimal over algebraically closed fields, our results (Corollary 6.18) over tensors with $d > 2$ are much larger than the existential bounds of Lemma 3.13. Can these hitting sets be improved to size $\mathcal{O}(\mathsf{poly}(d)nr^k)$ for $k = \mathcal{O}(1)$? As mentioned in the preliminaries (Lemma 3.14), any such hitting set with $k < 2$ would yield improved tensor-rank lower bounds (and thus circuit lower bounds) for odd d such as $d = 3$. However, as the best tensor-rank lower bounds for $d = 3$ are $\Theta(n)$ and our hitting set (over infinite fields) yields this bound (with a smaller constant), even improving our hitting set for $d = 3$ by constant factors could yield interesting new results. Specifically, for $d = 3$ can one construct (say over infinite fields) a hitting set of size $\leq nr^2/10$ for $[\![n]\!]^3$ tensors of rank $\leq r$?

Better Variable Reduction Theorem 5.1 shows that a bivariate polynomial with bounded individual degrees can be identity tested by identity testing a collection of univariate polynomials, where the size of this collection depends on the rank of bivariate polynomial. This naturally led to our hitting sets for matrices. We generalized this to d-variate polynomials in Theorem 6.6, but the collection of univariate polynomials has a size with a much worse dependence on the tensor-rank of the d-variate polynomial and is much less explicit. Can the size of the collection be reduced, or can the explicitness of this set be only polynomially larger than its size? We note that according to Lemma 3.14 a more explicit hitting set will yield lower bounds on tensor rank, however for tensors of high degrees such lower bounds are known [NW96].

Large Field Simulation The results of Section 6.3 show that hitting sets (and low-rank recovery sets) that involve tensors over an extension field imply hitting sets (and LRR sets) over the base field. While Proposition 6.16 shows that we can preserve the rank-1 property of these tensors while doing so, it introduces an $\exp(d)$ factor in the size of the hitting set. Can this be improved?

Acknowledgements

References

[AFT11]   B. Alexeev, M. Forbes, and J. Tsimerman. Tensor rank: some lower and upper bounds. In *IEEE Conference on Computational Complexity*, pages 283–291. IEEE Computer Society, Feb 2011. 14

[Agr05]   M. Agrawal. Proving lower bounds via pseudo-random generators. In *Proceedings of the 25th FSTTCS*, volume 3821 of *LNCS*, pages 92–105, 2005. 2, 14

[AV08]   M. Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *Proceedings of the 49th Annual FOCS*, pages 67–75, 2008. 2

[BBB+00] A. Beimel, F. Bergadano, N. H. Bshouty, E. Kushilevitz, and S. Varricchio. Learning functions represented as multiplicity automata. *J. ACM*, 47(3):506–530, 2000. 4

[BBV96]   F. Bergadano, N. H. Bshouty, and S. Varricchio. Learning multivariate polynomials from substitution and equivalence queries. *ECCC*, 3(8), 1996. 4

[BD80]   M. R. Brown and D. P. Dobkin. An improved lower bound on polynomial multiplication. *IEEE Trans. Computers*, 29(5):337–340, 1980. 14

[CP09]   E. J. Candes and Y. Plan. Matrix Completion With Noise. *ArXiv e-prints*, March 2009. 3

[CP11]   E. J. Candés and Y. Plan. Tight oracle inequalities for low-rank matrix recovery from a minimal number of noisy random measurements. *IEEE Trans. Inform. Theory*, 57(4):2342–2359, 2011. 3

[CRT05]   E. Candes, J. Romberg, and T. Tao. Stable Signal Recovery from Incomplete and Inaccurate Measurements. *ArXiv Mathematics e-prints*, March 2005. 3

[CSw]   http://dsp.rice.edu/cs. 2

[CT09]   E. J. Candes and T. Tao. The Power of Convex Relaxation: Near-Optimal Matrix Completion. *ArXiv e-prints*, March 2009. 3

[Del78]   Ph. Delsarte. Bilinear forms over a finite field, with applications to coding theory. *J. Combin. Theory Ser. A*, 25(3):226–241, 1978. 4, 6, 7, 32

[dP95]   G. C. F. M. Riche de Prony. Essai éxperimental et analytique: sur les lois de la dilatabilité de fluides élastique et sur celles de la force expansive de la vapeur de l'alkool, à différentes températures. *Journal de l'école Polytechnique*, 1:24–76, 1795. 8, 34

[DS08]   Z. Dvir and A. Shpilka. Towards dimension expanders over finite fields. In *Proceedings of the 23rd Annual IEEE Conference on Computational Complexity (CCC)*, pages 304–310, 2008. 3

[ENP11]   Y. C. Eldar, D. Needell, and Y. Plan. Unicity conditions for low-rank matrix recovery. *arXiv:1103.5479*, 2011. 3

[Gab85a]   E. M. Gabidulin. Optimal array error-correcting codes. *Probl. Peredach. Inform.*, 21(2):102–106, 1985. 4, 6, 7, 32

[Gab85b] E. M. Gabidulin. Theory of codes with maximum rank distance. *Probl. Inform. Transm.*, 21(1):1–12, 1985. 4, 6, 7, 32

[GK72] E. M. Gabidulin and V. I. Korzhik. Codes correcting lattice-pattern errors. *Zzvestiya VUZ. Radioelektronika*, 1972. 4, 6, 7, 32

[GLF$^+$10] David Gross, Yi-Kai Liu, Steven T. Flammia, Stephen Becker, and Jens Eisert. Quantum state tomography via compressed sensing. *Phys. Rev. Lett.*, 105:150401, Oct 2010. 3

[GR08] A. Gabizon and R. Raz. Deterministic extractors for affine sources over large fields. *Combinatorica*, 28(4):415–440, 2008. 5, 8, 14, 15, 17

[Gro09] D. Gross. Recovering low-rank matrices from few coefficients in any basis. *ArXiv e-prints*, October 2009. 3

[Hås90] J. Håstad. Tensor rank is np-complete. *J. Algorithms*, 11(4):644–654, 1990. 7

[HS80] J. Heintz and C. P. Schnorr. Testing polynomials which are easy to compute (extended abstract). In *Proceedings of the 12th annual STOC*, pages 262–272, 1980. 2, 14

[IPW11] P. Indyk, E. Price, and D. P. Woodruff. On the Power of Adaptivity in Sparse Recovery. *Foundations of Computer Science*, October 2011. 3, 32

[KI04] V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004. 2

[KOH11] A. Khajehnejad, S. Oymak, and B. Hassibi. Subspace expanders and matrix rank minimization. *arXiv:1102.3947v1*, 2011. 3

[KS06] A. Klivans and A. Shpilka. Learning restricted models of arithmetic circuits. *Theory of computing*, 2(10):185–206, 2006. 4

[KS08] Z. S. Karnin and A. Shpilka. Deterministic black box polynomial identity testing of depth-3 arithmetic circuits with bounded top fan-in. In *Proceedings of the 23rd Annual CCC*, pages 280–291, 2008. 8

[lrr] http://perception.csl.uiuc.edu/matrix-rank/. 3

[LZ08] A. Lubotzky and Y. Zelmanov. Dimension expanders. *J. Algebra*, 319(2):730–738, 2008. 3

[Mes95] R. Meshulam. Spaces of Hankel matrices over finite fields. *Linear Algebra Appl.*, 218:73–76, 1995. 4

[NW96] N. Nisan and A. Wigderson. Lower bound on arithmetic circuits via partial derivatives. *Computational Complexity*, 6:217–234, 1996. 2

[PCM88] G.M. Pitstick, J.R. Cruz, and R.J. Mulholland. A novel interpretation of Prony's method. *Proceedings of the IEEE*, 76(8):1052 –1053, aug 1988. 34

[Raz10] R. Raz. Tensor-rank and lower bounds for arithmetic formulas. In *Proceedings of the 42nd Annual STOC*, pages 659–666, 2010. 14

[RFP10] B. Recht, M. Fazel, and P. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010. 3, 6, 32

[Rot91]   R. M. Roth. Maximum-rank array codes and their application to crisscross error correction. *IEEE Transactions on Information Theory*, 37(2):328–336, 1991. 4, 6, 7, 13, 32, 49

[Rot96]   R. M. Roth. Tensor codes for the rank metric. *IEEE Transactions on Information Theory*, 42(6):2146–2157, 1996. 4, 7, 32

[RS05]   R. Raz and A. Shpilka. Deterministic polynomial identity testing in non commutative models. *Computational Complexity*, 14(1):1–19, 2005. 1, 2, 12

[Sch80]   J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980. 13

[SS11]   N. Saxena and C. Seshadhri. Blackbox identity testing for bounded top fanin depth-3 circuits: the field doesn't matter. In *Proceedings of the 43rd Annual STOC*, pages 431–440, 2011. 2, 5, 8

[Str73]   V. Strassen. Vermeidung von divisionen. *J. of Reine Angew. Math.*, 264:182–202, 1973. 14

[SY10]   A. Shpilka and A. Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5(3-4):207–388, 2010. 2, 4, 11

[TBD11]   V. Y. F. Tan, L. Balzano, and S. C. Draper. Rank minimization over finite fields: Fundamental limits and coding-theoretic interpretations. *arXiv:1104.4302v2*, 2011. 6

[Zip79]   R. Zippel. Probabilistic algorithms for sparse polynomials. In *Symbolic and algebraic computation*, pages 216–226. 1979. 13

A   Cauchy-Binet Formula

For completeness we give the proof of the Cauchy-Binet formula here.

Lemma A.1 (Cauchy-Binet Formula). *Let $m \geq n \geq 1$. Let $A \in \mathbb{F}^{n \times m}$, $B \in \mathbb{F}^{m \times n}$. For $S \subseteq [\![m]\!]$, let A_S be the $n \times |S|$ matrix formed from A by taking the columns with indices in S. Let B_S be defined analogously, but with rows. Then*

$$\det(AB) = \sum_{S \in \binom{[\![m]\!]}{n}} \det(A_S) \det(B_S)$$

Proof. Let C be an $m \times m$ diagonal matrix with the variables $x_1, \ldots, x_m$ on the diagonal. Define the polynomial $f(x_1, \ldots, x_m) \stackrel{\text{def}}{=} \det(ACB)$, so that $f(1, \ldots, 1) = \det(AB)$. Every entry of ACB is a homogeneous linear function in $x_1, \ldots, x_m$, which implies (as the determinant is homogeneous of degree n) that f is homogeneous of degree n, or zero. Let $S \in \binom{[\![m]\!]}{n}$ and consider all monomials only containing variables in $\{x_i \mid i \in S\}$. Note that also consider monomials with individual degrees above 1. Each monomial of degree n (and thus each monomial with non-zero coefficient in f) must be associated with some such S.

Define ρ_S to be the vector of variables when the substitution $x_i \mapsto 0$ is performed for $i \notin S$. It follows then that $f(\rho_S) = \det(A_S C_S B_S) = \det(A_S) \det(B_S) \cdot \prod_{i \in S} x_i$, where the last equality follows as A_S, B_S and C_S are all $n \times n$ matrices. By the above reasoning, this implies that the only monomials with non-zero coefficients in f are monomials of the form $\prod_{i \in S} x_i$ and such monomials have coefficient $\det(A_S) \det(B_S)$. Thus $f = \sum_{S \in \binom{[\![m]\!]}{n}} \det(A_S) \det(B_S) \prod_{i \in S} x_i$, and so $\det(AB) = f(1, \ldots, 1) = \sum_{S \in \binom{[\![m]\!]}{n}} \det(A_S) \det(B_S)$, yielding the claim. $\qquad \square$