2016 IEEE 57th Annual Symposium on Foundations of Computer Science

NP-Hardness of Reed-Solomon Decoding and the Prouhet-Tarry-Escott Problem

Venkata Gandikota*, Badih Ghazi † and Elena Grigorescu‡
* *Department of Computer Science, Purdue University, West Lafayette, IN 47906*
Email: vgandiko@purdue.edu
† *Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge MA 02139*
Email: badih@mit.edu
‡*Department of Computer Science, Purdue University, West Lafayette, IN 47906*
Email: elena-g@purdue.edu

***Abstract*—Establishing the complexity of *Bounded Distance Decoding* for Reed-Solomon codes is a fundamental open problem in coding theory, explicitly asked by Guruswami and Vardy (IEEE Trans. Inf. Theory, 2005). The problem is motivated by the large current gap between the regime when it is NP-hard, and the regime when it is efficiently solvable (i.e., the Johnson radius).**

We show the first NP-hardness results for asymptotically smaller decoding radii than the maximum likelihood decoding radius of Guruswami and Vardy. Specifically, for Reed-Solomon codes of length N and dimension $K = O(N)$, we show that it is NP-hard to decode more than $N - K - O(\frac{\log N}{\log \log N})$ errors. Moreover, we show that the problem is NP-hard under quasipolynomial-time reductions for an error amount $> N - K - c \log N$ (with $c > 0$ an absolute constant).

An alternative natural reformulation of the Bounded Distance Decoding problem for Reed-Solomon codes is as a *Polynomial Reconstruction* problem. In this view, our results show that it is NP-hard to decide whether there exists a degree K polynomial passing through $K + O(\frac{\log N}{\log \log N})$ points from a given set of points $(a_1, b_1), (a_2, b_2) \ldots, (a_N, b_N)$. Furthermore, it is NP-hard under quasipolynomial-time reductions to decide whether there is a degree K polynomial passing through $K + c \log N$ many points (with $c > 0$ an absolute constant).

These results follow from the NP-hardness of a generalization of the classical Subset Sum problem to higher moments, called *Moments Subset Sum*, which has been a known open problem, and which may be of independent interest.

We further reveal a strong connection with the well-studied Prouhet-Tarry-Escott problem in Number Theory, which turns out to capture a main barrier in extending our techniques. We believe the Prouhet-Tarry-Escott problem deserves further study in the theoretical computer science community.

***Keywords*-Reed-Solomon codes; Bounded distance decoding; Moment Subset Sum;**

I. Introduction

Despite being a classical problem in the study of error-correcting codes, the computational complexity of decoding Reed-Solomon codes [1] in the presence of large amounts of error is not fully understood. In the Bounded Distance Decoding problem, the goal is to recover a message corrupted by a bounded amount of error. Motivated by the large gap between the current efficient decoding regime, and the NP-hard regime for Reed-Solomon codes, we study the NP-hardness of Bounded Distance Decoding for asymptotically smaller error radii than previously known. In this process, we unravel a strong connection with the Prouhet-Tarry-Escott, a famous problem from number theory that has been studied for more than two centuries.

A Reed-Solomon (RS) code of length N, dimension K, defined over a finite field F, is the set of vectors (called *codewords*) corresponding to evaluations of low-degree univariate polynomials on a given set of evaluation points $\mathcal{D} = \{\alpha_1, \alpha_2, \ldots, \alpha_N\} \subseteq \mathsf{F}$. Formally, $RS_{\mathcal{D},K} = \{\langle p(\alpha_1), \ldots, p(\alpha_N)\rangle : p \in \mathsf{F}[x]$ is a univariate polynomial of degree $< K\}$. The Hamming distance between $x, y \in \mathsf{F}^N$ is $\Delta(x, y) := |\{i \in [N] : x_i \neq y_i\}|$. In the *Bounded Distance Decoding (BDD) problem*, given a target vector $y \in \mathsf{F}^N$ and a distance parameter λ, the goal is to output $c \in \mathcal{C}$ such that $\Delta(c, y) \leq \lambda$.

It is well-known that if the number of errors is $\lambda \leq (N - K)/2$, there is a unique codeword within distance λ from the message, which can be found efficiently [2], [3]. Further, Sudan [4] and Guruswami and Sudan [5] show efficient decoding up to $\lambda = N - \sqrt{NK}$ errors (the "Johnson radius"), a setting in which the algorithm may output a small list of possible candidate messages. At the other extreme, if the number of errors is at least $N - K$ (the covering radius), finding one close codeword becomes trivial, amounting to interpolating a degree $K-1$ polynomial through $\leq K$ points. However, just below that radius, namely at $N - K - 1$ errors, the problem becomes NP-hard, a celebrated result of Guruswami and Vardy [6]. The proof approach of [6] is only applicable to the Maximum Likelihood Decoding setting of $N - K - 1$ errors, prompting the fundamental problem of understanding the complexity of BDD in the wide remaining range between $N - \sqrt{KN}$ and $N - K - 1$:

> [6] *"It is an extremely interesting problem to show hardness of bounded distance decoding of Reed-Solomon codes for smaller decoding radius."*

0272-5428/16 $31.00 © 2016 IEEE
DOI 10.1109/FOCS.2016.86

Some partial progress on improving the NP-hardness regime was shown in a recent result by the same authors [7] for $N-K-2$ and $N-K-3$ errors. The only other work addressing the hardness of decoding RS codes are due to Cheng and Wan [8], [9] who show randomized reductions from the Discrete Log problem over finite fields, which is not believed to be NP-hard.

In this work, we study the complexity of the decision version of BDD, where the number of errors is parametrized by $d \geq 0$, as formalized next:

> **Problem** *Bounded Distance Decoding of Reed-Solomon codes with parameter d (RS-BDD(d))*
> **Input** $\mathcal{D} = \{\alpha_1, \alpha_2, \ldots, \alpha_N\} \subseteq \mathsf{F}$, where $\alpha_i \neq \alpha_j$ for all $i \neq j$, target $y = (y_1, y_2, \ldots, y_N)$, and integer $K < N$
> **Goal** Decide if there exists $p \in RS_{\mathcal{D},K}$ such that $\Delta(y,p) \leq (N-K) - d$

We emphasize that the BDD problem above is in fact the basic and natural Polynomial Reconstruction problem, where the input is a set of points $\mathcal{D} = \{(\alpha_1, y_1), (\alpha_2, y_2), \ldots, (\alpha_N, y_N)\} \subseteq \mathsf{F} \times \mathsf{F}$, and the goal is to decide if there exists a polynomial p of degree $< K$ that passes through at least $K + d$ points in $\mathcal{D}$.

We state our main result in both forms.

A. Contributions

Our main technical contribution is the first NP-hardness result for BDD of RS codes, for a number of errors that is asymptotically smaller than $N-K$, and its alternative view in terms of polynomial reconstruction.

Theorem I.1. *For every $1 \leq d \leq O(\frac{\log N}{\log\log N})$, the RS-BDD($d$) problem for Reed-Solomon codes of length N, dimension $K = N/2 - d + 1$ and field size $|\mathsf{F}| = 2^{\mathsf{poly}(N)}$ is NP-hard. Furthermore, there exists $c > 0$, such that for every $1 \leq d \leq c \cdot \log N$, RS-BDD($d$) over fields of size $|\mathsf{F}| = 2^{N^{O(\log\log N)}}$ does not have $N^{O(\log\log N)}$-time algorithms unless NP has quasi-polynomial time algorithms.*

Equivalently, for every $1 \leq d \leq O(\frac{\log N}{\log\log N})$, it is NP-hard to decide whether there exists a polynomial of degree $< K = N/2 - d + 1$ passing through $K + d$ many points from a given set $\mathcal{D} = \{(\alpha_1, y_1), (\alpha_2, y_2), \ldots, (\alpha_N, y_N)\} \subseteq \mathsf{F} \times \mathsf{F}$, with $|\mathsf{F}| = 2^{\mathsf{poly}(N)}$. Furthermore, there exists $c > 0$, such that for every $1 \leq d \leq c \cdot \log N$, the same interpolation problem over fields of size $|\mathsf{F}| = 2^{N^{O(\log\log N)}}$ does not have $N^{O(\log\log N)}$-time algorithms unless NP has quasi-polynomial time algorithms.

Our results significantly extend [6], [7], which only show NP-hardness for $d \in \{1, 2, 3\}$. As in [6], [7], we require the field size to be exponential in N.

The bulk of the proof of Theorem I.1 is showing the NP-hardness of a natural generalization of the classic Subset Sum problem to higher moments, that may be of independent interest.

> **Problem** *Moments Subset Sum with parameter d, over a field F (MSS(d))*
> **Input** Set $A \subseteq \mathsf{F}$ of size $|A| = N$, integer k, elements $m_1, m_2, \ldots, m_d \in \mathsf{F}$
> **Goal** Decide if there exists $S \subseteq A$ such that $\sum_{s \in S} s^\ell = m_\ell$, for all $\ell \in [d]$, and $|S| = k$.

We note that the reduction from $\mathrm{MSS}(d)$ to $\mathrm{RS\text{-}BDD}(d)$ uses the equivalence between elementary symmetric polynomials and moments polynomials, when the field is of characteristic larger than $\Omega(d!)$(see, e.g., [7] for a formal reduction.)

We point out that the Moments Subset Sum problem has natural analogs over continuous domains in the form of generalized moment problems and truncated moments problems, which arise frequently in economics, operations research, statistics and probability [10].

In this work, we prove NP-hardness of the Moments Subset Sum problem for large degrees.

Theorem I.2. *For every $1 \leq d \leq O(\frac{\log N}{\log\log N})$, the Moments Subset Sum problem MSS(d) over prime fields of size $|\mathsf{F}| = 2^{\mathsf{poly}(N)}$ is NP-hard. Furthermore, there exists $c > 0$, such that for every $1 \leq d \leq c \cdot \log N$, the Moments Subset Sum problem MSS(d) over fields of size $|\mathsf{F}| = 2^{N^{O(\log\log N)}}$ does not have $N^{O(\log\log N)}$-time algorithms unless NP has quasi-polynomial time algorithms.*

Furthermore, we reveal a connection with the famous Prouhet-Tarry-Escott (PTE) problem in Diophantine Analysis, which is the main barrier for extending Theorem I.2 and Theorem I.1 to $d = \omega(\log N)$, as explained shortly.

The PTE problem [11], [12], [13] first appeared in letters between Euler and Goldbach in 1750-1751, and it is a important topic of study in classical number theory (see, e.g., the textbooks of Hardy and Wright [14] and Hua [15]). It is also related to other classical problems in number theory, such as variants of the Waring problem and problems about minimizing the norm of cyclotomic polynomials, considered by Erdös and Szekeres [16], [17].

In the Prouhet-Tarry-Escott problem, given $k \geq 1$, the goal is to find disjoint sets of integers $\{x_1, x_2, \ldots, x_t\}$ and $\{y_1, y_2, \ldots, y_t\}$ satisfying the system:

$$\begin{array}{rcl} x_1 + x_2 + \cdots + x_t & = & y_1 + y_2 + \cdots + y_t \\ x_1^2 + x_2^2 + \cdots + x_t^2 & = & y_1^2 + y_2^2 + \cdots + y_t^2 \\ & \cdots & \\ x_1^k + x_2^k + \cdots + x_t^k & = & y_1^k + y_2^k + \cdots + y_t^k. \end{array}$$

We call t the size of the PTE solution. It turns out that the completeness proof of our reduction in Theorem I.2 relies on *explicit* solutions to this system for degree $k = d$ and of size $t = 2^k$. As explained next, despite significant efforts that have been devoted to constructing PTE solutions during the last 100 years, no explicit solutions of size $t = o(2^k)$

are known. This constitutes the main barrier to extending our Theorem I.2 and Theorem I.1 to $d = \omega(\log N)$.

The main open problem that has been tackled in the PTE literature is constructing solutions of small size t compared to the degree k. It is relatively easy to show that $t \geq k+1$, and straightforward (yet non-constructive!) pigeonhole counting arguments show the existence of solutions with $t = O(k^2)$. If we further impose the constraint that the system is not satisfied for degree $k+1$ (which is a necessary constraint for our purposes), then solutions of size $t = O(k^2 \log k)$ are known to exist [15]. However, these results are non-constructive, and the only general explicit solutions have size $t = O(2^k)$ (e.g., [13], [17]). A special class of solutions studied in the literature is for $t = k+1$ (of minimum possible size). Currently there are known explicit parametric constructions of infinitely many minimum-size solutions for $k \leq 12$ (e.g., [17], [18]), and finding such solutions often involves numerical simulations and extensive computer-aided searches [18].

From a computational point of view, an important open problem is to understand whether PTE solutions of size $O(k^2)$ (which are known to exist) can be *efficiently constructed*, i.e., in time $\mathsf{poly}(k)$.

We identify the following generalization of the PTE problem as a current barrier to extending our results:

Problem I.3. *Given a field* F*, integer* d*, and* $a, b \in \mathsf{F}$*, efficiently construct* $x_1, \ldots, x_t, y_1, \ldots, y_t \in \mathsf{F}$*, with* $t = o(2^d)$*, satisfying:*

$$x_1 + x_2 + \cdots + x_t = y_1 + y_2 + \cdots + y_t$$
$$a^i + \sum_{j=1}^{t} x_j^i = b^i + \sum_{j=1}^{t} y_j^i \quad \forall i \in \{2, \ldots, d\}$$

We believe that this question is worth further study in the theoretical computer science community.

In the next section, we outline the proof of Theorem I.2, and in the process, we explain how PTE solutions of degree d naturally arise when studying the computational complexity of MSS(d).

B. Proof Overview

To prove Theorem I.2, we begin with the classical reduction from 1-in-3-SAT to Subset-Sum, in which one needs to construct a set of integers such that there is a subset whose sum equals a given target m_1, if and only if there is an assignment that satisfies exactly one literal of each clause of the 3-SAT formula (we refer the reader to Section III for more details about this standard reduction). Extending this reduction so that the 2nd moment also hits target m_2 raises immediate technical hurdles, since we have very little handle on the extra moment. In [7], the authors manage to handle a reduction for 2nd and 3rd moments via ad-hoc arguments and identities tailored to the degree-2 and degree-3 cases. The problem becomes much more complex as we need to ensure both completeness and soundness for a large number of moments. In this work, we achieve such a reduction where the completeness will rely on explicit solutions to "inhomogeneous PTE instances" and the soundness will rely on a delicate balancing of the magnitudes of these explicit solutions. We now describe the details of this reduction.

For each 1-in-3-SAT variable, we create a collection of *explicit* auxiliary numbers which "stabilize" the contribution of this variable to all i-th moment equations with $2 \leq i \leq d$, while having no net effect on the 1st moment equation. Concretely, if a and b are the numbers corresponding to the two literals of the given variable, then we need to find numbers $x_1, \ldots, x_t, y_1, \ldots, y_t$ satisfying:

$$x_1 + x_2 + \cdots + x_t = y_1 + y_2 + \cdots + y_t$$
$$a^i + \sum_{j=1}^{t} x_j^i = b^i + \sum_{j=1}^{t} y_j^i \quad \forall i \in \{2, \ldots, d\} \qquad (\dagger)$$

Note that in order for the overall reduction to run in polynomial-time, the above auxiliary variables should be *efficiently constructible*. Moreover, we observe that (†) is an inhomogeneous PTE instance: for $a = b$, it reduces to a PTE instance of degree d. Of course, in our case a and b will not be equal, and (†) is a more general system (and is hence harder to solve) than PTE instances. Nevertheless, as we will see shortly, solving (†) can be essentially reduced to finding explicit PTE solutions of degrees $k \leq d$.

In addition, we need to ensure that the added auxiliary numbers satisfy some "bimodality" property regarding their magnitudes, which would allow the recovery of a satisfying 1-in-3-SAT assignment from any solution to the MSS(d) instance:

Property I.4 (Bimodality (informal))**.** *Every subset* S *of the auxiliary variables is such that either* $|\sum_{s \in S} s|$ *is tiny, or* $|\sum_{s \in S} s|$ *is huge.*

We note that the existence of explicit and efficiently constructible solutions of small size $t = O(d)$ to system (†) (and hence to a PTE system too) would at least ensure the completeness of a reduction with $d = O(N)$. If soundness can also be ensured for such solutions, then our techniques would extend to radii closer to the Johnson Bound radius.

Overview of procedure for solving system (†): We build the variables x_i and y_i recursively, by reducing the construction for degree i to a solution to degree $i-1$. Towards this goal, we design a sub-procedure, called AtomicSolver, that takes as inputs an integer $i \in \{2, 3, \ldots, d\}$, and a number R_i, and outputs 2^i rational[1] numbers $\{x_{i,j}, y_{i,j}\}_{j \in [2^{i-1}]}$ that satisfy a PTE system of degree $i-1$, along with a non-homogeneous equation of degree i:

[1] In our case, we can afford having *rational* solutions to Equations (2a) and (2b). Note that this system is still a generalization of the PTE problem since we can always scale the rational solutions by their least common denominator to get a PTE solution of degree $i-1$.

$$\sum_{\ell=1}^{2^{i-1}}(x_{i,\ell}^j - y_{i,\ell}^j) = 0 \quad \forall\ 2 \leq j < i, \tag{2a}$$

$$\sum_{\ell=1}^{2^{i-1}}(x_{i,\ell}^i - y_{i,\ell}^i) = R_i. \tag{2b}$$

We can then run AtomicSolver sequentially on inputs $i \in \{2, \ldots, d\}$ with the R_i input corresponding to a "residual" term that accounts for the contributions to the degree-i equation of the outputs of AtomicSolver(j,R_j) for all $2 \leq j < i$, namely,

$$R_i = b^i - a^i + \sum_{2 \leq j < i} \sum_{\ell=1}^{2^{j-1}} (y_{j,\ell}^i - x_{j,\ell}^i). \tag{3}$$

Note that the aim of the AtomicSolver(i,R_i) procedure is to satisfy the degree-i equation (2b) without affecting the lower-degree equations (2a).

We then argue that the union $\cup_{2 \leq i \leq d}\{x_{i,j}, y_{i,j}\}_{j \in [2^{i-1}]}$ of all output variables satisfies the polynomial constraints in (†) with $t = \exp(d)$.

Specifics of the AtomicSolver*:* We next illustrate the AtomicSolver procedure by describing its operation in the particular case where $i = d = 4$. In what follows, we drop "$i = 4$ subscripts" and denote $R = R_4$, $x_\ell = x_{4,\ell}$ and $y_\ell = y_{4,\ell}$ for all $1 \leq \ell \leq 8$. Then, Equation (2b) above that we need to satisfy becomes

$$\sum_{\ell=1}^{8}(x_\ell^4 - y_\ell^4) = R. \tag{4}$$

First, we let α be a constant parameter (to be specified later on) and we set

$$x_1 - y_1 = \alpha \tag{5a}$$

$$y_2 - x_2 = \alpha \tag{5b}$$

Namely, in Equations (5a) and (5b), we "*couple*" the ordered pairs (x_1, y_1) and (y_2, x_2) in the same way. Then, using Equations (5a) and (5b), we substitute $y_1 = x_1 - \alpha$ and $x_2 = y_2 - \alpha$, and the sum of the $\ell = 1$ and $\ell = 2$ terms in Equation (4) can be written as

$$(x_1^4 - y_1^4) - (y_2^4 - x_2^4) = p_\alpha(x_1) - p_\alpha(y_2) \tag{6}$$

where p_α is a *cubic* polynomial. If we set $x_1 - y_2 = \beta$, then (6) further simplifies to

$$p_\alpha(x_1) - p_\alpha(y_2) = q_{\alpha,\beta}(x_1) \tag{7}$$

where $q_{\alpha,\beta}$ is a *quadratic* polynomial[2].

In the next step, we couple the ordered tuple (y_3, x_3, y_4, x_4) in the same way that we have so far coupled the tuple (x_1, y_1, x_2, y_2). The sum of the first four terms in the LHS of (4) then becomes

$$\begin{aligned}\sum_{\ell=1}^{4}(x_\ell^4 - y_\ell^4) &= (x_1^4 - y_1^4 + x_2^4 - y_2^4) - (y_3^4 - x_3^4 + y_4^4 - x_4^4) \\ &= q_{\alpha,\beta}(x_1) - q_{\alpha,\beta}(y_3).\end{aligned} \tag{8}$$

As before, we set $x_1 - y_3 = \gamma$ and (8) further simplifies to

$$q_{\alpha,\beta}(x_1) - q_{\alpha,\beta}(y_3) = w_{\alpha,\beta,\gamma}(x_1) \tag{9}$$

where $w_{\alpha,\beta,\gamma}(x_1)$ is a *linear* polynomial in x_1. Finally, we couple the ordered tuple $(y_5,\ x_5,\ y_6,\ x_6,\ y_7, x_7, y_8, x_8)$ in the same way that we have so far coupled the tuple $(x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4)$, and we obtain that the following equation is equivalent to Equation (4) above:

$$w_{\alpha,\beta,\gamma}(x_1) - w_{\alpha,\beta,\gamma}(y_5) = R. \tag{10}$$

Setting $x_1 - y_5 = \theta$, Equation (10) further simplifies to

$$\theta \cdot h_{\alpha,\beta,\gamma} = R, \tag{11}$$

where $h_{\alpha,\beta,\gamma}$ is the coefficient of x_1 in the linear polynomial $w_{\alpha,\beta,\gamma}(x_1)$. We conclude that to satisfy (4), it suffices to choose $\alpha,\ \beta, \gamma$ such that $h_{\alpha,\beta,\gamma} \neq 0$, and to then set $\theta = R/h_{\alpha,\beta,\gamma}$.

It is easy to see that there exist α, β, γ such that $h_{\gamma,\beta,\alpha} \neq 0$, and that the above recursive coupling of the variables guarantees that (2a) is satisfied. The more difficult part will be to choose α, β, γ in a way that ensures the soundness of the reduction. This is briefly described next.

Bimodality of solutions: In the above description of the particular case where $i = d = 4$, it can be seen that the produced solutions are $\{0, \pm 1\}$-linear combinations of $\{\alpha, \beta, \gamma, \theta\}$, which are required to satisfy (11). It turns out that in this case $h_{\alpha,\beta,\gamma} = 24 \cdot \alpha \cdot \beta \cdot \gamma$, and so (11) becomes

$$\theta \cdot \alpha \cdot \beta \cdot \gamma = \frac{R}{24}. \tag{12}$$

So assuming we can upper bound $|R|$,[3] we would be able to set θ to a sufficiently large power of 10 while letting α, β and γ to have tiny absolute values and satisfy (12). Using the fact that the auxiliary x_i and y_i variables are set to $\{0, \pm 1\}$-linear combinations of $\{\alpha, \beta, \gamma, \theta\}$, this implies that the bimodality property is satisfied. In Section III, we show that the bimodality property ensures that in any feasible solution to MSS(d), the auxiliary variables should have no net contribution to the degree-1 moment equation (Proposition III.3), which then implies the soundness of the reduction.

[2] Intuitively, we can think the LHS of (7) (along with the setting $x_1 - y_2 = \beta$) as being a *"derivative operator"*. This explains the fact that we are starting from a cubic polynomial $p_\alpha(\cdot)$ and getting a quadratic polynomial $q_{\alpha,\beta}(\cdot)$. This intuition was also used (twice) in (6), and will be again used in (9) and (10) in order to reduce the degree further.

[3] which we will do by inductively upper bounding $|R_i|$.

General finite fields: We remark that as described above, our solution works over the rational numbers, and, by scaling appropriately, over the integers. By taking the integer solution modulo a large prime p (i.e., $p = 2^{\mathsf{poly}(N)}$) the same arguments extend to F_p. Moving to general finite fields $\mathsf{F} = \mathsf{F}_{p^\ell}$, we first observe that system (†) (and thus a PTE system too) has non-constructive solutions of size $O(d)$, which follows from the Weil bound . Our reduction in the proof of Theorem I.2 also extends to general fields $\mathsf{F} = \mathsf{F}_{p^\ell}$, where p is a prime $p = \Omega(d!)$, and $\ell = \mathsf{poly}(N, d!)$. The reduction now uses a representation of field elements in a polynomial basis $\{1, \gamma, \gamma^2, \ldots, \gamma^{\ell-1}\} \subseteq \mathsf{F}$, instead of decimal representations. See the full version for the changes that need to be made to the proof over the integers, as well as for all the missing proofs from this extended abstract.

C. Related Work

A number of fundamental works address the polynomial reconstruction problem in various settings. In particular, Goldreich et al. [19] show that that the polynomial reconstruction problem is NP-complete for univariate polynomials p over large fields. Håstad's celebrated results [20] imply NP-hardness for linear multivariate polynomials over finite fields. Gopalan et al. [21] show NP-hardness for multivariate polynomials of larger degree, over the field F_2.

We note that in general, the polynomial reconstruction problem does not require that the evaluation points are all distinct (i.e., $x_i \neq x_j$ whenever $i \neq j$). This distinction is crucial to the previous results on polynomial reconstruction (eg. [19], [21]). It is this distinction that prevents those results from extending to the setting of Reed-Solomon codes, and to their multivariate generalization, Reed-Muller codes.

On the algorithmic side, efficient algorithms for decoding of Reed-Solomon codes and their variants are well-studied. As previously mentioned, [4], [5] gave the first efficient algorithms in the list-decoding regime. Paravaresh and Vardy [22] and Guruswami and Rudra [23] construct capacity achieving codes based on variants of RS codes. Koetter and Vardy [24] propose soft decision decoders for RS codes. More recently, Rudra and Wooters [25] prove polynomial list-bounds for random RS codes.

A related line of work is the study of BDD and of Maximum Likelihood Decoding in general codes, possibly under randomized reductions, and when an unlimited amount of preprocessing of the code is allowed. These problems have been extensively studied under diverse settings, e.g., [26], [27], [28], [29], [30], [31], [6], [32].

II. Preliminaries

We start by recalling the formal definition of the MSS(d) problem.

Definition II.1 (Moments Subset-Sum: MSS(d)). *Given a set $A = \{a_1, \ldots, a_n\}$, $a_i \in \mathsf{F}$, integer t, and $m_1, \ldots, m_d \in \mathsf{F}$, decide if there exists a subset $S \subseteq A$ of size t, satisfying $\sum_{a \in S} a^i = m_i$ for all $i \in [d]$. We call t the size of the MSS(d) instance.*

We next recall the reduction from MSS(d) to RS-BDD(d).

Lemma II.2 ([7]). *MSS(d) is polynomial-time reducible to RS-BDD(d). Moreover, the reduction maps instances of MSS(d) on N numbers and of size t to Reed-Solomon codes of block length $N+1$ and of dimension $t-d+1$. The reduction holds over prime fields F_p where $p = 2^{\mathsf{poly}(N)}$.*

We will use the 1-in-3-SAT problem in which we are given a 3-SAT formula ϕ on n variables and m clauses and are asked to determine if there exists an assignment $z \in \{0,1\}^n$ satisfying exactly one literal in each clause. It is known that this problem is NP-hard even for $m = O(n)$ [33]. We will use $[n]$ to denote the set $\{1, 2, \ldots, n\}$.

III. Reduction from 1-in-3-SAT to MSS(d)

We start proving Theorem I.2 by describing the reduction from from 1-in-3-SAT to MSS(d) and its properties. Henceforth, we denote by 1^ℓ the concatenation of ℓ ones, and we let $(1^\ell)_{10}$ denote the positive integer whose decimal representation is 1^ℓ.

Subset Sum Reduction: We start by recalling the reduction from 1-in-3-SAT to Subset-Sum which will be used in our reduction to MSS(d). In that reduction, each variable $(z_t, \overline{z_t})$, $t \in [n]$ is mapped to 2 integers a'_t (corresponding to z_t) and b'_t (corresponding to $\overline{z_t}$). The integers a'_t and b'_t and the target B have the following decimal representation of length-$(n+m)$:

- The decimal representations of a'_t and b'_t consist of two parts: a variable region consisting of the leftmost n digits and a clause region consisting of the (remaining) rightmost m digits.
- In the variable region, a'_t and b'_t have a 1 at the t-th digit and 0's at the other digits. Denote that by $(a_t)^{'v}$.
- In the clause region, for every $j \in [m]$, a'_t (resp. b'_t) has a 1 at the jth location if z_t (resp. $\overline{z_t}$) appears in clause j, and a 0 otherwise. We denote the clause part of a'_t by $(a_t)^{'c}$.
- We define $a'_t = 10^m a_t^{'v} + a_t^{'c}$. We define b'_t similarly.
- The target B is set to the integer whose decimal representation is the all 1's, i.e., we set $B = 10^m (1^n)_{10} + (1^m)_{10}$.

See Figure 1 for an illustration of the decimal representations. This reduction to Subset-Sum is complete and sound. Indeed given a satisfying assignment to the 3-SAT formula $\phi(z)$, the subset $S = \{a'_t \mid t \in [n], z_t = 1\} \cup \{b'_t \mid t \in [n], z_t = 0\}$ is seen to satisfy that $\sum\limits_{s \in S} s = \sum\limits_{\substack{t \in [n] \\ z_t = 1}} a'_t + \sum\limits_{\substack{t \in [n] \\ z_t = 0}} b'_t = B$. Conversely, given a subset $S \subseteq \{a'_t, b'_t \mid t \in [n]\}$ such that $\sum\limits_{s \in S} s = B$, a satisfying assignment to $\phi(z)$ is constructed from it by setting $z_i = 1$ if $a'_t \in S$ and 0 otherwise.

Our Reduction from 1-in-3-SAT to MSS(d): An instance of MSS(d) consists of a tuple $\langle A, B_1, \dots, B_d \rangle$. In this reduction, each variable $(z_t, \overline{z_t})$ is mapped to $2^{d+1} - 2$ distinct rationals: $\{a_t\} \cup \{x_{t,i} \mid i \in [2^d - 2]\}$ (corresponding to z_t) and $\{b_t\} \cup \{y_{t,i} \mid i \in [2^d - 2]\}$ (corresponding to $\overline{z}_t$). Let $\{a'_t, b'_t : t \in [n]\}$ be the integers produced by the above reduction to Subset-Sum. We denote by $a_t^{'v}$ (resp. $a_t^{'c}$) the variable (resp. clause) region of a'_t. Let ν be a natural number to be specified later on. Define:

$$\begin{aligned} a_t &:= 10^{\nu}(10^m a_t^{'v} + a_t^{'c}) \quad \text{and,} \\ b_t &:= 10^{\nu}(10^m b_t^{'v} + b_t^{'c}). \end{aligned} \tag{13}$$

For each $t \in [n]$, we will explicitly construct two sets of $2^d - 2$ *auxiliary variables*, $X_t = \{x_{t,i} \mid i \in [2^d - 2]\}$ and $Y_t = \{y_{t,i} \mid i \in [2^d - 2]\}$ which satisfy the following properties:

Property (1): $\sum_{x \in X_t} x = \sum_{y \in Y_t} y = 0$.

Property (2): $\sum_{x \in X_t} x^k - \sum_{y \in Y_t} y^k = b_t^k - a_t^k$ for every $k \in \{2, \dots, d\}$.

Property (3): For any subset $S \subseteq \bigcup_{t \in [n]} (X_t \cup Y_t)$, either $\left|\sum_{s \in S} s\right| > 10^{m+2n+\nu}$ or $\left|\sum_{s \in S} s\right| < 10^{\nu}$.

Property (4): Every rational number of $\bigcup_{t \in [n]} (X_t \cup Y_t)$ can be written as a fraction whose numerator and denominator are integers of magnitudes at most $10^{\mathsf{poly}(n,d!)}$. Moreover, $\left|\bigcup_{t \in [n]} (X_t \cup Y_t)\right| = n \cdot (2^{d+1} - 4)$.

Properties (1) and (2) will be used to ensure completeness, Property (3) will be used to ensure soundness, and Property (4) will guarantee the polynomial running-time. Constructing such auxiliary variables forms the crux of the reduction.

Define the set $A = \bigcup_{t \in [n]} (\{a_t\} \cup \{b_t\} \cup X_t \cup Y_t)$. We will observe that $|A| = n(2^{d+1} - 2)$ by showing that all the variables $\{a_t\}, \{b_t\}$ and those in X_t and Y_t for $t \in [n]$ are distinct.

Let $N = |A| = n(2^{d+1} - 2)$. The targets $B_1, \dots, B_d$ are defined as follows:

$$\begin{aligned} B_1 &:= 10^{\nu}(10^m (1^n)_{10} + (1^m)_{10}), \\ B_k &:= \sum_{t=1}^{n} a_t^k + \sum_{t=1}^{n} \sum_{x \in X_t} x^k \text{ for every } k \in \{2, \dots, d\}. \end{aligned} \tag{14}$$

Note that a_t (and b_t and B_1, respectively) defined above are obtained by inserting ν zeros to the right of the decimal representation of a'_t (resp. b'_t and B). Therefore, $a_t = 10^{\nu} \cdot a'_t$. Similarly, $b_t = 10^{\nu} \cdot b'_t$ and $B_1 = 10^{\nu} \cdot B$ (see Figure 2 for a pictorial illustration). The following fact is immediate from the definitions,

Fact III.1. *For any $x \in \{a_t, b_t \mid t \in [n]\} \cup B_1$, we have*

$$10^{\nu} < |x| < 10^{m+n+\nu+1}$$

In Section III-A, we will show how to construct variables satisfying Properties (1), (2), (3) and (4). The proof of Theorem I.2 will follow from the next lemma and Property (4). The proof of Theorem I.1 will then follow from Theorem I.2 and Lemma II.2.

Lemma III.2. *(Main) There exists a satisfying assignment to a 3-SAT instance $\phi(z_1, \dots, z_n)$ if and only if there exists a subset $S \subseteq A$ of size $|S| = n(2^d - 1)$ such that for every $k \in [d]$,*

$$\sum_{s \in S} s^k = B_k.$$

Proof of Theorem I.2: Recall that $N = n(2^{d+1} - 2)$, and so $|S| = |A|/2 = N/2$. From Property (4) above, we know that every element constructed in the instance of MSS(d) has $\mathsf{poly}(n, d!)$ digit representation. Therefore, for $d = O(\log n / \log\log n)$, the reduction runs in $\mathsf{poly}(n)$ time.

The NP-hardness of MSS(d) for $d \leq O(\log N / \log\log N)$ (under polynomial-time reductions) and for $d < c \log N$ (under quasipolynomial time reductions, and with $c > 0$ being a sufficiently small absolute constant) then follows from Lemma III.2. ■

Proof of Theorem I.1: By Property (4) above, we deduce the same hardness results for MSS(d) over prime fields of size $2^{\mathsf{poly}(N)}$. This – along with Lemma II.2 – imply Theorem I.1. ■

We now prove Lemma III.2.

Proof of Lemma III.2: We start by proving the completeness of our reduction. We show that given a satisfying assignment z to the 3-SAT instance $\phi(z_1, \dots, z_n)$, there exists a subset $S \subseteq A$ such that for every $k \in [d]$,

$$\sum_{s \in S} s^k = B_k.$$

Consider the following subset S of variables:

$$S \triangleq \bigcup_{t \in [n], z_t = 1} \{a_t\} \bigcup_{t \in [n], z_t = 1} X_t \bigcup_{t \in [n], z_t = 0} \{b_t\} \bigcup_{t \in [n], z_t = 0} Y_t.$$

Note that $|S| = n(2^d - 1) = \frac{N}{2}$ since the number of auxiliary variables included in S corresponding to each $t \in [n]$ is exactly $2^d - 2$.

For every $k \in [d]$, we have that

$$\sum_{s \in S} s^k = \sum_{\substack{t \in [n] \\ z_t = 1}} \left(a_t^k + \sum_{x \in X_t} x^k \right) + \sum_{\substack{t \in [n] \\ z_t = 0}} \left(b_t^k + \sum_{y \in Y_t} y^k \right) \tag{15}$$

By Property (2) of the auxiliary variables, we have that for any $t \in [n]$ and any $k \in \{2, 3, \ldots, d\}$,

$$\sum_{x\in X_t} x^k - \sum_{y\in Y_t} y^k = b_t^k - a_t^k.$$

Summing this equation over all $t \in [n]$, such that $z_t = 0$, we get

$$\sum_{\substack{t\in[n]\\ z_t=0}} \left(b_t^k + \sum_{y\in Y_t} y^k \right) = \sum_{\substack{t\in[n]\\ z_t=0}} \left(a_t^k + \sum_{x\in X_t} x^k \right) \qquad (16)$$

From 15 and 16, we conclude that for every $k \in \{2, 3, \ldots, d\}$,

$$\sum_{s\in S} s^k = \sum_{t=1}^{n} \left(a_t^k + \sum_{x\in X_t} x^k \right) = B_k$$

For $k = 1$, Property (1) implies that for every $t \in [n]$, $\sum\limits_{x\in X_t} x = 0$ and $\sum\limits_{y\in Y_t} y = 0$. Therefore,

$$\sum_{s\in S} s = \sum_{\substack{t\in[n]\\ z_t=1}} a_t + \sum_{\substack{t\in[n]\\ z_t=0}} b_t \qquad (17)$$

Recall the variables a'_t, b'_t and B from the Subset Sum reduction defined at the beginning of the proof. Note that $(\sum\limits_{\substack{t\in[n]\\ z_t=1}} a'_t + \sum\limits_{\substack{t\in[n]\\ z_t=0}} b'_t) = B$. Therefore, we can rewrite Equation (17) as:

$$\sum_{s\in S} s = 10^{\nu} \cdot \left(\sum_{\substack{t\in[n]\\ z_t=1}} a'_t + \sum_{\substack{t\in[n]\\ z_t=0}} b'_t \right) = 10^{\nu} \cdot B = B_1.$$

We now prove the soundness of our reduction. Let S be a solution to the MSS(d) instance. That is, $S \subseteq A$ is such that $\sum\limits_{s\in S} s^k = B_k$ for every $k \in [d]$. Proposition III.3 – which is stated below – shows that the auxiliary variables in S should sum to 0. Therefore, there exists a subset $S' \subseteq \{a_t, b_t \mid t \in [n]\}$ such that $\sum\limits_{s\in S'} s = B_1$. By definition of a_t, b_t and B_1, it follows that there exists a subset of $\{a'_t, b'_t \mid t \in [n]\}$ which sums to B, and the soundness of our reduction then follows from the soundness of the Subset Sum reduction.

Proposition III.3. *Let $S \subseteq A$ be such that $\sum\limits_{s\in S} s = B_1$. Let $D = \bigcup\limits_{t\in[n]} (X_t \cup Y_t)$ be the set of all the auxiliary variables. Then,*

$$\sum_{y\in S\cap D} y = 0.$$

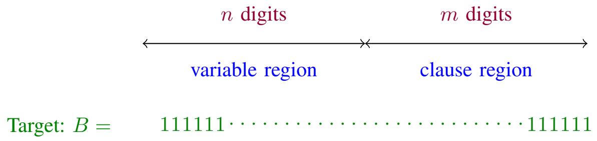

Figure 1. Decimal representations in the original reduction from 1-in-3-SAT to Subset-Sum.

Proof of Proposition III.3: Since $\sum\limits_{s\in S} s = B_1$, we have that

$$\sum_{y\in S\cap D} y + \sum_{s\in S\setminus D} s = B_1.$$

Note that $S \setminus D \subseteq \{a_t, b_t \mid t \in [n]\}$. Since the ν least significant digits of B_1 and those of each element of $S \setminus D$ are all equal to 0, either $\left| B_1 - \sum\limits_{s\in S\setminus D} s \right| = 0$ or $\left| B_1 - \sum\limits_{s\in S\setminus D} s \right| > 10^{\nu}$. If $\left| B_1 - \sum\limits_{s\in S\setminus D} s \right| = 0$, then we are done. Henceforth, we assume that $\left| B_1 - \sum\limits_{s\in S\setminus D} s \right| > 10^{\nu}$. By Fact III.1, the elements of $S \setminus D$ as well as B_1 all have magnitudes at most $10^{m+n+\nu+1}$. Therefore, $\left| B_1 - \sum\limits_{s\in S\setminus D} s \right| \leq (2n+1) \cdot 10^{m+n+\nu+1} < 10^{m+2n+\nu}$. On the other hand, by Property (3) of the auxiliary variables, we know that either $\left| \sum\limits_{y\in S\cap D} y \right| > 10^{m+2n+\nu}$ or $\left| \sum\limits_{y\in S\cap D} y \right| < 10^{\nu}$. Since $\left| \sum\limits_{y\in S\cap D} y \right| = \left| B_1 - \sum\limits_{s\in S\setminus D} s \right|$, we get a contradiction. Therefore, $\sum\limits_{y\in S\cap D} y = 0$.

■

■

A. *Constructing the auxiliary variables X_t, Y_t*

We now show how to construct the auxiliary variables, starting from the a_t, b_t variables described before, for every $t \in [n]$. We do so in Algorithm 1, the AUXILIARYVARIABLEGENERATOR. For every $t \in [n]$, we construct $2(2^d - 2)$ distinct auxiliary variables which satisfy the Properties 1, 2, 3 and 4 stated above. The AUXILIARYVARIABLEGENERATOR outputs the union of the variables generated in Algorithm 2, the ATOMICSOLVER, using the recursive coupling idea described in Section I-B. We use $\mathtt{1}^{\ell}$ (and $\mathtt{0}^{\ell}$) to denote a column vector of ℓ 1's (0's) respectively. For any vector v, let v^T denote its transpose.

```
Algorithm 1: AuxiliaryVariableGenerator:
Input: $\bigcup_{t\in[n]} \{a_t, b_t\}$
Output: Sets of auxiliary variables $X_t, Y_t$ for every $t \in [n]$.
for $t \in [n]$ do
    $X_t = \emptyset$
    $Y_t = \emptyset$
    for $i \in \{2, \dots, d\}$ do
        if $i = 2$ then
            $R_{t,i} = b_t^2 - a_t^2$
        else
            $R_{t,i} = (b_t^i - a_t^i) + \sum_{y\in Y_t} y^i - \sum_{x\in X_t} x^i$
        end if
        Let $\{x_{t,i,j} \mid j \in [2^{i-1}]\} \bigcup \{y_{t,i,j} \mid j \in [2^{i-1}]\}$
        =AtomicSolver($t$,$i$, $R_{t,i}$)
        Let $X_t = X_t \bigcup \{x_{t,i,j} \mid j \in [2^{i-1}]\}$ and
        $Y_t = Y_t \bigcup \{y_{t,i,j} \mid j \in [2^{i-1}]\}$
    end for
end for
```

We now give the details of AtomicSolver(t,i, $R_{t,i}$) for any $t \in [n]$ and $i \in \{2, 3, \dots, d\}$. Let $\nu = n^2$, and $M = m + \nu + n + 1$. For every $t \in [n], i \in \{2, 3, \dots, d\}$ and $r \in [i]$, we define the functions $f(t,i) := (i-1)! \cdot \nu_t$ and $g(t,i,r) := (t-1)d^2 + (i-1)i + r$, where ν_t is the t^{th} prime integer greater than n^4. Note that $M = O(n^3)$ and $10^M > B_1$, by Fact III.1. We will use the fact that ν_t is much larger than M later. Using the Prime Number Theorem [34], it follows that the number of primes in the interval $[n^4, n^5]$ is larger than n, and thus $\nu_n < n^5$. Moreover, these n primes can be found in deterministic polynomial time [35].

We will implement the recursive coupling idea of the AtomicSolver described in Section I-B, in terms of matrix algebra. For example, recall that in the first step of the variable coupling, we set $x_1 - y_1 = \alpha$, $y_2 - x_2 = \alpha$ and $x_1 - y_2 = \beta$. We can then express x_1, x_2, y_1, y_2 as a linear combination of α, β, where we use the extra degree of freedom to choose $x_1 = -x_2$, as follows: $(x_1, x_2)^T = \frac{1}{2}\begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \cdot (\alpha, \beta)^T$, and $(y_1, y_2)^T = \frac{1}{2}\begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \cdot (\alpha, \beta)^T$. In general, the polynomial equations give rise to $2^i - 1$ linear constraints on 2^i unknowns $(x_1, \cdots, x_{2^{i-1}}, y_1, \cdots, y_{2^{i-1}})$. The extra degree of freedom allows us to preserve the symmetry of the solution, which enables us to describe the algorithm and its analysis in a clean form.

Acknowledgements

We would like to thank Madhu Sudan for very helpful discussions that led to the proof of existence of inhomogeneous PTE solutions over finite fields. We would also like to thank Venkatesan Guruswami and Swastik Kopparty for helpful comments and conversations. Finally, we would like to thank Andrew Sutherland and Colin Ingalls for helpful correspondence.

```
Algorithm 2: AtomicSolver($t$,$i$, $R_{t,i}$):
Input: $i, t, R_{t,i}$
Output: Set of auxiliary variables,
$\{x_{t,i,j} \mid j \in [2^{i-1}]\} \bigcup \{y_{t,i,j} \mid j \in [2^{i-1}]\}$
Let $\nu_t$ be the $t^{th}$ prime integer greater than $n^4$
Let $f(t,i) = (i-1)! \cdot \nu_t$
Let $g(t,i,r) = (t-1)d^2 + (i-1)i + r$ for all
$1 < r < i$
$\alpha_{t,i,1} = 10^{f(t,i)}$
$\alpha_{t,i,r} = 10^{g(t,i,r)}$ for all $1 < r < i$
$\alpha_{t,i,i} = R_{t,i}/(i! \prod_{r\in[i-1]} \alpha_{t,i,r})$
$\alpha_{t,i} = [\alpha_{t,i,1}, \dots, \alpha_{t,i,i}]^T$
if $i = 2$ then
    $A_2 = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}$ and $B_2 = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$
else
    $A_i = \begin{bmatrix} A_{i-1} & \mathbf{1}^{2^{i-2}} \\ B_{i-1} & -\mathbf{1}^{2^{i-2}} \end{bmatrix}$ and $B_i = \begin{bmatrix} B_{i-1} & \mathbf{1}^{2^{i-2}} \\ A_{i-1} & -\mathbf{1}^{2^{i-2}} \end{bmatrix}$
end if
$[x_{t,i,1}, \dots, x_{t,i,2^{i-1}}]^T = \frac{1}{2} \cdot A_i \cdot \alpha_{t,i}$
$[y_{t,i,1}, \dots, y_{t,i,2^{i-1}}]^T = \frac{1}{2} \cdot B_i \cdot \alpha_{t,i}$
Return $\{x_{t,i,j} \mid j \in [2^{i-1}]\} \bigcup \{y_{t,i,j} \mid j \in [2^{i-1}]\}$
```

References

[1] I. S. Reed and G. Solomon, “Polynomial codes over certain finite fields,” *Journal of the society for industrial and applied mathematics*, vol. 8, no. 2, pp. 300–304, 1960.

[2] W. W. Peterson, “Encoding and error-correction procedures for the Bose-Chaudhuri codes,” *IRE Transactions on Information Theory*, vol. 6, no. 4, pp. 459–470, 1960. [Online]. Available: http://dx.doi.org/10.1109/TIT.1960.1057586

[3] E. Berlekamp and L. Welch, “Error correction for algebraic block codes,” 1986, uS Patent 4,633,470. [Online]. Available: http://www.google.com/patents/US4633470

[4] M. Sudan, “Decoding of Reed Solomon codes beyond the error-correction bound,” *J. Complexity*, vol. 13, no. 1, pp. 180–193, 1997. [Online]. Available: http://dx.doi.org/10.1006/jcom.1997.0439

[5] V. Guruswami and M. Sudan, “Improved decoding of Reed-Solomon and algebraic-geometry codes,” *IEEE Trans. Inf. Theory*, vol. 45, no. 6, pp. 1757–1767, 1999. [Online]. Available: http://dx.doi.org/10.1109/18.782097

[6] V. Guruswami and A. Vardy, “Maximum-likelihood decoding of Reed-Solomon codes is NP-hard,” *IEEE Trans. Inf. Theory*, vol. 51, no. 7, pp. 2249–2256, 2005. [Online]. Available: http://dx.doi.org/10.1109/TIT.2005.850102

[7] V. Gandikota, B. Ghazi, and E. Grigorescu, "On the NP-hardness of bounded distance decoding of Reed-Solomon codes," in *Information Theory (ISIT), 2015 IEEE International Symposium on*. IEEE, 2015, pp. 2904–2908.

[8] Q. Cheng and D. Wan, "On the list and bounded distance decodability of Reed-Solomon codes," *SIAM J. Comput.*, vol. 37, no. 1, pp. 195–209, 2007. [Online]. Available: http://dx.doi.org/10.1137/S0097539705447335

[9] ——, "Complexity of decoding positive-rate primitive Reed-Solomon codes," *IEEE Trans. Inf. Theory*, vol. 56, no. 10, pp. 5217–5222, 2010. [Online]. Available: http://dx.doi.org/10.1109/TIT.2010.2060234

[10] J. B. Lasserre, *Moments, positive polynomials and their applications*. World Scientific, 2009, vol. 1.

[11] E. Prouhet, "Mémoire sur quelques relations entre les puissances des nombres," *CR Acad. Sci. Paris*, vol. 33, no. 225, p. 1851, 1851.

[12] L. E. Dickson, *History of the Theory of Numbers, Volume II: Diophantine Analysis*. Courier Corporation, 2013, vol. 2.

[13] E. M. Wright, "Prouhet's 1851 solution of the Tarry-Escott problem of 1910," *The American Mathematical Monthly*, vol. 66, no. 3, pp. 199–201, 1959.

[14] G. H. Hardy and E. M. Wright, *An introduction to the theory of numbers*, ser. Oxford Science Publications. Oxford: Clarendon Press, 1979.

[15] L. K. Hua, *Introduction to number theory*. Springer, 1982.

[16] P. Erdos and G. Szekeres, "On the product∏ n k= 1 (1- zak), acad," *Serbe Sci. Publ. Inst. Math*, vol. 13, pp. 29–34, 1959.

[17] P. Borwein and C. Ingalls, "The Prouhet-Tarry-Escott problem revisited," *Enseign. Math*, vol. 40, pp. 3–27, 1994.

[18] P. Borwein, P. Lisonek, and C. Percival, "Computational investigations of the Prouhet-Tarry-Escott problem," *Math. Comput.*, vol. 72, no. 244, pp. 2063–2070, 2003.

[19] O. Goldreich, R. Rubinfeld, and M. Sudan, "Learning polynomials with queries: The highly noisy case," *SIAM J. Discrete Math.*, vol. 13, no. 4, pp. 535–570, 2000. [Online]. Available: http://dx.doi.org/10.1137/S0895480198344540

[20] J. Håstad, "Some optimal inapproximability results," *J. ACM*, vol. 48, no. 4, pp. 798–859, 2001.

[21] P. Gopalan, S. Khot, and R. Saket, "Hardness of reconstructing multivariate polynomials over finite fields," *SIAM J. Comput.*, vol. 39, no. 6, pp. 2598–2621, 2010. [Online]. Available: http://dx.doi.org/10.1137/070705258

[22] F. Parvaresh and A. Vardy, "Correcting errors beyond the guruswami-sudan radius in polynomial time," in *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, 2005, pp. 285–294. [Online]. Available: http://dx.doi.org/10.1109/SFCS.2005.29

[23] V. Guruswami and A. Rudra, "Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy," *IEEE Transactions on Information Theory*, vol. 54, no. 1, pp. 135–150, 2008. [Online]. Available: http://dx.doi.org/10.1109/TIT.2007.911222

[24] R. Koetter and A. Vardy, "Algebraic soft-decision decoding of Reed-Solomon codes," *IEEE Transactions on Information Theory*, vol. 49, no. 11, pp. 2809–2825, 2003. [Online]. Available: http://dx.doi.org/10.1109/TIT.2003.819332

[25] A. Rudra and M. Wootters, "Every list-decodable code for high noise has abundant near-optimal rate puncturings," in *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, 2014, pp. 764–773. [Online]. Available: http://doi.acm.org/10.1145/2591796.2591797

[26] A. Vardy, "Algorithmic complexity in coding theory and the minimum distance problem," in *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, 1997, pp. 92–109.

[27] S. Arora, L. Babai, J. Stern, and Z. Sweedyk, "The hardness of approximate optima in lattices, codes, and systems of linear equations," *J. Comput. Syst. Sci.*, vol. 54, no. 2, pp. 317–331, 1997.

[28] I. Dinur, G. Kindler, R. Raz, and S. Safra, "Approximating CVP to within almost-polynomial factors is NP-hard," *Combinatorica*, vol. 23, no. 2, pp. 205–243, 2003.

[29] I. Dumer, D. Micciancio, and M. Sudan, "Hardness of approximating the minimum distance of a linear code," *IEEE Trans. Information Theory*, vol. 49, no. 1, pp. 22–37, 2003.

[30] U. Feige and D. Micciancio, "The inapproximability of lattice and coding problems with preprocessing," *J. Comput. Syst. Sci.*, vol. 69, no. 1, pp. 45–67, 2004.

[31] O. Regev, "Improved inapproximability of lattice and coding problems with preprocessing," *IEEE Trans. Information Theory*, vol. 50, no. 9, pp. 2031–2037, 2004.

[32] Q. Cheng, "Hard problems of algebraic geometry codes," *IEEE Trans. Information Theory*, vol. 54, no. 1, pp. 402–406, 2008.

[33] T. J. Schaefer, "The complexity of satisfiability problems," in *STOC*. ACM, 1978, pp. 216–226.

[34] V. Shoup, *A computational introduction to number theory and algebra*. Cambridge university press, 2009.

[35] M. Agrawal, N. Kayal, and N. Saxena, "PRIMES is in P," *Annals of mathematics*, pp. 781–793, 2004.

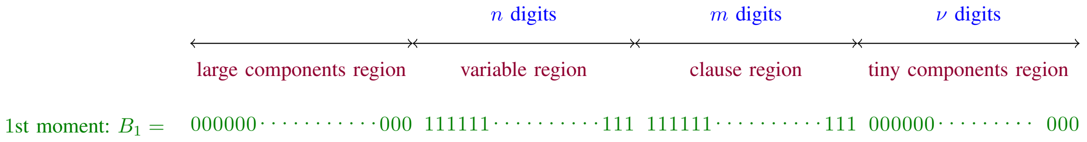
n digits
m digits
ν digits
large components region
variable region
clause region
tiny components region
1st moment: $B_1 =$
$000000\cdots\cdots\cdots\cdots 000$ $111111\cdots\cdots\cdots\cdots 111$ $111111\cdots\cdots\cdots\cdots 111$ $000000\cdots\cdots\cdots\cdots 000$

Figure 2. Decimal representations in the reduction from 1-in-3-SAT to MSS(d). The "large components region" only contains zeros in $\{a_t, b_t : t \in [n]\}$ but contains non-zeros in $\{|x_{t,i}|, |y_{t,i}| : t \in [n], i \in [2^d - 2]\}$.

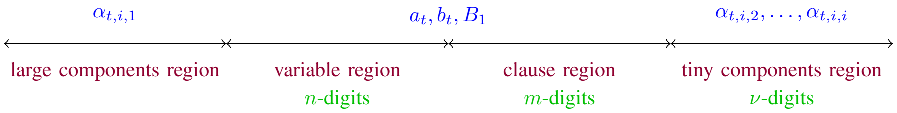

Figure 3. Relative distribution of $\alpha_{t,i,r}$ for any $i \in \{2, \cdots, d\}$ with respect to a_t, b_t and B_1.