Reduce and Prange: Revisiting Prange's ISD for Solving LPN/RSD over Large Fields

Jiseung Kim[1] and Changmin Lee[2]

[1] Jeonbuk National University, Jeonju, Rep. of Korea
jiseungkim@jbnu.ac.kr
[2] Korea Institute for Advanced Study, Seoul, Rep. of Korea
changminlee@kias.re.kr

Abstract. The Learning Parity with Noise (LPN) problem and its specific form, LPN with regularity, are widely utilized in crafting cryptographic primitives. Recently, extending these problems to operate over larger fields has been considered to enhance applications. Despite the broad analysis available for traditional LPN approaches, the exploration of LPN over large fields remains underdeveloped. This gap in research has led to the development of improved attacks, suggesting that primitives based on LPN over large fields may not meet necessary security standards.

We have developed an algorithm that enhances the efficiency of solving the LPN over large fields. This method innovatively modifies the Gaussian elimination attack, traditionally known as Prange's information set decoding algorithm. Our key advancement involves the selective use of partial Gaussian elimination rather than employing the full Gaussian algorithm throughout, which we have termed the "Reduce and Prange's (RP) algorithm." Additionally, we apply the RP algorithm to address the LPN problem over large fields with regular noise. Our RP highlights two key aspects: the vulnerability of existing schemes and the superiority of our approach compared to recent analyses.

Our findings reveal that cryptographic applications, including Syndrome Decoding in the Head frameworks (Crypto'22, Asiacrypt'23, Eurocrypt'24) and Scalable Multiparty Garbling (CCS'23), are not secure enough. To be precise, the schemes fail to achieve their intended bit-security.

Compared to the previous analysis by Liu et al. (Eurocrypt'24), we show that for LPN over large fields (e.g., 128-bit field size), the bit-security is reduced by 5-11 bits . Furthermore, our RP algorithm for solving LPN with regular noise outperforms recent results by Liu et al., Briaud, and Øygard (Eurocrypt'23) under certain parameter choices, leading to a reduction in bit-security by 5-20 bits for LPN with regular noise.

Keywords: LPN (over large fields), LPN with regular noise, Concrete security

1 Introduction

As the central problem of learning theory and coding theory, the learning parity with noise (LPN) problem has affected numerous cryptographic primitives such

as secure arithmetic computations [7, 9, 17, 19, 20, 28, 36, 41, 44, 56, 62, 70], zero knowledge proofs [10, 32, 39, 66] and more [4–6, 14, 25, 29, 35, 45, 46, 48, 51, 71, 72]. The LPN problem is defined as follows.

Problem 1 (Learning Parity with Noise (LPN)). Let m, n, t be positive integers and $\mathcal{R}$ be a ring. Let $\mathcal{C}$ be a probabilistic code generation algorithm such that $\mathcal{C}(m, n, \mathcal{R})$ returns a matrix $\mathbf{A} \in \mathcal{R}^{m \times n}$. Let $\chi(\mathcal{R}) = \{\chi_{m,t}\}_{m,t \in \mathbb{N}}(\mathcal{R})$ be a family of distributions over $\mathcal{R}^m$ that returns a vector in $\mathcal{R}^m$, where the number of nonzero coefficients in the vector is t.

The computational Learning Parity with Noise (LPN) problem with respect to parameters m, n and t involves obtaining a secret vector $\mathbf{s}$ given instances

$$(\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod \mathcal{R})$$

where $\mathbf{A} \leftarrow \mathcal{C}(m, n, \mathcal{R}), \mathbf{s} \leftarrow \mathcal{R}^n$ and $\mathbf{e} \leftarrow \chi_{m,t}(\mathcal{R})$. We simply say the problem (m, n, t)-LPN problem over $\mathcal{R}$.[1]

To the best of our knowledge, $\mathbb{F}_2, \mathbb{F}_q$ and ring $\mathbb{Z}_{2^\lambda}$ for some $\lambda > 1$ has been primarily considered. Recently, [50] proved that (m, n, t)-LPN over $\mathbb{Z}_{2^\lambda}$ can be reduced to $(m, n, t/2)$-LPN over $\mathbb{F}_2$, which suffices to analyze the security of LPN over both fields $\mathbb{F}_2$ and $\mathbb{F}_q$, respectively. If $\mathcal{R} = \mathbb{F}_2$, then it is called the standard LPN problem or simply LPN. If $\mathcal{R} = \mathbb{F}_q$ with a prime power $q \gg 2$, then we call it 'LPN over large fields', where it is a natural variant of the (standard) LPN. On the other hand, for post-quantum signatures, LPN over $\mathbb{F}_{2^8}$ is usually considered [1–3, 39].

The PCG-based primitives sometimes assume the regularity of the error distribution, which yields significant improvements in the design of efficient cryptographic primitives with practical applications [9, 10, 18, 20, 22, 28, 30, 31, 33, 34, 43, 47, 57, 60–62, 64–70]. Such a variant of LPN is called, LPN with regular noise (for short regular-LPN) defined as below. Even more, these protocols additionally employ a low-noise setting that $t/m = 1/n^\varepsilon$ for some constant $\varepsilon > 0$ with bounded samples.

Problem 2 (LPN with regular noise, regular-LPN). Let m, n, t, β be positive integers with $m = t \cdot \beta$ and $\mathcal{R}$ be a ring. Let $\mathcal{C}$ be a probabilistic code generation algorithm such that $\mathcal{C}(m, n, \mathcal{R})$ returns a matrix $\mathbf{A} \in \mathcal{R}^{m \times n}$. Let $\tau(\mathcal{R}) = \{\tau_{m,t}\}_{m,t \in \mathbb{N}}(\mathcal{R})$ be a family of distributions over $\mathcal{R}^m$. Given m, t, a distribution $\tau_{m,t}$ returns a vector $\mathbf{e} = (\mathbf{e}_1 \| \dots \| \mathbf{e}_t) \in \mathcal{R}^m$ such that $\mathbf{e}_i \in \mathcal{R}^\beta$ is of Hamming weight $|\mathbf{e}_i| = 1$ for $1 \le i \le t$.

[1] Following the previous work [19, 20, 50], we adapt the definition for our purpose. We note that the definition of LPN originally states that the goal is to find $\mathbf{s}$ using oracle access to $\mathcal{O}$, which returns a LPN instance. To be specific, we only consider an LPN where the number of oracle queries are limited to m. (The problem can be considered as a variant of decoding linear code.) This constraint is reasonable because most cryptographic primitives built on LPN use the parameter regime where the number of oracle queries are bounded.

The computational LPN with regular noise (regular-LPN) involves obtaining a secret vector $\mathbf{s}$ given instances

$$(\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod \mathcal{R})$$

where $\mathbf{A} \leftarrow \mathcal{C}(m, n, \mathcal{R}), \mathbf{s} \leftarrow \mathcal{R}^n$ and $\mathbf{e} \leftarrow \tau_{m,t}(\mathcal{R})$. We simply say the problem (m, n, t)-regular-LPN problem over $\mathcal{R}$.

Note that LPN instances of the form $(\mathbf{A}, \mathbf{b}) = (\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod \mathcal{R})$ can be easily transformed into a problem of finding $\mathbf{e}$ when there exists a matrix $\mathbf{H}$ such that $\mathbf{H} \cdot \mathbf{e} = \mathbf{H} \cdot \mathbf{b}$, and vice versa, as demonstrated in [55, Lemma 4.9]. Therefore, we can restrict our attention to $(\mathbf{A}, \mathbf{b}) = (\mathbf{A}, \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod \mathcal{R})$. In particular, this paper mainly focuses on $\mathcal{R} = \mathbb{F}_q$ with $q \gg 2$.

Concrete Analysis for LPN over large fields. While various applications have been developed using LPN over large fields, the focus of cryptanalysis has predominantly been on standard LPN. Accurately estimating the security level of cryptographic primitives is critical for assessing their robustness.

While the Blum-Kalai-Wasserman (BKW) algorithm [15] is highly effective in solving standard LPN problems, its efficiency diminishes when applied to LPN over large fields, particularly when q significantly exceeds 2. This discrepancy arises because the BKW algorithm is optimized for scenarios where the field size is smaller, and it relies on a subexponential number of samples for optimal performance. As a result, its application to larger fields like $\mathbb{F}_q$ is not as effective.

Boyle et al. [17] initiates a study of a concrete analysis of LPN over $\mathbb{F}_q$ in order to instantiate PCG-based protocols.The subsequent work by Liu et al. [50] shows that the analysis of [17] is oversimplified, which yields the time cost of actual attacks are overestimated. Moreover, they demonstrate that to solve LPN over $\mathbb{F}_2$ and $\mathbb{Z}_{2^\lambda}$, advanced algorithms such as [12, 16, 52, 53] yield the most effective results, as expected. Nevertheless, when these techniques are applied to solving LPN over $\mathbb{F}_q$ with large q, their performance, which is considerable for solving LPN over $\mathbb{F}_2$, diminishes as the value of q increases.

Concrete Analysis for regular-LPN (over large fields). In the case of regular-LPN, the work proposed by [21] describes a novel algebraic attack that exploits regular noise distributions on $\mathbb{F}_q$ for any $q \geq 2$. Specifically, the regular noise distribution is used to generate multivariate polynomials, which can reduce the concrete security of regular-LPN. Subsequently, Esser and Santini [38] modified the well-known algorithms for solving LPN into specified algorithms for solving regular-LPN on $\mathbb{F}_2$. They mainly exploit the specific structure derived from the regular distribution of the error vector. These results demonstrate the importance of addressing the regular noise distribution in LPN. Unfortunately, similar to a LPN case, if $\mathcal{R} = \mathbb{F}_q$ with large q, algorithms in [38] are not suitable for solving regular-LPN, except for the permutation-based algorithm.[2]

[2] This permutation-based algorithm is identical to a Prange's ISD algorithm except for the guessing probability induced by regular noise assumption.

These two results have in common that, despite powerful applications in the case of LPN over large fields, more advanced analysis algorithms are still lacking. According to [21,24,38,50,54,58], when the field size is sufficiently large, Prange's Information Set Decoding (ISD) algorithm has a comparable cost to other algorithms.

1.1 Our Contribution

In this paper, we revisit Prange's ISD algorithm to improve its performance by reducing the cost of the Gaussian elimination part. We named the modified Prange's algorithm as Reduce and Prange algorithm (RP). As a natural application of RP algorithm, we further provide an algorithm: RP for specifying regular-LPN, so-called regular-RP. We implement a SageMath [63] script to search for near-optimal parameterization for our attacks. The main impact of our algorithm is then two-fold:

Direct Impacts on Primitives. Our findings demonstrate that recent applications based on LPN over large fields—referenced in [2,3,11,39,42] are not secure enough. Specifically, while these applications are typically designed to provide 80/128-bit security, our algorithm reveals that their actual security falls below this threshold. This result is represented in Table 1.

Table 1. Target schemes and parameters.

Scheme	Parameters				Previous	RP
	$\mathbb{F}_q$	m	n	t		
SDitH [2,3,39][3]	$\mathbb{F}_{2^8}$	256	128	80	128	115
Scalable Multiparty Garbling [11]	[4]	555	127	139	80	73
		785	214	197	128	119
[42]	$\mathbb{F}_{2^8}$	220	101	90	128	125
	$\mathbb{F}_{2^9}$	207	93	90	128	126
	$\mathbb{F}_{2^{10}}$	196	92	84	128	124

Implicit Impacts on PCG-like protocols. Recently, Liu et al. [50] provides a bit-security estimation for solving LPN over $\mathbb{F}_q$, especially focusing on PCG protocols. They re-calculate all parameters in [17] and reproduce more accurate security estimation to tightly choose parameters. Furthermore, based on their

[3] In case of [39], the parameter is used in a variant 3 scheme. Other schemes are based on the hardness of LPN over $\mathbb{F}_2$.

[4] [11] does not provide an exact field size, but explicitly mentioned that their scheme is based on the hardness of LPN over large field.

estimation, they also provide recommendation parameters for PCG applications to achieve 128-bit security.

By applying RP to such parameters, we demonstrate that RP can surpass existing methods for solving the LPN problem over large fields in PCG parameter settings. The estimated results for various parameters are given by Table 2. Indeed, the bit-security of LPN over $\mathbb{F}_{2^{128}}$ is reduced by 5-11 bits when $\log q = 128$ compared to [50].

For regular-LPN over large fields, the results show that our algorithm outperforms the recent work proposed in [21,50] for certain parameter settings with relatively small n.[5] Specifically, the bit-security is reduced by 5-20 bits for small (m, n) and $\log q = 128$. However, when t/n is sufficiently small, the approach presented in [21] outperforms regular-RP. For detailed numerical results on solving LPN with regular noise, please refer to Table 4.

Moreover, for recommended parameters to achieve 128-bit security in [50], we provide numerical results for our RP and regular-RP algorithms. For LPN defined over $\mathbb{F}_q$ with $\log q = 128$, our algorithm outperforms the previous works and reduces the bit-security by 5-9 bits. The regular-RP can reduce the bit-security of regular-LPN by 8-11 bits for small m. However for large m, it becomes worse quickly. We refer to Table 3 and Table 5, respectively.

1.2 Technical Idea

We first recall Prange's Information Set Decoding (ISD) algorithm to describe our idea. To this end, we introduce some notations: Given any vector $\mathbf{x} \in \mathbb{F}_q^m$, $|\mathbf{x}|$ is denoted by the Hamming weights of $\mathbf{x}$. Let $(\mathbf{A}, \mathbf{b}) \in \mathbb{F}_q^{m\times n} \times \mathbb{F}_q^m$ be LPN instances and $\mathcal{I} \subset [m]$ an index set of size n. Then, we can obtain $\mathbf{A}_\mathcal{I}$ (Resp. $\mathbf{b}_\mathcal{I}$) by a submatrix of $\mathbf{A}$ (Resp. subvector of $\mathbf{b}$) by collecting i-th row of $\mathbf{A}$ (Resp. i-th coordinate of $\mathbf{b}$) for $i \in \mathcal{I}$.

Under notations, Prange's Information Set Decoding algorithm is given in Algorithm 1.

Algorithm 1 Prange's ISD

Input: (m, n, t)-LPN over $\mathbb{F}_q$ instances $(\mathbf{A}, \mathbf{b}) \in \mathbb{F}_q^{m\times n} \times \mathbb{F}_q^m$.
Output: The secret $\mathbf{s}$ such that $|\mathbf{b} - \mathbf{A} \cdot \mathbf{s} \bmod \mathbb{F}_q| = t$.

1: **repeat**
2: Choose random index subsets $\mathcal{I} \subset [m]$ of size n
3: Generate a submatrix $(\mathbf{A}_\mathcal{I}, \mathbf{b}_\mathcal{I})$ collecting i-th row of $\mathbf{A}$ (Resp. $\mathbf{b}$) for $i \in \mathcal{I}$
4: **until** $\mathbf{A}_\mathcal{I}$ is invertible and $|\mathbf{b} - \mathbf{A} \cdot \mathbf{s} \bmod \mathbb{F}_q| = t$ where $\mathbf{s} = \mathbf{A}_\mathcal{I}^{-1} \cdot \mathbf{b}_\mathcal{I} \bmod \mathbb{F}_q$
5: **return** $(\mathbf{s}, \mathcal{I})$

It is clear that Prange's algorithm terminates in $\mathcal{G}/\mathcal{P}$, where $\mathcal{P}$ denotes the probability of collecting n error-free instances (i.e., $b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle$) and $\mathcal{G}$ denotes

[5] Note that [38] mainly focuses on how to efficiently solve regular-LPN over the binary field. Indeed, their algorithms are not suit for regular-LPN over large fields.

time cost of the Gaussian elimination algorithm. Although the Gaussian elimination operation has a time complexity of only $O(n^\omega)$ for the linear algebra constant ω, where it is typically disregarded in the context of asymptotic analysis, its concrete time complexity is of significant consequence in estimating the bit-security. For instance, we consider the $(1024, 652, 57)$-LPN over $\mathbb{F}_q$. According to [50], the bit-size of the total cost associated with this problem is reported to be 111, with the Gaussian elimination accounting for a bit-size of 23 when we take $O(n^\omega) = n^{2.8}$. This highlights that the Gaussian elimination contributes to approximately one-fifth of the total complexity exponent involved in solving the problem. Consequently, to improve Prange's ISD, it is essential to reduce the cost of the Gaussian algorithm $\mathcal{G}$ since the probability of collecting n error-free samples cannot be changed.

Canteaut and Chabaud (CC) [23] and Bernstein, Lange and Peters (BLP) [13] proposed algorithms for solving LPN through reducing the cost of Gaussian elimination via so-called *reusing existing pivots* and *forcing more existing pivots*. In this work, we propose a method to extend these algorithms. The simplest RP is performed by two steps:

1. Guess N zero positions in $\mathbf{e}$ and transform from (m, n, t)-LPN instances to $(m - N, n - N, t)$-LPN instances.
2. Solve the $(m - N, n - N, t)$-LPN problem using Prange's ISD algorithm.

Since the secret dimension is $n - N$, the Gaussian elimination algorithm is conducted over a matrix of size $n - N$, which reduces the cost of the Gaussian elimination. This approach is performed iteratively, with the number of iterations being referred to as a *level*. As a result, the Reduce and Prange algorithms, which are defined by their level, are naturally defined. For example, when $k = 2$, then we first guess N_1 positions, to generate $(m - N_1, n - N_1, t)$-LPN instances. Then, we additionally guess N_2 positions to get $(m - N_1 - N_2, n - N_1 - N_2, t)$-LPN instances. As the final step, we solve the LPN problem using Prange's ISD algorithm.

Advantages of Reduce and Prange's ISD. The Reduce and Prange's ISD algorithm offers a key advantage by lowering the cost per iteration compared to the original Prange's ISD. While the original algorithm requires n^ω cost for each iteration due to performing Gaussian elimination on an n-dimensional matrix, the Reduce and Prange's algorithm strategically divides zero indices into sets, $\mathcal{I}_1$ and $\mathcal{I}_2$. If an error occurs in $\mathcal{I}_2$, only this subset is replaced rather than all n indices, reducing the iteration cost to $(n - N_1)^\omega$. Similarly, when $\mathcal{I}_1$ is incorrectly guessed, the Reduce and Prange's algorithm only repeats the Step 1 with a complexity $\mathcal{C}_{m,N_1}$. This approach significantly lowers the (concrete) iteration cost, while maintaining the overall success probability, as the algorithm still collects n zero indices.

Comparison to previous works [13,23]. We provide a conceptual comparison between previous works and our algorithm in terms of $\mathcal{G}$. The key idea behind the algorithms in [13,23] involves reusing the pivots from the previous iteration.

In a nutshell, suppose that $(\mathbf{A}, \mathbf{b})$ with $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod \mathbb{F}_q$ is given. The BLP/CC algorithms [13, 23] transform this into a pair $(\mathbf{A}', \mathbf{b}') \in \mathbb{F}_q^{m-n\times(n+1)}$ where $\mathbf{b}' = \mathbf{A}' \cdot \mathbf{e}' + \mathbf{e}''$ mod $\mathbb{F}_q$ by reducing $\mathbf{s}$. If the partial noise vector $\mathbf{e}' \in \mathbb{Z}_q^n$ is zero, then $\mathbf{b}'$ has a Hamming weight of t, leading to the algorithm's termination. If not, some components of $\mathbf{e}'$ will be zero, allowing the reuse of the $n - c$ zero coordinates in $\mathbf{e}'$ and collection of c new samples to obtain n error-free instances.

When guessing that $n - c$ coordinates of $\mathbf{e}'$ are zero, the corresponding columns of $\mathbf{A}'$ are not needed. As a result, the pair $(\mathbf{A}', \mathbf{b}')$ becomes $(\mathbf{A}'', \mathbf{b}') \in \mathbb{Z}_q^{m-n\times(c+1)}$. Since Gaussian elimination is required over a smaller dimension $c < n$, the overall complexity is reduced compared to Prange's algorithm. This optimization can be seen as collecting error-free instances from two separate lists, one of size $n - c$ and the other of size c.

Our RP algorithm starts from an extension of the BLP/CC observation: In the perspective, we aim at collecting n error-free instances using 'multi' separated lists $(N_1, N_2, \ldots, N_k)$ such that $\sum_{i=1}^{k} N_i = n$, where k is called a level. Given the multi-size set, the RP collects N_1 error-free instances and converts (m, n, t)-LPN instances into (m_1, n_1, t)-LPN instances with $m_1 = m - N_1, n_1 = n - N_1$ by reducing the secret vector from the N_1 error-free instances. This technique can be iteratively applied for multiple rounds with the given set. The time complexity of each iteration is then less than that of $\mathcal{G}$. By selecting $\{N_i\}_{i=1}^{k}$ properly, it improves the overall performance of the algorithm. For the detailed algorithmic description, we refer Section 2.

2 Reduce and Prange Technique

This section introduces a new approach for solving LPN over large fields, simply called the Reduce and Prange. For the algorithm description, we suppose that LPN instances of the form $(\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod \mathbb{F}_q)$ with $\mathbf{A} \in \mathbb{F}_q^{m\times n}, \mathbf{s} \in \mathbb{F}_q^n$ and $\mathbf{e} \leftarrow \chi_{t,m}$ are given.

It is important to note that the probability of collecting n error-free polynomials is $\frac{\binom{m-n}{t}}{\binom{m}{t}}$. Therefore, the Prange's ISD algorithm 1 ensures that the desired secret vector can be recovered with $\frac{\binom{m}{t}}{\binom{m-n}{t}}$ iterations. Therefore, the time complexity of Prange's ISD algorithm is estimated by

$$O\left(\frac{\binom{m}{t}}{\binom{m-n}{t}} \cdot n^{\omega}\right)$$

where n^{ω} is a cost of the Gaussian elimination with the linear algebra constant $2 \leq \omega \leq 3$.

2.1 Reduce and Prange's ISD

The intuition of the attack is that we first guess N_1 error-free samples to obtain LPN samples themselves of less dimension. We then apply Prange's algorithm to the LPN of $n - N_1$ dimensions.

Suppose that LPN instances of the form $(\mathbf{A}, \mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod \mathbb{F}_q)$ with $\mathbf{A} \in \mathbb{F}_q^{m \times n}, \mathbf{s} \in \mathbb{F}_q^n$ and $\mathbf{e} \leftarrow \chi_{t,m}$. We let $\mathbf{a}_i$ (Resp. b_i and e_i) denote the i-th vector of $\mathbf{A}$ (Resp. $\mathbf{b}$ and $\mathbf{e}$).

Step 1. Guess an index set $\mathcal{I}_1 \subset [m]$ of the size $N_1 (< n)$, which corresponds to the zero positions in $\mathbf{e}$. For an easy explanation, we will assume that the last N_1 positions in $\mathbf{e}$ are zero. In general, the zero positions can be shifted to the end through permutation. Let $\mathbf{A}_{\mathsf{top}}$ and $\mathbf{A}_{\mathsf{bot}}$ be a submatrix of $\mathbf{A}$ which consists of the first $m - N_1$ rows of $\mathbf{A}$ and the last N_1 rows, respectively. Analogously to the notation, we define $\mathbf{b}_{\mathsf{top}}$, $\mathbf{b}_{\mathsf{bot}}$, $\mathbf{e}_{\mathsf{top}}$, and $\mathbf{e}_{\mathsf{bot}}$. The above assumption then says that $\mathbf{e}_{\mathsf{bot}} = \mathbf{0}$. Under the notation, it holds that

$$\begin{pmatrix} \mathbf{b}_{\mathsf{top}} \\ \mathbf{b}_{\mathsf{bot}} \end{pmatrix} = \begin{pmatrix} \mathbf{A}_{\mathsf{top}} \\ \mathbf{A}_{\mathsf{bot}} \end{pmatrix} \cdot \mathbf{s} + \begin{pmatrix} \mathbf{e}_{\mathsf{top}} \\ \mathbf{0} \end{pmatrix} \bmod \mathbb{F}_q$$

We then apply the Gaussian elimination algorithm to obtain LPN of less dimension. For this purpose, We parse $\mathbf{s}$ to $(\mathbf{s}^1 \| \mathbf{s}^2)$ such that $\mathbf{s}^1 \in \mathbb{F}_q^{n-N_1}$ and $\mathbf{s}^2 \in \mathbb{F}_q^{N_1}$. Similarly to the $\mathbf{s}$, we split $\mathbf{A}_{\mathsf{top}}$ (Resp. $\mathbf{A}_{\mathsf{bot}}$) into $(\mathbf{A}_{\mathsf{top}}^{\mathsf{left}} \| \mathbf{A}_{\mathsf{top}}^{\mathsf{right}})$ (Resp. $(\mathbf{A}_{\mathsf{bot}}^{\mathsf{left}} \| \mathbf{A}_{\mathsf{bot}}^{\mathsf{right}})$) as well. Assuming that the matrix $\mathbf{A}_{\mathsf{bot}}^{\mathsf{right}}$ is invertible over $\mathbb{F}_q$, (If not, we change the order of columns until $\mathbf{A}_{\mathsf{bot}}^{\mathsf{right}}$ is invertible), one can get the following LPN samples:

$$\begin{aligned} \mathbf{b}' &:= \left(\mathbf{I}_{m-N_1} \| -\mathbf{A}_{\mathsf{top}}^{\mathsf{right}} \cdot (\mathbf{A}_{\mathsf{bot}}^{\mathsf{right}})^{-1} \right) \cdot \begin{pmatrix} \mathbf{b}_{\mathsf{top}} \\ \mathbf{b}_{\mathsf{bot}} \end{pmatrix} \\ &= \left(\mathbf{I}_{m-N_1} \| -\mathbf{A}_{\mathsf{top}}^{\mathsf{right}} \cdot (\mathbf{A}_{\mathsf{bot}}^{\mathsf{right}})^{-1} \right) \cdot \left(\begin{pmatrix} \mathbf{A}_{\mathsf{top}} \\ \mathbf{A}_{\mathsf{bot}} \end{pmatrix} \cdot \mathbf{s} + \begin{pmatrix} \mathbf{e}_{\mathsf{top}} \\ \mathbf{0} \end{pmatrix} \right) \bmod \mathbb{F}_q \\ &= \underbrace{(\mathbf{A}_{\mathsf{top}}^{\mathsf{left}} - \mathbf{A}_{\mathsf{top}}^{\mathsf{right}} \cdot (\mathbf{A}_{\mathsf{bot}}^{\mathsf{right}})^{-1} \cdot \mathbf{A}_{\mathsf{bot}}^{\mathsf{left}})}_{=:\mathbf{A}'} \cdot \mathbf{s}^1 + \mathbf{e}_{\mathsf{top}} \bmod \mathbb{F}_q. \end{aligned}$$

We denote it as $(\mathbf{A}', \mathbf{b}') \leftarrow \mathsf{Conv}((\mathbf{A}, \mathbf{b}), \mathcal{I}_1)$ since this conversion technique is employed several times.

Complexity. In each guessing, this algorithm computes the matrix $\mathbf{A}'$, which consists of one matrix inversion and two matrix multiplications so it takes in time $\mathcal{C}_{m,N_1} := O(N_1^\omega + N_1^2 \cdot (m - N_1) + N_1 \cdot (m - N_1) \cdot (n - N_1))$, where ω is the linear algebra constant. The probability that attackers can correctly guess N_1 positions is $\mathcal{P}_{m,t,N_1} := \frac{\binom{m-t}{N_1}}{\binom{m}{N_1}}$, which directly implies that the time complexity to get LPN samples of dimension $n - N_1$ is $\frac{1}{\mathcal{P}_{m,t,N_1}} \cdot \mathcal{C}_{m,N_1}$.

Step 2. For ease of representation, we assume that $(m - N_1, n - N_1, t)$-LPN instances $(\mathbf{A}', \mathbf{b}')$ such that $\mathbf{b}' = \mathbf{A}' \cdot \mathbf{s}^1 + \mathbf{e}_{\mathsf{top}} \bmod \mathbb{F}_q$ are given, where $(\mathbf{A}', \mathbf{b}') \leftarrow \mathsf{Conv}(\mathbf{A}, \mathbf{b}), \mathcal{I}_1), \mathbf{s}^1 \in \mathbb{F}_q^{n-N_1}$ and $\mathbf{e}_{\mathsf{top}} \in \chi_{t,m-N_1}$. In addition, the index of $\mathbf{e}_{\mathsf{top}}$ is given as $[m] \setminus \mathcal{I}_1$.

The second step just runs Prange's algorithm. That is, one first collects an index set $\mathcal{I}_2 \subset [m] \setminus \mathcal{I}_1$ of size $n - N_1$ from given $m - N_1$ LPN instances.

Then, one can recover the secret $\mathbf{s}^1$ via the Gaussian elimination algorithm. After recovering $\mathbf{s}^1$, one can recover the vector $\mathbf{s}^2$ from the identity $\mathbf{b}_{\mathsf{bot}} = \mathbf{A}_{\mathsf{bot}} \cdot (\mathbf{s}^1 \| \mathbf{s}^2)$.

Complexity. It is obvious to compute the complexity of Prange's algorithm for solving LPN over with dimension $n - N_1$, samples $m - N_1$ and t nonzero:

$$\mathcal{S}_{m-N_1,n-N_1,t} \stackrel{def}{=} \frac{\binom{m-N_1}{n-N_1}}{\binom{m-N_1-t}{n-N_1}} \cdot (n-N_1)^\omega = \frac{1}{\mathcal{P}_{m-N_1,t,n-N_1}} \cdot (n-N_1)^\omega,$$

where the linear algebra constant ω. Putting it together, the LPN problem is solved within

$$\frac{1}{\mathcal{P}_{m,t,N_1}} \cdot (\mathcal{C}_{m,N_1} + \mathcal{S}_{m-N_1,n-N_1,t})$$

In the complexity, the term $\frac{1}{\mathcal{P}_{m,t,N_1}} \cdot \mathcal{S}_{m-N_1,n-N_1,t} = \frac{\binom{m}{n}}{\binom{m-t}{n}} \cdot (n-N_1)^\omega$ is a dominating term. Compared to the original Prange's ISD algorithm $\frac{\binom{m-t}{n}}{\binom{m}{n}} \cdot n^\omega$, the fractional part is shared but one can see that the multiplier factor, $(n-N_1)^\omega$, is smaller for the Reduce and Prange algorithm. This can be expected to lower the overall complexity of the algorithm.

Impact of the Reduce and Prange's ISD. We present the advantages of the Reduce and Prange's ISD algorithm. The main difference lies in the cost of each iteration. To be precise, the original Prange's ISD algorithm incurs a cost of n^ω, as Gaussian elimination on an n-dimensional square matrix is performed in each iteration.

The Reduce and Prange's ISD algorithm reduces the cost per iteration. Intuitively, the algorithm collects sets of zero indices, $\mathcal{I}_1$ and $\mathcal{I}_2$, such that $\#\mathcal{I}_1 + \#\mathcal{I}_2 = n$ with $\#\mathcal{I}_1 = N_1$. Suppose that $\mathcal{I}_2$ includes an incorrect zero index. In this case, the Reduce and Prange's algorithm only needs to replace $\mathcal{I}_2$, rather than all n zero indices. This step corresponds to Step 2. This adjustment results in a per-iteration cost of only $(n-N_1)^\omega$. Similarly, if $\mathcal{I}_1$ is incorrectly guessed, the Reduce and Prange's algorithm repeats Step 1 with a complexity of $\mathcal{C}_{m,N_1}$.

In summary, the Reduce and Prange's algorithm reduces the cost per iteration. However, the overall probability remains unchanged since the algorithm still collects sets of n zero indices.

2.2 Iterative Reduce and Prange Technique

'Step 1' in Section 2 can be interpreted as a self-reduction of LPN. As a natural extension, one can run Step 1 reduction repeatedly to get LPN samples of less dimension. Thereafter, one can solve the reduced LPN problems by Step 2. With respect to the number of reductions, we call this algorithm the level-k Reduce and Prange algorithm.

For instance, suppose $k = 2$, the level-2 Reduce and Prange algorithm corresponds as follows: First, guess three index sets $\{\mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3\} \subset [m]$ of size $\#\mathcal{I}_i = N_i$ and $\sum_{i=1}^{3} N_i = n$. Then, one can get LPN samples $(\mathbf{A}^{(2)}, \mathbf{b}^{(2)})$ of dimension N_3 from $(\mathbf{A}^{(2)}, \mathbf{b}^{(2)}) \leftarrow \mathsf{Conv}((\mathbf{A}^{(1)}, \mathbf{b}^{(1)}), \mathcal{I}_2)$ and $(\mathbf{A}^{(1)}, \mathbf{b}^{(1)}) \leftarrow \mathsf{Conv}((\mathbf{A}, \mathbf{b}), \mathcal{I}_1)$. Then it can be solved via the Prange algorithm on the less dimension N_3. The interesting fact is that since this auxiliary step proceeds after the first step occurs, the overall time complexity for level-2 Reduce and Prange algorithm is given as

$$\frac{1}{\mathcal{P}_{m,t,N_1}} \cdot \left(\mathcal{C}_{m,N_1} + \frac{1}{\mathcal{P}_{m-N_1,t,N_2}} \left(\mathcal{C}_{m-N_1,N_2} + \mathcal{S}_{m-N_1-N_2,N_3,t} \right) \right).$$

As a generalization, the time complexity for level-k Reduce and Prange algorithm is then obviously computed. Given a set $\{N_i\}_{i=1}^{k+1}$ and sample an index set $\{\mathcal{I}_i\}_{i=1}^{k+1} \subset [m]$ such that $\#\mathcal{I}_i = N_i$ and $\sum_{i=1}^{k+1} \mathcal{I}_i = n$. For ease of exposition, we define simplified notations $\mathcal{P}(j), \mathcal{C}(j), \mathcal{S}$ as follows:

$$\begin{aligned} \mathcal{P}(j) &= \mathcal{P}_{m-\mu_{j-1},t,N_j} \\ \mathcal{C}(j) &= \mathcal{C}_{m-\mu_{j-1},N_j} \\ \mathcal{S} &= \frac{1}{\mathcal{P}_{m-\mu_k,t,n-\mu_k}} \cdot (n-\mu_k)^{\omega}, \end{aligned}$$

where $\mu_j = \sum_{i=1}^{j} N_i$ for every $j \geq 1$. Then, under the notation, the total cost of our algorithm is

$$\mathcal{T} = \frac{1}{\mathcal{P}(1)} \left(\mathcal{C}(1) + \frac{1}{\mathcal{P}(2)} \left(\mathcal{C}(2) + \cdots + \frac{1}{\mathcal{P}(k)} \left(\mathcal{C}(k) + \mathcal{S} \right) \right) \right)$$

To simplify it, we also define $\mathcal{Q}(j)$ as $\mathcal{Q}(j) = \prod_{i=1}^{j} \mathcal{P}(i)$. We then denote the total cost as

$$\mathcal{T} = \sum_{j=1}^{k} \frac{\mathcal{C}(j)}{\mathcal{Q}(j)} + \frac{\mathcal{S}}{\mathcal{Q}(k)}.$$

An algorithm to construct a set $\{N_i\}_{i=1}^{k+1}$, optimizing the complexity of the RP algorithm, will be described later. The whole algorithm is then described as follows.

Algorithm 2 Reduce and Prange's ISD

Input: (m, n, t)-LPN over $\mathbb{F}_q$ instances $(\mathbf{A}, \mathbf{b}) \in \mathbb{F}_q^{m \times (n+1)}$ and an index size $\{N_i\}_{i=1}^{k+1}$.
Output: The secret vector $\mathbf{s}$ such that $|\mathbf{b} - \mathbf{A} \cdot \mathbf{s} \bmod \mathbb{F}_q| = t$.

repeat
  for $i = 1$ up to k **do**
    Choose a random index set $\mathcal{I}_i$ of size N_i
    Update $(\mathbf{A}, \mathbf{b}) \leftarrow \mathsf{Conv}((\mathbf{A}, \mathbf{b}), \mathcal{I}_i)$
    $(\mathbf{s}', \mathcal{I}_{k+1}) \leftarrow$ *Algorithm 1*$((\mathbf{A}, \mathbf{b}), t)$
    $\mathcal{I} := \bigcup_{i=1}^{k+1} \mathcal{I}_i$ and $(\mathbf{A}_\mathcal{I}, \mathbf{b}_\mathcal{I}) \leftarrow [\mathbf{A}, \mathbf{b}]_\mathcal{I}$
  end for
until $\mathbf{A}_\mathcal{I}$ is invertible and $|\mathbf{b} - \mathbf{A} \cdot \mathbf{s} \bmod \mathbb{F}_q| = t$ where $\mathbf{s} = \mathbf{A}_\mathcal{I}^{-1} \cdot \mathbf{b}_\mathcal{I}$
return $(\mathbf{s}, \mathcal{I})$

Optimal index size. To (approximately) optimize the cost, we observe $\mathcal{T}$ as two perspectives. First of all, for every j, we observe

$$\mathcal{Q}(j) = \frac{\binom{m-t}{\mu_j}}{\binom{m}{\mu_j}}$$

where $\mu_j = \sum_{i=1}^{j} N_i$. From the observation, $\frac{\mathcal{C}(1)}{\mathcal{Q}(1)}$ is only determined by the N_1 when $j = 1$. Inductively, when the $\sum_{i=1}^{j-1} N_i$ is fixed once, the ratio $\frac{\mathcal{C}(j)}{\mathcal{Q}(j)}$ is only determined by the N_j.

Next, we also have

$$\begin{aligned}\frac{\mathcal{S}}{\mathcal{Q}(k)} &= \frac{\binom{m}{\mu_j}}{\binom{m-t}{\mu_j}} \cdot \mathcal{S} \\ &= \frac{\binom{m}{\mu_j}}{\binom{m-t}{\mu_j}} \cdot \frac{(n-\mu_k)^\omega}{\mathcal{P}_{m-\mu_k, t, n-\mu_k}}\end{aligned}$$

where $\mu_j = \sum_{i=1}^{j} N_j$ for each j. Therefore, it satisfies that

$$\frac{\mathcal{S}}{\mathcal{Q}(k)} = (n-\mu_k)^\omega \cdot \frac{\binom{m}{n}}{\binom{m-t}{n}},$$

where the quantity is invariant under the level k. This means that the cost of the last term is written as $\frac{\binom{m-t}{n}}{\binom{m}{n}} \cdot \alpha$ for some $\alpha = (n-\mu_k)^\omega$. Obviously, α gets smaller as μ_k gets larger.

Since the overall cost of a positive addition will be optimized when all values are similar, we let the threshold be $\frac{\binom{m-t}{n}}{\binom{m}{n}} \cdot \Delta$ for some integer $\Delta \in \mathbb{Z}$ and choose the largest integer N_i such that each $\frac{\mathcal{C}(i)}{\mathcal{Q}(i)}$ term is less than the threshold, starting at $i = 1$. If there is no integer N_j for an index j, we let $N_j = n - \mu_{j-1}$ and

operate the Prange's algorithm of LPN of dimension N_j. Then the total cost is given as $\sum_{i=1}^{j-1} \frac{\mathcal{C}(j)}{\mathcal{Q}(j)} + \frac{\binom{m}{n}}{\binom{m-t}{n}} N_j^{\omega}$.

We try it for all Δ smaller than $n^{2.8}$, and give the most optimized cost. The full algorithm for estimating the Reduce and Prange will be given by Algorithm 3.

Algorithm 3 RP Estimation

Input: LPN parameters (m, n, t)
Output: The total cost of the Reduce and Prange algorithm.
Set $\mathsf{T} = \frac{\binom{m}{n}}{\binom{m-t}{n}}$ // T is a threshold.
Set $Cost = \mathsf{T} \cdot n^{2.8}$
for $\Delta = 1$ up to $n^{2.8}$ **do**
  Set $k = 1$
  while $\exists N_k$ such that $\frac{\mathcal{C}(k)}{\mathcal{Q}(k)} < \Delta \cdot \mathsf{T}$ **do**
    Set largest N_k such that $\frac{\mathcal{C}(k)}{\mathcal{Q}(k)} < \Delta \cdot \mathsf{T}$
    Set $k = k + 1$
  end while
  Set $N_{k+1} = n - \sum_{i=1}^{k} N_i$
  $Cost = \min\left\{Cost, \sum_{i=1}^{k} \frac{\mathcal{C}(i)}{\mathcal{Q}(i)} + \mathsf{T} \cdot N_{k+1}^{2.8}\right\}$
end for
Return $Cost$ and $\{N_i\}_{i=1}^{k+1}$

2.3 Graphical Results and Its Discussion

This section provides graphical results for our RP to provide the improvement scale with the parameter size. For this purpose, we have tried to find an asymptotic time complexity represented by $\frac{1}{\mathcal{P}} \cdot n^{C \cdot 2.8}$ with $\mathcal{P} = \frac{\binom{m-n}{t}}{\binom{m}{t}}$. For example, it is obvious that $C = 1$ when we exploit the original Prange's ISD. If C is strictly less than 1, then one concludes RP outperforms Prange's ISD. This is because $\mathcal{P}$ in RP and $\mathcal{P}$ in Prange's ISD are essentially the same. Thus, C directly measures how much RP improves the performance rather than Prange's ISD.

Given two parameters $(m, n, t) = (2^{10}, 652, 106)$ and $(m, n, t) = (2^{12}, 1589, 172)$, we provide plots of the change in C by changing only one parameter. For example, Figure 1 and Figure 2 show changes of C when increasing m with a fixed (n, t). According to these figures, we observe that when m goes to infinity, C is also close to 1.

On the other hand, from Figure 3 and Figure 4, we also observe that when n is bigger than, RP is more better with a fixed (m, t). Similarly, from Figure 5 and Figure 6 argues that given a fixed (m, n), whenever t is bigger, the performance of RP is better than that of Prange's ISD.

When we find an approximation of C very close to each figure, then it would be helpful to estimate the performance of the asymptotic complexity of RP. We unfortunately fail to find the closeness of C. However, we believe that it would be helpful to graphically provide the improvement of RP.

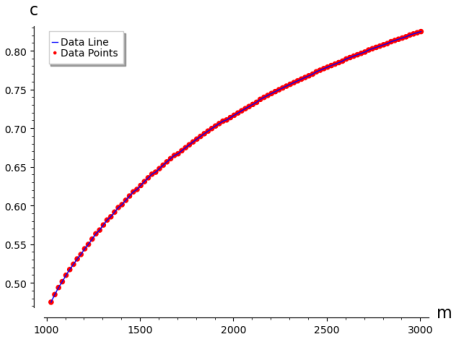

Fig. 1. Graphical results by increasing m with a fixed $(n, t) = (652, 106)$

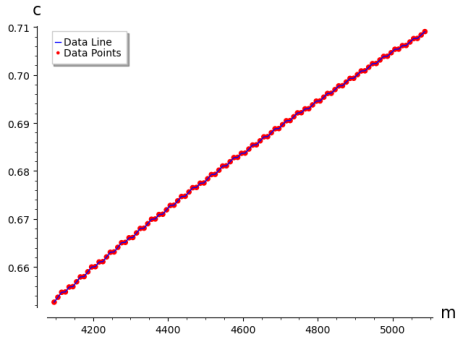

Fig. 2. Graphical results by increasing m with a fixed $(n, t) = (1589, 172)$

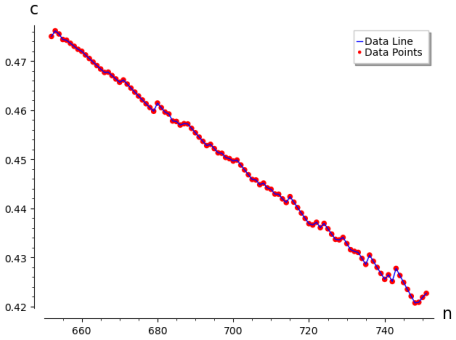

Fig. 3. Graphical results by increasing n with a fixed $(m, t) = (2^{10}, 106)$

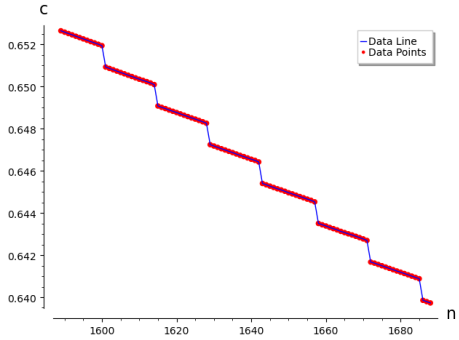

Fig. 4. Graphical results by increasing n with a fixed $(m, t) = (2^{12}, 172)$

3 Regular-RP: Application to regular-LPN

This section provides how to apply the RP to the LPN with regular noise, which is equivalent to the regular syndrome decoding problem. Throughout this section, we set $\mathcal{R} = \mathbb{F}_q$ with $q \geq 2$.

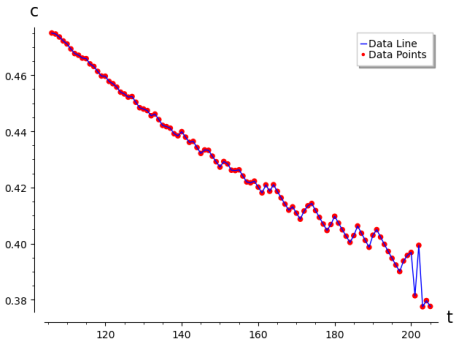

Fig. 5. Graphical results by increasing t with a fixed $(m, n) = (2^{10}, 652)$

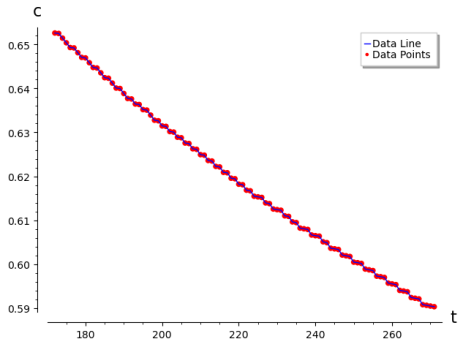

Fig. 6. Graphical results by increasing t with a fixed $(m, n) = (2^{12}, 1589)$

We first introduce a new problem, called the regular syndrome decoding (RSD). RSD was originated [8], and it has a lot of applications in PGC-like protocols [9, 18, 20, 22, 28, 31, 40, 60–62, 66, 70].

Problem 3 (Regular Syndrome Decoding over $\mathbb{F}_q$). Let m, n, t, β be positive integers with $m = t \cdot \beta$ and $\mathbb{F}_q$ be a finite field. Sample a full rank matrix $\mathbf{H} \leftarrow \mathbb{F}_q^{(m-n)\times m}$ and a column vector $\mathbf{e} := (\mathbf{e}_1 \| \mathbf{e}_2 \| \dots \| \mathbf{e}_t) \leftarrow \mathbb{F}_q^m$ such that $\mathbf{e}_i \in \mathbb{F}_2^\beta$ is of Hamming weight $|\mathbf{e}_i| = 1$ for $1 \le i \le t$. Given $(\mathbf{H}, \mathbf{y} = \mathbf{H} \cdot \mathbf{e})$, recover the error $\mathbf{e}$.

3.1 Algorithm for regular-LPN over Large Fields

Esser and Santini [38] proposed an algorithm to solve regular-LPN, with the permutation-based ISD algorithm leveraging the regular structure to identify error-free instances. We incorporate this technique into our algorithm to better estimate the security of regular-LPN over large fields.

This approach optimizes the guessing probability in the RP algorithm to solve regular-LPN more efficiently in large fields, which we refer to as *regular-RP*. Except for the probability, the strategy of RP is used.

Regular-RP. As the standard RP, suppose that $\{N_i\}_{i=1}^{k+1}$ is given and sample $\{\mathcal{I}_i\}_{i=1}^{k+1}$ such that $\#\mathcal{I}_i = N_i$ and $\sum_{i=1}^{k+1} N_i = n$. Specifically, in the initial step, we sample $\mathcal{I}_1$ to set $(\mathbf{A}', \mathbf{b}') \leftarrow \mathsf{Conv}((\mathbf{A}, \mathbf{b}), \mathcal{I}_1)$. We now focus on how to sample $\mathcal{I}_1$ to leverage the regular constraint. Here, we set N_i as a multiple of t for a simple description.

By definition of regular-LPN, it holds that each $\mathbf{e}_j$ has exactly one nonzero element. It immediately implies that the probability of guessing a zero coordinate of $\mathbf{e}_1$ equals to $\frac{\beta-1}{\beta}$. To guess the second zero position, one considers two cases:

- E_1: Find the zero coordinate in $\mathbf{e}_1$.
- E_2: Find the zero coordinate in $\mathbf{e}_j$ for $j \neq 1$.

The probability of event E_1 is $\frac{\beta-2}{\beta-1}$, whereas the probability of event E_2 is $\frac{\beta-1}{\beta}$. Thus, the best strategy for sampling $\mathcal{I}_1$ is to guess N_1/t-zero coordinates in each $\mathbf{e}_j$. Consequently, the probability of selecting N_1 zero positions is

$$\mathcal{P}^R_{m,t,N_1} = \left(1 - \frac{N_1/t}{\beta}\right)^t = \left(1 - \frac{N_1}{m}\right)^t.$$

Similarly, when sampling $\mathcal{I}_i$, guessing N_i/t zero coordinates in each $\mathbf{e}_j$ yields an optimal probability. To describe it formally, we denote a coordinate set of $\mathbf{e}_j$ as $[\mathbf{e}_j]$ and parse $\mathcal{I}_i = \cup_{j=1}^{t} \mathcal{I}_{ij}$ such that $\mathcal{I}_{ij} \subset [\mathbf{e}_j] \backslash \cup_{\ell=1}^{i-1} \mathcal{I}_{\ell j}$ of size $\#\mathcal{I}_{ij} = N_i/t$. We refer to the RP algorithm using this customized sampling method as regular-RP.

Algorithm 4 Regular Reduce and Prange's ISD

Input: (m, n, t)-regular LPN over $\mathbb{F}_q$ instances $(\mathbf{A}, \mathbf{b}) \in \mathbb{F}_q^{m\times(n+1)}$ and an index set $\{N_i\}_{i=1}^{k+1}$.
Output: The secret vector $\mathbf{s}$ such that $|\mathbf{b} - \mathbf{A}\cdot\mathbf{s} \bmod \mathbb{F}_q| = t$.

repeat
  for $i = 1$ up to k **do**
    Choose a random index set $\mathcal{I}_{ij}$ of size N_i/t for $1 \le j \le t$
    Update $(\mathbf{A}, \mathbf{b}) \leftarrow \mathsf{Conv}((\mathbf{A}, \mathbf{b}), \mathcal{I}_i)$
    $(\mathbf{s}', \mathcal{I}_{k+1}) \leftarrow$ *Algorithm 1*$((\mathbf{A}, \mathbf{b}), t)$
    $\mathcal{I} := \bigcup_{i=1}^{k+1} \mathcal{I}_i$ and $(\mathbf{A}_\mathcal{I}, \mathbf{b}_\mathcal{I}) \leftarrow [\mathbf{A}, \mathbf{b}]_\mathcal{I}$
  end for
until $\mathbf{A}_\mathcal{I}$ is invertible and $|\mathbf{b} - \mathbf{A}\cdot\mathbf{s} \bmod \mathbb{F}_q| = t$ where $\mathbf{s} = \mathbf{A}_\mathcal{I}^{-1}\cdot\mathbf{b}_\mathcal{I}$
return $(\mathbf{s}, \mathcal{I})$

Complexity. Based on the modified probability, we can also calculate the time costs of regular-RP. To this end, we define $\mathcal{P}^R(j), \mathcal{C}(j)$ and $\mathcal{S}^R$ as follows:

$$\begin{aligned}
\mathcal{P}^R(j) &= \mathcal{P}^R_{m-\mu_{j-1},t,N_j} \\
\mathcal{C}(j) &= \mathcal{C}_{m-\mu_{j-1},N_j} \\
\mathcal{S}^R &= \frac{1}{\mathcal{P}^R_{m-\mu_k,t,n-\mu_k}} \cdot (n-\mu_k)^\omega
\end{aligned}$$

where $\mu_j = \sum_{i=1}^{j} N_i$ for every j. Then, under the notation, the total cost of level-k regular-RP algorithm, denoted by $\mathcal{T}^R$, can be expressed as

$$\frac{1}{\mathcal{P}^R(1)}\left(\mathcal{C}(1) + \frac{1}{\mathcal{P}^R(2)}\left(\mathcal{C}(2) + \cdots + \frac{1}{\mathcal{P}^R(k)}\left(\mathcal{C}(k) + \mathcal{S}^R\right)\right)\right)$$

To simplify it, we also define $\mathcal{Q}^R(j)$ as $\mathcal{Q}^R(j) = \prod_{i=1}^{j} \mathcal{P}^R(i)$. We then denote the total cost as

$$\mathcal{T}^R = \sum_{j=1}^{k} \frac{\mathcal{C}(j)}{\mathcal{Q}^R(j)} + \frac{\mathcal{S}^R}{\mathcal{Q}^R(k)}.$$

4 Concrete Estimation of LPN via Reduce and Prange algorithm

In this section, we provide the estimated results of our algorithms for solving the LPN over large fields. Especially, we mainly provide security estimations for LPN over large fields used in PCG applications. The results are given by

- Table 2: Comparison results between RP and previous works for solving LPN over $\mathbb{F}_{2^{128}}$
- Table 3: Numerical results of RP on recommendation parameters for 128-bit security in [50]
- Table 4: Comparison results between regular-RP and previous works for solving regular-LPN
- Table 5: Numerical results of regular-RP on recommendation parameters for 128-bit security in [50]

We further compare the results with recent attacks [21, 38, 50] depending on problems in each tables. Throughout this section, time complexity means the number of arithmetic operations in each field. According to the results in Table 3, we observe that for LPN over large fields, our algorithm is always better than previous works. For regular LPN over large fields, which achieves 128-bit security, our algorithm tends to be better when m, n is small, but the algorithm quickly becomes worse for large parameters. For this purpose, we first recall the parameters for LPN over large fields.

- q: the size of fields $= 2^{128}$.
- m: the number of LPN instances.
- n: the dimension of LPN secret $\mathbf{s}$.
- t: the number of nonzero entries in $\mathbf{e}$.

Our focus is mainly on the parameter regime $(m, q, t) = (poly(n), n^{w(1)}, o(m))$, which has been widely used in several cryptographic applications including those presented in [9, 10, 17, 19, 20, 28, 32, 62, 66, 70].

While setting optimal parameters provides the lowest computational cost for solving the LPN problem, empirically determining the optimal index size $\{N_i\}_{i=1}^{k+1}$ can be time-consuming. Furthermore, the level parameter k is usually set at a minimum of 2000, which adds to the complexity. As an alternative, we propose a quicker estimation method using the RP algorithm with $k = 3$. We configure it by setting $N_1 = 0.4 \cdot n, N_2 = 0.3 \cdot n, N_3 = 0.2 \cdot n, N_4 = 0.1 \cdot n$. Heuristically, this setup of the RP cost on these index size yields a nearly optimal cost, with only a 2-3 bit difference in performance. This approach significantly reduces the setup time while maintaining close to optimal performance levels. We say the RP on the index size quick-RP. Our analysis primarily utilizes the RP algorithm. However, for concrete estimations, we occasionally employ the quick-RP approach and will specifically note this when used.

For a detailed examination of the computational experiments conducted in this study, the complete source code has been made available. The repository

includes implementations of the RP estimation and its faster variant, quick-RP, applied to both LPN and regular-LPN problems. The code can be accessed via the following link: RP and Quick-RP Estimations.

To assess our algorithm, we compare its performance with ISD and its variants. We begin by recalling the results from [50]:

- For the parameter regime $\mathcal{R} = \mathbb{F}_q$ with prime power $q = n^{w(1)}$, $t = o(m)$ and $(1+\beta)\cdot n \leq m = poly(n)$ for some constant β, the statistical decoding including the 2.0 variant [26] needs more cost than Prange's algorithm [59].
- For the above parameter setup, information set decoding and its variants outperform other algorithms.

Therefore, ISD and its variants serve as the most relevant benchmarks for evaluating our results. In the following, we then describe the complexity model of the algorithms.

For fair comparisons, the time complexity of the Gaussian elimination algorithm on a $k \times k$ matrix is based on the same cost model. Indeed, we utilize a cost model of $k^{2.8}$ as in the previous works [17, 50].[6]

Cost models of the Previous algorithms. We present the cost models for known algorithms that solve the LPN problem over large fields, focusing on three key algorithms: Gauss ISD [59], ISD frameworks, and Statistical Decoding [26, 50].

Gauss Algorithm. The Gauss algorithm is identical to Prange's algorithm (see Algorithm 1), but it covers an additional case: LPN instances $(\mathbf{A}, \mathbf{b}) = (\mathbf{A}, \mathbf{A}\cdot\mathbf{s}+\mathbf{e})$ can be transformed into the problem of finding $\mathbf{e}$ when $\mathbf{H}\cdot\mathbf{e} = \mathbf{H}\cdot\mathbf{b}$ for a parity check matrix $\mathbf{H}$, and vice versa, as shown in [55, Lemma 4.9].

In this scenario, the cost is given by:$(m-n)^{2.8} \cdot \left(\frac{\binom{m}{t}}{\binom{m-n}{t}}\right)$.Combining these factors, the Gauss algorithm's cost model is defined as follows:

$$\texttt{Gauss\_cost\_model} = \log\left(\min\{n^{2.8}, (m-n)^{2.8}\} \cdot \left(\frac{\binom{m}{t}}{\binom{m-n}{t}}\right)\right)$$

Information Set Decoding. For LPN over large fields and regular-LPN problems, Liu et. al. [50] showed that SD-ISD is the most efficient algorithm. As a result, they estimate the hardness of LPN over large fields and regular-LPN based on the cost of SD-ISD. Thus, we introduce only a cost of SD-ISD attack as a representative of ISD frameworks.

Lemma 4.1 (Cost of SD-ISD attack [50]) *The (m, n, t)-LPN problem over $\mathbb{F}_q$ can be solved by the SD-ISD variant in expected time*

$$T_{SD}(m, n, t) = \min_{p,\ell} \frac{1}{\mathcal{P}(p, \ell)} \left(T_{\mathsf{Gauss}} + 2L \cdot m \left(1 + \frac{L}{q^\ell}\right)\right)$$

[6] The exponent is inspired by the Strassen matrix multiplication algorithm.

where $T_{\mathsf{Gauss}} = \frac{(m-n-\ell)\cdot m^2}{\log(m-n-\ell)}$, $L = \binom{\frac{n+\ell}{2}}{p/2}\cdot(q-1)^{p/2}$ *and* $\mathcal{P}(p,\ell) = \frac{\binom{m-n-\ell}{t-p}}{\binom{m}{t}}\cdot\frac{L^2}{(q-1)^p}$.[7]

Statistical Decoding framework. While ISD algorithms aim to recover the secret element $\mathbf{s}$ directly, the Low Weight Parity Check (LPC) algorithm focuses on distinguishing between random instances and LPN instances. In other words, it determines whether $\mathbf{b}$ is equal to $\mathbf{A}\cdot\mathbf{s}+\mathbf{e} \bmod \mathbb{F}_q$ or just a random element. Although this algorithm is not well-suited for solving LPN over large fields, we still provide its attack complexity as a comparison for decision problems. For $q \geq 4t$, the adapted SD 2.0 algorithm is proposed in [50].

Lemma 4.2 ([26, 50]) *The adapted SD 2.0 algorithm can solve the decisional* (m,n,t)*-LPN over* $\mathbb{F}_q$ *in time* T *with constant advantage, where* $w = w(s) \in \mathbb{N}$ *and* $q \geq 4t$*. The formula for* T *is given by*

$$T = \min_s \left(T_1 \cdot \left(\frac{\binom{m}{t}}{\binom{m-w}{t}} \cdot \frac{q}{q-1} \right)^2 + s \cdot \log q \cdot q^s \right).$$

Here, T_1 *represents the time it takes to find one parity check vector. Following [17, 50], we also set* $w = n - s + 1$ *and* $T_1 = n + 1$*. As a result, we have*

$$T = \min_s \left((n+1) \cdot \left(\frac{\binom{m}{t}}{\binom{m-n+s-1}{t}} \cdot \frac{q}{q-1} \right)^2 + s \cdot \log q \cdot q^s \right).$$

4.1 Numerical Results of LPN over large fields

This section presents the numerical results obtained from our attack estimation for different parameter settings. We developed a SageMath [63] script to search for near-optimal parameterization for our attack. The original algorithm would have examined the attack complexity that is minimized for all Δ values, but we examined the values via the binary search with depth ≤ 12 to find an approximate minimum quickly with a constraint $\Delta \leq n^{2.8}$. The cost we found also provides a sufficiently small value compared to the original cost. We also provide comparison results between [50] and ours for a few parameters. The results are given by Table 2.

Our results indicate that for the parameters used in [50], the bit-security of LPN over large fields is reduced by 5-11 bits when $\log q = 128$. Consequently, the parameters described in [50] do not achieve 128-bit security. To ensure 128-bit security, we propose new parameter settings, as shown in Table 3.

[7] As the paper [50] only presents an estimation of LPN over $\mathbb{F}_2$, we rely on the formulation provided by their Python script to describe our approach.

[8] [50] also presents numerical results for the statistical decoding 2.0 attack [27]. However, we did not take these results into account because they showed lower performance compared to than Gauss and ISD results in these parameters in the table.

Table 2. Comparison results with previous results [50, Tab. 2] and ours. All values are logarithmic scale of arithmetic operations over $\mathbb{F}_q$ with the field size $\log q = 128$. The column SD indicates the result of algebraic attacks In our works, 'RP' column presents the estimated result of Reduce and Prange algorithm. † indicates estimation results obtained by quick-RP.

Parameters			[50][8]			This work
m	n	t	Gauss	ISD	SD	
2^{10}	652	57	111	111	184	**102**
2^{12}	1589	98	100	100	151	**91**
2^{14}	3482	198	101	101	149	**94**
2^{16}	7391	389	103	103	147	**96**
2^{18}	15536	760	105	105	146	**101**
2^{20}	32771	1419	107	145	**102**	**102**
2^{22}	67440	2735	108	144	**104**	**104**
2^{12}	$3/4 \cdot 2^{12}$	44	117	117	189	**109**
2^{14}	$3/4 \cdot 2^{14}$	39	111	111	170	**105**
2^{16}	$3/4 \cdot 2^{16}$	34	107	107	151	**102**
2^{18}	$3/4 \cdot 2^{18}$	32	108	108	145	**104**†
2^{20}	$3/4 \cdot 2^{20}$	31	112	112	143	**107**†
2^{22}	$3/4 \cdot 2^{22}$	30	116	116	141	**111**†
2^{24}	$3/4 \cdot 2^{24}$	29	119	119	139	**115**†

Remark 1. Some readers might be curious about the configuration of the sub-dimension sets $\{N_i\}_{i=1}^{k}$. Given our observations, with k typically ranging from over 20 to as high as 2000, accurately defining the set can be challenging. Concurrently, it has been noted that the size of N_1 often exceeds $N/3$. Similar occurrences are also observed in the context of the regular-LPN.

4.2 Numerical Results of regular-LPN over large fields

This section provides numerical results for solving regular-LPN for various parameters through a regular-RP algorithm, and gives comparison results with [21, 50]. Here, as in the previous section, we mainly cover regular-LPN over large fields parameters. Thus, we do not cover estimations in [25, 38] specified on regular-LPN over a binary field.

We first note that [21, 50] provides numerical results of the bit-security of regular-LPN over $\mathbb{F}_q$ with $\log q = 128$ since both algorithms rarely depend on the field size. Conversely, [38] solely provides the bit-security of regular-LPN over $\mathbb{F}_2$. Thus, we revisit the algorithms in [38] and re-estimate the bit-security of regular-LPN over $\mathbb{F}_{2^{128}}$.

[38] proposes three types of algorithms: permutation-based regular-ISD (P-ISD), enumeration-based regular-ISD (E-ISD), and representation-based regular-

Table 3. Comparison results with previous results [50, Tab. 3] and ours. All values are logarithmic scale of arithmetic operations over $\mathbb{F}_q$ with the field size $\log q = 128$. In our works, 'RP' column presents the estimated result of Reduce and Prange algorithm. RP for recommended parameters takes at least 2^{128} arithmetic operations.

Parameters			[50]	RP	Recommend Parameters					
					Increase n			Increase t		
m	n	t			m	n	t	m	n	t
2^{12}	1321	172	128	**119**	2^{12}	1400	172	2^{12}	1321	188
2^{14}	2895	338		**120**	2^{14}	2895	338	2^{14}	2895	367
2^{16}	6005	667		**121**	2^{16}	6450	667	2^{16}	6005	720
2^{18}	12160	1312		**122**	2^{18}	12900	1312	2^{18}	12160	1400
2^{20}	25346	2467		**123**	2^{20}	26900	2467	2^{20}	25346	2617
2^{22}	50854	4788		**123**	2^{22}	53490	4788	2^{22}	50854	5050

ISD (R-ISD), each adapting well-known algorithms [12, 52, 59] for the specific task of solving regular-LPN. Among these, we specifically focus on the P-ISD algorithm. This is because the other algorithms require the collection of a large number of collision vectors over $\mathbb{Z}_q$, which significantly increase the concrete complexity. Indeed, numerical results from [50, Tab 3. Tab. 4] indicate that the Gauss costs are always comparable to ISD costs for (regular-)LPN over $\mathbb{F}_{2^{128}}$, although ISD consistently surpasses the Gauss in terms of cost for (regular-)LPN over $\mathbb{F}_2$.

Revisiting P-ISD for regular-LPN over $\mathbb{F}_q$ with $q \gg 2$. As discussed in Section 3.1 and [37, Sec. 4.1], we can just compute the time cost as follows

$$\left(1 - \frac{\binom{n}{t}}{\binom{m}{t}}\right)^{-t} \cdot \min\{n^{2.8}, (m-n)^{2.8}\} = \left(1 - \frac{n}{m}\right)^{-t} \cdot \min\{n^{2.8}, (m-n)^{2.8}\}$$

for solving (m, n, t)-regular-LPN. We further note that in [37,38], they computed the time complexity

$$\left(1 - \frac{n-t}{m}\right)^{-t} \cdot \min\{n^{2.8}, (m-n)^{2.8}\}$$

because one can directly obtain extra t error-free polynomials as in Appendix A. However, in case of regular-LPN over $\mathbb{F}_q$, it does not occur. Hence, we re-estimate the bit-security of regular LPN over $\mathbb{F}_{2^{128}}$ as follows:

$$\left(1 - \frac{n}{m}\right)^{-t} \cdot \min\{n^{2.8}, (m-n)^{2.8}\}.$$

The numerical results are reported in Table 4, which demonstrates that our attack outperforms several parameter settings. For the regular-LPN parameters in [50] that meet 128-bit security, we provide a new parameter setup in Table 5.

However, unlike RP for LPN over large fields, regular-RP may perform worse than existing attacks in some cases, and for these cases, we do not provide new parameters.

Table 4. Numerical results for regular-LPN over $\mathbb{F}_{2^{128}}$. All values are logarithmic scale of the number of the number of arithmetic operations on $\mathbb{F}_{2^{128}}$.

Parameters			[49, 50]	[21]	[38][9]	**This work**
m	n	t				
2^{10}	652	106	194	179	178	**159**
2^{12}	1589	172	155	150	151	**140**
2^{14}	3482	338	150	150	149	**139**
2^{16}	7391	667	151	150	151	**142**
2^{18}	15336	1312	153	**133**	154	147
2^{20}	32771	2467	155	**131**	155	148
2^{22}	67440	4788	152	**110**	156	151
2^{10}	652	57	111	111	107	**98**
2^{12}	1589	98	100	107	99	**89**
2^{14}	3482	198	101	110	101	**94**
2^{16}	7391	389	103	111	103	**97**
2^{18}	15336	760	105	107	105	**100**
2^{20}	32771	1419	107	**102**	107	103
2^{22}	67440	2735	108	**104**	108	105
2^{12}	$3/4 \cdot 2^{12}$	44	117	127	116	**106**
2^{14}	$3/4 \cdot 2^{14}$	39	111	127	111	**105**
2^{16}	$3/4 \cdot 2^{16}$	34	107	128	107	**102**
2^{18}	$3/4 \cdot 2^{18}$	32	108	132	108	**104** †
2^{20}	$3/4 \cdot 2^{20}$	31	112	139	112	**107**†
2^{22}	$3/4 \cdot 2^{22}$	30	116	145	116	**111**†
2^{24}	$3/4 \cdot 2^{24}$	29	119	152	119	**115**†

References

1. C. Aguilar-Melchor et al. The syndrome decoding in the head (sd-in-the-head) signature scheme. submission to the nist call for additional post-quantum signatures (2023). https://csrc.nist.gov/csrc/media/Projects/pqc-dig-sig/documents/round-1/spec-files/SDitH-spec-web.pdf.

[9] This is a re-estimated result of permutation-based algorithm that only suits for regular-LPN over large fields. In particular,

Table 5. Comparison results with previous results [50, Tab. 3] and ours for Numerical results for regular-LPN. All values are logarithmic scale of arithmetic operations over $\mathbb{F}_q$ with the field size $\log q = 128$. The regular-RP for recommend parameters takes at least 2^{128} arithmetic operations.

Parameters			[50]	RP	Recommend Parameters					
					Increase n			Increase t		
m	n	t			m	n	t	m	n	t
2^{12}	1377	172	128	**120**	2^{12}	1460	172	2^{12}	1377	187
2^{14}	2909	338		**118**	2^{14}	3165	338	2^{14}	2909	373
2^{16}	6091	667		**121**	2^{16}	6500	667	2^{16}	6091	713
2^{18}	14796	1312		**142**						
2^{20}	30978	2467		**142**						
2^{22}	75396	4788		**164**						

2. C. Aguilar-Melchor, N. Gama, J. Howe, A. Hülsing, D. Joseph, and D. Yue. The return of the sdith. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 564–596. Springer, 2023.
3. C. Aguilar-Melchor, A. Hülsing, D. Joseph, C. Majenz, E. Ronen, and D. Yue. Sdith in the qrom. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 317–350. Springer, 2023.
4. M. Alekhnovich. More on average case vs approximation complexity. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 298–307. IEEE, 2003.
5. B. Applebaum, B. Barak, and A. Wigderson. Public-key cryptography from different assumptions. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 171–180, 2010.
6. B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 595–618. Springer, 2009.
7. B. Applebaum, I. Damgård, Y. Ishai, M. Nielsen, and L. Zichron. Secure arithmetic computation with constant computational overhead. In *Annual International Cryptology Conference*, pages 223–254. Springer, 2017.
8. D. Augot, M. Finiasz, and N. Sendrier. A family of fast syndrome based cryptographic hash functions. In *Mycrypt*, volume 3715, pages 64–83. Springer, 2005.
9. C. Baum, L. Braun, A. Munch-Hansen, and P. Scholl. Moz$\mathbb{Z}_2^k$arella: efficient vector-ole and zero-knowledge proofs over z 2 k. In *Advances in Cryptology–CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part IV*, pages 329–358. Springer, 2022.
10. C. Baum, A. J. Malozemoff, M. B. Rosen, and P. Scholl. Mac'n'cheese mac′ n′ cheese: Zero-knowledge proofs for boolean and arithmetic circuits with nested disjunctions. In *Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part IV 41*, pages 92–122. Springer, 2021.

11. G. Beck, A. Goel, A. Hegde, A. Jain, Z. Jin, and G. Kaptchuk. Scalable multiparty garbling. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 2158–2172, 2023.
12. A. Becker, A. Joux, A. May, and A. Meurer. Decoding random binary linear codes in 2 n/20: How 1+ 1= 0 improves information set decoding. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 520–536. Springer, 2012.
13. D. J. Bernstein, T. Lange, and C. Peters. Attacking and defending the mceliece cryptosystem. In J. Buchmann and J. Ding, editors, *Post-Quantum Cryptography*, pages 31–46, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.
14. A. Blum, M. L. Furst, M. J. Kearns, and R. J. Lipton. Cryptographic primitives based on hard learning problems. In *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*, pages 278–291. Springer, 1993.
15. A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.
16. L. Both and A. May. Decoding linear codes with high error rate and its impact for lpn security. In *International Conference on Post-Quantum Cryptography*, pages 25–46. Springer, 2018.
17. E. Boyle, G. Couteau, N. Gilboa, and Y. Ishai. Compressing vector OLE. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 896–912. ACM, 2018.
18. E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, L. Kohl, N. Resch, and P. Scholl. Correlated pseudorandomness from expand-accumulate codes. In *Advances in Cryptology–CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part II*, pages 603–633. Springer, 2022.
19. E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, L. Kohl, P. Rindal, and P. Scholl. Efficient two-round ot extension and silent non-interactive secure computation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 291–308, 2019.
20. E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, L. Kohl, and P. Scholl. Efficient pseudorandom correlation generators: Silent ot extension and more. In *Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part III 39*, pages 489–518. Springer, 2019.
21. P. Briaud and M. Øygarden. A new algebraic approach to the regular syndrome decoding problem and implications for pcg constructions. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 391–422. Springer, 2023.
22. D. Bui and G. Couteau. Private set intersection from pseudorandom correlation generators. *IACR Cryptol. ePrint Arch.*, 2022:334, 2022.
23. A. Canteaut and F. Chabaud. A new algorithm for finding minimum-weight words in a linear code: Application to mceliece's cryptosystem and to narrow-sense bch codes of length 511. *IEEE Transactions on Information Theory*, 44(1):367–378, 1998.
24. R. Canto Torres and N. Sendrier. Analysis of information set decoding for a sublinear error weight. In *International Workshop on Post-Quantum Cryptography*, pages 144–161. Springer, 2016.

25. E. Carozza, G. Couteau, and A. Joux. Short signatures from regular syndrome decoding in the head. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 532–563. Springer, 2023.
26. K. Carrier, T. Debris-Alazard, C. Meyer-Hilfiger, and J.-P. Tillich. Statistical decoding 2.0: Reducing decoding to lpn. In *Advances in Cryptology–ASIACRYPT 2022: 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5–9, 2022, Proceedings, Part IV*, pages 477–507. Springer, 2022.
27. K. Carrier, T. Debris-Alazard, C. Meyer-Hilfiger, and J.-P. Tillich. Statistical decoding 2.0: Reducing decoding to lpn. *Asiacrpyt 2022*, 2022.
28. G. Couteau, P. Rindal, and S. Raghuraman. Silver: silent vole and oblivious transfer from hardness of decoding structured ldpc codes. In *Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part III*, pages 502–534. Springer, 2021.
29. I. Damgård and S. Park. Is public-key encryption based on LPN practical? *IACR Cryptol. ePrint Arch.*, 2012:699, 2012.
30. I. Damgård, V. Pastro, N. Smart, and S. Zakarias. Multiparty computation from somewhat homomorphic encryption. In *Advances in Cryptology–CRYPTO 2012: 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, pages 643–662. Springer, 2012.
31. S. Dittmer, Y. Ishai, S. Lu, and R. Ostrovsky. Authenticated garbling from simple correlations. In *Advances in Cryptology–CRYPTO 2022: 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15–18, 2022, Proceedings, Part IV*, pages 57–87. Springer, 2022.
32. S. Dittmer, Y. Ishai, S. Lu, and R. Ostrovsky. Improving line-point zero knowledge: Two multiplications for the price of one. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 829–841, 2022.
33. S. Dittmer, Y. Ishai, S. Lu, and R. Ostrovsky. Improving line-point zero knowledge: Two multiplications for the price of one. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 829–841, 2022.
34. S. Dittmer, Y. Ishai, and R. Ostrovsky. Line-point zero knowledge and its applications. *Cryptology ePrint Archive*, 2020.
35. Y. Dodis, E. Kiltz, K. Pietrzak, and D. Wichs. Message authentication, revisited. In D. Pointcheval and T. Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*, pages 355–374. Springer, 2012.
36. N. Döttling, S. Ghosh, J. B. Nielsen, T. Nilges, and R. Trifiletti. Tinyole: Efficient actively secure two-party computation from oblivious linear function evaluation. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS*, page 2263–2276, New York, NY, USA, 2017. Association for Computing Machinery.
37. A. Esser and P. Santini. Not just regular decoding: Asymptotics and improvements of regular syndrome decoding attacks. Cryptology ePrint Archive, Paper 2023/1568, 2023. https://eprint.iacr.org/2023/1568.
38. A. Esser and P. Santini. Not just regular decoding: Asymptotics and improvements of regular syndrome decoding attacks. In *Annual Cryptology Conference*. Springer-Verlag, 2024.

39. T. Feneuil, A. Joux, and M. Rivain. Syndrome decoding in the head: Shorter signatures from zero-knowledge proofs. In *Annual International Cryptology Conference*, pages 541–572. Springer, 2022.
40. N. Franzese, J. Katz, S. Lu, R. Ostrovsky, X. Wang, and C. Weng. Constant-overhead zero-knowledge for ram programs. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 178–191, 2021.
41. S. Ghosh, J. B. Nielsen, and T. Nilges. Maliciously secure oblivious linear function evaluation with constant overhead. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 629–659. Springer, 2017.
42. S. Gueron, E. Persichetti, and P. Santini. Designing a practical code-based signature scheme from zero-knowledge proofs with trusted setup. *Cryptography*, 6(1):5, 2022.
43. C. Hazay, P. Scholl, and E. Soria-Vazquez. Low cost constant round mpc combining bmr and oblivious transfer. *Journal of Cryptology*, 33(4):1732–1786, 2020.
44. Y. Ishai, M. Prabhakaran, and A. Sahai. Secure arithmetic computation with no honest majority. In *Theory of Cryptography Conference*, pages 294–314. Springer, 2009.
45. A. Jain, S. Krenn, K. Pietrzak, and A. Tentes. Commitments and efficient zero-knowledge proofs from learning parity with noise. In *Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings*, volume 7658 of *Lecture Notes in Computer Science*, pages 663–680. Springer, 2012.
46. A. Jain, H. Lin, and A. Sahai. Indistinguishability obfuscation from well-founded assumptions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 60–73, 2021.
47. M. Keller, E. Orsini, and P. Scholl. Mascot: faster malicious arithmetic secure computation with oblivious transfer. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 830–842, 2016.
48. E. Kiltz, K. Pietrzak, D. Venturi, D. Cash, and A. Jain. Efficient authentication from hard learning problems. *J. Cryptol.*, 30(4):1238–1275, 2017.
49. H. Liu, X. Wang, K. Yang, and Y. Yu. The hardness of lpn over any integer ring and field for pcg applications. Cryptology ePrint Archive, Paper 2022/712, 2022. https://eprint.iacr.org/2022/712.
50. H. Liu, X. Wang, K. Yang, and Y. Yu. The hardness of lpn over any integer ring and field for pcg applications. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 149–179. Springer, 2024.
51. V. Lyubashevsky and D. Masny. Man-in-the-middle secure authentication schemes from LPN and weak prfs. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 308–325. Springer, 2013.
52. A. May, A. Meurer, and E. Thomae. Decoding random linear codes in $\tilde{O}(2^{0.054n})$. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 107–124. Springer, 2011.
53. A. May and I. Ozerov. On computing nearest neighbors with applications to decoding of binary linear codes. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 203–228. Springer, 2015.

54. A. Meurer. *A coding-theoretic approach to cryptanalysis*. PhD thesis, Verlag nicht ermittelbar, 2013.
55. D. Micciancio and P. Mol. Pseudorandom knapsacks and the sample complexity of lwe search-to-decision reductions. In *Advances in Cryptology–CRYPTO 2011: 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings 31*, pages 465–484. Springer, 2011.
56. M. Naor and B. Pinkas. Oblivious polynomial evaluation. *SIAM Journal on Computing*, 35(5):1254–1281, 2006.
57. J. B. Nielsen, P. S. Nordholt, C. Orlandi, and S. S. Burra. A new approach to practical active-secure two-party computation. In *Advances in Cryptology–CRYPTO 2012: 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, pages 681–700. Springer, 2012.
58. C. Peters. Information-set decoding for linear codes over $\mathfrak{U}_q$. In *International Workshop on Post-Quantum Cryptography*, pages 81–94. Springer, 2010.
59. E. Prange. The use of information sets in decoding cyclic codes. *IRE Transactions on Information Theory*, 8(5):5–9, 1962.
60. S. Raghuraman and P. Rindal. Blazing fast psi from improved okvs and subfield vole. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2505–2517, 2022.
61. P. Rindal and P. Schoppmann. Vole-psi: fast oprf and circuit-psi from vector-ole. In *Advances in Cryptology–EUROCRYPT 2021: 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part II*, pages 901–930. Springer, 2021.
62. P. Schoppmann, A. Gascón, L. Reichert, and M. Raykova. Distributed vector-ole: improved constructions and implementation. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1055–1072, 2019.
63. The Sage Developers. *SageMath, the Sage Mathematics Software System (Version x.y.z)*, YYYY. `https://www.sagemath.org`.
64. X. Wang, S. Ranellucci, and J. Katz. Authenticated garbling and efficient maliciously secure two-party computation. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 21–37, 2017.
65. X. Wang, S. Ranellucci, and J. Katz. Global-scale secure multiparty computation. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 39–56, 2017.
66. C. Weng, K. Yang, J. Katz, and X. Wang. Wolverine: fast, scalable, and communication-efficient zero-knowledge proofs for boolean and arithmetic circuits. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1074–1091. IEEE, 2021.
67. C. Weng, K. Yang, Z. Yang, X. Xie, and X. Wang. Antman: Interactive zero-knowledge proofs with sublinear communication. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2901–2914, 2022.
68. K. Yang, P. Sarkar, C. Weng, and X. Wang. Quicksilver: Efficient and affordable zero-knowledge proofs for circuits and polynomials over any field. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 2986–3001, 2021.
69. K. Yang, X. Wang, and J. Zhang. More efficient mpc from improved triple generation and authenticated garbling. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 1627–1646, 2020.

70. K. Yang, C. Weng, X. Lan, J. Zhang, and X. Wang. Ferret: Fast extension for correlated ot with small communication. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 1607–1626, 2020.
71. Y. Yu and J. P. Steinberger. Pseudorandom functions in almost constant depth from low-noise LPN. In *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 154–183. Springer, 2016.
72. Y. Yu and J. Zhang. Cryptography with auxiliary input and trapdoor from constant-noise LPN. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 214–243. Springer, 2016.

A Algorithm for Regular-LPN over Binary Field

According to [21], the regularity of an error distribution τ provides an extra relations: For any $1 \leq i \leq t$, we observe that

$$1 - \sum_{j=1}^{\beta} e_{i,j} = 0,$$

where $e_{i,j}$ is the j-th entry of a vector $\mathbf{e}_i$. Consequently, the attack can easily get t linear equations from the regularity of $\tau_{m,t}$.

We can parse $\mathbf{b} = (\mathbf{b}_1 \| \cdots \| \mathbf{b}_t)$ where $\mathbf{b}_i \in \mathbb{F}_2^{\beta}$, and our attack exploits this fact as follows: For every $1 \leq i \leq t$,

$$\begin{aligned}\sum_{j=1}^{\beta} b_{i,j} &= (\sum_{j=1}^{\beta} \mathbf{a}_j^T) \cdot \mathbf{s} + \sum_{j=1}^{\beta} e_{i,j} \bmod 2 \\ &= (\sum_{j=1}^{\beta} \mathbf{a}_j^T) \cdot \mathbf{s} + 1 \bmod 2\end{aligned}$$

Here, $b_{i,j}$ is the j-th entry of $\mathbf{b}_i$, and $\mathbf{a}_j^T$ is the j-th row vector of $\mathbf{A}$. Thus, we automatically obtain an error-free polynomial g_i of the form

$$g_i(\mathbf{x}) = \sum_{j=1}^{\beta} b_{i,j} - 1 - (\sum_{j=1}^{\beta} \mathbf{a}_j^T) \cdot \mathbf{x}.$$

This exploitation can significantly enhance our attack. Specifically, the purpose of RP is to collect error-free polynomials, and the t error-free polynomials obtained from the regularity of $\tau_{m,t}$ reduce the number of error-free polynomials that we need to obtain. Consequently, we need only $n - t$ error-free polynomials to solve regular-LPN.

The remaining part is to obtain $n - t$ error-free polynomial through RP in Section 3.1.