A Review of “Post-Quantum Homomorphic Encryption: A Case for Code-Based Alternatives”

AmirMatin Shahnazi

Sharif University of Technology, Tehran, Iran

Email: *matin.shahnazi@sharif.edu*

Abstract

This overview summarizes the key contributions of the paper titled *Post-Quantum Homomorphic Encryption: A Case for Code-Based Alternatives*.[11] The original work explores the landscape of post-quantum homomorphic encryption (PQHE), emphasizing the potential of error-correcting code-based schemes as alternatives to the dominant lattice-based constructions. It outlines the mathematical foundations of code-based cryptography, compares its security and efficiency with lattice-based approaches, and identifies core challenges that have limited its adoption. The paper concludes by proposing five research directions to advance code-based PQHE.

1 Introduction

1.1 Motivation

Homomorphic encryption (HE) has emerged as a transformative cryptographic primitive, enabling computation directly on encrypted data without requiring decryption. This property unlocks secure outsourcing of computation, privacy-preserving machine learning, and encrypted database queries — all while maintaining data confidentiality. As digital ecosystems increasingly rely on cloud-based and distributed architectures, HE offers a compelling solution to the tension between utility and privacy. However, the security of many widely deployed cryptosystems, including RSA and ECC, hinges on the hardness of problems like integer factorization and discrete logarithms. These assumptions are fundamentally threatened by quantum algorithms such as Shor’s and Grover’s algorithms, which can solve these problems in polynomial time. The advent of scalable quantum computers would render traditional cryptographic schemes obsolete, necessitating a shift toward post-quantum cryptography (PQC). Among the leading candidates for PQC are lattice-based and code-based cryptographic constructions. Lattice-based schemes, particularly those built on the Learning With Errors (LWE) and Ring-LWE problems, have gained prominence due to their versatility and efficiency. However, recent developments have raised concerns about their long-term viability. In a 2024 paper, Yilei Chen proposed a polynomial-time quantum algorithm for solving LWE and RLWE under certain parameter regimes[16]. Although a critical flaw was later identified in the algorithm's Step 9, invalidating the main claim, the episode underscores the risk of relying exclusively on lattice-based assumptions. This motivates a broader exploration of alternative post-quantum foundations. Code-based cryptography, rooted in NP-hard problems such as decoding random linear codes (DP) and syndrome decoding problem (SDP), offers a compelling complement. These problems have withstood decades of cryptanalytic scrutiny and remain resistant to known quantum attacks. By revisiting code-based constructions in the context of homomorphic encryption, researchers can diversify the cryptographic landscape and mitigate systemic risks associated with over-reliance on any single hardness assumption.While lattice-based homomorphic encryption is currently the focus of much research due to its maturity and optimization, code-based approaches offer a complementary perspective.

2 Preliminaries

This section provides an overview of homomorphic encryption (HE), explaining its basic principles and levels of capability. Homomorphic encryption allows computations on encrypted data without decryption. Each of the different types: partially homomorphic encryption (PHE), somewhat homomorphic encryption (SWHE), and fully homomorphic encryption (FHE), offers varying degrees of flexibility and efficiency.

Notation

For an integer $n \geq 1$, we denote by $[n]$ the set of integers $1, \dots, n$. In the following, $\mathbb{F}$ will be some arbitrary finite field.

2.1 Homomorphic Encryption: Definitions

Homomorphic encryption (HE) is a cryptographic technique that allows computations to be performed on encrypted data without the need for decryption. This property enables the processing and analysis of sensitive information while preserving its confidentiality and integrity.

2.1.1 Formal definition of Homomorphic Encryption

Formally, a homomorphic encryption scheme can be defined as a tuple of probabilistic polynomial-time (PPT) algorithms, denoted as $\text{HE} = (\text{KeyGen}, \text{Enc}, \text{Eval}, \text{Dec})$ [5]. Each algorithm is described in detail below:

- **Key Generation (KeyGen):** $\text{HE.KeyGen}(1^\lambda) \to (pk, sk, ek)$: Given a security parameter λ that determines the level of security, the key generation algorithm outputs a public key pk, a secret key sk, and an evaluation key ek. The public key is used for encryption, the secret key for decryption, and the evaluation key for performing homomorphic operations.

- **Encryption (Enc):** $\text{HE.Enc}(pk, m) \to c$: Given a message m and the public key pk, the encryption algorithm outputs a ciphertext c. The ciphertext c is an encrypted version of the message m, which can be processed homomorphically without revealing the underlying plaintext.

- **Evaluation (Eval):** $\text{HE.Eval}(ek, f, c, c') \to c_{\text{eval}}$: Given two ciphertexts c and c', an evaluation key ek, and a homomorphic function f, the evaluation algorithm outputs an evaluated ciphertext $c_{\text{eval}} = f(c, c')$. The evaluation key ek is used to enable homomorphic operations, and it plays a crucial role in the bootstrapping process, which refreshes the ciphertext to allow for further computations.

- **Decryption (Dec):** HE.Dec$(sk, c) \to m$: Given a ciphertext c encrypted under the public key pk, the decryption algorithm outputs the original message m using the corresponding secret key sk. This ensures that only the holder of the secret key can access the decrypted data.

2.1.2 Mathematical definition of Homomorphic Encryption

Mathematically, a homomorphic encryption (HE) scheme is defined as follows:

Definition 1. *A homomorphic encryption (HE) scheme is an encryption scheme that satisfies the following property: For all ciphertexts $C_1, C_2 \in \mathcal{C}$, all plaintexts $M_1, M_2 \in \mathcal{P}$, and for any key K, if*

$$C_1 = \text{Enc}(M_1) \quad \textit{and} \quad C_2 = \text{Enc}(M_2),$$

then

$$\text{Dec}\big(C_1 \circ C_2\big) = M_1 \odot M_2. \tag{1}$$

Here, $\circ$ and $\odot$ denote the group operations in the ciphertext space $\mathcal{C}$ and the plaintext space $\mathcal{P}$, respectively.

addition and field addition.

2.2 Linear Codes

2.2.1 Hamming Weight and Distance

Definition 2 (**Hamming Weight**)**.** *The Hamming Weight $w_H(\mathbf{v})$ of a vector $\mathbf{v} \in \mathbb{F}$ is the number of non-zero components.*

Definition 3 (**Hamming Distance**)**.** *The Hamming Distance between two vector $\mathbf{v}, \mathbf{u} \in \mathbb{F}$ is $d_H(\mathbf{u}, \mathbf{v}) = w_H(\mathbf{u} - \mathbf{v})$.*

Definition 4 (**Minimum Hamming Distance**)**.** *The Minimum Hamming Distance d_{min} of a code $\mathcal{C}$ is the smallest Hamming distance between any two distinct codewords in $\mathcal{C}$. It determines the error-detecting and correcting capability:*

- *Detects up to t errors if $d_{\min} \geq t + 1$,*
- *Corrects up to t errors if $d_{\min} \geq 2t + 1$.*

2.2.2 Code

Definition 5 (**Linear Codes**)**.** *A $[n, k, d]$ linear code (with $k < d$) is a k-dimensional linear subspace $\mathcal{C}$ of $\mathbb{F}^n$ with minimum Hamming distance d where $\mathbb{F}$ is a finite field. That is, it holds for all $c, c' \in \mathcal{C}$ that $c + c' \in \mathcal{C}$ and $|c + c'| > d$. We call vectors $\mathbf{c} \in \mathcal{C}$ as* (error-free) codewords *and vectors $\mathbf{c} \in \mathbb{F} \backslash 0$ as erroneous codewords. Erroneous codewords can be written as $\mathbf{c} + \mathbf{e}$ where $\mathbf{e} \in \mathbb{F}^n \backslash 0$ is called the error vector. The* bad locations*, that is where an error occurred, are* supp(e) *and the* good locations*, that is the error-free locations, are* $[n]\backslash\text{supp}(e)$*. For a subset* $I \in [n]$*, we define:*

$$\mathcal{C}(I) := \{\mathbf{c} + \mathbf{e} | \mathbf{c} \in \mathcal{C}, \mathbf{e} \in \mathbb{F}^n, \text{supp}(\mathbf{e}) \subseteq [n]\backslash I\}$$

Definition 6 (Generator Matrix)**.** *Let* $\mathcal{C} \subseteq \mathbb{F}_q^n$ *be a linear code of dimension* k*. A* generator matrix $\mathbf{G} \in \mathbb{F}_q^{k \times n}$ *is a matrix whose rows form a basis for* $\mathcal{C}$*. Every codeword* $\mathbf{c} \in \mathcal{C}$ *can be expressed as:*

$$\mathbf{c} = \mathbf{m} \cdot \mathbf{G}$$

where $\mathbf{m} \in \mathbb{F}_q^k$ *is the message vector. If* $\mathbf{G}$ *is in standard form, it takes the shape* $\mathbf{G} = [\mathbf{I}_k \mid \mathbf{P}]$*, where* $\mathbf{I}_k$ *is the identity matrix and* $\mathbf{P}$ *is a parity component.*

Definition 7 (Parity-Check Matrix)**.** *A* parity-check matrix $\mathbf{H} \in \mathbb{F}_q^{(n-k) \times n}$ *defines the set of linear constraints that valid codewords must satisfy. A vector* $\mathbf{c} \in \mathbb{F}_q^n$ *is a codeword in* $\mathcal{C}$ *if and only if:*

$$\mathbf{H} \cdot \mathbf{c}^T = \mathbf{0}$$

If the generator matrix $\mathbf{G}$ *is in standard form* $[\mathbf{I}_k \mid \mathbf{P}]$*, then the corresponding parity-check matrix is given by:*

$$\mathbf{H} = [-\mathbf{P}^T \mid \mathbf{I}_{n-k}]$$

In binary codes (i.e., over $\mathbb{F}_2$*), negation is equivalent to identity, so* $-\mathbf{P}^T = \mathbf{P}^T$*.*

2.3 Hardness Assumptions and Computational Problems

In this section, we outline and formally define the computational assumptions and foundational problems that underpin the encryption schemes discussed throughout the paper.

- **Sparse Subset Sum Problem (SSSP):** Given a set of integers $S = \{a_1, \ldots, a_n\} \subseteq \mathbb{Z}$, determine whether there exists a sparse subset $I \subseteq \{1, \ldots, n\}$ such that $\sum_{i \in I} a_i = 0$.
- **Bounded Distance Decoding Problem (BDD):** Given a lattice and a target vector that is close to the lattice, find the closest lattice vector.
- **Ideal Shortest Vector Problem (Ideal SVP):** Find the shortest non-zero vector in an ideal lattice.
- **Approximate Greatest Common Divisor (AGCD) Problem:** Given a set of integers $x_i = p \cdot q_i + r_i$, where p is a secret odd integer and r_i are small noise terms, the AGCD problem requires recovering p. This problem is believed to be computationally hard, even for quantum computers.
- **Circular Security:** The Circular Security means that the encryption of the secret key p does not compromise the overall security of the scheme.
- **Decisional Small Polynomial Ratio (DSPR) Problem:** This problem involves distinguishing between a random polynomial and a ratio of two small polynomials in the ring R. The DSPR problem is believed to be computationally hard, even for quantum computers.
- **γ-Approximate Shortest Vector Problem (SVP_γ):**

  - Given a lattice L and an approximation factor $\gamma \geq 1$, the goal is to find a non-zero vector $v \in L$ such that:
$$\|v\| \leq \gamma \cdot \lambda_1(L),$$
where $\lambda_1(L)$ is the length of the shortest non-zero vector in L.

- **Decisional Shortest Vector Problem ($GapSVP_{\gamma,r}$):**
  - Given a lattice L, an approximation factor $\gamma \geq 1$, and a bound $r > 0$, the goal is to decide whether:
$$\lambda_1(L) \leq r \quad \text{or} \quad \lambda_1(L) \geq \gamma \cdot r.$$

- **γ-Unique Shortest Vector Problem ($uSVP_\gamma$):**
  - Given a lattice L and an approximation factor $\gamma \geq 1$, the goal is to find the shortest non-zero vector $v \in L$ under the condition that:
$$\lambda_1(L) < \gamma \cdot \lambda_2(L),$$
where $\lambda_2(L)$ is the length of the second shortest vector in L.

- **γ-Approximate Closest Vector Problem (CVP_γ):**
  - Given a lattice L, a target vector $t \in \mathbb{R}^n$, and an approximation factor $\gamma \geq 1$, the goal is to find a vector $v \in L$ such that:
$$\text{dist}(t, v) \leq \gamma \cdot \text{dist}(t, L),$$
where $\text{dist}(t, L)$ is the distance between t and the lattice L.

- **Decisional Closest Vector Problem ($DCVP_{\gamma,r}$):**
  - Given a lattice L, a target vector $t \in \mathbb{R}^n$, an approximation factor $\gamma \geq 1$, and a bound $r > 0$, the goal is to decide whether:
$$\text{dist}(t, L) \leq r \quad \text{or} \quad \text{dist}(t, L) \geq \gamma \cdot r.$$

- **α-Bounded Distance Decoding (BDD_α):**
  - Given a lattice L, a target vector $t \in \mathbb{R}^n$, and a parameter $\alpha \leq 1$, the goal is to find the closest lattice vector $v \in L$ under the condition that:
$$\text{dist}(t, L) < \alpha \cdot \lambda_1(L).$$

- **Shortest Integer Solution (SIS) Problem ($SIS_{q,m,\beta}$):**
  - Given a matrix $A \in \mathbb{Z}_q^{m \times n}$ and a bound $\beta < q$, the goal is to find a non-zero integer vector $x \in \mathbb{Z}^m$ such that:
$$xA \equiv 0 \pmod{q} \quad \text{and} \quad \|x\| \leq \beta.$$

- **LWE Problem:**

  - Given a matrix $A \in \mathbb{Z}_q^{m \times n}$, a secret vector $s \in \mathbb{Z}_q^n$, and a noise vector $e \in \mathbb{Z}_q^m$ sampled from a discrete Gaussian distribution, the LWE instance is:

$$b = As + e \pmod{q}.$$

    The goal is to recover the secret vector s given (A, b).

- **Ring-LWE Problem:**
  - Given a polynomial ring $R_q = \mathbb{Z}[x]/\langle f(x) \rangle$, a secret polynomial $s(x) \in R_q$, and a noise polynomial $e(x) \in R_q$, the Ring-LWE instance is:

$$b(x) = a(x)s(x) + e(x) \pmod{f(x)},$$

    where $a(x) \in R_q$ is a random polynomial. The goal is to recover the secret polynomial $s(x)$ given $(a(x), b(x))$.

3 Background on Homomorphic Encryption

3.1 Levels of Homomorphism

HE schemes are often classified into three levels based on the types and number of homomorphic operations that can be supported. These are:

- **Partially Homomorphic Encryption (PHE)**: Supports only one operation (e.g., addition or multiplication).
- **Somewhat Homomorphic Encryption (SHE)**: Supports all operations, but has limitations on the number of operations of a certain type.
- **Fully Homomorphic Encryption (FHE)**: Enables unlimited arbitrary computations on encrypted data.

Table 1 compares the salient features of the different levels of homomorphic encryption.

3.2 The evolution of FHE schemes

FHE schemes were classified in 2017 into *generations* based on computational efficiency and the techniques used to manage noise during ciphertext evaluation [4]. This section briefly outlines the four generations of FHE schemes, listing the representative algorithms and their underlying security assumptions. For a comprehensive discussion of each scheme's construction, performance, and limitations, readers are referred to the original paper[11]. Table 2 provides a comparative summary of the four generations. Notably, code-based techniques have not been employed in any of these generations.

- **First Generation**
  - **Scheme(s):** Gentry (2009)
    - **Assumptions:** Ideal SVP, Sparse Subset Sum Problem (SSSP), BDD

Table 1: Features of Partial HE, Somewhat HE, and Fully HE.

Feature	**Partial Homomorphic Encryption (PHE)**	**Somewhat Homomorphic Encryption (SWHE)**	**Fully Homomorphic Encryption (FHE)**
Supported Operations	Supports one type of operation (either addition or multiplication).	Supports both addition and multiplication but only for a limited number of operations (shallow circuit depth).	Supports arbitrary computations (both addition and multiplication without a fixed limit).
Noise Growth	Minimal noise accumulation, making these schemes computationally efficient.	Noise accumulates with each operation (linear for additions; quadratic or even exponential for multiplications), limiting computation depth.	Requires advanced noise management techniques (bootstrapping, modulus switching, relinearization) to control error growth over arbitrary computations.
Complexity & Efficiency	Generally the most efficient and simplest in design (e.g., RSA, Paillier, ElGamal).	More complex than PHE; efficiency is affected by the need to manage noise, often limiting practical depth without additional techniques.	Most complex and computationally intensive due to elaborate noise control and the necessity for periodic refreshment of ciphertexts.
Common Examples	RSA (multiplicative), Paillier (additive), and ElGamal (multiplicative or adapted to additive in variants).	Early schemes such as Gentry's original SWHE, later improved in the Brakerski–Gentry–Vaikuntanathan (BGV) and Fan–Vercauteren (FV) schemes.	Advanced lattice-based schemes including Gentry's FHE, GSW, CKKS, among others, each incorporating techniques to enable fully homomorphic evaluation.
Security Assumptions	Relies on hardness problems such as the Integer Factorization Problem (IFP), Decisional Composite Residuosity (DCR), or the Discrete Logarithm Problem (DLP).	Based on lattice problems such as Learning With Errors (LWE) or Ring-LWE, with security intricately linked to noise management.	Built on advanced lattice assumptions (Ideal Lattice, LWE, RLWE, and variants) and often require additional measures to maintain security under extensive operations.

  - **Scheme(s):** DGHV
    - **Assumptions:** AGCD, SSSP, Circular Security

- **Second Generation**
  - **Scheme(s):** BV, BGV, FV
    - **Assumptions:** Learning With Errors (LWE), Ring Learning With Errors (RLWE)
  - **Scheme(s):** NTRU-Based FHE (LTV)
    - **Assumptions:** RLWE, DSPR, Circular Security

- **Third Generation**
  - **Scheme(s):** GSW
    - **Assumptions:** LWE

- **Fourth Generation**
  - **Scheme(s):** CKKS (HEAAN)
    - **Assumptions:** LWE, RLWE

3.3 Security Assumptions of current FHE schemes

The security of current lattice-based Fully Homomorphic Encryption (FHE) schemes is grounded in the hardness of problems such as SVP, CVP, SIS, LWE, and Ring-LWE. These problems are believed to be resistant to quantum attacks, making lattice-based cryptography a promising candidate for post-quantum security. While lattice reduction algorithms pose a potential threat, careful parameter selection and error distribution management can mitigate these risks, ensuring the robustness of lattice-based cryptographic schemes.

Lattice-based homomorphic encryption (HE) schemes, like those based on Learning With Errors (RLWE) and NTRU, have made notable strides in efficiency and security. However, they still face challenges such as large ciphertext expansion, high computational overhead, and susceptibility to quantum attacks. Concerns about their long-term security arise from their reliance on lattice problems. To address these limitations, researchers are turning to alternative hardness assumptions. Code-based cryptography, known for its strength in post-quantum encryption, offers potential benefits for HE, including robust security and efficiency. Exploring code-based HE schemes may help overcome the shortcomings of lattice approaches while maintaining resistance to quantum threats.

Table 2: Comparison of Fully Homomorphic Encryption (FHE) Models with respect to the underlying hard problem, the key techniques, advantages, disadvantages and whether bootstrapping is necessary.

FHE Model	**Underlying Assumption / Hard Problem**	**Key Techniques**	**Advantages**	**Disadvantages**	**Bootstrapping Requirements**
First Generation: Ideal Lattice FHE (Gentry's Scheme)	Ideal lattice problems (e.g., SVP, SSSP, BDD)	Squashing to reduce decryption circuit complexity; Bootstrapping	Foundational breakthrough; supports arbitrary computations	High circuit complexity; challenging noise management; computationally expensive	Bootstrapping enabled via squashing (adds overhead)
First Generation: AGCD-Based FHE (DGHV Scheme)	Approximate GCD, combined with SSSP	Integer arithmetic with modular noise reduction	Conceptually simple; non-lattice alternative	Large public key size; high computational cost; less efficient in practice	Bootstrapping is required to control noise accumulation
Second Generation: LWE/RLWE-Based FHE (BV, BGV, FV Schemes)	Learning With Errors (LWE) and its ring variant (RLWE)	Modulus switching, relinearization, batching, scale invariance	More practical and efficient; implemented in libraries such as Microsoft SEAL	Noise growth limits circuit depth without bootstrapping; parameter tuning can be complex	Leveled FHE variants may avoid bootstrapping for fixed-depth circuits; bootstrapping still used for deeper evaluations
Second Generation: NTRU-Based FHE	NTRU assumption and circular security	Bootstrapping and modulus switching adapted to NTRU structure	Initially faster encryption operations	Vulnerabilities discovered; requires larger parameters for security; largely deprecated	Bootstrapping is incorporated, but overall robustness is lower
Third Generation: GSW-Based FHE (Approximate Eigenvector Method)	LWE/RLWE with an approximate eigenvector approach	Bit decomposition; optimized bootstrapping with reduced error growth	Improved noise management; supports deeper circuits with less noise amplification	Increased communication cost due to larger ciphertext sizes; higher computational overhead	Bootstrapping is more efficient, reducing overall complexity
Fourth Generation: CKKS Scheme	RLWE tailored for approximate arithmetic	Leveled encryption; plaintext embedding into complex number vectors; modulus scaling	Efficient for real-valued and approximate computations (e.g., ML applications)	Inherent approximation errors; requires careful precision management	Designed primarily as a leveled scheme (bootstrapping is optional)

4 Code-Based Cryptography

Code-based cryptography builds its security foundation on the computational hardness of problems in coding theory, most notably the *decoding of a random linear code* and the *difficulty of proving equivalence of codes*, which are known to be NP-hard. In the post-quantum era, where traditional number-theoretic assumptions are vulnerable to quantum attacks, such coding-theoretic problems offer a resilient alternative. These schemes typically rely on the existence of a well-structured secret code equipped with an efficient decoding algorithm, referred to as the *trapdoor* s. Specifically, given a received word of the form $\mathbf{c} = \mathbf{m} + \mathbf{e}$, where $\mathbf{m} \in \mathbb{F}_q^n$ is the original message and $\mathbf{e} \in \mathbb{F}_q^n$ is an error vector of Hamming weight $\mathrm{wt}(\mathbf{e}) < t$, the legitimate receiver — possessing s — can efficiently recover $\mathbf{m}$. For adversaries, however, the code appears random, rendering the decoding problem computationally infeasible.

In this section we first introduce the NP-hard problems on which code-based cryptography is based. We will then introduce existing code-based ciphers.

4.1 NP-Hard Problems in Coding Theory

There are two categories of NP-hard problems in this area. The first is around the difficulty of decoding a random linear code and the second is around the difficulty of proving equivalence of codes.

4.1.1 The Decoding Problems in Coding Theory

There are three variations of decoding problems, each of which is described below:

Decoding Problem (DP):
Let $\mathbb{F}_q$ be a finite field and consider a code defined by a generator matrix

$$G \in \mathbb{F}_q^{k \times n}.$$

Given a received vector $r \in \mathbb{F}_q^n$ and an integer t (representing an error weight threshold), the task is to decide whether there exists a message $m \in \mathbb{F}_q^k$ and an error vector $e \in \mathbb{F}_q^n$ with

$$\mathrm{wt}(e) \leq t, \text{ such that, } r = mG + e.$$

This problem is fundamental in algebraic coding theory.

Syndrome Decoding Problem (SDP):
In the syndrome formulation, one is given a parity-check matrix

$$H \in \mathbb{F}_q^{(n-k) \times n},$$

a syndrome $s \in \mathbb{F}_q^{n-k}$, and an integer t. The goal is to find an error vector $e \in \mathbb{F}_q^n$ satisfying

$$eH^\top = s \quad \text{and} \quad \mathrm{wt}(e) \leq t.$$

By converting G into systematic form and obtaining the corresponding H, the DP can be recast as an SDP, and conversely, one can recover a DP instance from an SDP instance.

Given Weight Codeword Problem (GWCP):
Given a parity-check matrix H and an integer w, the problem asks whether there exists a codeword

$$c \in \mathbb{F}_q^n,$$

such that

$$cH^\top = 0 \quad \text{and} \quad \mathrm{wt}(c) = w.$$

By augmenting the generator matrix with the received vector, one shows that GWCP is equivalent to the DP (and hence to the SDP).

4.1.2 Code Equivalence Problems

Code equivalence problem is known as the indistinguishability of a random linear code. Similar to decoding problems, there are three varieties of code equivalence problems.

Permutation Equivalence Problem (PEP):
Given two generator matrices

$$G,\, G' \in \mathbb{F}_q^{k\times n},$$

find a permutation $\phi \in S_n$ (the symmetric group on n elements) such that

$$\phi(\langle G\rangle) = \langle G'\rangle.$$

This problem is a special case of the broader linear equivalence issues.

Linear Equivalence Problem (LEP):
For $G,\, G' \in \mathbb{F}_q^{k\times n}$, the goal is to find a mapping

$$\phi \in (\mathbb{F}_q^\star)^n \rtimes S_n,$$

that sends the code generated by G to that generated by G'.

Subcode Equivalence and the Permuted Kernel Problem (PKP/SEP):

- **PKP:** Given $G \in \mathbb{F}_q^{k\times n}$ and another matrix H' (typically related to a subcode), find a permutation matrix P such that
$$H'(GP)^\top = 0.$$
- **SEP:** Reformulated as the subcode equivalence problem, one seeks a permutation matrix P such that
$$\langle G'\rangle \subset \langle GP\rangle.$$

A relaxed version of PKP requires only finding a non-zero codeword (i.e. a subcode of dimension 1) that meets the equivalence condition.

In summary, the Decoding Problem (DP), the Syndrome Decoding Problem (SDP), and the Given Weight Codeword Problem (GWCP) are shown to be equivalent formulations central to algebraic coding theory. These problems underpin many cryptographic constructions based on error-correcting codes.

4.2 Frameworks in Code-Based Cryptography

This section describes the two existing cryptosystems built on NP-hard problems in coding theory. We will discuss the McEleice framework and the Niederiter Framework.In addition to these there are two variants of the Alekhnovich Framework[11], the Quasi-cyclic framework [1], Augot-Finiasz (AF) cryptosystem [8] and the GPT cryptosystem [19] which are widely in use and are derivative variations of the above three frameworks. Details of all the code-based frameworks can be found in [27].

4.2.1 McEliece Framework

McEliece [23] proposed a public key cryptosystem using a linear code, for example, a binary Goppa code. The secret key is chosen as one of the many possible generators of a chosen linear code. The public key is a new matrix created by adding randomness and permuting the generator. The new matrix looks like a random matrix and will not leak information about the actual generator. A sender will multiply this random matrix with the plaintext and the product will be added to a random binary error vector whose weight will be less than the error-correcting bound of the chosen linear code. The resulting ciphertext is sent back to the owner of the secret key. To decode the ciphertext, the receiver first performs the decoding algorithm to remove the error vector added in the encoding, then performs the inverse of the operations done to generate the random matrix and retrieve the original message. Without knowledge of the generator and the random transformation, an attacker observing the ciphertext will find it computationally hard to find the message.

The McEliece framework is described in Algorithm 1. The parameters of the framework are (q, n, k, t) where q is a prime or prime power, n the length of the codeword, k the length of the plaintext and t the error correcting capacity of the code. GL_k is the general linear group of order k.

Algorithm 1: McEliece Cryptosystem

```
Function KeyGen($1^\lambda$):
    Select linear code $C \subseteq \mathbb{F}_q^n$ with parameters $[n, k, t]$;
    Choose generator matrix $G$ for $C$;
    Pick random invertible $S \in GL_k(\mathbb{F}_q)$;
    Pick random permutation matrix $P \in \mathbb{F}_q^{n \times n}$;
    Compute $G' = SGP$;
    return Public key $(t, G')$, Private key $(G, S, P)$;
Function Enc($m, (t, G')$):
    Encode message $m \in \mathbb{F}_q^k$;
    Sample error $e \in \mathbb{F}_q^n$ with $\mathrm{wt}(e) \leq t$;
    Compute ciphertext $c = mG' + e$;
    return $c$;
Function Dec($c, (G, S, P)$):
    Compute $cP^{-1} = mSG + eP^{-1}$;
    Decode using $C$ to recover $mS$;
    Compute $m = (mS)S^{-1}$;
    return $m$;
```

4.2.2 Niederreiter Framework

The Niederreiter framework [24] is a code-based encryption scheme that uses a parity-check matrix instead of a generator matrix, while offering equivalent security. Niederreiter proposed GRS codes as the secret codes. In this scheme, the plaintext is a vector of length n with Hamming weight at most equal to the code's error-correcting capacity. The general construction is shown in the Algorithm 2.

Algorithm 2: Niederreiter Cryptosystem

```
Function KeyGen($1^\lambda$):
    Select a linear code $C \subseteq \mathbb{F}_q^n$ with parameters $[n, k, t]$;
    Choose an $(n - k) \times n$ parity-check matrix $\mathbf{H}$ for $C$;
    Pick random invertible $\mathbf{S} \in GL_k(\mathbb{F}_q)$;
    Pick random permutation matrix $\mathbf{P} \in \mathbb{F}_q^{n \times n}$;
    Compute $\mathbf{H}' = \mathbf{SGP}$;
    return Public key $(t, \mathbf{H}')$, Private key $(\mathbf{G}, \mathbf{S}, \mathbf{P})$;
Function Enc($m, (t, \mathbf{H}')$):
    Encode message $m \in \mathbb{F}_q^k$ with $\mathrm{wt}(m) \leq t$ ;
    Compute ciphertext $c^T = \mathbf{H}' m^T$;
    return $c$;
Function Dec($c, (\mathbf{G}, \mathbf{S}, \mathbf{P})$):
    Compute $\mathbf{S}^{-1} c^T = \mathbf{HP} m^T$;
    Decode using $C$ to recover $\mathbf{P} m^T$;
    Compute $m^T = \mathbf{P}^{-1}(\mathbf{P} m^T)$;
    return $m$;
```

Despite the security advantages of code-based ciphers the challenges that have impeded its practical application are the large size of the keys and ciphertext expansion, making it costly and inefficient to communicate.

4.3 Advantages in the Post-Quantum Era

Code-based cryptosystems offer strong resistance to quantum attacks, as no known quantum algorithm efficiently solves the Syndrome Decoding Problem (SDP) or distinguishes Goppa codes. They also rely on simpler arithmetic operations and avoid complex noise management, making them efficient and easier to implement compared to lattice-based schemes.

- **Quantum Resistance:** Security is based on the NP-hard SDP [9] and the Goppa Code Distinguishing Problem. Unlike RSA or ECC, which are vulnerable to Shor's algorithm, no quantum algorithm solves SDP sub-exponentially [10]. Classic McEliece, a NIST PQC finalist [25], exemplifies this long-standing robustness. Unlike lattice-based schemes relying on LWE/Ring-LWE [3], code-based systems are grounded in decades-old coding theory [26].
- **Efficiency:** Encryption uses finite field matrix operations (e.g., $\mathcal{O}(n^2)$ matrix-vector multiplications in McEliece variants) instead of more computationally intensive lattice-based polynomial ring operations ($\mathcal{O}(n^3)$) [22]. Martínez et al. [22] showed 1.8× faster encryp-

tion in code-based RLWE hybrids compared to pure lattice implementations. The absence of probabilistic decryption failures further reduces redundant computations [18].

- **Implementation Simplicity:** Noise is naturally managed through error-correcting codes rather than artificial noise sampling [7]. This avoids lattice-style noise flooding and modulus switching. McEliece-based HE uses predetermined error vectors, eliminating complex bootstrapping frameworks [20]. Chen et al. [15] reported 40% fewer code lines than lattice-based libraries. FPGA implementations achieve 2.3× better area-time product [21].

5 Code-Based Homomorphic Encryption Schemes

This section presents code-based HE constructions, along with their functionality and comparative analysis.

5.1 Bogdanov and Lee Homomorphic Encryption

This scheme[12] was constructed by combining the encryption structure of the local cryptosystem of Applebaum, Barak, and Wigderson [6] with a key scrambling of the McEliece cryptosystem [23]. The ABW PKE scheme is based on hardness-on-average assumptions for natural combinatorial NP-hard optimization problems with the following assumptions:

- It is infeasible to solve a random set of sparse linear equations mod 2, of which a small fraction is noisy.
- It is infeasible to distinguish between a random unbalanced bipartite graph and a graph in which a set S with only $|S|/3$ neighbors is planted at random on the large side.
- There is a pseudo-random generator, where every output depends on a random subset of the constant size of the input.

The basic idea here is to construct a generator matrix for the McEliece cryptosystem with the above assumptions. The encryption scheme can be described below:

- **KeyGen**
  Choose a uniformly random subset $S' \subseteq [n]$ of size s and an $n \times r$ matrix M from the following distribution. First, choose a set of uniformly random but distinct values $a_1, \cdots, a_n$ from F_q. Set the i^{th} row m_i^T to

$$m_i^T = \left\{ \begin{array}{ll} (a_i, a_i^2, \cdots, a_i^{s/3}, 0 \cdots, 0), & \text{if } i \in S' \\ (a_i, a_i^2, \cdots, a_i^{s/3}, a_i^{s/3+1}, \cdots, a_i^r) & \text{if } i \notin S' \end{array} \right\}$$

  The secret key is the pair (S', M) and the public key is the matrix $P = MR$, where R is a random $r \times r$ matrix over F_q with determinant 1.

- **Encryption**
  Given a public key P, to encrypt a message $m \in F_q$, choose a uniformly random $x \in F_q^r$ and an error vector $e \in F_q^n$ by choosing each of its entries independently at random from a random distribution χ. Output the ciphertext $c = Px + m1 + e$, where $1 \in F_q^n$ is the all ones vector.

- **Decryption**
  Given a secret key (S', M), to decrypt a ciphertext $c \in F_q^n$, first find a solution to the following system of $(s/3)+1$ linear equations over variables $y_i \in F_q, i \in S'$

$$\sum_{i \in S'} y_i m_i^T = 0,$$

$$\sum_{i \in S'} y_i = 1,$$

  with $y_i = 0$ when $i \notin S'$. Output the value $\langle y, c \rangle$.

- **Homomorphic Addition.** Let $c_1 = Px_1 + m_1 1 + e_1$ and $c_2 = Px_2 + m_2 1 + e_2$ be two ciphertexts encrypting messages m_1 and m_2, respectively. The homomorphic addition is performed component-wise:

$$c_{\text{add}} = c_1 + c_2 = P(x_1 + x_2) + (m_1 + m_2)1 + (e_1 + e_2)$$

  The resulting ciphertext c_{add} encrypts the sum $m_1 + m_2$ with aggregated noise $e_1 + e_2$. Correct decryption is possible as long as the combined noise remains within the decoding threshold.

In this step, we verify that the correctness property holds: Decode(Dec(Enc(m))) = m.

$$\begin{aligned} Dec(Enc(m)) = \langle y, c \rangle = \sum_{i=1}^{n} y_i c_i = \sum_{i=1}^{n} y_i (Px + m1 + e)_i \\ = \sum_{i=1}^{n} y_i (MRx)_i + \sum_{i=1}^{n} y_i (m1)_i + \sum_{i=1}^{n} y_i e_i \\ = \sum_{i=1}^{n} y_i m_i^T Rx + \sum_{i=1}^{n} y_i m + \sum_{i=1}^{n} y_i e_i \\ = Rx \sum_{i=1}^{n} y_i m_i^T + m \sum_{i=1}^{n} y_i + e' = m + e' \end{aligned}$$

hence, The message can be recovered by executing the decoding algorithm using s.

5.2 Armknecht Scheme

Armknecht [7] provided a generic construction of a symmetric key homomorphic encryption scheme that can evaluate multivariate polynomials up to a fixed degree μ.

1. **Keygen(s,μ, L)**
   The input s represents the security parameter, L is the expected total number of encryptions, and μ is the maximum degree of supported polynomials. The setup algorithm will then select a codeword support x, a message support y, and two special evaluation codes C and C' in such a way that $C^\mu \subseteq C'$, and the length of codewords is at least L. The choice of appropriate codes and parameters will vary depending on the coding scheme. Keygen will generate a set, I, of size T and a subset of $[n]$, where $[n]$ is the set $\{1, 2 \cdots n\}$. T depends on the above parameter and the deployed code. I represents the good locations for the generated encryptions and serves as the secret key of the scheme. The final output will be the secret key $k = (x, y, I)$.

2. **Encrypt(m,k)**
   The inputs are a plaintext message $m \in \mathbb{F}$, and a secret key $k = (x, y, I)$. Encrypt first chooses a random encoding $w \in C$ of m, using the Encode algorithm and the knowledge of the supports x and y. Then, it samples a uniformly random error vector $e \in \mathbb{F}^n$, such that $\mathit{supp}(e) \subseteq [n] \backslash I$ and computes $c = w + e$. Finally, the ciphertext is defined as the pair $(c, 1)$ where the first entry is an erroneous codeword in $C(I)$ that encodes the plaintext m while the second entry, the integer, is a counter to keep track of the number of multiplications. (see section 2.2 of Preliminaries for this notation)

3. **Decrypt($(c, \gamma), k)$)**
   Decrypt gets as input the secret key $k = (x, y, I)$ and a pair (c, γ) with $c \in C(I)$ and $\gamma \leq \mu$. It outputs $m = \mathit{Decode}(c, I)$ where Decode is used with respect to x and y.

4. **Add($(c_1, \gamma_1), (c_2, \gamma_2)$)**, outputs $(c_1 + c_2, \max(\gamma_1, \gamma_2))$.

5. **Mult($(c_1, \gamma_1), (c_2, \gamma_2)$**, outputs $(c_1 \cdot c_2, \gamma_1 + \gamma_2)$.

5.3 Comparison of Code-based HE with Lattice-Based HE

5.3.1 Efficiency

As shown in Table 3, code-based HE schemes typically require very large keys—ranging from megabytes to gigabytes—due to their matrix-based design (e.g., McEliece). Although operations such as XOR and matrix-vector multiplications can be parallelized, the overall computational cost remains high. In contrast, lattice-based schemes, especially those built on Ring-LWE, achieve much smaller keys (1–10 KB) and faster arithmetic by leveraging the Number Theoretic Transform (NTT). Furthermore, ciphertext expansion is generally larger in code-based schemes, whereas lattice-based schemes offer more moderate growth; for instance, CKKS expands ciphertexts by only about 8× compared to the plaintext.

Table 3: Efficiency Comparison of code-based vs lattice-based Homomorphic encryption.

Metric	**Code-Based HE**	**Lattice-Based HE**
Key Sizes	Large (e.g., McEliece keys: ≈1 MB–1 GB due to matrix-based structures). Recent code-based FHE schemes still struggle with key size reduction.	Smaller (e.g., Ring-LWE keys: ≈ 1-10 KB). Optimizations like NTT (Number Theoretic Transform) enable compact representations.
Computation Speed	Matrix/vector operations are parallelizable but computationally heavy. Simpler arithmetic (e.g., XOR-based operations in some schemes).	Faster due to polynomial ring optimizations (e.g., NTT accelerates multiplication in CKKS/BGV [17]).
Ciphertext Expansion	High (e.g., ciphertexts are large matrices or vectors).	Moderate (e.g., CKKS ciphertexts expand 8x plaintext size).

5.3.2 Noise Management

Noise management plays a central role in HE schemes. As shown in Table 4, code-based HE benefits from the inherent error correction of underlying codes (e.g., Goppa codes), which helps keep noise growth bounded in additive scenarios. In these schemes, noise typically grows linearly [7]. By contrast, lattice-based HE experiences polynomial noise growth with each multiplication, requiring techniques such as modulus switching and bootstrapping. While bootstrapping, introduced by Gentry and later optimized in schemes like BGV [13], effectively controls noise accumulation, it comes at the cost of additional computation.

Table 4: Noise Management Comparison of code-based vs lattice-based homomorphic encryption.

Aspect	**Code-Based HE**	**Lattice-Based HE**
Noise Growth	Inherits error-correction properties: errors are intentionally added but bounded by code distance. Noise grows linearly in some additive schemes (e.g., [7]).	Noise grows polynomially with multiplicative operations. Requires frequent management (e.g., modulus switching).
Bootstrapping	Limited progress, with high overhead.	Mature techniques (e.g., Gentry's bootstrapping in BGV [13]). Optimized via sparse embeddings or hybrid key-switching.
Error Correction	Built-in error correction can mitigate noise.	Relies on probabilistic decryption; no inherent error correction.

5.3.3 Security

The security of code-based HE relies on NP-hard problems such as the Syndrome Decoding Problem (SDP) and the Linear Code Distinguishing Problem, for which no quantum speedups are known. These strong foundations make code-based schemes resistant to quantum attacks; however, their adoption has been limited by large key sizes and relatively immature FHE implementations (Table 5). In contrast, lattice-based HE is built on more recent assumptions like Learning With Errors (LWE) and Ring-LWE (RLWE), which enjoy worst-case hardness guarantees and have been widely integrated into NIST-backed PQC standards. While lattice-based schemes benefit from extensive industrial testing, their security depends on ongoing advances in lattice reduction methods and careful protection against side-channel attacks.

Table 5: Security Assumptions and Trade-offs between code-based vs lattice-based homomorphic encryption.

Criterion	**Code-Based HE**	**Lattice-Based HE**
Core Hard Problems	Syndrome Decoding Problem (SDP), Linear Code Distinguishing Problem (NP-hard).	Learning With Errors (LWE), Ring-LWE (reductions to worst-case lattice problems).
Quantum Resistance	SDP has no known quantum speedup; robust against Shor's/Grover's algorithms.	LWE/Ring-LWE are quantum-resistant but rely on newer assumptions.
Standardization	Limited adoption (e.g., Classic McEliece is a NIST PQC finalist but not HE-focused).	Dominates NIST PQC standards (e.g., Kyber, Dilithium). HE schemes (CKKS/BGV) are industry-tested.
Attack Surface	Structural attacks (e.g., weak code choice) and ISD attacks (exponential time).	Side-channel attacks, decryption failures, and novel lattice reductions (e.g., BKZ).

Table 6: Summary of Trade-offs between code-based vs lattice-based homomorphic encryption.

Category	**Code-Based HE**	**Lattice-Based HE**
Strengths	- Provable NP-hard security. - Built-in error correction. - Simpler operations.	- Efficient bootstrapping. - Optimized libraries (e.g., Microsoft SEAL). - NIST-backed.
Weaknesses	- Large keys/ciphertexts. - Immature FHE implementations.	- Complex noise management. - Relies on newer security assumptions.

Table 6 highlights the trade-offs between code-based and lattice-based HE schemes. Code-based approaches provide strong NP-hard security guarantees, inherent error correction, and simpler operational models, but face challenges such as very large key and ciphertext sizes and less mature implementations. In contrast, lattice-based schemes benefit from efficient bootstrapping, optimized libraries (e.g., Microsoft SEAL), and strong support from NIST standardization efforts, though they require complex noise management and rely on newer security assumptions still under scrutiny.

The security landscape of HE remains an active research area. Lattice-based schemes, typically grounded in the Learning With Errors (LWE) problem, have been central to the development of fully homomorphic encryption. However, proposed quantum algorithms [16] suggest potential vulnerabilities in LWE under certain parameter regimes, raising questions about long-term security, despite flaws identified in such algorithms. By contrast, code-based schemes offer alternative security foundations, such as the Ideal Rank Syndrome Decoding (IRSD) problem, which remains difficult even for quantum adversaries. Recent work [2] demonstrated a somewhat homomorphic scheme based on IRSD, further highlighting the potential of code-based HE to withstand quantum threats.

6 Conclusion

Code-based homomorphic encryption shows strong potential for post-quantum cryptography. Its security relies on NP-hard problems, such as the Syndrome Decoding Problem, and it benefits from inherent error-correction that strengthens resistance against both classical and quantum attacks. While current challenges include efficiency limitations and noise management, ongoing research and standardization efforts are steadily improving practicality. As advancements continue, code-based homomorphic encryption is positioned to play a key role in safeguarding sensitive computations in the quantum era.

6.1 Challenges in Code-Based HE

Code-based homomorphic encryption still faces several challenges before achieving practical deployment. Key issues include improving computational efficiency, reducing the large key and ciphertext sizes, and better managing noise growth. Although built-in error correction helps control noise, significant refinements are required for high-performance applications.

Braverski [14] showed that LPN-based HE schemes suffer from structural weaknesses due to their reliance on linear algebra. Repeated homomorphic operations can expose linear relations among ciphertexts, enabling key recovery via algebraic attacks. Similar concerns apply to code-based HE, as McEliece-style constructions also depend on linear codes: homomorphic additions preserve linearity, which may reveal the structure of the generator matrix [7]. Addressing these vulnerabilities likely requires new frameworks for homomorphic encryption that avoid such exploitable algebraic structures.

References

[1] Carlos Aguilar-Melchor, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, and Gilles Zémor. Efficient encryption from random quasi-cyclic codes. *IEEE Transactions on Information Theory*, 64(5):3927–3943, 2018.

[2] Carlos Aguilar-Melchor, Victor Dyseryn, and Philippe Gaborit. Somewhat homomorphic encryption based on random codes. *Designs, Codes and Cryptography*, pages 1–25, 2025.

[3] Martin Albrecht. Lwe without modular reduction and improved side-channel attacks. *CRYPTO*, 2018.

[4] Martin Albrecht, Melissa Chase, Hao Chen, Jintai Ding, Shafi Goldwasser, Sergey Gorbunov, Shai Halevi, Jeffrey Hoffstein, Kim Laine, Kristin Lauter, Satya Lokam, Daniele Micciancio, Dustin Moody, Travis Morrison, Amit Sahai, and Vinod Vaikuntanathan. Homomorphic encryption standard. Cryptology ePrint Archive, Paper 2019/939, 2019.

[5] A. Aloufi, P. Hu, Y. Song, and K. Lauter. Computing blindfolded on data homomorphically encrypted under multiple keys: A survey. *ACM Computing Surveys (CSUR)*, 54(9):1–37, 2021.

[6] Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptography from different assumptions. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 171–180, 2010.

[7] Frederik Armknecht, Daniel Augot, Ludovic Perret, and Ahmad-Reza Sadeghi. On constructing homomorphic encryption schemes from coding theory. In *IMA International Conference on Cryptography and Coding*, pages 23–40. Springer, 2011.

[8] Daniel Augot and Matthieu Finiasz. A public key encryption scheme based on the polynomial reconstruction problem. In Eli Biham, editor, *Advances in Cryptology — EUROCRYPT 2003*, pages 229–240, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[9] Alexander Barg. Complexity of decoding for the general error pattern. *IEEE Transactions on Information Theory*, 1998.

[10] Daniel J. Bernstein. Classic mceliece: Conservative code-based cryptography. In *NIST PQC Standardization Conference*, 2017.

[11] Siddhartha Siddhiprada Bhoi, Arathi Arakala, Amy Beth Corman, and Asha Rao. Post-quantum homomorphic encryption: A case for code-based alternatives. *arXiv preprint arXiv:2504.16091*, 2025.

[12] Andrej Bogdanov and Chi-Hoon Lee. Homomorphic encryption from codes. Technical Report 2011/622, IACR Cryptology ePrint Archive, 2011.

[13] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In *Annual cryptology conference*, pages 868–886. Springer, 2012.

[14] Zvika Braverski et al. When homomorphism becomes a liability. *Journal of Cryptology*, 35(3):1–34, 2022.

[15] Abel CH Chen. Homomorphic encryption based on post-quantum cryptography. In *2023 IEEE International Conference on Machine Learning and Applied Network Technologies (ICMLANT)*, pages 1–5. IEEE, 2023.

[16] Yilei Chen. Quantum algorithms for lattice problems. *Cryptology ePrint Archive*, 2024.

[17] Jung Hee Cheon, A. Kim, M. Kim, and Y. Song. Homomorphic encryption for arithmetic of approximate numbers. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology–ASIACRYPT 2017*, Cham, Switzerland, 2017. Springer.

[18] Nico Döttling et al. Code-based cryptography: A unifying framework. *TCC*, 2019.

[19] Ernst M Gabidulin, Aleksei Vladimirovich Paramonov, and OV Tretjakov. Ideals over a non-commutative ring and their application in cryptology. In *Advances in Cryptology—EUROCRYPT'91: Workshop on the Theory and Application of Cryptographic Techniques Brighton, UK, April 8–11, 1991 Proceedings 10*, pages 482–489. Springer, 1991.

[20] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 169–178, 2009.

[21] Qian Guo et al. Hardware acceleration of code-based cryptography. *IEEE Transactions on Computers*, 2021.

[22] Ramon Martínez et al. Code-based rlwe: A new path to post-quantum he. In *ASIACRYPT*, 2022.

[23] Robert J McEliece. A public-key cryptosystem based on algebraic. *Coding Thv*, 4244(1978):114–116, 1978.

[24] Harald Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Prob. Contr. Inform. Theory*, 15(2):157–166, 1986.

[25] NIST. Nist post-quantum cryptography standardization. *CSRC*, 2022.

[26] Nicolas Sendrier. Structural cryptanalysis of mceliece variants with compact keys. *IEEE Transactions on Information Theory*, 2007.

[27] Violetta Weger, Niklas Gassner, and Joachim Rosenthal. A survey on code-based cryptography. *arXiv preprint arXiv:2201.07119*, 2022.