cryptography

IMPACT FACTOR 2.1

CITESCORE 5.0

Review

A Survey on Classical Lattice Algorithms

Tongchen Shen and Xiangxue Li

MDPI

https://doi.org/10.3390/cryptography10020017

cryptography

MDPI

Review

A Survey on Classical Lattice Algorithms

Tongchen Shen [1] and Xiangxue Li [2,*]

[1] Software Engineering Institute, East China Normal University, Shanghai 200241, China; 52205902014@stu.ecnu.edu.cn
[2] School of Cryptology, East China Normal University, Shanghai 200241, China
* Correspondence: xxli@cs.ecnu.edu.cn

Abstract

The rapid advancement of quantum computing poses a severe threat to traditional public key cryptosystems. Lattice-based cryptography has emerged as a core candidate for post-quantum cryptography due to its presumed quantum resistance, robust security foundations, and functional versatility, with its concrete security relying on the computational hardness of lattice problems. Existing lattice-based cryptography surveys mainly focus on cryptosystem design, scheme comparisons, and post-quantum cryptography standardization progress, with only cursory coverage of classical lattice algorithms that underpin the concrete security of lattice-based cryptography. We present the first systematic survey of classical lattice algorithms, focusing on two core categories of algorithms for solving lattice problems: approximate algorithms and exact algorithms. The approximate algorithms cover mainstream lattice basis reduction methods such as Lenstra–Lenstra–Lovász (LLL), Block Korkine–Zolotarev (BKZ), and General Sieve Kernel (G6K) algorithms, as well as alternative frameworks. The exact algorithms encompass dominant techniques like enumeration and sieving algorithms, along with alternative strategies. We systematically trace the evolutionary trajectory and inherent logical connections of various algorithms, clarify their core mechanisms, and identify promising future research directions. This survey not only serves as an introductory guide for beginners but also provides a valuable reference for seasoned researchers, facilitating the concrete security evaluation of lattice-based cryptosystems and the design of novel lattice algorithms.

Keywords: lattice cryptanalysis; lattice basis reduction; enumeration; sieving

1. Introduction

In the past few decades, propelled by the unprecedented advancements in quantum computing, many computational problems that were once considered computationally intractable under classical computing models have been shown to contain efficient solutions within the quantum computing paradigm. Notably, Shor's algorithm [1] enables polynomial time factorization of large integers and polynomial time solutions to the discrete logarithm problem, while Grover's quantum search algorithm [2] delivers a quadratic speedup for unstructured search tasks. These breakthroughs underscore the critical risk that traditional public key cryptosystems, whose security relies exactly on the hardness of these problems, may face an existential threat once large-scale, practical quantum computers become a reality.

In this context, post-quantum cryptography (PQC) has risen to the forefront of cryptographic research, encompassing diverse approaches such as isogeny-based, multivariate polynomial-based, code-based, hash-based, and lattice-based cryptography. Among these,

Academic Editor: Christoforos Ntantogian

Received: 19 January 2026
Revised: 22 February 2026
Accepted: 4 March 2026
Published: 6 March 2026

lattice-based cryptography has emerged as one of the most promising candidates, particularly evident from its prominence in NIST's PQC standardization [3]. Beyond its conjectured resilience against quantum attacks, lattice-based cryptography offers several distinctive advantages that differentiate it from other PQC alternatives:

- STRONG SECURITY FOUNDATIONS. Its security rests on the hardness of fundamental lattice problems, including the Shortest Vector Problem (SVP) and Closest Vector Problem (CVP). Both SVP [4] and CVP [5] are proven to be NP-hard. For more information about the complexity of NP-hard problems, please refer to [6,7]. In fact, lattice-based cryptosystems achieve provable security under the worst case hardness of (variants of) these problems [8,9], which is a property rarely found in other cryptosystems. This provides a solid theoretical grounding for its security assurances.
- REMARKABLE VERSATILITY. Unlike other PQC candidates, lattice-based cryptography supports a comprehensive set of practical cryptographic primitives, far beyond public key encryption (PKE) [10], key encapsulation mechanisms (KEM) [11], and digital signatures (SIG) [12]. For instance, it enables the construction of advanced cryptographic tools, such as attribute-based encryption (ABE) [13] and fully homomorphic encryption (FHE) [14,15], which are essential for emerging applications like privacy-preserving machine learning and secure cloud computing. This versatility makes lattice-based cryptography adaptable to a wide range of scenarios.

1.1. Concrete Security of Lattice-Based Cryptosystems

In order to promote the real-world deployment of lattice-based cryptography, researchers have increasingly focused on their concrete security in recent years. Concrete security quantifies the exact computational resources required to break the scheme. The evaluation of concrete security is largely determined by the hardness of core lattice problems [16–18], with the most fundamental being SVP, i.e., finding a non-zero lattice vector with the smallest Euclidean norm, and CVP, i.e., finding the lattice vector closest to a given target vector in Euclidean space, along with their various variants.

In particular, its security against message recovery, key recovery, or forgery attacks relies on the hardness of (variants of) Learning With Error (LWE) [9] and Short Integer Solution (SIS) [8] problems. For example, given a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ and a vector $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^m$, where each coefficient of $\mathbf{A}$ and $\mathbf{s}$ is uniformly sampled from $\mathbb{Z}_q$, while each coefficient of the error $\mathbf{e}$ is sampled from a small distribution χ over $\mathbb{Z}_q$, the search-LWE problem is to recover the secret $\mathbf{s}$ from the tuple $(\mathbf{A}, \mathbf{b})$. For simplicity, we use the standard Learning With Error (LWE) problem as an illustrative example here. In fact, the most effective attacks against other algebraically structured LWE variants, e.g., Ring-LWE [19] and Module-LWE [20], still involve reducing them to the standard LWE problem for attack [10,11]. The same logic applies to algebraically structured Short Integer Solution (SIS) variants [12]. There exist multiple strategies for reducing the Learning With Error (LWE) problem to lattice problems, including the primal attack [21], dual attack [22], and hybrid attack [23]. These strategies transform an LWE instance into (variants of) SVP and CVP problems, which are subsequently solved using lattice algorithms.

1.2. Lattice Algorithms and the Paper Organization

In this subsection, we present the classification of mainstream lattice algorithms and the organization of subsequent sections of this paper. There are two categories of algorithms for solving lattice problems: approximate and exact algorithms.

Approximate Algorithms. Approximate algorithms encompass a range of lattice basis reduction algorithms. They transform a given "bad basis" into a "good basis", which can then be leveraged to solve various approximate lattice problems. Currently, the three most

widely used lattice basis reduction algorithms are LLL, BKZ, and G6K. These algorithms can be seen as successive extensions of their predecessors. In particular, the celebrated LLL algorithm [24] revolutionized lattice research by efficiently generating a reduced basis. Subsequently, BKZ [25], and the subsequent BKZ 2.0 [26], extended the two-dimensional projected sublattice considered in LLL to a β-dimensional projected sublattice. Lastly, G6K [27] replaced the SVP oracle used in BKZ with sieving algorithms and introduced alternative iterative strategies. On the other hand, there exist several other frameworks for lattice basis reduction, such as dual-based and random sampling-based approaches. Their unique technical routes may hold potential for further improvements in the future.

Exact Algorithms. Lattice reduction algorithms often require internal calls to exact algorithms. Currently, the mainstream exact lattice algorithms are enumeration and sieving algorithms. Dating back to the 1980s [28], lattice enumeration algorithms work by systematically searching all lattice points within a specific geometric region to identify a shortest or closest lattice vector. Though they boast favorable polynomial space complexity, making them feasible in resource-constrained environments, the state-of-the-art enumeration algorithms [29–31] still have a super-exponential time complexity of $2^{\Theta(n \log n)}$. Lattice sieving algorithms [32–34] represent the other key class of exact methods. They mitigate the time inefficiency of enumeration algorithms by trading higher space consumption for single-exponential time and space complexities, making them the preferred option for high-dimensional exact lattice problems. Beyond these, alternative strategies exist for solving exact lattice problems. Voronoi cell-based algorithms [35–37] leverage lattice geometric properties to tackle such problems, but they typically require substantial preprocessing time for the lattice. Discrete Gaussian sampling-based algorithms [38,39] are provably the fastest exact algorithms, with a time complexity of $2^{n+o(n)}$. Yet they remain purely theoretical. These techniques may hold potential for integration into enumeration or sieving algorithms in the future.

To facilitate readers' quick grasp of the complexity and performance of mainstream lattice algorithms, Table 1 provides a concise overview of the time complexity, space complexity, and key performance metrics of the most prominent classical lattice algorithms in current research. Specifically, the first two rows belong to approximate algorithms, whose performance is quantified by the root Hermite factor (RHF). Formally, the RHF of a lattice basis $\mathbf{B}$ is defined as $\mathrm{rhf}(\mathbf{B}) = \left(\frac{\|\mathbf{b}_1\|}{\mathrm{vol}(\mathcal{L})^{1/n}}\right)^{1/n}$. It is evident that for the BKZ algorithm, as the block size β increases, the theoretical RHF value decreases monotonically, indicating a progressively improved quality of the reduced lattice basis. The remaining four rows are classified as exact algorithms. Among these, sieving algorithms represent the state-of-the-art heuristic exact methods to date, achieving the most favorable time–space complexity trade-offs in practical high-dimensional lattice instances. For a comprehensive technical description of the underlying principles and theoretical analyses of these approximate and exact algorithms, interested readers are referred to Section 3 and Section 4, respectively, where we elaborate on their core mechanisms and performance characteristics in depth.

Table 1. Quick grasp of representative lattice algorithms.

Algorithm	Time Complexity	Space Complexity	Performance
LLL [24]	$\mathrm{poly}(n)$	$\mathrm{poly}(n)$	$\mathrm{rhf}(\mathbf{B}) \leq \left(\frac{4}{4\delta-1}\right)^{1/4}$
BKZ [25,26]	$\mathrm{poly}(n)$ calls to β-dim SVP solver	$\mathrm{poly}(n)$	$\mathrm{rhf}(\mathbf{B}) \sim \left(\frac{\beta}{2\pi e}(\pi\beta)^{1/\beta}\right)^{\frac{1}{2(\beta-1)}}$
Enumeration [28,29]	$n^{\frac{n}{2e}+o(n)}$	$\mathrm{poly}(n)$	Exact algorithm
Sieving [32–34]	$2^{0.292n+o(n)}$	$2^{0.208n+o(n)}$	Exact algorithm
DGS-Based [38,39]	$2^{n+o(n)}$	$2^{n+o(n)}$	Exact algorithm
Voronoi Cell-Based [35,37]	$2^{2n+o(n)}$	$2^{n+o(n)}$	Exact algorithm

Paper Organization. In Section 2, we present the necessary background for lattice algorithms. In Section 3, we elaborate on mainstream lattice basis reduction algorithms, including LLL, BKZ, and G6K, and we also explore other basis reduction frameworks. Section 4 is dedicated to exact lattice algorithms, including the two mainstream exact algorithms, enumeration and sieving algorithms, and other alternative strategies. Finally, in Section 5, we summarize the paper and propose some promising future research directions related to classical lattice algorithms.

1.3. Contributions

Most existing surveys provide a comprehensive yet preliminary overview of lattice-based cryptography, focusing primarily on the design of cryptosystems, comparisons between different schemes, and the progress of PQC standardization, while lattice algorithms are only mentioned in a cursory manner. In contrast, this paper is the first survey to systematically elaborate on classical lattice algorithms. Our contributions are threefold. First, it serves as a basic tutorial for researchers and practitioners to understand the core mechanisms of classical lattice algorithms. Second, it systematically traces the evolution of classical lattice algorithms, clarifying the inherent logical connections among major algorithms. Third, it identifies promising future research directions in the field. We aim to bridge the gap between existing lattice-based cryptography surveys and lattice algorithm research, providing a useful reference for both beginners and seasoned researchers in related fields.

1.4. Related Work

Lattice-based cryptography and PQC have been the subject of many survey papers, each with distinct focuses and scopes. These existing surveys can be categorized based on their core research objectives:

- **Foundational theoretical exploration.** Specifically, early foundational surveys laid the groundwork for the mathematical underpinnings of lattice-based schemes, exploring weaker worst-case assumptions central to cryptographic security but omitting practical algorithmic implementations [40].
- **PQC standardization and scheme comparison.** Surveys focused on PQC standardization provided a broad perspective on the evolution of PQC, analyzing the selection process for PKE, KEM and emerging research trends, yet they did not conduct in-depth analysis of lattice algorithms or their optimization [41].
- **Specialized analysis of lattice-based primitives.** Specialized surveys categorized lattice-based digital signature schemes and discussed their theoretical implications, while others reviewed lattice-based cryptosystems in the context of PQC standardization, covering design paradigms, key proposals, and cryptanalytic methods, but only mentioned lattice algorithms as auxiliary tools without systematic elaboration [42,43].
- **Introductory overviews of lattice cryptography mathematics.** Additionally, some surveys analyzed recent advances in PQC to identify cryptographic weaknesses, and introductory works focused on the underlying mathematics of lattice cryptography, both lacking attention to the algorithmic foundations of lattice problem-solving [44,45].

A critical common limitation of all these prior surveys is their lack of exclusive and systematic focus on lattice algorithms. None of the existing works trace the historical evolution of core classical lattice algorithms, e.g., LLL, BKZ, G6K, enumeration, and sieving, clarify their inherent logical connections, or comprehensively discuss their performance trade-offs in solving approximate and exact lattice problems. To the best of our knowledge, this survey is the first to fill this gap by centering entirely on classical lattice algorithms, thereby

forming a critical complement to existing surveys and addressing a key underexplored perspective in the field.

On the other hand, efficient lattice-based cryptosystems currently rely heavily on structured lattice problems. For example, their security often hinges on the hardness of variants of algebraically structured LWE, such as Ring-LWE [19] or Module-LWE [20]. Nowadays, researchers generally agree that reducing these variants to the standard LWE problem remains the most effective attack strategy [10,11]. However, it remains unclear whether their algebraic structure truly prevents practical attacks, which is a crucial open problem in the field.

On the other hand, lattice-based cryptosystems are designed to resist quantum attacks, yet this survey only covers classical lattice algorithms. In fact, the main impact of quantum computing on lattice-related attacks currently lies in Grover's quantum search algorithm [2], which provides a square-root speedup for search tasks [46]. Whether quantum algorithms targeting lattice problems can be designed to achieve better complexity is another critical open problem.

In particular, lattice reduction algorithms can efficiently achieve an approximation factor of $2^{O(n)}$ for any lattice on classical computers, while quantum computers can attain a much smaller approximation factor of $2^{O(\sqrt{n})}$ in polynomial time for ideal lattices in cyclotomic fields [47,48]. However, this quantum attack does not generalize to module lattices of ranks bigger than one, which may imply that Module-LWE is more resilient to severe attacks than Ring-LWE. On the other hand, the LLL algorithm can be generalized to module lattices [49], and the same generalization can also be extended to the BKZ algorithm [50].

Finally, there exist other non-lattice attacks against lattice-based cryptosystems. For example, the Arora-Ge method [51,52] transforms the LWE problem into a system of polynomial equations, which is then solved using standard linearization techniques. The BKW method [53,54], by contrast, leverages a sufficiently high number of LWE samples to construct specialized new samples for solving the problem. However, these attacks are only feasible when the support of the error distribution is sufficiently small and the number of LWE samples is exponential. How to alleviate these limitations remains an interesting open question.

2. Preliminaries

2.1. Lattices

Let bold lowercase letters denote vectors, which are understood as column vectors. For a vector $\mathbf{v}$, its Euclidean norm is defined as

$$\|\mathbf{v}\| = \sqrt{\langle \mathbf{v}, \mathbf{v} \rangle}.$$

Capital bold letters denote matrices. Given a set of linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_m \in \mathbb{R}^n$, the lattice generated by the lattice basis $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_m) \in \mathbb{R}^{n \times m}$ is given by

$$\mathcal{L}(\mathbf{B}) = \{\mathbf{Bz} \mid \mathbf{z} \in \mathbb{Z}^m\}.$$

We typically consider full-rank lattices, where the dimension of the lattice equals the number of basis vectors, i.e., $n = m$. Any lattice of a rank greater than two has infinitely many bases. Two lattice bases $\mathbf{B}_1$ and $\mathbf{B}_2$ generate the same lattice if and only if there exists a unimodular matrix $\mathbf{U} \in \mathbb{Z}^{m \times m}$ such that $\mathbf{B}_1 = \mathbf{B}_2 \cdot \mathbf{U}$. One of the most important invariants of a lattice is its volume (or determinant), defined as

$$\mathrm{vol}(\mathcal{L}(\mathbf{B})) = \mathrm{vol}(\mathbf{B}) = \sqrt{\det(\mathbf{B}^T\mathbf{B})}.$$

Another key invariant is its successive minima. Given a lattice Λ of dimension n, the i-th successive minima $\lambda_i(\Lambda)$ (for $i \leq n$) is the smallest radius $r > 0$ such that there exist i linearly independent lattice vectors whose norms are at most r. We are particularly interested in the first minima $\lambda_1(\Lambda)$. On the other hand, for any point $\mathbf{t} \in \mathbb{R}^n$, the distance from $\mathbf{t}$ to a lattice $\Lambda \subseteq \mathbb{R}^n$ is defined as

$$\mathrm{dist}(\mathbf{t}, \Lambda) = \min_{\mathbf{w} \in \Lambda} \|\mathbf{w} - \mathbf{t}\|.$$

2.2. Computational Problems

The two most fundamental computational problems on lattices are finding a non-zero lattice vector with minimal Euclidean norm and finding a lattice vector closest to a given target vector. We formally define these problems and their approximate variants as follows.

Definition 1 (Shortest Vector Problem (SVP))**.** *Given a lattice basis* $\mathbf{B}$*, the Shortest Vector Problem (SVP) requires finding a non-zero lattice vector* $\mathbf{w} \in \mathcal{L}(\mathbf{B})$ *such that*

$$\|\mathbf{w}\| = \lambda_1(\mathcal{L}(\mathbf{B})).$$

Definition 2 (Closest Vector Problem (CVP))**.** *Given a lattice basis* $\mathbf{B}$ *and a target vector* $\mathbf{t} \in \mathrm{span}(\mathcal{L}(\mathbf{B}))$*, the Closest Vector Problem (CVP) asks for a lattice vector* $\mathbf{w} \in \mathcal{L}(\mathbf{B})$ *satisfying*

$$\|\mathbf{w} - \mathbf{t}\| = \mathrm{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})).$$

Both SVP [4] and CVP [5] are known to be NP-hard. For cryptographic applications, their approximate versions are of central importance.

Definition 3 (γ-Approximate Shortest Vector Problem (γ-SVP))**.** *Given a lattice basis* $\mathbf{B}$ *and an approximation factor* $\gamma \geq 1$*, the* γ*-approximate Shortest Vector Problem (*γ*-SVP) requires finding a non-zero lattice vector* $\mathbf{w} \in \mathcal{L}(\mathbf{B})$ *such that*

$$\|\mathbf{w}\| \leq \gamma \cdot \lambda_1(\mathcal{L}(\mathbf{B})).$$

Definition 4 (γ-Approximate Closest Vector Problem (γ-CVP))**.** *Given a lattice basis* $\mathbf{B}$*, a target vector* $\mathbf{t} \in \mathrm{span}(\mathcal{L}(\mathbf{B}))$*, and an approximation factor* $\gamma \geq 1$*, the* γ*-approximate Closest Vector Problem (*γ*-CVP) asks for a lattice vector* $\mathbf{w} \in \mathcal{L}(\mathbf{B})$ *such that*

$$\|\mathbf{w} - \mathbf{t}\| \leq \gamma \cdot \mathrm{dist}(\mathbf{t}, \mathcal{L}(\mathbf{B})).$$

Most lattice-based cryptosystems rely on the conjectured hardness of γ-SVP and γ-CVP with the approximation factor $\gamma = \mathrm{poly}(n)$. On the other hand, the first successive minimum $\lambda_1(\mathcal{L}(\mathbf{B}))$ is typically not known in advance, so it is often impossible to directly verify whether a vector $\mathbf{w}$ satisfies the condition of γ-SVP. In practice, one usually uses the lattice volume as an alternative metric.

Definition 5 (γ-Approximate Hermite Shortest Vector Problem (γ-HSVP))**.** *Given a lattice basis* $\mathbf{B}$ *and an approximation factor* $\gamma \geq 1$*, the* γ*-approximate Hermite Shortest Vector Problem (*γ*-HSVP) requires finding a non-zero lattice vector* $\mathbf{w} \in \mathcal{L}(\mathbf{B})$ *such that*

$$\|\mathbf{w}\| \leq \gamma \cdot \mathrm{vol}(\mathcal{L}(\mathbf{B}))^{1/n}.$$

The approximate SVP and approximate Hermite SVP problems are closely related. Any algorithm solving γ-SVP can also solve $(\gamma \cdot \gamma_n)$-HSVP by Minkowski's theorem, where

γ_n denotes the n-dimensional Hermite constant. Conversely, any algorithm for γ-HSVP can solve γ^2-SVP with at most polynomially many invocations [55].

2.3. Estimating the First Minima

Minkowski's theorem was the first to provide an upper bound on the first minima of a lattice. Specifically, it states that for any n-dimensional lattice Λ, the following inequality holds:

$$\lambda_1(\Lambda) \leq \left(\prod_{i=1}^{n} \lambda_i(\Lambda)\right)^{\frac{1}{n}} \leq \sqrt{n} \cdot \mathrm{vol}(\Lambda)^{\frac{1}{n}}.$$

Note that this upper bound is not tight. On the other hand, Gaussian Heuristic is widely employed in lattice algorithms to estimate the first minima in the average case. It approximates the number of lattice points in a "nice" region by dividing the volume of this region, denoted as $\mathrm{vol}(S)$, by the volume of the lattice, denoted as $\mathrm{vol}(\mathcal{L})$. In particular, the first minima $\lambda_1(\mathcal{L})$ of a lattice $\mathcal{L}$ is estimated as the radius of an n-dimensional ball with volume $\mathrm{vol}(\mathcal{L})$, i.e.,

$$\lambda_1(\mathcal{L}) \approx \mathrm{GH}(\mathcal{L}) = \frac{\mathrm{vol}(\mathcal{L})^{1/n}}{\mathrm{vol}(\mathcal{B}_n(1))^{1/n}},$$

where $\mathrm{vol}(\mathcal{B}_n(1))$ denotes the volume of the n-dimensional unit ball. In many applications, Gaussian Heuristic has been empirically validated for specific regions, such as when analyzing the runtime of enumeration algorithms [29,30] and the output quality of lattice reduction algorithms [26,56].

2.4. Gram–Schmidt Orthogonalization

Projection operators are among the most essential tools in lattice algorithm research. For a lattice basis $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$, let π_i be the orthogonal projection onto the orthogonal complement in the span of $(\mathbf{b}_1, \ldots, \mathbf{b}_{i-1})$, i.e., $\pi_i : \mathbb{R}^n \to \mathrm{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{i-1})^{\perp}$. Let $\mathbf{B}^* = (\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*)$ denote the Gram–Schmidt orthogonalization of $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$. Specifically, the basis $\mathbf{B}$ can be uniquely decomposed as $\mathbf{B}^T = \mathbf{M} \cdot \mathbf{D} \cdot \mathbf{O}$, where

$$\mathbf{M} = \begin{pmatrix} 1 & 0 & \ldots & 0 \\ \mu_{2,1} & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mu_{n,1} & \ldots & \mu_{n,n-1} & 1 \end{pmatrix}, \quad \mathbf{D} = \begin{pmatrix} \|\mathbf{b}_1^*\| & 0 & \ldots & 0 \\ 0 & \|\mathbf{b}_2^*\| & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \ldots & \ldots & \|\mathbf{b}_n^*\|, \end{pmatrix}, \quad \mathbf{O} = \begin{pmatrix} \frac{\mathbf{b}_1^{*T}}{\|\mathbf{b}_1^*\|} \\ \frac{\mathbf{b}_2^{*T}}{\|\mathbf{b}_2^*\|} \\ \vdots \\ \frac{\mathbf{b}_n^{*T}}{\|\mathbf{b}_n^*\|} \end{pmatrix},$$

and $\mathbf{O}$ is an orthonormal matrix. The Gram–Schmidt coefficients are defined as $\mu_{i,j} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle}$ for $i > j$, with $\mu_{i,i} = 1$ by convention. The volume of the lattice $\mathcal{L}(\mathbf{B})$ is given by

$$\mathrm{vol}(\mathcal{L}(\mathbf{B})) = \prod_{i=1}^{n} \|\mathbf{b}_i^*\|.$$

Note that $\mathbf{b}_i^* = \pi_i(\mathbf{b}_i)$. For indices $1 \leq i \leq j \leq n$, we further define $\mathbf{B}_{[i,j]} = (\pi_i(\mathbf{b}_i), \ldots, \pi_i(\mathbf{b}_j))$ and $\mathcal{L}_{[i,j]} = \mathcal{L}(\mathbf{B}_{[i,j]})$.

3. Approximate Algorithms

This section discusses various lattice basis reduction algorithms, which can directly yield solutions to γ-SVP and γ-HSVP. These algorithms transform a "bad basis" into a "good basis" for the same lattice. See Figure 1 for an illustration. Specifically, this figure shows a bad basis $(\mathbf{b}_1, \mathbf{b}_2)$ and a good basis $(\mathbf{c}_1, \mathbf{c}_2)$ that generate the identical lattice,

i.e., $\mathcal{L}(\mathbf{b}_1, \mathbf{b}_2) = \mathcal{L}(\mathbf{c}_1, \mathbf{c}_2)$. Vectors in a bad basis are typically long, with pairwise angles far from orthogonal. In contrast, vectors in a good basis are generally short, and their pairwise angles are close to orthogonal.

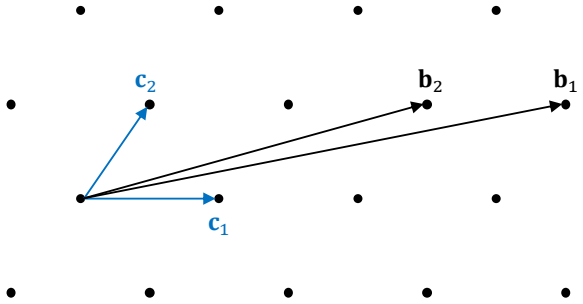

Figure 1. A "bad" basis $(\mathbf{b}_1, \mathbf{b}_2)$ and a "good" basis $(\mathbf{c}_1, \mathbf{c}_2)$ of the same lattice.

We typically evaluate the performance of a lattice basis reduction algorithm by the length of the first vector of its output basis. The approximation factor of a basis $\mathbf{B}$ is defined as $\mathrm{app}(\mathbf{B}) = \frac{\|\mathbf{b}_1\|}{\lambda_1(\mathcal{L}(\mathbf{B}))}$, while the Hermite factor of a basis $\mathbf{B}$ is defined as $\mathrm{hf}(\mathbf{B}) = \frac{\|\mathbf{b}_1\|}{\mathrm{vol}(\mathcal{L})^{1/n}}$. These metrics quantify the quality of the basis: A smaller factor $\mathrm{app}(\mathbf{B})$ or $\mathrm{hf}(\mathbf{B})$ indicates that the first basis vector $\mathbf{b}_1$ is closer to the shortest non-zero lattice vector. Since the Hermite factor achievable by almost all existing lattice basis reduction algorithms is exponential in the lattice dimension n [26,56], the root Hermite factor (RHF) is often used in the literature [27,57] to simplify the characterization of basis quality. It is defined as $\mathrm{rhf}(\mathbf{B}) = \left(\frac{\|\mathbf{b}_1\|}{\mathrm{vol}(\mathcal{L})^{1/n}}\right)^{1/n}$. On the one hand, the n-th root of the root Hermite factor is attributed to the current design techniques and proof frameworks of lattice basis reduction algorithms. On the other hand, it is corroborated by experimental results. The output basis generally adheres to the Geometric Series Assumption (GSA), as discussed in Section 3.2, which justifies the use of the root Hermite factor as a concise metric for overall basis quality. Also, note that some authors adopt an alternative definition using the $(n-1)$-th root: $\mathrm{rhf}(\mathbf{B}) = \left(\frac{\|\mathbf{b}_1\|}{\mathrm{vol}(\mathcal{L})^{1/n}}\right)^{1/(n-1)}$. However, the difference between the two definitions becomes very small for large n.

3.1. LLL

The renowned LLL algorithm, named after Lenstra, Lenstra, and Lovasz [24], is a central tool in the algorithmic study of lattice cryptography. It can be regarded as a high-dimensional generalization of the Euclidean algorithm, which acts on a pair of integers. The LLL algorithm seeks the local optimal solution for each two-dimensional projected sublattice $\mathbf{B}_{[i,i+1]}$ for $i \in \{1, \ldots, n-1\}$. It iteratively leverages these local optimal solutions to construct a globally approximate solution for the original lattice. That is, once a short vector is found for a projected sublattice $\mathbf{B}_{[i,i+1]}$, the corresponding vector of the full lattice is swapped forward in the basis. Formally, the lattice basis output by the LLL algorithm satisfies the Lovasz condition, meaning

$$\delta \|\mathbf{b}_i^*\| \leq \|\pi_i(\mathbf{b}_{i+1})\| = \left\|\mathbf{b}_{i+1}^* + \mu_{i+1,i}\mathbf{b}_i^*\right\|, \text{ for all } 1 \leq i < n,$$

where the parameter $\delta \in \left(\frac{1}{4}, 1\right)$ determines the quality of the output basis. Moreover, the LLL algorithm enforces that the pairwise angles between the currently considered basis vectors are kept as small as possible. In other words, its output basis is size-reduced, meaning

$$\left|\mu_{i,j}\right| \leq \frac{1}{2}, \text{ for all } 1 \leq j < i \leq n,$$

where $\mu_{i,j} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle}$ denotes the Gram–Schmidt coefficient. Therefore, the LLL algorithm outputs a δ-LLL-reduced basis that satisfies both the size reduction condition and the Lovasz condition. It can be shown that a δ-LLL-reduced basis $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$ can achieve an exponential Hermite factor [24]:

$$\mathrm{hf}(\mathbf{B}) = \frac{\|\mathbf{b}_1\|}{\mathrm{vol}(\mathcal{L})^{1/n}} \leq \left(\frac{4}{4\delta - 1}\right)^{\frac{n-1}{4}}.$$

The pseudocode of the LLL algorithm is given in Algorithm 1, and it can be proven that the algorithm terminates in polynomial time using the potential-based proof technique [24].

Algorithm 1 LLL (High Level)

Require: A set of lattice basis $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n)$.
Ensure: A set of δ-LLL-reduced basis.
1: Compute the Gram–Schmidt orthogonalization $\mathbf{B}^* = (\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*)$ of $\mathbf{B}$.
2: Initialize $k \leftarrow 2$.
3: **while** $k \leq n$ **do**
4: Size-reduce the vector $\mathbf{b}_k$ to ensure $|\mu_{k,i}| \leq \frac{1}{2}$ for all $i < k$.
5: **if** $\|\mathbf{b}_k^*\|^2 < (\delta - \mu_{k,k-1}^2)\|\mathbf{b}_{k-1}^*\|^2$ **then**
6: Swap $\mathbf{b}_k$ and $\mathbf{b}_{k-1}$.
7: Recompute affected Gram–Schmidt coefficients.
8: $k \leftarrow \max(k-1, 2)$.
9: **else**
10: $k \leftarrow k + 1$.
11: **end if**
12: **end while**
13: **return** the LLL-reduced basis $\mathbf{B}$.

Notably, in practice, the LLL algorithm can sometimes yield a solution of fairly high quality. Especially for very low-dimensional lattices, for instance, when $n < 20$, it can even directly produce an exact solution to SVP. Furthermore, a slightly modified variant of the LLL algorithm [58] is commonly adopted in lattice basis reduction algorithms to eliminate the linear dependence of a generating set, where a generating set refers to a collection of linearly dependent vectors that generates the target lattice.

Numerous refinements have been developed for the LLL algorithm. In practical applications, the LLL algorithm is often required to process lattice bases with large entry sizes [59,60]. A substantial speedup can be achieved by reducing the runtime dependence on such entry sizes, provided that we control the precision of floating-point arithmetic while preserving the correctness of the algorithm. A prominent variant is the L^2 algorithm [61], which is implemented in the `fplll` library [62]. It can be proven that this floating-point variant still outputs a valid LLL-reduced basis.

At ASIACRYPT 2025, Ducas et al. [63] presented a highly efficient LLL implementation named `BLASter`, which unifies multiple theoretical optimizations for lattice reduction. In detail, this implementation leverages the segmented recursive strategy [64], Seysen's size reduction method [65], and state-of-the-art parallelization techniques [66]. They experimentally verify that `BLASter` achieves a substantial speedup over `fplll`, with no degradation in the quality of the resulting LLL-reduced basis.

Another optimization is the deep insertion technique, namely the DeepLLL algorithm [25]. Unlike the standard LLL algorithm that only swaps adjacent basis vectors, DeepLLL permits a basis vector $\mathbf{b}_k$ to be inserted between $\mathbf{b}_{i-1}$ and $\mathbf{b}_i$ for indices satisfying $i < k - 1$. As such, it is anticipated to produce a lattice basis of superior quality to LLL, and in practical applications, DeepLLL indeed often identifies a shorter vector than the vanilla

LLL algorithm [56]. Nevertheless, no rigorous proof has yet established that DeepLLL is a polynomial-time algorithm, and its actual runtime does not seem to exhibit polynomial complexity [56].

3.2. BKZ

Variants of BKZ algorithms [26,67] are widely used in the concrete security estimation of lattice-based cryptosystems. Schnorr and Euchner [25] were the first to propose the Block Korkine–Zolotarev (BKZ) algorithm as a natural generalization of LLL. Unlike LLL, which only considers the two-dimensional projected sublattice $\mathbf{B}_{[i,i+1]}$, the BKZ algorithm computes a short vector $\mathbf{v}$ in the β-dimensional projected sublattice $\mathbf{B}_{[i,i+\beta]}$ and then inserts $\mathbf{v}$ into $\mathbf{B}$ at position i. Traditionally, enumeration algorithms, as discussed in Section 4.1, are employed as exact SVP solvers for this task, owing to their excellent practical performance in low-dimensional lattices. The pseudocode of the BKZ algorithm is provided in Algorithm 2.

Algorithm 2 BKZ (High Level)

Require: A set LLL-reduced basis $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n)$, and a block size parameter $\beta \geq 2$.
Ensure: A set of β-BKZ-reduced basis.

```
repeat
    for i = 1, 2, ..., n − 1 do
        Perform LLL over the projected basis B_[i,min{i+β,n}].
        Compute a shortest non-zero vector v ∈ L_[i,min{i+β,n}].
        Insert v into B at position i and eliminate the linear dependencies with LLL.
    end for
until no more change over the basis is made.
return the β-BKZ-reduced basis B.
```

Note that the LLL algorithm is used to eliminate the linear dependence after inserting the short vector $\mathbf{v}$ into the basis $\mathbf{B}$, as stated in Section 3.1. All considered input lattice bases are assumed to have undergone LLL preprocessing, an approach that is extremely common in the field of lattice algorithms. On the other hand, the last few projected sublattices, i.e., $\mathbf{B}_{[i,n]}$ where $n < i + \beta$, are more special than other parts. In fact, the tail part of basis $\mathbf{B}$ is Hermite–Korkine–Zolotarev (HKZ)-reduced, which is generally regarded as the "best" reduced basis achievable. That is,

$$\|\mathbf{b}_i^*\| = \lambda_1\left(\mathcal{L}\left(\mathbf{B}_{[i:n]}\right)\right) \text{ for all } n - \beta + 1 \leq i \leq n.$$

To begin with, we analyze the quality of the lattice basis output by the BKZ algorithm. Schnorr [68] provided an intuitive heuristic characterization for a β-BKZ-reduced basis $\mathbf{B}$, namely the Geometric Series Assumption (GSA). This assumption directly models the Gram–Schmidt norms $\|\mathbf{b}_1^*\|, \ldots, \|\mathbf{b}_n^*\|$ as a sequence decaying in a strict geometric progression, i.e.,

$$\frac{\|\mathbf{b}_i^*\|}{\|\mathbf{b}_{i+1}^*\|} = \alpha_\beta \text{ for all } i = 1, \ldots, n-1,$$

where the parameter $\alpha_\beta \approx \left(\frac{\beta}{2\pi e}\right)^{1/\beta}$ [26] dictates the quality of the reduced basis. Specifically, according to GSA, a larger block size β leads to a smaller parameter α_β, which in turn yields a reduced basis of higher quality. In the context of GSA, this relationship manifests as follows: A larger block size β produces a flatter decay profile of Gram–Schmidt norms, whereas a smaller block size β results in a steeper decay trend. This assumption is relatively precise in specific scenarios, for instance, when $50 < \beta \ll n$ [26,69]. Furthermore, under the

combined framework of GSA and the Gaussian Heuristic, Chen [70] derived the asymptotic limit of the root Hermite factor for BKZ-reduced bases with the block size β:

$$\lim_{n\to\infty} \mathrm{rhf}(\mathbf{B}) = \left(\frac{\beta}{2\pi e}(\pi\beta)^{\frac{1}{\beta}}\right)^{\frac{1}{2(\beta-1)}}.$$

This is a result that is widely adopted in the concrete security evaluation of lattice-based cryptosystems.

We typically partition the Gram–Schmidt norms into three parts: the head, the body, and the tail. Generally, the body segment aligns more closely with the GSA line than the other two parts. The tail curve phenomenon can be attributed to the fact that the tail segment is HKZ-reduced rather than BKZ-reduced [70]. In contrast, the head concavity phenomenon can be modeled via the probabilistic version of the Gaussian Heuristic, as elaborated by Bai, Stehle, and Wen [57].

Building on the foundational framework of the classic BKZ algorithm, a series of practical variants have been proposed to enhance its computational efficiency while maintaining or improving the basis quality. At ASIACRYPT 2011, Chen and Nguyen [26] proposed a highly practical variant of BKZ, namely the BKZ 2.0 algorithm. Apart from the early-abort technique, several key optimizations are incorporated into BKZ 2.0. In particular, BKZ 2.0 employs a more robust preprocessing step than the standard LLL algorithm. This step significantly reduces the computational overhead of subsequent enumeration calls, thereby cutting down the overall computational cost of the algorithm. Additionally, BKZ 2.0 integrates the extreme pruning technique [29] and optimizes the search radius of the enumeration-based SVP solver; these techniques collectively yield a substantial reduction in the overhead of each enumeration invocation.

In enumeration-based lattice basis reduction, reduction steps and enumeration steps are inherently intertwined. On the one hand, enumeration algorithms are employed as the underlying SVP solver within the reduction framework. On the other hand, the computational complexity of enumeration algorithms heavily depends on the quality of the basis being enumerated, as discussed in Section 4.1, thus requiring a stronger reduction as the preprocessing step. Building on the paradigm of adopting more robust preprocessing than the standard LLL algorithm [26], Aono et al. [67] proposed another BKZ variant, namely the Progressive BKZ algorithm. Unlike traditional BKZ, which employs a fixed block size β, Progressive BKZ conducts reduction starting with a small block size and gradually increasing the block size, where the sequence $\beta_1 < \beta_2 < \beta_3 < \ldots$ is optimized through extensive experiments to reduce the overall runtime. Each reduction round with a smaller block size can be regarded as a preprocessing step for the subsequent reduction with a larger block size.

Going a step further, Albrecht et al. [31,71] boosted the computational efficiency of enumeration-based lattice basis reduction algorithms by incorporating both the extended preprocessing technique and the relaxed enumeration technique. Their approach seeks to mitigate the adverse effects induced by the tail curve, which deviates from the GSA line and performs preprocessing on an extended projected sublattice. Notably, this optimized BKZ variant achieves a root Hermite factor of $\mathrm{rhf}(\mathbf{B}) = \beta^{\frac{1}{2\beta}}$, with a time complexity of $\beta^{\frac{\beta}{8}+o(\beta)}$, narrowing the performance gap between enumeration-based and sieving-based lattice basis reduction algorithms, as discussed in Section 3.3.

To predict the practical performance and runtime of these BKZ-like variants, simulating the lattice basis reduction process has proven to be a valuable approach. Chen and Nguyen [26] developed a BKZ 2.0 simulator based on the Gaussian Heuristic, which can efficiently model the evolution of Gram–Schmidt norms throughout each iteration of the

algorithm, called a tour of BKZ. Through extensive experiments with block sizes $\beta \geq 50$, they demonstrated that their simulator is capable of approximating the quality of the output basis and estimating the corresponding runtime. As a side result, these experiments also validated the applicability of the Gaussian Heuristic in the context of lattice basis reduction algorithms.

3.3. G6K

Over the past two decades, sieving algorithms have undergone tremendous advancements both theoretically and practically, as discussed in Section 4.2. It is a trend that has reshaped the landscape of lattice basis reduction. This rapid progress naturally motivates the idea of replacing the enumeration-based SVP oracle in the classic BKZ algorithm with sieving algorithms. From a computational complexity perspective, even the earliest sieving algorithms with rigorous theoretical guarantees, such as the AKS-sieve [33] and List-sieve [34], exhibit a single-exponential time complexity of $2^{\Theta(n)}$. This stands in stark contrast to the super-exponential $2^{\Theta(n \log n)}$ time complexity of all enumeration algorithms, meaning that sieving algorithms hold an inherent asymptotic edge in high-dimensional lattices. However, the practical performance of sieving algorithms remained lackluster for years, plagued by prohibitive memory overhead and complex implementation details. It was not until roughly the past decade that a series of substantial refinements [32,72–74] propelled sieving into a viable alternative to enumeration.

At EUROCRYPT 2019, Albrecht et al. [27] proposed a landmark sieving-based lattice basis reduction algorithm: the General Sieve Kernel (G6K). Beyond its core innovation of adopting sieving algorithms as SVP oracles, G6K deviates from BKZ in several fundamental aspects that underpin its superior performance. It supports a diverse set of reduction strategies, rather than being confined to the rigid tour framework that defines BKZ's reduction strategies [25]. Further, unlike BKZ, which treats its SVP oracle, i.e., the enumeration algorithm, as the "black box" that outputs only a single short vector per call, G6K introduces an auxiliary short vector database L. This database stores the large volume of short vectors generated by each sieving invocation. Critically, L is not a static repository. The database accelerates subsequent sieving calls by reusing precomputed short vectors to generate other possibly short vectors and is dynamically updated as the projected sublattice $\mathbf{B}_{[i,j]}$ slides across the full lattice basis, ensuring its contents remain relevant to the current reduction task.

The design philosophy of this auxiliary database draws primarily from two key techniques: the progressive sieving technique [74] and the dimension-for-free technique [73], both of which are seamlessly integrated into G6K's framework. In detail, the progressive sieving technique proposed by Laarhoven and Mariano [74] first performs sieving on a small-dimensional sublattice to generate a list of short vectors. It then incrementally incorporates basis vectors corresponding to the remaining dimensions, sieving the expanded sublattice and updating the short vector list at each step. Since most computations are performed on low-dimensional sublattices, where sieving is far more efficient, this technique drastically reduces the practical space and time overhead of the algorithm. On the other hand, the dimension-for-free technique proposed by Ducas [73] delivers another practical insight. Solving the SVP for an n-dimensional lattice does not require sieving the full n dimensions. Instead, a few sieving invocations on sublattices of dimension less than $n - k$, where $k = \Theta(n/\log n)$, are sufficient to find the shortest vector in the original lattice. Importantly, both techniques are compatible with nearly all sieving algorithms. While they do not improve the asymptotic complexity of sieving, their synergy yields subexponential gains in practical runtime and memory usage. These techniques can accelerate sieving algorithms by tens of times in real-world implementations.

The practical impact of G6K's design is profound. Albrecht et al. [27] used G6K to solve the SVP for 151-dimensional lattices, achieving a 400-fold speedup over the previous record. This breakthrough was not an endpoint. At EUROCRYPT 2021, Ducas, Stevens, and Woerden [75] further pushed the boundaries by solving the SVP for lattices of dimensions up to 180 using a GPU-accelerated variant of G6K. This milestone not only demonstrated G6K's exceptional power but also provided critical insights for the concrete security evaluation of lattice-based cryptosystems. G6K thus established itself as a cornerstone tool in modern lattice cryptanalysis, bridging the gap between theoretical advances in sieving and practical lattice reduction.

3.4. Other Basis Reduction Algorithms

There also exist many lattice basis reduction algorithms under alternative frameworks, including dual lattice-based reduction [76,77] and random sampling-based reduction [68,78–80]. In the following, we briefly introduce dual lattice-based reduction algorithms. Although they are not the most mainstream at present, their unique design strategies hold potential reference value for the development of other lattice reduction algorithms.

The lattice basis reduction algorithms discussed in previous sections, such as LLL, BKZ, and G6K, operate directly on the primal lattice to find short vectors and reduce bases. Beyond these primal lattice-based paradigms, dual lattice-based reduction algorithms form another important branch. These algorithms leverage the intrinsic connection between a lattice and its dual lattice to design reduction strategies, yielding unique advantages in theoretical analysis.

At STOC 2008, Gama and Nguyen [76] proposed the slide reduction algorithm, which capitalizes on a key property of dual lattices: Maximizing the last Gram–Schmidt norm $\left\|\mathbf{b}_\beta^*\right\|$ for a β-dimensional lattice $\Lambda = \mathcal{L}(\mathbf{b}_1, \ldots, \mathbf{b}_\beta)$ is equivalent to finding the shortest non-zero vector in the dual lattice $\widehat{\Lambda}$ of Λ. Thus, we can iteratively invoke SVP oracles on both primal and dual bases to smooth the GSA line as much as possible.

Beyond leveraging dual lattice properties, slide reduction differs significantly from BKZ-like algorithms in one critical aspect. During both primal and dual reduction, slide reduction minimizes the overlap of projected sublattices, whereas BKZ-like algorithms maximize such overlap. This design choice renders slide reduction highly amenable to complexity analysis. Gama and Nguyen directly adopted the modified potential-based proof technique, which is originally used in the complexity analysis of LLL [24], to derive elegant theoretical results. In contrast, BKZ-like algorithms are notoriously hard to analyze, even when employing the proof technique based on discrete dynamic systems [81,82]. However, despite its theoretical elegance, slide reduction underperforms compared to BKZ in practical applications. It seems that maximizing the overlap of projected sublattices leads to higher-quality reduced bases, while minimizing the overlap leads to lower-quality reduced bases.

Building on this theoretical foundation, Micciancio and Walter enhanced the practicality of dual lattice-based methods at EUROCRYPT 2016 [77] by proposing Self-Dual BKZ (SDBKZ). This algorithm combines elements of both BKZ and slide reduction. Its practical performance is comparable to that of BKZ 2.0, yet its theoretical efficiency is easier to analyze. Despite substantial research efforts devoted to improving BKZ-like algorithms, little progress has been made in advancing dual lattice-based reduction algorithms. It is plausible that more theoretical and practical improvements can be achieved by drawing insights from such alternative frameworks.

4. Exact Algorithms

In this section, we present several mainstream algorithms for solving exact lattice problems. In Section 4.1, we discuss enumeration algorithms. As the earliest class of exact lattice algorithms, they dominated the field in terms of practical performance for a long period, particularly for lattices of small-to-medium dimensions. However, with the advent of sieving algorithms, as discussed in Section 4.2, enumeration algorithms were surpassed by the latter in both asymptotic complexity and practical performance. In fact, sieving algorithms are currently widely employed to evaluate the concrete security of lattice cryptosystems. On the other hand, there exist alternative approaches to solving exact lattice problems. In Section 4.3, we also introduce alternative algorithms for solving exact lattice problems, including the discrete Gaussian sampling-based algorithms and Voronoi cell-based algorithms.

4.1. Enumeration

Lattice enumeration algorithms follow a straightforward idea. Given a lattice basis $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$, to solve SVP, one enumerates all lattice points within the n-dimensional hyperball $\mathcal{B}_n(R)$. The shortest non-zero lattice point among these is the solution to the SVP. According to the Gaussian Heuristic, the radius R is typically set to $c \cdot \mathrm{GH}(\mathcal{L}(\mathbf{B}))$ for some constant c to guarantee the presence of non-zero lattice points inside the hyperball while minimizing the radius.

Generally, a lattice does not possess an orthogonal basis. To enumerate all lattice points within a given region, one employs the Gram–Schmidt orthogonalized basis $\mathbf{B}^* = (\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*)$ to decouple each dimension and analyze the coefficients of each individual dimension. In particular, suppose a lattice vector $\mathbf{w} = w_1\mathbf{b}_1 + \ldots + w_n\mathbf{b}_n$ has a norm of $\|\mathbf{w}\| = \lambda_1(\mathcal{L}(\mathbf{B})) \leq R$. Then, we can express $\mathbf{w}$ as

$$\mathbf{w} = \sum_{i=1}^{n} w_i \mathbf{b}_i = \sum_{j=1}^{n} \sum_{i=j}^{n} \mu_{i,j} w_i \mathbf{b}_j^*.$$

The orthogonal projection of $\mathbf{w}$ satisfies, for $1 \leq k \leq n$,

$$\|\pi_{n+1-k}(\mathbf{w})\|^2 = \sum_{j=n+1-k}^{n} \left[\left(w_j + \sum_{i=j+1}^{n} \mu_{i,j} w_i \right)^2 \cdot \|\mathbf{b}_j^*\|^2 \right] \leq R^2.$$

This allows us to perform an exhaustive search for the coordinates $(w_n, \ldots, w_1) \in \mathbb{Z}^n$ (in the reverse order) by rewriting the inequality, i.e., for $k = n, n-1, \ldots, 1$,

$$\left(w_{n+1-k} + \sum_{i=n+2-k}^{n} \mu_{i,n+1-k} w_i \right)^2 \leq \frac{R^2 - \sum_{j=n+2-k}^{n} \left[\left(w_j + \sum_{i=j+1}^{n} \mu_{i,j} w_i \right)^2 \cdot \|\mathbf{b}_j^*\|^2 \right]}{\|\mathbf{b}_{n+1-k}^*\|^2}.$$

The space complexity is trivially $\mathrm{poly}(n)$, as the enumeration algorithm merely requires storing information about the current lattice point, the smallest length encountered, and its corresponding coefficient vector. Notably, almost all other exact lattice algorithms demand exponential space. Thus, space complexity constitutes a key advantage of enumeration algorithms.

Analyzing the time complexity requires considering the enumeration tree. Specifically, the enumeration process can be viewed as a depth-first search (DFS) of this enumeration tree, where each vector in the projected lattice $\mathcal{L}_{[n+1-k,n]}$ corresponds to a node at depth k of the tree. It is worth emphasizing that the quality of the lattice basis is paramount to the time complexity of enumeration algorithms. Generally, the higher the quality of the

given lattice basis, the lower the time complexity of the algorithm. For LLL-reduced bases, the time complexity is upper-bounded by $2^{O(n^2)}$ [83], whereas for quasi-HKZ-reduced bases, this upper bound is tightened to $n^{\frac{n}{2e}+o(n)}$ [84]. A natural question arises: Can we further reduce the time complexity by employing an even higher-quality reduced basis? Nguyen provided a negative answer, proving that the time complexity cannot be improved beyond $n^{\frac{n}{8}+o(n)}$, regardless of how high the quality of the lattice basis used is [85].

Pruning techniques, first proposed by Schnorr and Horner [86], stand as the core optimization methods for enumeration algorithms. The techniques have significantly enhanced the practical performance of enumeration algorithms, enabling them to outperform sieving algorithms in practical terms for lattices up to approximately 70∼80 dimensions [73]. From the perspective of the enumeration tree, the goal is to cut the subtrees with a low probability of yielding a solution. Thus, pruned enumeration cannot guarantee finding an optimal solution. But the loss in success probability is often acceptable compared to the gains in time complexity. For instance, the success probability may drop to $\frac{1}{2^3}$ while only requiring $\frac{1}{2^{10}}$ of the original time. Geometrically, pruned enumeration targets a subregion of $\Lambda \cap \mathcal{B}_n(R)$. In particular, the enumeration region of cylinder pruning [29] is the intersection of hyper-cylinders, whereas that of discrete pruning [30] is the intersection of the union of many disjointed hyper-boxes with the hyperball. On the other hand, unsurprisingly, lattice enumeration algorithms can be quadratically accelerated on quantum computers with their time complexity reduced from T to approximately $\sqrt{T}$ [87].

Very recently, Chen et al. introduced a time–memory trade-off technique for lattice enumeration [88]. Their approach performs enumeration separately on the head and tail blocks of the lattice basis. Then, after lifting the vectors enumerated from the tail blocks, the algorithm search for the shortest difference vector between the lifted vectors from the tail block enumeration and the vectors generated from the head block enumeration is performed. This difference vector serves as a solution to SVP. In a certain sense, it may be regarded as a hybrid of enumeration and sieving algorithms. Notably, their approach reduced the time complexity to $n^{\frac{n}{4e}+o(n)}$, in exchange for a space complexity of the same asymptotic order $n^{n/(4e)+o(n)}$.

4.2. Sieving

A natural question arises: Can we sample points, hopefully efficient, within $\mathcal{L} \cap \mathcal{B}(R)$, for some $R = O(\lambda_1(\mathcal{L}))$? Suppose we can sample enough points relatively uniformly and efficiently in this region, for instance, $2^{O(n)}$ points, then there is a considerable probability of directly sampling the shortest vector. However, we currently have no idea how to construct such an efficient sampling algorithm. In fact, this is itself a very challenging problem related to the discrete Gaussian sampling (DGS) problem (see Section 4.3. We can only efficiently sample lattice points that are exponentially farther away, i.e., $\mathcal{L} \cap \mathcal{B}(R)$ with $R = 2^{O(n)} \cdot \lambda_1(\mathcal{L})$. The core design idea of sieving algorithms is to continuously perform pairwise reduction on these long vectors to obtain shorter ones. For example, if $\|\mathbf{v}_1 \pm \mathbf{v}_2\| < \max\{\|\mathbf{v}_1\|, \|\mathbf{v}_2\|\}$, then replace the vector corresponding to $\max\{\|\mathbf{v}_1\|, \|\mathbf{v}_2\|\}$ with $\|\mathbf{v}_1 \pm \mathbf{v}_2\|$. It is worth noting that if $\mathbf{v}_1$ and $\mathbf{v}_2$ are both lattice points, then $\mathbf{v}_1 \pm \mathbf{v}_2$ is also a lattice point.

The development of lattice sieving algorithms dates back to the early 2000s, with the AKS-sieve representing the first pivotal advancement in this field. The AKS-sieve algorithm, proposed by Ajtai, Kumar, and Sivakumar at STOC 2001 [33], stands as the first lattice sieving algorithm equipped with rigorous theoretical foundations. It achieves time and space complexities of $2^{5.90n+o(n)}$ and $2^{2.95n+o(n)}$, respectively [89]. Note that this marks a substantial improvement over the enumeration methods prevalent at the time, improving the time complexity from super-exponential $n^{O(n)}$ to single-exponential $2^{O(n)}$.

A distinctive advancement in the AKS-sieve resides in its invention of the perturbation mechanism, which underpins rigorous theoretical validations of both its complexity and correctness. Nevertheless, this perturbation-based approach proves computationally costly in real-world implementations. Leveraging the foundational framework of the AKS-sieve, Nguyen and Vidick [89] abandoned the inefficient perturbation technique and developed a heuristic alternative known as the NV-sieve algorithm. Though it depends on some heuristic assumptions concerning vector distribution, the NV-sieve delivered a notable efficiency gain, boasting time and space complexities of $2^{0.415n+o(n)}$ and $2^{0.2075n+o(n)}$, respectively.

At SODA 2010, Micciancio and Voulgaris [34] proposed a new class of lattice sieving, called the MV-like sieving paradigm, deviating from the AKS-like paradigm, encompassing the List-sieve and Gauss-sieve algorithms. The List-sieve can be rigorously proved to have the time and space complexities of $2^{3.20n+o(n)}$ and $2^{1.33n+o(n)}$, respectively. In contrast, the heuristic Gauss-sieve employs a greedier reduction strategy akin to the classical Gauss–Lagrange reduction. It matches the NV-sieve's time and space complexities yet offers better practical memory efficiency.

Later, researchers prioritized enhancing heuristic sieve algorithms, as their practicality made them critical to the concrete security estimation of lattice-based cryptography. One of the optimization techniques was refining the iteration method. The Level-sieve algorithms proposed by Wang et al. [90] and Zhang et al. [91] pushed time and space complexities to $2^{0.378n+o(n)}$ and $2^{0.283n+o(n)}$, respectively. Derived from the NV-sieve, its core logic is dividing each sieve iteration round into multiple levels by the length of vectors. This drastically cuts the total number of vector pairs to check, thus reducing time complexity.

Another notable innovative advancement involved moving away from traditional pairwise vector reduction toward multi-vector reduction. Building on the List-sieve, Bai, Laarhoven, and Stehle [92] put forward the Tuple-sieve algorithm, with its time complexity $2^{0.481n+o(n)}$ and space complexity $2^{0.189n+o(n)}$. One standout characteristic of the Tuple-sieve is that it performs simultaneous reductions on k-tuples of vectors, where $k > 2$, instead of restricting reduction to the conventional method of processing only vector pairs. Seeking to further boost the Tuple-sieve's efficiency, Herold et al. [93,94] developed an optimized variant named the HKL-Tuple-sieve algorithm. This improved algorithm refined the short vector discovery mechanism of the original Tuple-sieve by optimizing the strategy for tuple selection and integrating more streamlined reduction rules. These targeted adjustments led to a noticeable balanced complexity. The HKL-Tuple-sieve's time complexity is $2^{0.330n+o(n)}$, while its space complexity is $2^{0.208n+o(n)}$.

In the advancement of lattice sieve algorithms, a parallel strand of optimization targeted speeding up the nearest neighbor search (NNS) subroutine, which is a common bottleneck in sieve iterations. At CRYPTO 2015, Laarhoven proposed the Hash-sieve algorithm [72], built upon the Gauss-sieve. This algorithm adopted the Locality-Sensitive Hashing (LSH) technique to accelerate the brute-force search phase within each sieve iteration. Subsequently, at SODA 2016, Becker, Ducas, Gama, and Laarhoven [32] introduced the BDGL-sieve algorithm. It employed the Locality-Sensitive Filter (LSF) technique, one analogous to LSH, to further boost the efficiency of the brute-force search phase. The algorithm achieves time and space complexities of $2^{0.292n+o(n)}$ and $2^{0.208n+o(n)}$, respectively. This stands as the state-of-the-art (heuristic) classical lattice sieve algorithm and also serves as the state-of-the-art exact classical SVP solver. Beyond classical computing frameworks, Laarhoven [95] further reduced the time complexity to $2^{0.265n+o(n)}$ leveraging Grover's algorithm [2] under the quantum computing model.

At ANTS 2014, Becker, Gama, and Joux [96] introduced a distinct sieving approach called the Overlattice-sieve algorithm, boasting time and space complexities of $2^{0.3774n}$ and $2^{0.2925n}$, respectively. It lags behind the BDGL-sieve in complexity results yet holds a special

advantage. It functions as an inherent exact CVP solver, capable of addressing both CVP and SVP. The complexities of the aforementioned sieving algorithms are summarized in Table 2, which is sorted by time complexity.

Table 2. Summary of complexities of lattice sieving algorithms.

Algorithms	Authors	Conference/Journal (Year)	Time	Space
AKS-sieve [33]	Ajtai, Kumar, Sivakumar	STOC (2001)	$2^{5.90n+o(n)}$	$2^{2.95n+o(n)}$
List-sieve [34]	Micciancio, Voulgaris	SODA (2010)	$2^{3.20n+o(n)}$	$2^{1.33n+o(n)}$
Tuple-sieve [92]	Bai, Laarhoven, Stehlé	LMS J. Comput. Math. (2016)	$2^{0.481n+o(n)}$	$2^{0.189n+o(n)}$
NV-sieve [89]	Nguyen, Vidick	J. Math. Crypt. (2008)	$2^{0.415n+o(n)}$	$2^{0.2075n+o(n)}$
Gauss-sieve [34]	Micciancio, Voulgaris	SODA (2010)	$2^{0.415n+o(n)}$	$2^{0.2075n+o(n)}$
Level-sieve [90,91]	Wang et al.; Zhang et al.	AsiaCCS (2011); SAC (2013)	$2^{0.378n+o(n)}$	$2^{0.283n+o(n)}$
Overlattice-sieve [96]	Becker, Gama, Joux	ANTS (2014)	$2^{0.3774n+o(n)}$	$2^{0.2925n+o(n)}$
Hash-sieve [72]	Laarhoven	CRYPTO (2015)	$2^{0.337n+o(n)}$	$2^{0.208n+o(n)}$
HKL-Tuple-sieve [93,94]	Herold, Kirshanova, Laarhoven	PKC (2017); PKC (2018)	$2^{0.330n+o(n)}$	$2^{0.208n+o(n)}$
BDGL-sieve [32]	Becker, Ducas, Gama, Laarhoven	SODA (2016)	$2^{0.292n+o(n)}$	$2^{0.208n+o(n)}$
BDGL-sieve with quantum speedup [95]	Laarhoven	PhD Thesis (2016)	$2^{0.265n+o(n)}$	$2^{0.208n+o(n)}$

A natural and critical question arises in the field: Is there still room for further improvements in sieving algorithms, and can we be confident that the current state-of-the-art sieving algorithms will remain unrivaled in the decades to come? This question gains urgency as PQC standards solidify, requiring long-term security guarantees. At CRYPTO 2021, Kirshanova and Laarhoven [97] provided a partial answer by proving that the LSF technique employed in the BDGL-sieve has achieved the optimal asymptotic complexity for the NNS problem in the lattice cryptographic setting, which is a core subroutine in both AKS-like and MV-like sieving frameworks. Consequently, researchers can largely rule out the possibility of further asymptotic improvements that solely target the NNS subroutine within these two mainstream frameworks.

On the other hand, very recently, Shen, Li, and Wang [98] demonstrated that the Overlattice-sieve [96] has already achieved the complexity lower bound within the overlattice-based sieving paradigm. Combined with Kirshanova and Laarhoven's lower-bound result [97], these findings jointly set the lower bound of the complexity for existing lattice sieving algorithms. Specifically, no incremental refinements under current technical sieving frameworks can attain a fundamental leap in time complexity. Any significant advancement will necessitate revolutionary breakthroughs in the core technologies of lattice sieving.

Beyond asymptotic complexity improvements, researchers have also focused on enhancing the practical implementation effectiveness of sieve methods. Techniques such as the progressive sieve [74], dimension-for-free technique [73], and the G6K framework [27] have all significantly optimized the practical execution performance of sieve methods, as well as the performance of lattice basis reduction algorithms, as discussed in Section 3.3. Notably, the ideas behind these improved sieving algorithms are largely derived from other lattice algorithms. For example, the progressive sieve [74] can be seen as drawing inspiration from the Progressive BKZ [67], while the dimension-for-free technique [73] originates from the lift operation in enumeration algorithms. Among these, G6K, as described in Section 3.3, has become one of the primary tools for analyzing the concrete security of lattice-based cryptosystems, widely used in evaluating candidates for NIST's PQC standardization.

Not only do sieving algorithms stand as the state-of-the-art exact SVP solvers, but they also rank as the most effective practical methods for high-dimensional lattice problems. Nowadays, one of the key bottlenecks in tackling high-dimensional SVP instances via sieving algorithms lies in the cost of random memory access. In 2024, Zhao, Ding, and Yang [99]

enhanced the memory access efficiency of the BGJ-sieve [100], reduced the storage of vector coordinates, and thereby lowered overall memory requirements. These improvements enabled them to successfully solve the 200-dimensional SVP Challenge [101]. Very recently, in 2026, Ding and Zhao further leveraged sieving algorithms to tackle an even higher-dimensional instance: They successfully solved the 210-dimensional SVP Challenge [101]. Currently, they have not disclosed the specific technical details of their implementation.

4.3. Other Exact Algorithms

In this section, we present two other categories of exact lattice algorithms: discrete Gaussian sampling (DGS)-based algorithms and Voronoi cell-based algorithms.

The core idea of DGS-based algorithms is analogous to that of sieving algorithms. Informally, from the perspective of individual vectors, sieving algorithms focus on the length of each single lattice point and employ multiple sieve rounds to shorten the length of these points. In contrast, DGS-based algorithms adopt the perspective of discrete Gaussian distributions, consider the "width" of these distributions, and utilize multiple sieve-like rounds to narrow this width. Notably, DGS-based algorithms are the fastest provable exact lattice algorithms to date. They are capable of solving both exact SVP [38] and CVP [39], with time and space complexities of $2^{n+o(n)}$ and $2^{n+o(n)}$. However, DGS-based algorithms hold far greater theoretical value than practical utility. To the best of the authors' knowledge, there is no practical implementations of such algorithms.

In particular, discrete Gaussian distributions feature a critical width parameter s. Generally, when s is much larger than the smoothing parameter of the lattice [102], the discrete Gaussian distribution is highly smooth, i.e., closely resembling the continuous Gaussian distribution, and thus admits efficient sampling. Conversely, discrete Gaussian distributions with a much smaller width parameter are highly discrete, rendering the sampling process very challenging. DGS-based algorithms firstly sample an exponential number of lattice points that conform to discrete Gaussian distributions with a very large width s. They then iteratively "sieve" these points to narrow the width s of the resulting discrete Gaussian distributions. Ultimately, exact lattice problems can be tackled via the final list corresponding to a very small width s.

In the following, we present another category of exact lattice algorithms: Voronoi cell-based algorithms. This approach is grounded in the geometric properties of Voronoi cells. The Voronoi cell $\mathcal{V}(\mathbf{w})$ corresponding to a lattice point $\mathbf{w}$ comprises all points in the Euclidean space $\mathrm{span}(\Lambda)$ that are closest to $\mathbf{w}$, i.e.,

$$\mathcal{V}(\mathbf{w}) = \{\mathbf{x} \in \mathbb{R}^n \mid \forall \mathbf{v} \in \Lambda,\ \|\mathbf{x} - \mathbf{w}\| \leq \|\mathbf{x} - \mathbf{v}\|\}.$$

For the exact CVP problem, the goal is to find a lattice point $\mathbf{w}$ that is closest to a given target vector $\mathbf{t}$. This is equivalent to identifying a lattice vector $\mathbf{w}$ such that $\mathbf{t}' = \mathbf{t} - \mathbf{w} \in \mathcal{V}(\mathbf{0})$, i.e., $\mathbf{t} \in \mathcal{V}(\mathbf{w})$. All Voronoi cells are congruent, meaning we only need to analyze the properties of $\mathcal{V} := \mathcal{V}(\mathbf{0})$. Specifically, $\mathcal{V}$ is a centrally symmetric polytope with at most $2(2^n - 1)$ facets. We define $\mathrm{VR}(\Lambda)$, the set of Voronoi-relevant vectors of lattice Λ, as the lattice vectors that induce the facets of $\mathcal{V}$. An illustration of Voronoi-relevant vectors is provided in Figure 2.

Voronoi cell-based algorithms follow an approach consisting of two steps. First, compute the set of Voronoi-relevant vectors $\mathrm{VR}(\Lambda)$ for the target lattice Λ. Second, utilize $\mathrm{VR}(\Lambda)$ to identify a lattice vector $\mathbf{w}$, satisfying $\mathbf{t}' = \mathbf{t} - \mathbf{w} \in \mathcal{V}(\mathbf{0})$. This is equivalent to solving the Closest Vector Problem with Preprocessing (CVPP) with the additional hint of all Voronoi-relevant vectors $\mathrm{VR}(\Lambda)$. Micciancio [103] proved that CVPP is also NP-hard when the preprocessing advice is of polynomial size. At STOC 2010, Micciancio and Voulgaris [35] introduced a deterministic algorithm to construct a lattice's Voronoi cell

in $2^{2n+o(n)}$ time and $2^{n+o(n)}$ space, which allows for solving CVPP in $2^{2n+o(n)}$ time and $2^{n+o(n)}$ space. Subsequently, at SODA 2015, Dadush and Bonifas presented a randomized algorithm to solve CVPP in $2^{n+o(n)}$ time and $2^{n+o(n)}$ space. However, their algorithm did not improve the process of computing $\mathrm{VR}(\Lambda)$, meaning the time complexity for solving exact CVP remains unchanged at $2^{2n+o(n)}$.

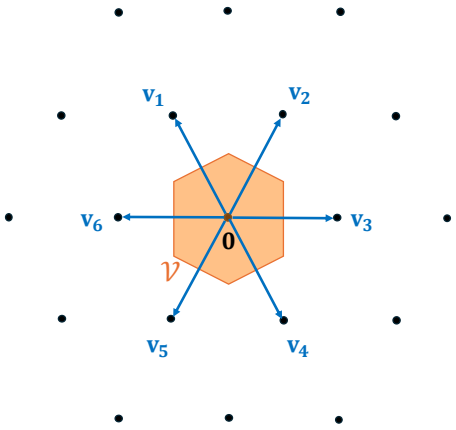

Figure 2. The set of Voronoi-relevant vectors $\mathrm{VR}(\Lambda) = \{\mathbf{v}_1, \ldots, \mathbf{v}_6\}$ decides the Voronoi cell $\mathcal{V}$.

On the other hand, to make the Voronoi cell-based approach more practical, it is generally beneficial to avoid using the entire set of relevant vectors, a strategy widely adopted in practical lattice algorithms. In fact, an n-dimensional lattice has at most $2^{n+1} - 2$ relevant vectors, and this number is attained with overwhelming probability if the lattice basis is chosen at random from a continuous distribution [104]. Doulgerakis et al. [105] and Laarhoven [37] considered the possibility of using only part of all relevant vectors. In other words, the preprocessing computes an approximation Voronoi cell, instead of the exact one. These approximate representations of Voronoi cells are lossy but are also smaller and faster to process. Under certain heuristics, they proposed the randomized version of the iterative slicer algorithm to solve CVP. At PKC 2020, Ducas et al. [106] derived sharp asymptotic bounds on the success probability of the randomized slicer and characterized time–space complexity trade-offs. In particular, under several heuristic assumptions, they proved that the iterative slicer can solve a single CVPP instance in $2^{0.264n+o(n)}$ time and $2^{0.185n+o(n)}$ space when employing their optimized nearest neighbor data structures [106].

5. Discussions

Since the advent of the LLL algorithm [24] and enumeration algorithms [28,83] in the 1980s, classical lattice algorithms have undergone dramatic transformations. A wealth of algorithms for solving both approximate and exact lattice problems has emerged continuously. In this work, we systematically review the development of current mainstream classical lattice algorithms, and we hope this survey will be informative to readers interested in these topics.

In the following, we summarize our key insights on the development pattern of classical lattice algorithms. In recent years, the development of classical lattice algorithms has gradually exhibited a significant trend of technological convergence. Although the progress of many single technical routes has largely hit a bottleneck, the insights and advantages gained from one route can often be leveraged to improve other technical routes. For instance, replacing the enumeration-based SVP oracles in BKZ [25] with the sieving algorithms in G6K [27] has significantly boosted the efficiency of lattice reduction. Similarly, the idea behind progressive sieve [74] draws inspiration from Progressive BKZ [67]. The dimension-for-free technique [73] also originates from the rank reduction technique in enu-

meration algorithms. The Locality-Sensitive Filter (LSF) technique in the BDGL-sieve [32] is derived from nearest neighbor search (NNS) algorithms in other fields. These examples demonstrate that cross-technique integration has become a crucial and effective way to break through performance bottlenecks of individual algorithms and will continue to be an important driving force for the future advancement of classical lattice algorithms.

According to the current development status and remaining challenges, we highlight several important future research directions for classical lattice algorithms.

- First, beyond framework-level innovations, the approximation and precise estimation of lower-order complexity terms represent a meaningful research direction. The accurate estimation of lower-order factors in the complexity of lattice algorithms has not been thoroughly explored, yet it is crucial for the concrete security evaluation of lattice-based cryptography.
- Second, it is necessary to carefully examine the practical validity of heuristic assumptions employed in various lattice algorithms. Many state-of-the-art algorithms, especially sieving algorithms, rely on several heuristic assumptions whose rationality and stability still lack comprehensive verification, which deserves further study.
- Third, the efficient engineering implementation of these algorithms and the adoption of GPU-accelerated computing remain practically significant challenges. Improving real-world efficiency and scalability is essential for applying these algorithms in practical security evaluation of post-quantum cryptosystems.
- Fourth, the generalization of classical lattice algorithms to complex structured lattices requires in-depth investigation. Future research should extend and adapt classical lattice algorithms to such structured lattices to support more accurate security analysis of lattice-based cryptosystems.

Author Contributions: Conceptualization, T.S. and X.L.; Methodology, T.S.; Formal Analysis, T.S. and X.L.; Validation, X.L.; Writing—original draft preparation, T.S.; Writing—review and editing, X.L.; Supervision, T.S. and X.L.; Project Administration, X.L.; Funding Acquisition, X.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by NSFC (62541315) and Shanghai Municipal Education Commission (2021-01-07-00-08-E00101).

Data Availability Statement: No new data were generated in this study. All supporting materials are available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

PQC	Post-Quantum Cryptography
SVP	Shortest Vector Problem
CVP	Closest Vector Problem
PKE	Public Key Encryption
KEMs	Key Encapsulation Mechanisms
SIG	Digital Signatures
ABE	Attribute-Based Encryption
FHE	Fully Homomorphic Encryption
LWE	Learning With Error
SIS	Short Integer Solution
HSVP	Hermite Shortest Vector Problem
RHF	Root Hermite Factor

GSA	Geometric Series Assumption
BKZ	Block Korkine–Zolotarev
HKZ	Hermite–Korkine–Zolotarev
G6K	General Sieve Kernel
DFS	Depth-First Search
DGS	Discrete Gaussian Sampling
LSH	Locality-Sensitive Hashing
LSF	Locality-Sensitive Filters
CVPP	Closest Vector Problem with Preprocessing

References

1. Shor, P.W. Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In Proceedings of the 35th Annual Symposium on Foundations of Computer Science (SFCS), Santa Fe, NM, USA, 20–22 November 1994; pp. 124–134.
2. Grover, L.K. A Fast Quantum Mechanical Algorithm for Database Search. In Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC), Philadelphia, PA, USA, 22–24 May 1996; pp. 212–219.
3. Post-Quantum Cryptography: Selected Algorithms. Available online: https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/selected-algorithms (accessed on 1 January 2026).
4. Ajtai, M. The Shortest Vector Problem in L_2 is *NP*-hard for Randomized Reductions (Extended Abstract). In Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, TX, USA, 23–26 May 1998; pp. 10–19.
5. Boas, P.V.E. Another NP-complete partition problem and the complexity of computing short vectors in a lattice. *Inf. Process. Lett.* **1981**, *12*, 212–215.
6. Abrams, D.S.; Lloyd, S. Nonlinear quantum mechanics implies polynomial-time solution for NP-complete and #P problems. *Phys. Rev. Lett.* **1998**, *81*, 3992–3995.
7. Zheng, C. Universal quantum simulation of single-qubit nonunitary operators using duality quantum algorithm. *Sci. Rep.* **2021**, *11*, 3960. [CrossRef] [PubMed]
8. Ajtai, M. Generating Hard Instances of Lattice Problems. In Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC), Philadelphia, PA, USA, 22–24 May 1996; pp. 99–108.
9. Regev, O. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. In Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC), Baltimore, MD, USA, 22–24 May 2005; pp. 84–93.
10. Bos, J.W.; Ducas, L.; Kiltz, E.; Lepoint, T.; Lyubashevsky, V.; Schanck, J.M.; Schwabe, P.; Seiler, G.; Stehlé, D. CRYSTALS—Kyber: A CCA-Secure Module-Lattice-Based KEM. In Proceedings of the 2018 IEEE European Symposium on Security and Privacy (EuroS&P 2018), London, UK, 24–26 April 2018; pp. 353–367.
11. Alkim, E.; Ducas, L.; Pöppelmann, T.; Schwabe, P. Post-quantum Key Exchange—A New Hope. In Proceedings of the 25th USENIX Security Symposium (USENIX Security 16), Austin, TX, USA, 10–12 August 2016; pp. 327–343.
12. Ducas, L.; Kiltz, E.; Lepoint, T.; Lyubashevsky, V.; Schwabe, P.; Seiler, G.; Stehlé, D. CRYSTALS-Dilithium: A Lattice-Based Digital Signature Scheme. *IACR Trans. Cryptogr. Hardw. Embed. Syst.* **2018**, *2018*, 238–268. [CrossRef]
13. Gorbunov, S.; Vaikuntanathan, V.; Wee, H. Attribute-Based Encryption for Circuits. In Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC), Palo Alto, CA, USA, 1–4 June 2013; pp. 545–554.
14. Chillotti, I.; Gama, N.; Georgieva, M.; Izabachène, M. Improving TFHE: Faster Packed Homomorphic Encryption. In Proceedings of the 23rd International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), Hong Kong SAR, China, 3–7 December 2017; pp. 3–33.
15. Cheon, J.H.; Kim, A.; Kim, M.; Song, Y. Homomorphic Encryption for Arithmetic of Approximate Numbers. In Proceedings of the 23rd International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), Hong Kong SAR, China, 3–7 December 2017; pp. 409–437.
16. Albrecht, M.R.; Player, R.; Scott, S. On the Concrete Hardness of Learning with Errors. *J. Math. Cryptol.* **2015**, *9*, 169–203. [CrossRef]
17. Albrecht, M.R.; Curtis, B.R.; Deo, A.; Davidson, A.; Player, R.; Postlethwaite, E.W.; Virdia, F.; Wunderer, T. Estimate all the LWE, NTRU schemes! In Proceedings of the 16th International Conference on Security and Cryptography (SCN), Amalfi, Italy, 5–7 September 2018; pp. 1–28.
18. Wenger, E.; Saxena, E.; Malhou, M.; Thieu, E.; Lauter, K. Benchmarking Attacks on Learning with Errors. In Proceedings of the 45th IEEE Symposium on Security and Privacy (S&P), San Francisco, CA, USA, 20–23 May 2024; pp. 279–297.
19. Lyubashevsky, V.; Peikert, C.; Regev, O. On Ideal Lattices and Learning with Errors over Rings. In Proceedings of the 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2010), French Riviera, France, 30 May–3 June 2010; pp. 1–23.

20. Peikert, C.; Pepin, Z. Algebraically Structured LWE, Revisited. In Proceedings of the 17th International Conference on Theory of Cryptography (TCC 2019), Nuremberg, Germany, 1–5 December 2019; pp. 1–23.
21. Bernstein, D.J. *Post-Quantum Cryptography*; Springer Nature: Cham, Switzerland, 2008; Volume 549, pp. 188–194.
22. Carrier, K.; Meyer-Hilfiger, C.; Shen, Y.; Tillich, J.-P. Assessing the Impact of a Variant of MATZOV's Dual Attack on Kyber. In Proceedings of the 45th Annual International Cryptology Conference (CRYPTO 2025), Santa Barbara, CA, USA, 17–21 August 2025; pp. 444–476.
23. Karenin, A.; Kirshanova, E.; May, A.; Nowakowski, J. Fast Slicer for Batch-CVP: Making Lattice Hybrid Attacks Practical. In Proceedings of the 31st International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2025), Melbourne, VIC, Australia, 8–12 December 2025; pp. 100–132.
24. Lenstra, A.K.; Lenstra, H.W.; Lovász, L. Factoring Polynomials with Rational Coefficients. *Math. Ann.* **1982**, *261*, 515–534. [CrossRef]
25. Schnorr, C.-P.; Euchner, M. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Program.* **1994**, *66*, 181–199. [CrossRef]
26. Chen, Y.; Nguyen, P.Q. BKZ 2.0: Better Lattice Security Estimates. In Proceedings of the 19th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), Seoul, Republic of Korea, 4–8 December 2011; pp. 1–20.
27. Albrecht, M.R.; Ducas, L.; Herold, G.; Kirshanova, E.; Postlethwaite, E.W.; Stevens, M. The General Sieve Kernel and New Records in Lattice Reduction. In Proceedings of the 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques (Eurocrypt), Darmstadt, Germany, 19–23 May 2019; pp. 719–752.
28. Kannan, R. Minkowski's Convex Body Theorem and Integer Programming. *Math. Oper. Res.* **1987**, *12*, 415–440.
29. Gama, N.; Nguyen, P.Q.; Regev, O. Lattice Enumeration Using Extreme Pruning. In Proceedings of the 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), Nice, France, 30 May–3 June 2010; pp. 257–278.
30. Albrecht, M.R.; Nguyen, P.Q. Random Sampling Revisited: Lattice Enumeration with Discrete Pruning. In Proceedings of the 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), Paris, France, 23–27 April 2017; pp. 535–563.
31. Albrecht, M.R.; Bai, S.; Fouque, P.-A.; Kirchner, P.; Stehlé, D.; Wen, W. Faster Enumeration-Based Lattice Reduction: Root Hermite Factor Bounds in Time $2^{0.207n}$. In Proceedings of the 40th Annual International Cryptology Conference (CRYPTO), Santa Barbara, CA, USA, 17–21 August 2020; pp. 585–615.
32. Becker, A.; Ducas, L.; Gama, N.; Laarhoven, T. New directions in nearest neighbor searching with applications to lattice sieving. In Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), Arlington, VA, USA, 10–12 January 2016; pp. 10–24.
33. Ajtai, M.; Kumar, R.; Sivakumar, D. A sieve algorithm for the shortest lattice vector problem. In Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC), Heraklion, Crete, Greece, 13–15 July 2001; pp. 601–610.
34. Micciancio, D.; Voulgaris, P. Faster Exponential Time Algorithms for the Shortest Vector Problem (SVP). In Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), Austin, TX, USA, 17–19 January 2010; pp. 1468–1480.
35. Micciancio, D.; Voulgaris, P. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC 2010), Cambridge, MA, USA, 5–8 June 2010; pp. 351–358.
36. Dadush, D.; Bonifas, N. Short Paths on the Voronoi Graph and Closest Vector Problem with Preprocessing. In Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2015), San Diego, CA, USA, 4–6 January 2015; pp. 295–314.
37. Laarhoven, T. Approximate Voronoi Cells for Lattices, Revisited. *J. Math. Cryptol.* **2021**, *15*, 60–71.
38. Aggarwal, D.; Dadush, D.; Regev, O.; Stephens-Davidowitz, N. Solving the Shortest Vector Problem in 2^n Time Using Discrete Gaussian Sampling: Extended Abstract. In Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing (STOC 2015), Portland, OR, USA, 14–17 June 2015; pp. 733–742.
39. Aggarwal, D.; Dadush, D.; Stephens-Davidowitz, N. Solving the Closest Vector Problem in 2^n Time—The Discrete Gaussian Strikes Again! In Proceedings of the IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS 2015), Berkeley, CA, USA, 17–20 October 2015; pp. 563–582.
40. Mihăiţă, A.; Simion, E. A survey on lattice-based cryptography. *Acta Univ. Apulensis Math.-Inform.* **2012**, *29*, 53–64.
41. Dam, D.-T.; Tran, T.-H.; Hoang, V.-P.; Pham, C.-K.; Hoang, T.-T. A survey of post-quantum cryptography: Start of a new race. *Cryptography* **2023**, *7*, 40. [CrossRef]
42. Liu, F.; Zheng, Z.; Gong, Z.; Tian, K.; Zhang, Y.; Hu, Z.; Li, J.; Xu, Q. A survey on lattice-based digital signature. *Cybersecurity* **2024**, *7*, 7. [CrossRef]
43. Wang, A.; Xiao, D.; Yu, Y. Lattice-based cryptosystems in standardisation processes: A survey. *IET Inf. Secur.* **2023**, *17*, 227–243.

44. Cisneros, M.; Olazabal, J. Lattice-based cryptography in the quantum era: A survey. *Interfaces* **2023**, *2023*, 281–299.
45. Wang, X.; Xu, G.; Yu, Y. Lattice-based cryptography: A survey. *Chin. Ann. Math. B* **2023**, *44*, 945–960. [CrossRef]
46. Albrecht, M.R.; Gheorghiu, V.; Postlethwaite, E.W.; Schanck, J.M. Estimating Quantum Speedups for Lattice Sieves. In Proceedings of the 26th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2020), Daejeon, Republic of Korea, 7–11 December 2020; pp. 583–613.
47. Cramer, R.; Ducas, L.; Peikert, C.; Regev, O. Recovering Short Generators of Principal Ideals in Cyclotomic Rings. In Proceedings of the 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2016), Vienna, Austria, 8–12 May 2016; pp. 559–585.
48. Cramer, R.; Ducas, L.; Wesolowski, B. Mildly Short Vectors in Cyclotomic Ideal Lattices in Quantum Polynomial Time. *J. ACM* **2021**, *68*, 8. [CrossRef]
49. Lee, C.; Pellet-Mary, A.; Stehlé, D.; Wallet, A. An LLL Algorithm for Module Lattices. In Proceedings of the 25th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2019), Kobe, Japan, 8–12 December 2019; pp. 59–90.
50. Mukherjee, T.; Stephens-Davidowitz, N. Lattice Reduction for Modules, or How to Reduce ModuleSVP to ModuleSVP. In Proceedings of the 40th Annual International Cryptology Conference (CRYPTO 2020), Santa Barbara, CA, USA, 17–21 August 2020; pp. 213–242.
51. Arora, S.; Ge, R. New Algorithms for Learning in Presence of Errors. In Proceedings of the 38th International Colloquium on Automata, Languages and Programming (ICALP 2011), Zurich, Switzerland, 4–8 July 2011; pp. 403–415.
52. Steiner, M.J. The Complexity of Algebraic Algorithms for LWE. In Proceedings of the 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2024), Zurich, Switzerland, 26–30 May 2024; pp. 375–403.
53. Albrecht, M.R.; Cid, C.; Faugère, J.-C.; Fitzpatrick, R.; Perret, L. On the complexity of the BKW algorithm on LWE. *Des. Codes Cryptogr.* **2015**, *74*, 325–354.
54. Wei, Y.; Bi, L.; Lu, X.; Wang, K. Memory-Efficient BKW Algorithm for Solving the LWE Problem. In *Proceedings of the 28th IACR International Conference on Practice and Theory of Public-Key Cryptography (PKC 2025), Røros, Norway, 12–15 May 2025*; Jager, T., Pan, J., Eds.; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany; Volume 15675, pp. 331–362.
55. Lovasz, L. *An Algorithmic Theory of Numbers, Graphs and Convexity*; CBMS-NSF Regional Conference Series in Applied Mathematics; SIAM Publications: Philadelphia, PA, USA, 1986; Volume 50.
56. Gama, N.; Nguyen, P.Q. Predicting Lattice Reduction. In Proceedings of the 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2008), Istanbul, Turkey, 13–17 April 2008; pp. 31–51.
57. Bai, S.; Stehlé, D.; Wen, W. Measuring, Simulating and Exploiting the Head Concavity Phenomenon in BKZ. In Proceedings of the 24th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2018), Brisbane, QLD, Australia, 2–6 December 2018; pp. 369–404.
58. Pohst, M. A Modification of the LLL Reduction Algorithm. *J. Symb. Comput.* **1987**, *4*, 123–127. [CrossRef]
59. Lagarias, J.C.; Odlyzko, A.M. Solving Low-Density Subset Sum Problems. *J. ACM* **1985**, *32*, 229–246. [CrossRef]
60. Coppersmith, D. Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities. *J. Cryptol.* **1997**, *10*, 233–260. [CrossRef]
61. Nguyen, P.Q.; Stehlé, D. An LLL Algorithm with Quadratic Complexity. *SIAM J. Comput.* **2009**, *39*, 874–903. [CrossRef]
62. fplll. Available online: https://github.com/fplll/fplll (accessed on 1 January 2026).
63. Ducas, L.; Pulles, L.N.; Stevens, M. Towards a Modern LLL Implementation. In Proceedings of the 31st International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2025), Melbourne, VIC, Australia, 8–12 December 2025; pp. 65–99.
64. Neumaier, A.; Stehlé, D. Faster LLL-type Reduction of Lattice Bases. In Proceedings of the ACM International Symposium on Symbolic and Algebraic Computation (ISSAC 2016), Waterloo, ON, Canada, 19–22 July 2016; pp. 373–380.
65. Seysen, M. Simultaneous reduction of a lattice basis and its reciprocal basis. *Combinatorica* **1993**, *13*, 363–376. [CrossRef]
66. Kirchner, P.; Espitau, T.; Fouque, P.-A. Towards Faster Polynomial-Time Lattice Reduction. In Proceedings of the 41st Annual International Cryptology Conference (CRYPTO 2021), Virtual Event, 16–20 August 2021; pp. 760–790.
67. Aono, Y.; Wang, Y.; Hayashi, T.; Takagi, T. Improved Progressive BKZ Algorithms and Their Precise Cost Estimation by Sharp Simulator. In Proceedings of the 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2016), Vienna, Austria, 8–12 May 2016; pp. 789–819.
68. Schnorr, C.-P. Lattice Reduction by Random Sampling and Birthday Methods. In Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science (STACS 2003), Berlin/Heidelberg, Germany, 27 February–1 March 2003; pp. 145–156.
69. Yu, Y.; Ducas, L. Second Order Statistical Behavior of LLL and BKZ. In Proceedings of the 24th International Conference on Selected Areas in Cryptography (SAC 2017), Ottawa, ON, Canada, 16–18 August 2017; pp. 3–22.
70. Chen, Y. Réduction de Réseau et Sécurité Concrète du Chiffrement Complètement Homomorphe. Ph.D. Thesis, Paris 7, Paris, France, 2013.

71. Albrecht, M.R.; Bai, S.; Li, J.; Rowell, J. Lattice Reduction with Approximate Enumeration Oracles—Practical Algorithms and Concrete Performance. In Proceedings of the 41st Annual International Cryptology Conference (CRYPTO 2021), Virtual Event, 16–20 August 2021; pp. 732–759.
72. Laarhoven, T. Sieving for Shortest Vectors in Lattices Using Angular Locality-Sensitive Hashing. In Proceedings of the 45th Annual International Cryptology Conference (CRYPTO), Santa Barbara, CA, USA, 16–20 August 2015; pp. 520–543.
73. Ducas, L. Shortest Vector from Lattice Sieving: A Few Dimensions for Free. In Proceedings of the 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques (Eurocrypt), Tel Aviv, Israel, 29 April–3 May 2018; pp. 613–642.
74. Laarhoven, T.; Mariano, A. Progressive lattice sieving. In Proceedings of the 11th International Workshop on Post-Quantum Cryptography (PQC), Minneapolis, MN, USA, 15–17 October 2018; pp. 317–337.
75. Ducas, L.; Stevens, M.; van Woerden, W.P.J. Advanced Lattice Sieving on GPUs, with Tensor Cores. In Proceedings of the 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2021), Zagreb, Croatia, 17–21 October 2021; pp. 249–279.
76. Gama, N.; Nguyen, P.Q. Finding Short Lattice Vectors Within Mordell's Inequality. In Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC 2008), Victoria, BC, Canada, 17–20 May 2008; pp. 207–216.
77. Micciancio, D.; Walter, M. Practical, Predictable Lattice Basis Reduction. In Proceedings of the 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2016), Vienna, Austria, 8–12 May 2016; pp. 820–849.
78. Buchmann, J.; Ludwig, C. Practical Lattice Basis Sampling Reduction. In Proceedings of the 7th International Symposium on Algorithmic Number Theory (ANTS-VII), Berlin/Heidelberg, Germany, 23–28 July 2006; pp. 222–237.
79. Fukase, M.; Kashiwabara, K. An Accelerated Algorithm for Solving SVP Based on Statistical Analysis. *J. Inf. Process.* **2015**, *23*, 67–80. [CrossRef]
80. Teruya, T.; Kashiwabara, K.; Hanaoka, G. Fast Lattice Basis Reduction Suitable for Massive Parallelization and Its Application to the Shortest Vector Problem. In Proceedings of the 21st IACR International Conference on Practice and Theory of Public-Key Cryptography (PKC 2018), Rio de Janeiro, Brazil, 25–29 March 2018; pp. 437–460.
81. Hanrot, G.; Pujol, X.; Stehlé, D. Analyzing Blockwise Lattice Algorithms Using Dynamical Systems. In Proceedings of the 31st Annual Cryptology Conference (CRYPTO 2011), Santa Barbara, CA, USA, 14–18 August 2011; pp. 447–464.
82. Li, J.; Nguyen, P.Q. A Complete Analysis of the BKZ Lattice Reduction Algorithm. *J. Cryptol.* **2025**, *38*, 12. [CrossRef]
83. Fincke, U.; Pohst, M. A Procedure for Determining Algebraic Integers of Given Norm. In Proceedings of the European Computer Algebra Conference (EUROCAL '83), London, UK, 28–30 March 1983; pp. 194–202.
84. Hanrot, G.; Stehlé, D. Improved Analysis of Kannan's Shortest Lattice Vector Algorithm. In Proceedings of the 27th Annual International Cryptology Conference (CRYPTO 2007), Santa Barbara, CA, USA, 19–23 August 2007; pp. 170–186.
85. Nguyen, P.Q.; Vallée, B. (Eds.) *The LLL Algorithm—Survey and Applications*; Information Security and Cryptography; Springer: Berlin/Heidelberg, Germany, 2010.
86. Schnorr, C.-P.; Hörner, H.H. Attacking the Chor-Rivest Cryptosystem by Improved Lattice Reduction. In Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT '95), Saint-Malo, France, 21–25 May 1995; pp. 1–12.
87. Aono, Y.; Nguyen, P.Q.; Shen, Y. Quantum Lattice Enumeration and Tweaking Discrete Pruning. In Proceedings of the 24th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT 2018), Brisbane, QLD, Australia, 2–6 December 2018; pp. 405–434.
88. Chen, Y.; Chen, Z.; Guo, T.; Sun, C.; Wen, W.-Q.; Yu, Y. Time-Memory Trade-off for Enumeration. *IACR Cryptol. ePrint Arch.* **2025**, *2227*. Available online: https://eprint.iacr.org/2025/2227 (accessed on 3 March 2026).
89. Nguyen, P.Q.; Vidick, T. Sieve Algorithms for the Shortest Vector Problem (SVP) are Practical. *J. Math. Crypt.* **2008**, *2*, 181–207. [CrossRef]
90. Wang, X.; Liu, M.; Tian, C.; Bi, J. Improved Nguyen-Vidick Heuristic Sieve Algorithm for the Shortest Vector Problem. In Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security (AsiaCCS), Hong Kong SAR, China, 10–13 May 2011; pp. 357–368.
91. Zhang, F.; Pan, Y.; Hu, G. A Three-Level Sieve Algorithm for the Shortest Vector Problem. In Proceedings of the 21st International Conference on Selected Areas in Cryptography (SAC), Montreal, QC, Canada, 14–16 August 2013; pp. 335–352.
92. Bai, S.; Laarhoven, T.; Stehlé, D. Tuple lattice sieving. *LMS J. Comput. Math.* **2016**, *19*, 146–162. [CrossRef]
93. Herold, G.; Kirshanova, E. Improved Algorithms for the Approximate k-List Problem in Euclidean Norm. In Proceedings of the 20th IACR International Conference on Practice and Theory of Public-Key Cryptography (PKC), Yokohama, Japan, 20–23 February 2017; pp. 441–470.

94. Herold, G.; Kirshanova, E.; Laarhoven, T. Speed-Ups and Time–Memory Trade-Offs for Tuple Lattice Sieving. In Proceedings of the 21st IACR International Conference on Practice and Theory of Public-Key Cryptography (PKC), Jerusalem, Israel, 29 April–3 May 2018; pp. 551–580.
95. Laarhoven, T. Search Problems in Cryptography: From Fingerprinting to Lattice Sieving. Ph.D. Thesis, Eindhoven University of Technology, Eindhoven, The Netherlands, 2015.
96. Becker, A.; Gama, N.; Joux, A. Solving shortest and closest vector problems: The decomposition approach. In Proceedings of the 11th Algorithmic Number Theory Symposium (ANTS), Pohang, Republic of Korea, 11–15 August 2014; pp. 339–366.
97. Kirshanova, E.; Laarhoven, T. Lower Bounds on Lattice Sieving and Information Set Decoding. In Proceedings of the 41st Annual International Cryptology Conference (CRYPTO), Virtual Event, 16–20 August 2021; pp. 1–23.
98. Shen, T.; Li, X.; Wang, L. Lower Bound on the Overlattice-Based Sieve Algorithm. *Cryptography* **2026**, *10*, 5. [CrossRef]
99. Zhao, Z.; Ding, J.; Yang, B.-Y. BGJ15 Revisited: Sieving with Streamed Memory Access. *IACR Cryptol. ePrint Arch.* **2024**, *739*. Available online: https://eprint.iacr.org/2024/739 (accessed on 3 March 2026).
100. Becker, A.; Gama, N.; Joux, A. Speeding-up lattice sieving without increasing the memory, using sub-quadratic nearest search. *IACR Cryptol. ePrint Arch.* **2015**, *102*. Available online: https://eprint.iacr.org/2015/522 (accessed on 3 March 2026).
101. SVP Challenge. Available online: https://www.latticechallenge.org/svp-challenge/ (accessed on 1 January 2026).
102. Micciancio, D.; Regev, O. Worst-Case to Average-Case Reductions Based on Gaussian Measures. In Proceedings of the 45th Symposium on Foundations of Computer Science (FOCS 2004), Rome, Italy, 17–19 October 2004; pp. 372–381.
103. Micciancio, D. The Hardness of the Closest Vector Problem with Preprocessing. *IEEE Trans. Inf. Theory* **2001**, *47*, 1212–1215. [CrossRef]
104. Agrell, E.; Eriksson, T.; Vardy, A.; Zeger, K. Closest Point Search in Lattices. *IEEE Trans. Inf. Theory* **2002**, *48*, 2201–2214. [CrossRef]
105. Doulgerakis, E.; Laarhoven, T.; de Weger, B. Finding Closest Lattice Vectors Using Approximate Voronoi Cells. In Proceedings of the 10th International Conference on Post-Quantum Cryptography (PQCrypto 2019), Chongqing, China, 8–10 May 2019; pp. 3–22.
106. Ducas, L.; Laarhoven, T.; van Woerden, W.P.J. The Randomized Slicer for CVPP: Sharper, Faster, Smaller, Batchier. In Proceedings of the 23rd IACR International Conference on Practice and Theory of Public-Key Cryptography (PKC 2020), Edinburgh, UK, 4–7 May 2020; pp. 3–36.