Wagner's Algorithm Provably Runs in Subexponential Time for SIS$^\infty$

Léo Ducas[1,2], Lynn Engelberts[1,3](✉), and Johanna Loyer[1]

[1] Centrum Wiskunde & Informatica, Amsterdam, The Netherlands
{l.ducas,lynn.engelberts}@cwi.nl, johanna.loyer@inria.fr
[2] Leiden University, Leiden, The Netherlands
[3] QuSoft, Amsterdam, The Netherlands

Abstract. At CRYPTO 2015, Kirchner and Fouque claimed that a carefully tuned variant of the Blum-Kalai-Wasserman (BKW) algorithm (JACM 2003) should solve the Learning with Errors problem (LWE) in slightly subexponential time for modulus $q = n^{\Theta(1)}$ and narrow error distribution, when given enough LWE samples. Taking a modular view, one may regard BKW as a combination of Wagner's algorithm (CRYPTO 2002), run over the corresponding dual problem, and the Aharonov-Regev distinguisher (JACM 2005). Hence the subexponential Wagner step alone should be of interest for solving this dual problem – namely, the Short Integer Solution problem (SIS) – but this appears to be undocumented so far.

We re-interpret this Wagner step as walking backward through a chain of projected lattices, zigzagging through some auxiliary superlattices. We further randomize the bucketing step using Gaussian randomized rounding to exploit the powerful discrete Gaussian machinery. This approach avoids sample amplification and turns Wagner's algorithm into an approximate discrete Gaussian sampler for q-ary lattices.

For an SIS lattice with n equations modulo q, this algorithm runs in subexponential time $\exp(O(n/\log\log n))$ to reach a Gaussian width parameter of, say, $s = q/\mathrm{polylog}(n)$ only requiring $m = n+\omega(n/\log\log n)$ many SIS variables. For instance, this directly provides a provable algorithm for solving the Short Integer Solution problem in the infinity norm (SIS$^\infty$) for norm bounds $\beta = q/\mathrm{polylog}(n)$. This variant of SIS underlies the security of the NIST post-quantum cryptography standard ML-DSA, also known as Dilithium. Despite its subexponential complexity, Wagner's algorithm does not appear to threaten ML-DSA's concrete security.

Keywords: Wagner's algorithm · Discrete Gaussian sampling · Short Integer Solution problem (SIS) · Lattice-based cryptography · Cryptanalysis

1 Introduction

The Short Integer Solution problem (SIS) is a fundamental problem in lattice-based cryptography. It was introduced by Ajtai [Ajt96], along with an average-

Y. Tauman Kalai and S. F. Kamara (Eds.): CRYPTO 2025, LNCS 16000, pp. 353–384, 2025.
https://doi.org/10.1007/978-3-032-01855-7_12

case to worst-case reduction from SIS (in the average case) to the problem of finding a short basis in an arbitrary lattice (in the worst case). Since then, a plethora of cryptographic schemes have been based on the presumed average-case hardness of SIS. The SIS problem asks to find an integer vector of norm at most β that satisfies a set of n equations in m variables modulo q. The average-case to worst-case reductions [Ajt96, MR07] consider SIS in the Euclidean norm, with norm bound β significantly smaller than q. However, when it comes to practical cryptographic schemes, designers are often tempted to consider SIS instances outside of the asymptotic coverage of the reduction, or even to consider variants of the problem. This is the case for the new NIST standard ML-DSA (formerly known as Dilithium [DKL+18]), which considers SIS in the ℓ_∞-norm with norm bound β rather close to the modulus q.

The dual of the latter SIS variant is commonly known as the Learning with Errors problem (LWE) [Reg05] with narrow error distribution, say ternary errors. This version of LWE was proven to be solvable in slightly subexponential time by Kirchner and Fouque [KF15] using a variant of the Blum-Kalai-Wasserman algorithm (BKW) [BKW03], at least when the number of samples m is large enough. More precisely, it was claimed in [KF15] that m linear in n suffices, but a subsequent work of Herold, Kirshanova, and May [HKM18] found an issue in the proof. They resolved the issue only for $m \geq Cn \log n$ (for some constant $C > 0$), leaving open whether fewer samples would suffice. Whether this limitation on the number of samples is fundamental or an artifact of the proof remains unclear.

Our work was triggered by the folklore interpretation of BKW as a combination of Wagner's algorithm [Wag02], run over the dual lattice, and the Aharonov-Regev distinguisher [AR05]. This raises a natural question: using the tricks of Kirchner and Fouque but only in the Wagner part of the algorithm, can one (provably) solve interesting instances of SIS in subexponential time?

1.1 Contributions

We answer the above question positively, namely we prove that a variant of Wagner's algorithm solves SIS with n equations modulo $q = n^{\Theta(1)}$ in subexponential time $\exp(O(n/\log \log n))$ for ℓ_∞-norm bound $\beta = q/\mathrm{polylog}(n)$ and only requiring $m = n + \omega(n/\log \log n)$ many SIS variables. We also achieve subexponential complexity for ISIS and SIS$^\times$ for an ℓ_2-norm bound $\beta = \sqrt{n} \cdot q/\mathrm{polylog}(n)$. In fact, in each of these applications, the $\mathrm{polylog}(n)$ factor in the norm bound can be replaced, for instance, by $2^{\log(n)^c}$ for an arbitrary constant $c < 1$. All prior algorithms (including heuristic ones) for these problems had exponential asymptotic complexity $\exp(\Omega(n))$.

Beyond these applications, we also provide a significant revision of the ideas and techniques at hand. First, we propose a more abstract interpretation of Wagner for lattices using a zigzag through a chain of projected lattices and superlattices, depicted in Fig. 1. This emphasizes that the key principle of the algorithm is not tied to SIS, and its subexponential complexity really stems from the ease of constructing the appropriate chain of lattices in the case of SIS lattices.

Furthermore, we circumvent the sample-amplification technique [Lyu05], and instead follow in the footsteps of [ADRS15, ADS15, AS18, ALS21, ACKS21] by relying on discrete-Gaussian techniques to obtain provable results. Specifically, we use discrete Gaussian sampling and rounding to maintain precise control of the distribution of vectors throughout the chain of lattices.

Finally, we consider whether this approach threatens the concrete security of the NIST standard ML-DSA. As detailed in the full version [DEL25], despite its subexponential complexity and despite removing all the overhead introduced for provability, this algorithm is far less efficient against the concrete parameters of ML-DSA than standard lattice-reduction attacks.

$$\begin{tikzcd}
\Lambda'_r \arrow[d, phantom, "\cup\!\vert"] \arrow[dr] & \Lambda'_{r-1} \arrow[d, phantom, "\cup\!\vert"] \arrow[dr] & \cdots \arrow[dr] & \Lambda'_1 \arrow[d, phantom, "\cup\!\vert"] \arrow[dr] & \\
\Lambda = \Lambda_r \arrow[r, two heads] & \Lambda_{r-1} \arrow[r, two heads] & \cdots \arrow[r, two heads] & \Lambda_1 \arrow[r, two heads] & \Lambda_0 = \mathbb{Z}^{m-n}
\end{tikzcd}$$

Fig. 1. Wagner's algorithm interpreted over a chain of projected lattices Λ_i and auxiliary superlattices Λ'_i. Thick black double-arrows denote surjective orthogonal projections between lattices. The red curvy arrow denotes the path taken by the algorithm. It starts with (short) vectors in $\mathbb{Z}^{m-n}$, and follows the zigzag path until it generates (short) vectors in the lattice Λ, which, in our application, correspond to SIS solutions.

1.2 Technical Overview

The Short Integer Solution problem in the infinity norm is defined as follows.

Problem 1.1 *(*$\mathrm{SIS}^\infty_{n,m,q,\beta}$*).* Let $n, m, q \in \mathbb{N}$ with $m \geq n$, and let $\beta > 0$. Given a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, the problem $\mathrm{SIS}^\infty_{n,m,q,\beta}$ asks to find a nonzero vector $\mathbf{x} \in \mathbb{Z}^m$ satisfying $\mathbf{A}\mathbf{x} = \mathbf{0} \bmod q$ and $\|\mathbf{x}\|_\infty \leq \beta$.

This problem can be phrased as a short vector problem, as it is equivalent to finding a nonzero vector of norm at most β in the q-ary lattice $\Lambda_q^\perp(\mathbf{A}) := \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} = \mathbf{0} \bmod q\}$. It is trivial when $\beta > q$, as $(q, 0, \ldots, 0)$ is a solution, and becomes vacuously hard (i.e., typically no solution exists) when β is too small.

Consider an instance of the $\mathrm{SIS}^\infty_{n,m,q,\beta}$ problem for some given $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$. When $\mathbf{A}$ is of full rank[1] we can, without loss of generality, assume that $\mathbf{A}$ is written in systematic form, i.e., $\mathbf{A} = [\mathbf{A}' \mid \mathbf{I}_n]$ with $\mathbf{A}' \in \mathbb{Z}_q^{n\times(m-n)}$. With this writing of $\mathbf{A}$, we see that the problem of finding $\mathbf{x} \in \mathbb{Z}_q^m$ satisfying $\mathbf{A}\mathbf{x} = \mathbf{0} \bmod q$ and $\|\mathbf{x}\|_\infty \leq \beta$ is equivalent to finding $\mathbf{z} \in \mathbb{Z}_q^{m-n}$ satisfying $\|\mathbf{x}(\mathbf{z})\|_\infty \leq \beta$ where $\mathbf{x}(\mathbf{z}) := (\mathbf{z}; -\mathbf{A}'\mathbf{z})$.[2]

[1] This happens with overwhelming probability when $m-n = \omega(1)$: a uniformly random matrix in $\mathbb{Z}_q^{n\times m}$ (with $m \geq n$) is of full rank with probability at least $1 - 1/q^{m-n}$.

[2] We use the notation $(\mathbf{a}; \mathbf{b})$ for the vertical concatenation of vectors $\mathbf{a}$ and $\mathbf{b}$.

Wagner-Style Algorithms for SIS. Wagner's generalized birthday algorithm [Wag02] addresses the problem of finding elements in a list $L_0 \subseteq \mathbb{Z}_q^n$ that sum up to zero.[3] The same technique was independently used in [BKW03] to solve a dual problem.

Given an SIS instance $\mathbf{A} = [\mathbf{A}' \mid \mathbf{I}_n] \in \mathbb{Z}_q^{n\times m}$, divide the rows of the parity-check matrix $\mathbf{A}$ into r blocks of equal size n/r, as illustrated in Fig. 2 (for $b_j := n/r$). The algorithm then iteratively solves smaller SIS instances corresponding to the parity-check matrices $\mathbf{A}_i = [\mathbf{A}'_i \mid \mathbf{I}_{in/r}]$, for $i = 1$ up to r, where $\mathbf{A}'_i$ is defined by the first i blocks of rows of $\mathbf{A}'$. Specifically, the algorithm begins by filling an initial list L_0 with short vectors. Then, in iteration i, it computes, for each vector $\mathbf{x} \in L_{i-1} \subseteq \mathbb{Z}_q^{m-n+(i-1)n/r}$, the unique (modulo q) vector $\mathbf{y} \in \mathbb{Z}^{n/r}$ such that $\mathbf{A}_i(\mathbf{x};\mathbf{y}) = \mathbf{0} \bmod q$. This vector $\mathbf{y}$ is likely to be of high norm since it is uniformly distributed modulo q. The algorithm then searches for pairs of such vectors $\mathbf{x}'_1 = (\mathbf{x}_1;\mathbf{y}_1)$, $\mathbf{x}'_2 = (\mathbf{x}_2;\mathbf{y}_2)$ that satisfy $\mathbf{y}_1 = \mathbf{y}_2 \bmod q$, to then add $\mathbf{x}'_1 - \mathbf{x}'_2 = (\mathbf{x}_1 - \mathbf{x}_2;\mathbf{0})$ to the list L_i. This ensures that the difference vector $\mathbf{x}'_1 - \mathbf{x}'_2$ remains short, as $\mathbf{x}_1$ and $\mathbf{x}_2$ are short and the rest has norm zero. The final list L_r contains vectors $\mathbf{x} \in \mathbb{Z}_q^m$ that satisfy $\mathbf{A}\mathbf{x} = \mathbf{0} \bmod q$, providing potential nonzero and short solutions to the original SIS problem.

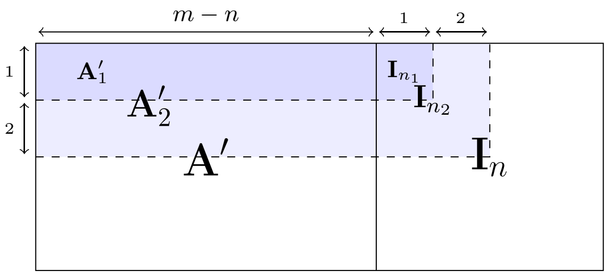

Fig. 2. Illustration of the parity-check matrices $\mathbf{A}_i := [\mathbf{A}'_i \mid \mathbf{I}_{n_i}]$, where $\mathbf{A}'_i$ is the matrix defined by the first $n_i := \sum_{j=1}^{i} {}_j$ rows of $\mathbf{A}'$. (Without the generalized lazy-modulus switching technique, the ${}_j$ are set equal to n/r.) Each iteration i of the algorithm solves an SIS instance given by parity-check matrix $\mathbf{A}_i$, until ultimately the whole matrix $\mathbf{A} = \mathbf{A}_r$ is covered.

Lazy-Modulus Switching. The aforementioned approach constructs the lists L_i by combining vectors that lie in the same *bucket*, where the bucket of $(\mathbf{x};\mathbf{y})$ for $\mathbf{x} \in L_{i-1}$ is labeled by the value of $\mathbf{y}$ modulo q. The vectors in L_i are then guaranteed to be of the form $(\mathbf{x}';\mathbf{0})$, but this is not necessary for the algorithm to succeed: a form like $(\mathbf{x}';\mathbf{y}')$ with both $\mathbf{x}'$ and $\mathbf{y}'$ being short suffices. Based on this observation, the authors of [AFFP14] consider using a smaller modulus

[3] Although Wagner originally considered the case $\mathbb{Z}_2^n$, the algorithm for $\mathbb{Z}_q^n$ follows the same principle.

$p < q$, and combine $(\mathbf{x}_1; \mathbf{y}_1)$ and $(\mathbf{x}_2; \mathbf{y}_2)$ if $\lfloor \frac{p}{q}\mathbf{y}_1 \rceil = \lfloor \frac{p}{q}\mathbf{y}_2 \rceil \bmod p$, where the operation $\lfloor \cdot \rceil$ denotes rounding each coordinate to its nearest integer modulo p. This modified condition for combining vectors may result in a nonzero 'rounding error' in the $\mathbf{y}'$-part, which is traded for a reduced number of buckets: $p^{n/r}$ instead of $q^{n/r}$.

Two concurrent works [KF15, GJMS17] generalize this *lazy-modulus switching* technique by considering different moduli $p_1, \ldots, p_r$ (not necessarily prime). In this approach, the matrix $\mathbf{A}$ is divided into r blocks of respective size $b_1, \ldots, b_r$. The rounding errors induced by iteration i are at most q/p_i, and may double in each subsequent iteration when the vectors are combined. Hence, it makes sense to use decreasing moduli p_i to balance the accumulation of rounding error. Each step requires (more than) $p_i^{b_i}$ vectors to find collisions in $\mathbb{Z}_{p_i}^{b_i}$, leading to increasing block sizes b_i. This increasing choice of b_i's is central to the subexponential complexity claim of [KF15].

A Naive Analysis. Algorithm 1 is a rephrased version of the algorithm of [KF15] without the LWE dual-distinguishing step. Let us attempt to analyze it in order to highlight the core of the issue.

Algorithm 1: Wagner-Style Algorithm for SIS

```
Input : Integers $n, m, q$;
        Full-rank matrix $\mathbf{A} = [\mathbf{A}' \mid \mathbf{I}_n] \in \mathbb{Z}_q^{n \times m}$;
        Integer parameters $N, r, (p_i)_{i=1}^r, (b_i)_{i=1}^r$ with $\sum_{i=1}^r b_i = n$
Output: List of vectors $\mathbf{x} \in \mathbb{Z}_q^m$ such that $\mathbf{Ax} = \mathbf{0} \bmod q$ and $\|\mathbf{x}\|_\infty \leq 2^r$

Initialize a list $L_0$ with $3^r N$ independent, uniform samples from $\{-1, 0, 1\}^{m-n}$
for $i = 1, \ldots, r$ do
    $L_i := \emptyset$
    Initialize empty buckets $B(\mathbf{c})$ for each $\mathbf{c} \in \mathbb{Z}_{p_i}^{b_i}$
    for $\mathbf{x} \in L_{i-1}$ do                                              // Bucketing
        Compute the unique $\mathbf{y} \in \mathbb{Z}_q^{b_i}$ satisfying $\mathbf{A}_i(\mathbf{x}; \mathbf{y}) = \mathbf{0} \bmod q$
        Compute $\mathbf{c} = \left\lfloor \frac{p_i}{q}\mathbf{y} \right\rceil \bmod p_i$
        Append $\mathbf{x}' := (\mathbf{x}; \mathbf{y})$ to $B(\mathbf{c})$
    for $\mathbf{c} \in \mathbb{Z}_{p_i}^{b_i}$ do                                 // Combining
        for each two vectors $\mathbf{x}_1', \mathbf{x}_2'$ in $B(\mathbf{c})$ do
            Append $\mathbf{x}_1' - \mathbf{x}_2'$ to the list $L_i$
            Remove $\mathbf{x}_1'$ and $\mathbf{x}_2'$ from $B(\mathbf{c})$
return $L_r$
```

List Construction. Since each vector in L_i is either paired or left alone in a bucket, it follows that $|L_i| \leq 2|L_{i+1}| + p_i^{b_i}$. Thus, by initializing a list L_0 of size $3^r N$, we obtain at least N vectors in L_r, assuming $N \geq \max_i p_i^{b_i}$.

Bucketing and Combining. At the beginning of each iteration $i \in \{1, \dots, r\}$, the algorithm initializes some empty lists $B(\mathbf{c})$, the *buckets*, each labeled by a vector (coset) $\mathbf{c} \in \mathbb{Z}_{p_i}^{b_i}$. Every iteration works in two phases. First, the 'for'-loop over the $\mathbf{x} \in L_{i-1}$ performs the *bucketing* phase: the vectors are added to a bucket according to their corresponding $\mathbf{y}$-value. In the *combining* phase, the 'for'-loop over each bucket representative $\mathbf{c} \in \mathbb{Z}_{p_i}^{b_i}$ takes differences of vectors $\mathbf{x}'_1, \mathbf{x}'_2$ belonging to the same bucket. Since $\mathbf{A}\mathbf{x}'_1 = \mathbf{0} \bmod q$ and $\mathbf{A}\mathbf{x}'_2 = \mathbf{0} \bmod q$, the difference vector $\mathbf{x}'_1 - \mathbf{x}'_2$ also satisfies $\mathbf{A}(\mathbf{x}'_1 - \mathbf{x}'_2) = \mathbf{0} \bmod q$ by linearity. Moreover, if two vectors $\mathbf{x}'_1 = (\mathbf{x}_1; \mathbf{y}_1)$, $\mathbf{x}'_2 = (\mathbf{x}_2; \mathbf{y}_2)$ belong to the same bucket $B(\mathbf{c})$, then they satisfy $\mathbf{c} = \lfloor \frac{p_i}{q} \mathbf{y}_1 \rceil = \lfloor \frac{p_i}{q} \mathbf{y}_2 \rceil \bmod p_i$. In particular, $\mathbf{y}_1$ and $\mathbf{y}_2$ are 'close' to each other, ensuring their difference is short: we have $\|\mathbf{y}_1 - \mathbf{y}_2\|_\infty \leq q/p_i$, and thus $\|\mathbf{x}'_1 - \mathbf{x}'_2\|_\infty \leq \max\{2\|\mathbf{x}_1\|_\infty, 2\|\mathbf{x}_2\|_\infty, q/p_i\}$. By induction over the r iterations, it follows that the algorithm outputs vectors $\mathbf{x} \in L_r$ satisfying

$$\|\mathbf{x}\|_\infty \leq \max_{0 \leq i \leq r} 2^{r-i} \frac{q}{p_i}, \tag{1}$$

where we define $p_0 = q$.

Time Complexity of Algorithm 1. Equation (1) shows that the choice of the number of iterations r and the moduli p_i influences the maximal norm of the vectors. These parameters also affect the algorithm's time complexity, along with the other parameters that need to be chosen. Although the algorithm parameters p_i, b_i, r and N should be integers to make sense, we consider them as real numbers for simplicity in this introductory section.

Parameter Selection. Let the target norm for the SIS problem be $\beta = \frac{q}{f}$ for some factor $f > 1$. Note that the problem is easiest when $f = 1$ (norm q) and harder when f increases (i.e., the norm q/f becomes shorter). The bound on the output norms given in Eq. (1) is minimized when both terms are balanced: $2^r = 2^{r-i} q/p_i$. Therefore we set $p_i := q/2^i$. One can generate $p_i^{b_i}$ distinct vectors modulo p_i with b_i coordinates, so to keep the number of samples comparable at each step i, we set $N = p_i^{b_i}$. This implies $b_i = \frac{\log N}{\log p_i} = \frac{\log_2 N}{\log_2(q) - i}$. By the choice of the p_i, the final vectors have a norm of at most 2^r. To ensure that it is at most $\beta = q/f$, we must choose r such that $2^r \leq \frac{q}{f}$. We set $r := \log_2(q/f) - 1$. We must also ensure that $n = n_r = \sum_{i=1}^r b_i$. Since the b_i increase with i, we can bound their sum by $n = \sum_{i=1}^r b_i \leq \int_1^{r+1} \frac{\log_2 N}{\log_2(q) - x} \mathrm{d}x = (\log_2 N) \cdot \ln \left(\frac{\log_2(q) - 1}{\log_2(q) - (r+1)} \right)$.[4] Thus,

$$n \leq \log_2(N) \cdot \ln \left(\frac{\log_2 q}{\log_2 f} \right).$$

Rewriting, we conclude that taking $\log_2(N) = \frac{n}{\ln\ln(q) - \ln\ln(f)}$ suffices. Up to rounding of the parameters (as they need to be integers), we would get the following statement.

[4] For f an increasing function, $\int_0^r f(x)\,\mathrm{d}x \leq \sum_{i=1}^r f(i) \leq \int_1^{r+1} f(x)\,\mathrm{d}x$. Also, for $A, B, a, b > 0$, $\int_a^b \frac{A}{B-x} \mathrm{d}x = A \ln \left(\frac{B-a}{B-b} \right)$.

Tentative Theorem. *Let $n, m, q \in \mathbb{N}$ and $f > 1$. There exists an algorithm that, given $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, returns a (possibly zero) vector $\mathbf{x} \in \mathbb{Z}_q^m$ such that $\mathbf{A}\mathbf{x} = \mathbf{0} \bmod q$ and $\|\mathbf{x}\|_\infty \leq \frac{q}{f}$ in time*

$$T = 2^{\frac{n}{\ln\ln(q) - \ln\ln(f)}} \cdot \mathrm{poly}(n, \log q).$$

Important Remark. There is a catch! The output list of the algorithm contains a SIS solution only if at least one of the output vectors is *nonzero*. So it remains to be proven that the vectors do not all cancel out to zero. At the first iteration, the set $\{-1, 0, 1\}^{m-n}$ is significantly larger than the number of buckets, hence differences of vectors from a same bucket are unlikely to be zero vectors. However, from the second iteration onward, specifying the vector distributions is far from straightforward.

This issue does not arise in the original works of [Wag02, BKW03], where m was assumed to be as large as the initial list size $|L_0|$, and the initial list is simply filled with standard unit vectors (i.e., vectors with one coordinate equal to 1 and zeros elsewhere). These vectors are linearly independent and hence cannot cancel each other out. In the case of solving ternary LWE with [BKW03] rather than SIS with [Wag02], this situation can be emulated using a sample-amplification technique from [Lyu05]. However, according to [HKM18], this requires at least $m = \Theta(n \log n)$ samples to start with; the argument of [KF15] that this should work for $m = \Theta(n)$ has been shown to be flawed in [HKM18].

Yet, at least heuristically, it is not clear why this approach should fail when $m = \Theta(n)$. There is enough entropy to avoid collision at the first step, and entropy should intuitively increase at a later stage: after all, the vectors are getting larger. But to adequately formalize this intuition, we shift our perspective on the algorithm.

From Parity-Check Perspective to Lattices. As mentioned before, the SIS problem is equivalent to a short vector problem which asks to find a nonzero vector of norm at most β in the lattice $\Lambda := \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} = \mathbf{0} \bmod q\}$. We consider the sequence of lattices $\Lambda_0 = \mathbb{Z}^{m-n}, \ldots, \Lambda_r = \Lambda$ with $\Lambda_i := \{\mathbf{x} \in \mathbb{Z}^{m-n+n_i} : \mathbf{A}_i\mathbf{x} = \mathbf{0} \bmod q\}$ for $i \geq 1$. Each parity-check matrix $\mathbf{A}_i$ adds a block of b_i new coordinates, making Λ_{i-1} a projection of Λ_i. It is easy to sample bounded vectors in $\Lambda_0 := \mathbb{Z}^{m-n}$; for example, one can sample in $\{-1, 0, 1\}^{m-n}$, as Algorithm 1 did. The goal is to use these initial samples to go back through the projections toward Λ (recall Fig. 1), while keeping the norms bounded.

While one can lift a vector from Λ_{i-1} to Λ_i, such a lifted vector would not be short because the lattice Λ_i we are lifting over is too sparse. Instead, what happens in Algorithm 1 is that we implicitly lift to a denser lattice $\Lambda_i' \supseteq \Lambda_i$, and then take the difference of two vectors in the same coset of the quotient Λ_i'/Λ_i to fall again in Λ_i. More explicitly, setting $\mathbf{c} = \lfloor \frac{p_i}{q}\mathbf{y} \rceil$ as in Algorithm 1, note that the vector $(\mathbf{x}, \mathbf{y} - \frac{q}{p_i}\mathbf{c})$ lives in the lattice $\Lambda_i' = \Lambda_i + (\{0\}^{m-n+n_{i-1}} \times \frac{q}{p_i}\mathbb{Z}^{b_i})$ which satisfies $|\Lambda_i'/\Lambda_i| = p_i^{b_i}$. Furthermore, the coset of that vector in the quotient Λ_i'/Λ_i is exactly determined by $\mathbf{c} \bmod p_i$.

This provides a clean and pleasant interpretation of the algorithm as walking in a commutative diagram shown in Fig. 1. It further provides the framework to deploy the full machinery of discrete Gaussian distributions over lattices: the initial samples can easily be made Gaussian over $\mathbb{Z}^{m-n}$, the lifting step as well using the randomized Babai algorithm [GPV08, EWY23], and the combination step preserves Gaussians using convolution lemmas [Pei10, MP13], as already used in [ADRS15, ADS15, ACKS21, AS18, ALS21] to solve short vector problems. There are further technicalities to control the independence between the samples, which we handle using the (conditional) similarity notion introduced in [ALS21]. This permits to control all the distributions throughout the algorithm, leading to provable conclusions. In particular, a careful choice of the algorithm's parameters guarantees, with high probability, that the final distribution is not concentrated at zero [PR06, Lemma 2.11].

1.3 Organization of the Paper

Section 2 provides the necessary background. In Sect. 3, we present our Gaussian sampler and analyze its asymptotic time complexity. In Sect. 4, we then use the Gaussian sampler to asymptotically solve several variants of SIS, carefully avoiding the 'canceling out to zero' issue. In the appendix of the full version [DEL25], we discuss the impact of the attack on the concrete security of ML-DSA.

2 Preliminaries

Notation. We write $\|\cdot\|_2$ and $\|\cdot\|_\infty$ for the Euclidean and infinity norm of a vector, respectively. For a positive integer N, we define $[N] := \{1, \ldots, N\}$. We use the notation $X = e^{\pm\delta}$ as shorthand for $X \in [e^{-\delta}, e^{\delta}]$. For $x = (x_1, \ldots, x_N)$ and $i \in [N]$, we define $x_{-i} := (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N)$, and we define $x_{-\{i,j\}}$ analogously. For events E_0, E_1, we use the convention that $\Pr[E_0 \mid E_1] = 0$ if $\Pr[E_1] = 0$.

Asymptotic Notation. Let f and g be functions that map positive integers to positive real numbers. We write $f(n) = O(g(n))$ if there exist constants $c, n_0 > 0$ such that $f(n) \le c \cdot g(n)$ for every integer $n \ge n_0$. Similarly, we write $f(n) = \Omega(g(n))$ if there exist constants $c, n_0 > 0$ such that $f(n) \ge c \cdot g(n)$ for every integer $n \ge n_0$. We say that $f(n) = \Theta(g(n))$ if both $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$. We define $\mathrm{poly}(f(n)) := f(n)^{O(1)}$ and $\mathrm{polylog}(f(n)) := \log(f(n))^{O(1)}$. We define $\tilde{O}(f(n)) := O\left(f(n) \cdot \mathrm{polylog}(f(n))\right)$ and $\tilde{\Omega}(f(n)) := \Omega\left(f(n)/\mathrm{polylog}(f(n))\right)$. We write $f(n) = o(g(n))$ if, for all constants $c > 0$, there exists $n_0 > 0$ such that $f(n) < c \cdot g(n)$ for every integer $n \ge n_0$. We write $f(n) = \omega(g(n))$ if, for all constants $c > 0$, there exists $n_0 > 0$ such that $f(n) > c \cdot g(n)$ for every integer $n \ge n_0$.

2.1 Similarity of Distributions

Inspired by [ALS21], we consider the notions of *similarity* and *conditional similarity*, which we define below, to measure the pointwise distance between two distributions. These concepts are slightly stronger than statistical distance (see Remark 2.1), and are particularly useful for handling small independencies arising from 'bucket-and-combine' type of algorithms (like each iteration of Wagner-style algorithms).

Definition 2.1 (Similar). *Let D be a probability distribution over a set $\mathcal{X}$, and let $\delta \geq 0$ be a real. A random variable $X \in \mathcal{X}$ is δ-similar to D if, for all $x \in \mathcal{X}$, it holds that*

$$\Pr_X[X = x] = e^{\pm\delta} \cdot \Pr_{Y \sim D}[Y = x].$$

Remark 2.1 (Similarity implies Closeness in Statistical Distance). The notion of δ-similarity is a stronger notion than being within statistical distance δ, where we recall that the statistical distance between two discrete random variables X, Y is defined as $\frac{1}{2}\sum_{x\in\mathcal{X}} |\Pr[X = x] - \Pr[Y = x]|$. Indeed, for all $\delta \in [0, 1]$, being δ-similar implies being within statistical distance δ, since for any such δ we have $[e^{-\delta}, e^{\delta}] \subseteq [1 - 2\delta, 1 + 2\delta]$.

The next definition is closely related to [ALS21, Definition 4.1], but we remark that we use different terminology and that we made the definition more general.

Definition 2.2 (Conditionally Similar). *Let D be a probability distribution over a set $\mathcal{X}$, and let $\delta \geq 0$ be a real. Discrete random variables $X_1, \ldots, X_N \in \mathcal{X}$ are* conditionally δ-similar to independent samples from D *if, for all $i \in [N]$ and $x \in \mathcal{X}^N$, it holds that*

$$\Pr_{X_1,\ldots,X_N}[X_i = x_i \mid X_{-i} = x_{-i}] = e^{\pm\delta} \cdot \Pr_{Y \sim D}[Y = x_i].$$

In particular, being conditionally 0-similar (i.e., $\delta = 0$) is equivalent to being independently distributed according to D.

Remark 2.2 (Conditional Similarity implies Marginal Similarity). For $N = 1$, conditional similarity coincides with Definition 2.1. More generally, for all $N \geq 1$ and $\delta \geq 0$, conditional δ-similarity implies marginal δ-similarity. Indeed, if $X_1, \ldots, X_N \in \mathcal{X}$ are conditionally δ-similar to independent samples from a distribution D on $\mathcal{X}$, then, for all $i \in [N]$ and $x \in \mathcal{X}$, we have $\Pr[X_i = x] = \sum_{y\in\mathcal{X}^{N-1}} \Pr[X_i = x|X_{-i} = y] \Pr[X_{-i} = y] = e^{\pm\delta} \cdot \Pr_{Y\sim D}[Y = x]$.

A useful property of similarity and conditional similarity is that these notions are closed under convex combinations (as already observed in [ALS21]).

Lemma 2.1. *Let D be a probability distribution over a set $\mathcal{X}$, and let $\delta \geq 0$ be a real. Let $\mathcal{E}$ be a finite or countably infinite set of mutually exclusive and collectively exhaustive events. If discrete random variables $X_1, \ldots, X_N \in \mathcal{X}$ satisfy*

$$\Pr_{X_1,\ldots,X_N}[X_i = x_i \mid X_{-i} = x_{-i} \text{ and } E] = e^{\pm\delta} \cdot \Pr_{Y \sim D}[Y = x_i]$$

for all $i \in [N]$, $x \in \mathcal{X}^N$, and $E \in \mathcal{E}$, then $X_1, \ldots, X_N$ are conditionally δ-similar to independent samples from D.

Proof. Suppose that the premise is true for random variables $X_1, \ldots, X_N$. Then, for all $i \in [N]$ and $x \in \mathcal{X}^N$, we have

$$\Pr_{X_1,\ldots,X_N}[X_i = x_i \mid X_{-i} = x_{-i}] = \sum_{E \in \mathcal{E}} \Pr_{X_1,\ldots,X_N}[X_i = x_i \text{ and } E \mid X_{-i} = x_{-i}]$$
$$= \sum_{E \in \mathcal{E}} \Pr_{X_1,\ldots,X_N}[X_i = x_i \mid E \text{ and } X_{-i} = x_{-i}] \cdot \Pr_{X_1,\ldots,X_N}[E \mid X_{-i} = x_{-i}]$$
$$= e^{\pm\delta} \cdot \Pr_{Y \sim D}[Y = x_i]$$

since $\sum_{E \in \mathcal{E}} \Pr[E \mid X_{-i} = x_{-i}] = \frac{\sum_{E \in \mathcal{E}} \Pr[E \text{ and } X_{-i} = x_{-i}]}{\Pr[X_{-i} = x_{-i}]} = 1$. □

The following property can be viewed as the data-processing inequality for conditional similarity.

Lemma 2.2. *Let D be a probability distribution over a set $\mathcal{X}$, and let $\delta \geq 0$ be a real. If discrete random variables $X_1, \ldots, X_N \in \mathcal{X}$ are conditionally δ-similar to independent samples from D, then*

$$\Pr_{X_1,\ldots,X_N}[f(X_i) = 1] = e^{\pm\delta} \cdot \Pr_{Y \sim D}[f(Y) = 1]$$

for all $i \in [N]$ and all functions $f \colon \mathcal{X} \to \{0, 1\}$.

Proof. Suppose that $X_1, \ldots, X_N$ are conditionally δ-similar to independent samples from D. Let $i \in [N]$ and let $f \colon \mathcal{X} \to \{0, 1\}$ be an arbitrary function. Define $f^{-1}(1) := \{x \in \mathcal{X} \colon f(x) = 1\}$. Then

$$\begin{aligned}
\Pr_{X_1,\ldots,X_N}[f(X_i) = 1] &= \sum_{x \in \mathcal{X}} \Pr_{X_1,\ldots,X_N}[f(X_i) = 1 \text{ and } X_i = x] \\
&= \sum_{x \in f^{-1}(1)} \Pr_{X_1,\ldots,X_N}[X_i = x] \\
&= e^{\pm\delta} \cdot \sum_{x \in f^{-1}(1)} \Pr_{Y \sim D}[Y = x] \\
&= e^{\pm\delta} \cdot \Pr_{Y \sim D}[f(Y) = 1].
\end{aligned}$$

□

2.2 Lattices

Given k linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_k \in \mathbb{R}^n$, let $\mathbf{B} \in \mathbb{R}^{n \times k}$ be the matrix whose columns are the $\mathbf{b}_i$. The *lattice* associated to $\mathbf{B}$ is the set $\mathcal{L}(\mathbf{B}) := \mathbf{B}\mathbb{Z}^k = \left\{\sum_{i=1}^{k} z_i \mathbf{b}_i \colon z_i \in \mathbb{Z}\right\} \subseteq \mathbb{R}^n$ of all integer linear combinations of these vectors. We say that $\mathbf{B}$ is a *basis* for a lattice $\mathcal{L}$ if $\mathcal{L} = \mathcal{L}(\mathbf{B})$. We say that $\mathcal{L}$ has *rank* k and *dimension* n. If $n = k$, then $\mathcal{L}$ is said to be of *full rank*. We define $\operatorname{span}_{\mathbb{R}}(\mathcal{L}(\mathbf{B})) = \operatorname{span}_{\mathbb{R}}(\mathbf{B}) = \{\mathbf{B}\mathbf{x} \colon \mathbf{x} \in \mathbb{R}^n\}$. We define the *dual* of $\mathcal{L}$ by $\mathcal{L}^* := \{\mathbf{y} \in \operatorname{span}_{\mathbb{R}}(\mathcal{L}) \colon \forall \mathbf{x} \in \mathcal{L}, \langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z}\}$, which is a lattice.

First Successive Minimum. Given a lattice $\mathcal{L}$, we write $\lambda_1(\mathcal{L}) := \inf\{\|\mathbf{x}\|_2 : \mathbf{x} \in \mathcal{L} \setminus \{\mathbf{0}\}\}$ for the Euclidean norm of a shortest lattice vector. We define $\lambda_1^\infty(\mathcal{L})$ similarly for the infinity norm.

Projections and Primitive Sublattices. For $S \subseteq \mathbb{R}^n$, we write π_S for the projection onto $\mathrm{span}_{\mathbb{R}}(S)$ and $\pi_S^\perp$ for the projection orthogonal to $\mathrm{span}_{\mathbb{R}}(S)$. We say that a sublattice $\mathcal{S}$ of a lattice $\mathcal{L} \subseteq \mathbb{R}^n$ is *primitive* if $\mathcal{S} = \mathrm{span}_{\mathbb{R}}(\mathcal{S}) \cap \mathcal{L}$. It implies that there exists a sublattice $\mathcal{C} \subseteq \mathcal{L}$ such that $\mathcal{S} \oplus \mathcal{C} = \mathcal{L}$ (i.e., $\mathcal{S} + \mathcal{C} = \mathcal{L}$ and $\mathcal{S} \cap \mathcal{C} = \{\mathbf{0}\}$). We then say that $\mathcal{C}$ is a *complement* to $\mathcal{S}$.

Relevant q-ary Lattices. We say that a lattice $\mathcal{L} \subseteq \mathbb{R}^n$ is *q-ary* if $q\mathbb{Z}^n \subseteq \mathcal{L} \subseteq \mathbb{Z}^n$. The two relevant types of q-ary lattices considered in this work are as follows: for $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, we define the full-rank q-ary lattices

$$\Lambda_q^\perp(\mathbf{A}) := \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} = 0 \bmod q\} \subseteq \mathbb{Z}^m,$$
$$\Lambda_q(\mathbf{A}) := \{\mathbf{y} \in \mathbb{Z}^m : \exists \mathbf{s} \in \mathbb{Z}_q^n, \mathbf{y} = \mathbf{A}^\top \mathbf{s} \bmod q\} = \mathbf{A}^\top \mathbb{Z}^n + q\mathbb{Z}^m \subseteq \mathbb{Z}^m.$$

They are duals up to appropriate scaling: namely, $\Lambda_q(\mathbf{A}) = q \cdot (\Lambda_q^\perp(\mathbf{A}))^*$.

If $m \geq n$ and $\mathbf{A}$ is of full rank, we can assume without loss of generality that $\mathbf{A} = [\mathbf{A}' \mid \mathbf{I}_n]$ for some $\mathbf{A}' \in \mathbb{Z}_q^{n \times (m-n)}$. Then a basis of $\Lambda_q^\perp(\mathbf{A})$ is given by

$$\begin{pmatrix} 0 & \mathbf{I}_{m-n} \\ q\mathbf{I}_n & -\mathbf{A}' \end{pmatrix}.$$

2.3 Discrete Gaussian Distribution and Smoothness

In the following, when the subscripts s and $\mathbf{c}$ are omitted, they are respectively taken to be 1 and $\mathbf{0}$.

For any real $s > 0$ and $\mathbf{c} \in \mathbb{R}^n$, we define the Gaussian function on $\mathbb{R}^n$ centered at $\mathbf{c}$ with parameter s by

$$\forall \mathbf{x} \in \mathbb{R}^n,\ \rho_{s,\mathbf{c}}(\mathbf{x}) := \exp(-\pi\|(\mathbf{x} - \mathbf{c})/s\|_2^2).$$

For any countable set A, we define $\rho_{s,\mathbf{c}}(A) = \sum_{\mathbf{x} \in A} \rho_{s,\mathbf{c}}(\mathbf{x})$. Note that $\rho_{s,\mathbf{c}}(\mathbf{x}) = \rho_s(\mathbf{x} - \mathbf{c})$, and thus $\rho_{s,\mathbf{c}}(A) = \rho_s(A - \mathbf{c})$.

For any real $s > 0$, $\mathbf{c} \in \mathbb{R}^n$, and full-rank lattice $\mathcal{L} \subseteq \mathbb{R}^n$, we define the discrete Gaussian distribution over $\mathcal{L}$ centered at $\mathbf{c}$ with parameter s by

$$\forall \mathbf{x} \in \mathcal{L},\ D_{\mathcal{L},s,\mathbf{c}}(\mathbf{x}) := \frac{\rho_{s,\mathbf{c}}(\mathbf{x})}{\rho_{s,\mathbf{c}}(\mathcal{L})} = \frac{\rho_s(\mathbf{x} - \mathbf{c})}{\rho_s(\mathcal{L} - \mathbf{c})}$$

and it is 0 for $\mathbf{x} \notin \mathcal{L}$.

Similarly, for any $\mathbf{t} \in \mathbb{R}^n$, we define $D_{\mathcal{L}-\mathbf{t},s,\mathbf{c}}(\mathbf{y}) := \frac{\rho_{s,\mathbf{c}}(\mathbf{y})}{\rho_{s,\mathbf{c}}(\mathcal{L}-\mathbf{t})}$ for $\mathbf{y} \in \mathcal{L} - \mathbf{t}$. (Note that $D_{\mathcal{L},s,\mathbf{c}} \equiv \mathbf{c} + D_{\mathcal{L}-\mathbf{c},s}$.)

Infinity Norm of Discrete Gaussian Samples. We can tail-bound the infinity norm of a discrete Gaussian sample using the following lemma, where we define $\mathcal{B}_n^\infty := \{\mathbf{x} \in \mathbb{R}^n \colon \|\mathbf{x}\|_\infty \leq 1\}$.

Lemma 2.3 ([Ban95, Lemma 2.10]). *For any full-rank lattice $\mathcal{L} \subseteq \mathbb{R}^n$ and real $R > 0$,*

$$\frac{\rho(\mathcal{L} \setminus R \cdot \mathcal{B}_n^\infty)}{\rho(\mathcal{L})} < 2n \cdot e^{-\pi R^2}.$$

Smoothness. The work of [MR07] introduced a lattice quantity known as the smoothing parameter. More precisely, for any full-rank lattice $\mathcal{L} \subseteq \mathbb{R}^n$ and real $\varepsilon > 0$, we define the *smoothing parameter* $\eta_\varepsilon(\mathcal{L})$ as the smallest real $s > 0$ such that $\rho_{1/s}(\mathcal{L}^* \setminus \{\mathbf{0}\}) \leq \varepsilon$.

Intuitively, it gives a lower bound on s such that $D_{\mathcal{L},s}$ 'behaves like' a continuous Gaussian distribution, in a specific mathematical sense. The following lemma justifies the name of the smoothing parameter.

Lemma 2.4 (Implicit in [MR07, Lemma 4.4]). *For any full-rank lattice $\mathcal{L} \subseteq \mathbb{R}^n$, real $\varepsilon \in (0,1)$, real $s \geq \eta_\varepsilon(\mathcal{L})$, and $\mathbf{c} \in \mathbb{R}^n$,*

$$\frac{\rho_{s,\mathbf{c}}(\mathcal{L})}{\rho_s(\mathcal{L})} \in \left[\frac{1-\varepsilon}{1+\varepsilon}, 1\right].$$

For s slightly above smoothing, we can upper bound the probability of the most likely outcome of the discrete Gaussian distribution.

Lemma 2.5 (Min-entropy [PR06, Lemma 2.11]). *For any full-rank lattice $\mathcal{L} \subseteq \mathbb{R}^n$, reals $\varepsilon > 0$ and $s \geq 2\eta_\varepsilon(\mathcal{L})$, center $\mathbf{c} \in \mathbb{R}^n$, and vector $\mathbf{x} \in \mathbb{R}^n$,*

$$\Pr_{X \sim D_{\mathcal{L},s,\mathbf{c}}}[X = \mathbf{x}] \leq \frac{1+\varepsilon}{1-\varepsilon} \cdot 2^{-n}.$$

In particular, for $\mathbf{x} = \mathbf{0}$, Lemma 2.5 gives an upper bound on the probability that a discrete Gaussian sample is zero (if the standard deviation s is large enough).

Sampling and Combining. In our algorithms, we sample from (scalings of) $\mathbb{Z}^n$ using the exact Gaussian sampler from [BLP+13].

Lemma 2.6 (Implicit in [BLP+13, Lemma 2.3]). *There is a randomized algorithm that, given a real $s \geq \sqrt{\ln(2n+4)/\pi}$ and $\mathbf{c} \in \mathbb{R}^n$, returns a sample from $D_{\mathbb{Z}^n,s,\mathbf{c}}$ in expected time* $\mathrm{poly}(n, \log s, \log \|\mathbf{c}\|_\infty)$.

Our analysis uses a variant of the convolution lemma [MP13, Theorem 3.3] (see also [Pei10, Theorem 3.1]) that bounds how similar the difference of two discrete Gaussians is to a discrete Gaussian. It is a slightly tighter result than [ALS21, Lemma 2.14].

Lemma 2.7 (Explicit Variant of Convolution Lemma). *Let $\mathcal{L} \subseteq \mathbb{R}^n$ be a full-rank lattice and let $s \geq \sqrt{2}\eta_\varepsilon(\mathcal{L})$ for some real $\varepsilon > 0$. For $i = 1, 2$, let $\mathcal{L} + \mathbf{c}_i$ be an arbitrary coset of $\mathcal{L}$ and Y_i an independent sample from $D_{\mathcal{L}+\mathbf{c}_i,s}$. Then the distribution of $Y_1 - Y_2$ satisfies*

$$\forall\, \mathbf{y} \in \mathcal{L} + \mathbf{c}_1 - \mathbf{c}_2,\ \Pr[Y_1 - Y_2 = \mathbf{y}] \in \left[\frac{1-\varepsilon}{1+\varepsilon}, \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot D_{\mathcal{L}+\mathbf{c}_1-\mathbf{c}_2,\sqrt{2}s}(\mathbf{y}).$$

It follows, for instance, that $Y_1 - Y_2$ is 3ε-similar to $D_{\mathcal{L}+\mathbf{c}_1-\mathbf{c}_2,\sqrt{2}s}$ when $\varepsilon \leq \frac{1}{2}$.

Proof. Let $D_1 := D_{\mathcal{L}+\mathbf{c}_1,s}$ and $D_2 := D_{\mathcal{L}+\mathbf{c}_2,s}$. For $Y_1 \sim D_1$ and $Y_2 \sim D_2$, the distribution of $Y_1 - Y_2$ has support $\mathcal{L} + \mathbf{c}_1 - \mathbf{c}_2$. For all $\mathbf{x} \in \mathcal{L} + \mathbf{c}_1 - \mathbf{c}_2$, we have

$$\begin{aligned}
\Pr_{\substack{Y_1 \sim D_1 \\ Y_2 \sim D_2}}[Y_1 - Y_2 = \mathbf{x}] &= \sum_{\mathbf{y}_1 \in \mathcal{L}+\mathbf{c}_1} \Pr_{\substack{Y_1 \sim D_1 \\ Y_2 \sim D_2}}[Y_1 = \mathbf{y}_1 \text{ and } Y_1 - Y_2 = \mathbf{x}] \\
&= \sum_{\mathbf{y}_1 \in \mathcal{L}+\mathbf{c}_1} \Pr_{Y_1 \sim D_1}[Y_1 = \mathbf{y}_1] \cdot \Pr_{\substack{Y_1 \sim D_1 \\ Y_2 \sim D_2}}[Y_1 - Y_2 = \mathbf{x} \mid Y_1 = \mathbf{y}_1] \\
&= \sum_{\mathbf{y}_1 \in \mathcal{L}+\mathbf{c}_1} \frac{\rho_s(\mathbf{y}_1)}{\rho_s(\mathcal{L}+\mathbf{c}_1)} \cdot \frac{\rho_s(\mathbf{y}_1 - \mathbf{x})}{\rho_s(\mathcal{L}+\mathbf{c}_2)} \qquad \text{(def. of } D_1, D_2) \\
&= \rho_{\sqrt{2}s}(\mathbf{x}) \cdot \sum_{\mathbf{y}_1 \in \mathcal{L}+\mathbf{c}_1} \frac{\rho_{s/\sqrt{2}}(\mathbf{y}_1 - \mathbf{x}/2)}{\rho_s(\mathcal{L}+\mathbf{c}_1) \cdot \rho_s(\mathcal{L}+\mathbf{c}_2)} \qquad (2) \\
&= \rho_{\sqrt{2}s}(\mathbf{x}) \cdot \frac{\rho_{s/\sqrt{2}}(\mathcal{L} + \mathbf{c}_1 - \mathbf{x}/2)}{\rho_s(\mathcal{L}+\mathbf{c}_1) \cdot \rho_s(\mathcal{L}+\mathbf{c}_2)}
\end{aligned}$$

where Eq. (2) holds since $\rho_s(\mathbf{v}_1) \cdot \rho_s(\mathbf{v}_1 - \mathbf{v}_2) = \rho_s(\mathbf{v}_2/\sqrt{2}) \cdot \rho_s(\sqrt{2}\mathbf{v}_1 - \mathbf{v}_2/\sqrt{2}) = \rho_{\sqrt{2}s}(\mathbf{v}_2) \cdot \rho_{s/\sqrt{2}}(\mathbf{v}_1 - \mathbf{v}_2/2)$ for all $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^n$. Since $s \geq \sqrt{2}\eta_\varepsilon(\mathcal{L})$, $\rho_{s/\sqrt{2}}(\mathcal{L} + \mathbf{c}_1 - \mathbf{x}/2) \in \left[\frac{1-\varepsilon}{1+\varepsilon}, 1\right] \cdot \rho_{s/\sqrt{2}}(\mathcal{L})$ by Lemma 2.4. Hence,

$$\Pr_{\substack{Y_1 \sim D_1 \\ Y_2 \sim D_2}}[Y_1 - Y_2 = \mathbf{x}] \in \left[\frac{1-\varepsilon}{1+\varepsilon}, 1\right] \cdot \rho_{\sqrt{2}s}(\mathbf{x}) \cdot \frac{\rho_{s/\sqrt{2}}(\mathcal{L})}{\rho_s(\mathcal{L}+\mathbf{c}_1) \cdot \rho_s(\mathcal{L}+\mathbf{c}_2)}.$$

Summing both sides implies $1 \in \left[\frac{1-\varepsilon}{1+\varepsilon}, 1\right] \cdot \rho_{\sqrt{2}s}(\mathcal{L} + \mathbf{c}_1 - \mathbf{c}_2) \cdot \frac{\rho_{s/\sqrt{2}}(\mathcal{L})}{\rho_s(\mathcal{L}+\mathbf{c}_1)\cdot\rho_s(\mathcal{L}+\mathbf{c}_2)}$, i.e., $\frac{\rho_{s/\sqrt{2}}(\mathcal{L})}{\rho_s(\mathcal{L}+\mathbf{c}_1)\cdot\rho_s(\mathcal{L}+\mathbf{c}_2)} \in \left[1, \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot \frac{1}{\rho_{\sqrt{2}s}(\mathcal{L}+\mathbf{c}_1-\mathbf{c}_2)}$. It follows that

$$\Pr_{\substack{Y_1 \sim D_1 \\ Y_2 \sim D_2}}[Y_1 - Y_2 = \mathbf{x}] \in \left[\frac{1-\varepsilon}{1+\varepsilon}, \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot \frac{\rho_{\sqrt{2}s}(\mathbf{x})}{\rho_{\sqrt{2}s}(\mathcal{L} + \mathbf{c}_1 - \mathbf{c}_2)}$$

as we wanted to show. □

2.4 Bounds on Smoothing Parameters of Relevant Lattices

The following lemma gives a bound on the smoothing parameter of $\mathbb{Z}^n$. (Note that it can also be viewed as a special case of Lemma 2.9 below.)

Lemma 2.8 (Special Case of [MR07, Lemma 3.3]). *For any real $\varepsilon > 0$,*

$$\eta_\varepsilon(\mathbb{Z}^n) \le \sqrt{\frac{\ln(2n(1+1/\varepsilon))}{\pi}}.$$

The next lemma gives an upper bound on $\eta_\varepsilon(\mathcal{L})$ in terms of $\lambda_1^\infty(\mathcal{L}^*)$. It will be used to obtain an upper bound on the smoothing parameter of the lattices Λ_i that we consider.

Lemma 2.9 (Part of [Pei08, Lemma 3.5]). *For any full-rank lattice $\mathcal{L} \subseteq \mathbb{R}^n$ and real $\varepsilon > 0$,*

$$\lambda_1^\infty(\mathcal{L}^*) \cdot \eta_\varepsilon(\mathcal{L}) \le \sqrt{\frac{\ln(2n(1+1/\varepsilon))}{\pi}}.$$

The lattices Λ_i considered in this work are of the form $\Lambda_i = \Lambda_q^\perp(\mathbf{A}_i)$ for some matrices $\mathbf{A}_i$, and we recall that their dual lattice is of the form $\frac{1}{q}\Lambda_q(\mathbf{A}_i)$. We obtain the following lower bound on $\lambda_1^\infty(\Lambda_q(\mathbf{A}))$ for random matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$.

Lemma 2.10 (Variant of [GPV08, Lemma 5.3]). *Let n, m, q be positive integers with q prime and $q^{1-n/m} \ge 6$. For a uniformly random $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$,*

$$\lambda_1^\infty(\Lambda_q(\mathbf{A})) > \frac{q^{1-n/m} \cdot 2^{-n/m}}{3}$$

except with probability $< 2^{-n}$.

In particular, if $m \ge n$, then the right-hand side is lower bounded by $\frac{q^{1-n/m}}{6}$.

Proof. For some positive real B to be determined, let $S := \{\mathbf{y} \in \mathbb{Z}^m : \|\mathbf{y}\|_\infty \le B\}$, and note that $|S| = (2B+1)^m$. For all $\mathbf{s} \in \mathbb{Z}_q^n \setminus \{\mathbf{0}\}$, $\Pr[\mathbf{A}^\top \mathbf{s} \bmod q \in S] = |S| \cdot q^{-m} = (2B+1)^m \cdot q^{-m}$. Taking the union bound over all $\mathbf{s} \in \mathbb{Z}_q^n \setminus \{\mathbf{0}\}$ gives

$$\begin{aligned}\Pr[\lambda_1^\infty(\Lambda_q(\mathbf{A})) \le B] &= \Pr[\exists \mathbf{s} \in \mathbb{Z}_q^n \setminus \{\mathbf{0}\} \text{ such that } \mathbf{A}^\top \mathbf{s} \bmod q \in S] \\ &\le |\mathbb{Z}_q^n \setminus \{\mathbf{0}\}| \cdot (2B+1)^m \cdot q^{-m} \\ &< (2B+1)^m \cdot q^{n-m}.\end{aligned}$$

Let $B := \frac{1}{3} q^{1-n/m} \cdot 2^{-n/m}$, and observe that $q^{1-n/m} \ge 6$ implies $q^{1-n/m} \cdot 2^{-n/m} \ge 3$ for all $m \ge n$. It follows that $B \ge 1$ and thus $\Pr[\lambda_1^\infty(\Lambda_q(\mathbf{A})) \le B] < (3B)^m \cdot q^{n-m} = 2^{-n}$ as desired. The last part is immediate. □

Lemma 2.11 (Smoothing Parameter of $\Lambda_q^\perp(\mathbf{A})$). *Let n, m, q be positive integers with q prime, $m \ge n$, and $q^{1-n/m} \ge 6$. Let $\varepsilon \le \frac{1}{4m}$ be a positive real. For a uniformly random $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$,*

$$\eta_\varepsilon(\Lambda_q^\perp(\mathbf{A})) < \sqrt{\frac{72\ln(1/\varepsilon)}{\pi}} \cdot q^{n/m}$$

except with probability $< 2^{-n}$.

Proof. Since $m \geq n$, for a uniformly random $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, Lemma 2.10 implies $\lambda_1^\infty(\Lambda_q(\mathbf{A})) > \frac{1}{6} q^{1-n/m}$, except with probability $< 2^{-n}$. Since the dual of $\mathcal{L} := \Lambda_q^\perp(\mathbf{A})$ is $\mathcal{L}^* = \frac{1}{q}\Lambda_q(\mathbf{A})$, we obtain $\lambda_1^\infty(\mathcal{L}^*) > \frac{1}{6} q^{-n/m}$. Furthermore, by Lemma 2.9 (recall that $\mathcal{L}$ is full-rank and has dimension m), we have

$$\lambda_1^\infty(\mathcal{L}^*) \cdot \eta_\varepsilon(\mathcal{L}) \leq \sqrt{\frac{\ln(2m(1+1/\varepsilon))}{\pi}}$$

which is $\leq \sqrt{\frac{2\ln(1/\varepsilon)}{\pi}}$ for $\varepsilon \leq \frac{1}{4m}$. The statement follows. □

3 Wagner-Style Gaussian Sampler

In Sect. 1.2, we presented a warm-up version of the Wagner-style algorithm, which returns N short vectors in $\Lambda_q^\perp(\mathbf{A})$. However, these vectors are possibly all equal to $\mathbf{0}$. We now show that a variant of that algorithm, using discrete Gaussians, allows us to avoid that issue. In particular, we present a Wagner-style algorithm for sampling N vectors from a distribution that is essentially $D_{\Lambda_q^\perp(\mathbf{A}),s}$ when s is sufficiently large. Such samples can be shown to be short and nonzero with high probability. Specifically, we present an algorithm for sampling from $D_{\Lambda_q^\perp(\mathbf{A}),s}$ in expected subexponential time for $m = n + \omega(n/\log\log n)$, $q = n^{\Theta(1)}$, and $s = q/f$ for some $f = \omega(1)$.

Recall that, for some $r \in \mathbb{N}$ and $b_1, \ldots, b_r$ such that $n = \sum_{i=1}^r b_i$, we define the q-ary lattices

$$\Lambda_0 = \mathbb{Z}^{m-n} \quad \text{and} \quad \Lambda_i = \Lambda_q^\perp(\mathbf{A}_i) = \{\mathbf{x} \in \mathbb{Z}^{m-n+n_i} \colon \mathbf{A}_i\mathbf{x} = \mathbf{0} \bmod q\} \qquad (3)$$

for $i = 1, \ldots, r$, where $\mathbf{A}_i \in \mathbb{Z}_q^{n_i \times (m-n+n_i)}$ is the matrix corresponding to the first $n_i := \sum_{j=1}^i b_j$ SIS equations (recall Fig. 2).

Our approach to sample N vectors from $D_{\Lambda_q^\perp(\mathbf{A}),s}$ for a given parameter s is to start from many vectors sampled from D_{Λ_0,s_0}, where s_0 is such that $s = \sqrt{2^r} s_0$. Then, we iteratively (for $i \in \{1, \ldots, r\}$) transform a list of vectors that are conditionally similar to independent samples from $D_{\Lambda_{i-1},\sqrt{2^{i-1}}s_0}$ into a list of samples that are conditionally similar to independent samples from $D_{\Lambda_i,\sqrt{2^i}s_0}$. Then, after the last iteration, the list contains samples that are conditionally similar to independent samples from $D_{\Lambda_q^\perp(\mathbf{A}),s}$, as desired. (Using Lemma 2.5, we can then bound the probability that one such sample is nonzero.)

As explained in Sect. 1.2, mapping vectors in Λ_{i-1} to vectors in Λ_i will be done by first lifting the vectors in Λ_{i-1} to vectors in a suitable superlattice $\Lambda_i' \supseteq \Lambda_i$, and then combining them into vectors in Λ_i. Specifically, for some $p_i \in \mathbb{N}$, the lattices Λ_i' are defined by $\Lambda_i' = \mathcal{L}(\mathbf{B}_i')$ for

$$\mathbf{B}_i' := \begin{pmatrix} \mathbf{0} & \mathbf{I}_{m-n} \\ \mathbf{D}_i & -\mathbf{A}_i' \end{pmatrix} \text{ with } \mathbf{D}_i := \begin{pmatrix} \mathbf{0} & q\mathbf{I}_{n_{i-1}} \\ \frac{q}{p_i}\mathbf{I}_{b_i} & \mathbf{0} \end{pmatrix} \qquad (4)$$

The first b_i columns of $\mathbf{B}'_i$ generate (an embedding into $\mathbb{R}^{m-n+n_i}$ of) $\frac{q}{p_i}\mathbb{Z}^{b_i}$, which is a primitive sublattice of Λ'_i that we denote by $\mathcal{S}$. Consider the projected lattice $\mathcal{P} = \pi^{\perp}_{\mathcal{S}}(\Lambda'_i)$, and note that it is (an embedding of) Λ_{i-1}. Thus, we can consider ways to lift from Λ_{i-1} to Λ'_i; in Sect. 3.1, we consider a randomized way of lifting that preserves discrete Gaussian distributions.

From there, we would like to produce samples in Λ_i, rather than Λ'_i. A natural approach is to bucket our samples according to their cosets in the quotient Λ'_i/Λ_i, and then take differences within those buckets. Using standard analysis of convolutions of discrete Gaussians, we show in Sect. 3.2 that these differences are still essentially discrete Gaussian, yet with a width parameter increased by a factor of $\sqrt{2}$.

In Sect. 3.3, we then lay out the resulting Gaussian variant of Wagner's algorithm, and demonstrate its correctness and time complexity, under certain smoothing constraints on the parameters. Finally, in Sect. 3.4 we provide a choice of parameters that satisfy these constraints and allow for a subexponential-time algorithm for sampling from $D_{\Lambda_q^{\perp}(\mathbf{A}),s}$.

3.1 Discrete-Gaussian Lifting

Algorithm 2 below lifts vectors from Λ_{i-1} to vectors in Λ'_i. It revisits the GPV sampling algorithm [GPV08] with a reinterpretation of the induction: rather than reducing the problem in dimension n to two instances in dimensions 1 and $n-1$, we consider arbitrary splits in dimensions n' and $n-n'$. Algorithm 2 can be viewed as a special case of [EWY23, Alg. 2], and our Lemma 3.1 is a variant of [EWY23, Theorem 1], where we analyze the conditional similarity of the output with respect to the discrete Gaussian (instead of merely looking at the statistical distance).

In particular, Lemma 3.1 (with $\delta = 0$ and $N = 1$) shows that Algorithm 2 turns a sample from $D_{\mathcal{P},s}$ into a sample that is 3ε-similar to (and thus within statistical distance 3ε from) $D_{\mathcal{L},s}$, whenever $s \geq \eta_{\varepsilon}(\mathcal{S})$ for $0 < \varepsilon \leq \frac{1}{2}$.

Lemma 3.1 (Complexity and Distribution of DGLift**).** *Let $\mathcal{L} \subseteq \mathbb{R}^n$ be a lattice and let $\mathcal{P} = \pi^{\perp}_{\mathcal{S}}(\mathcal{L})$ for a primitive sublattice $\mathcal{S} \subseteq \mathcal{L}$. For a real $s > 0$, let $\mathcal{A}$ be a randomized algorithm that, given $\mathbf{c} \in \operatorname{span}(\mathcal{S})$, returns a sample from $D_{\mathcal{S},s,\mathbf{c}}$. Then* $\mathrm{DGLift}(\mathcal{P}, \mathcal{L}, s, \cdot)$ *(Algorithm 2) is a randomized algorithm that, given a vector $\mathbf{x} \in \mathcal{P}$, outputs a vector $\mathbf{x}'$ in $\mathcal{L}$. It uses one query to $\mathcal{A}$, and all other operations run in polynomial time.*

Moreover, for any reals $\delta \geq 0$ and $0 < \varepsilon \leq \frac{1}{2}$ satisfying $s \geq \eta_{\varepsilon}(\mathcal{S})$, if $X_1, \ldots, X_N \in \mathcal{P}$ are conditionally δ-similar to independent samples from $D_{\mathcal{P},s}$, then the distribution of $X'_1, \ldots, X'_N$ for $X'_i :=$ $\mathrm{DGLift}(\mathcal{P}, \mathcal{L}, s, X_i)$ *is conditionally $(\delta + 3\varepsilon)$-similar to independent samples from $D_{\mathcal{L},s}$.*

We will invoke the above lemma with $\mathcal{S} = \frac{q}{p_i}\mathbb{Z}^{b_i}$, hence an exact polynomial-time sampler is available whenever $s \geq \frac{q}{p_i}\sqrt{\ln(2b_i+4)/\pi}$ by Lemma 2.6.

Algorithm 2: DGLift$(\mathcal{P}, \mathcal{L}, s, \mathbf{x})$

```
Input  : Lattices $\mathcal{P}, \mathcal{L}$, where $\mathcal{P} = \pi_{\mathcal{S}}^{\perp}(\mathcal{L})$ for a primitive sublattice
         $\mathcal{S} \subseteq \mathcal{L}$;
         Real $s > 0$;
         Vector $\mathbf{x} \in \mathcal{P}$
Output: Vector $\mathbf{x}' \in \mathcal{L}$ such that $\pi_{\mathcal{S}}^{\perp}(\mathbf{x}') = \mathbf{x}$

Let $\mathcal{C}$ be a complement to $\mathcal{S}$
Compute the unique $\mathbf{y} \in \mathcal{C}$ such that $\pi_{\mathcal{S}}^{\perp}(\mathbf{y}) = \mathbf{x}$      // Lifting
Sample $\mathbf{z} \sim D_{\mathcal{S},s,\mathbf{x}-\mathbf{y}}$                                  // Sampling
return $\mathbf{x}' := \mathbf{z} + \mathbf{y}$
```

Proof. Consider Algorithm 2, where we use algorithm $\mathcal{A}$ for the sampling step, and let $\mathcal{C}$ be the complement of $\mathcal{S}$ that it considers. Note that $\pi_{\mathcal{S}}^{\perp}$ induces a bijection from $\mathcal{C}$ to $\mathcal{P}$, and that both directions can be computed in polynomial time. The claim on time and query complexity of Algorithm 2 is thus immediate.

For correctness, we remark that any output vector $\mathbf{x}' = \mathbf{z} + \mathbf{y}$ belongs to $\mathcal{L}$, since $\mathbf{z} \in \mathcal{S} \subseteq \mathcal{L}$ and $\mathbf{y} \in \mathcal{C} \subseteq \mathcal{L}$. Furthermore, $\mathbf{x}' = \pi_{\mathcal{S}}(\mathbf{x}') + \pi_{\mathcal{S}}^{\perp}(\mathbf{x}')$ with $\pi_{\mathcal{S}}(\mathbf{x}') = \mathbf{z} + \pi_{\mathcal{S}}(\mathbf{y})$ and $\pi_{\mathcal{S}}^{\perp}(\mathbf{x}') = \pi_{\mathcal{S}}^{\perp}(\mathbf{y}) = \mathbf{x}$, so the output is as desired.

For the remainder of the proof, let $f(\mathbf{x}) := \text{DGLift}(\mathcal{P}, \mathcal{L}, s, \mathbf{x})$. We remark that $\mathcal{S} \oplus \mathcal{C} = \mathcal{L}$ implies that any $\mathbf{v} \in \mathcal{L}$ can be uniquely written as $\mathbf{v} = \mathbf{v}_{\mathcal{S}} + \mathbf{v}_{\mathcal{C}}$ for $\mathbf{v}_{\mathcal{S}} \in \mathcal{S}$ and $\mathbf{v}_{\mathcal{C}} \in \mathcal{C}$ (and we define $\mathbf{v}_{\mathcal{S}}, \mathbf{v}_{\mathcal{C}}$ as such).

We first observe that for all $\mathbf{x} \in \mathcal{P}$ and $\mathbf{x}' \in \mathcal{L}$, the probability that DGLift on input $\mathbf{x}$ outputs $\mathbf{x}'$ is

$$\begin{aligned}\Pr[f(\mathbf{x}) = \mathbf{x}'] &= \Pr_{Z \sim D_{\mathcal{S},s,\mathbf{x}-\mathbf{y}(\mathbf{x})}}[Z = \mathbf{x}' - \mathbf{y}(\mathbf{x})] \\ &= \begin{cases} \frac{\rho_s(\pi_{\mathcal{S}}(\mathbf{x}'))}{\rho_s(\mathcal{S}+\pi_{\mathcal{S}}(\mathbf{x}'_{\mathcal{C}}))} & \text{if } \mathbf{x} = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}') \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{5}$$

where $\mathbf{y}(\mathbf{x})$ denotes the unique $\mathbf{y} \in \mathcal{C}$ such that $\pi_{\mathcal{S}}^{\perp}(\mathbf{y}) = \mathbf{x}$. Indeed, note that $\mathbf{x}' - \mathbf{y}(\mathbf{x}) \in \mathcal{S}$ if and only if $\mathbf{x}'_{\mathcal{C}} = \mathbf{y}(\mathbf{x})$ if and only if $\mathbf{x} = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}'_{\mathcal{C}})$ if and only if $\mathbf{x} = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}')$. Therefore, $\Pr_{Z \sim D_{\mathcal{S},s,\mathbf{x}-\mathbf{y}(\mathbf{x})}}[Z = \mathbf{x}' - \mathbf{y}(\mathbf{x})] = \frac{\rho_s(\mathbf{x}'-\mathbf{x})}{\rho_s(\mathcal{S}-\mathbf{x}+\mathbf{y}(\mathbf{x}))} = \frac{\rho_s(\pi_{\mathcal{S}}(\mathbf{x}'))}{\rho_s(\mathcal{S}-\mathbf{x}+\mathbf{y}(\mathbf{x}))}$ if $\mathbf{x} = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}')$ and 0 otherwise. Since $\mathbf{x} = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}')$ implies $\mathbf{y}(\mathbf{x}) = \mathbf{x}'_{\mathcal{C}}$, and thus $\mathbf{y}(\mathbf{x}) - \mathbf{x} = \pi_{\mathcal{S}}(\mathbf{y}(\mathbf{x})) = \pi_{\mathcal{S}}(\mathbf{x}'_{\mathcal{C}})$, Eq. (5) follows.

Next, we prove the following intermediate claim.

Claim. For $s \geq \eta_\varepsilon(\mathcal{S})$, we have $\rho_s(\mathcal{P}) \cdot \rho_s(\mathcal{S}) \in \left[1, \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot \rho_s(\mathcal{L})$.

Proof (of Claim). By Eq. (5), if the input is a random variable $X \sim D$ for some distribution D on $\mathcal{P}$, then for all $\mathbf{x}' \in \mathcal{L}$, we have

$$\Pr_{X,Z}[f(X) = \mathbf{x}'] = \Pr_{X,Z}[X = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}')] \cdot \Pr_{X,Z}[f(X) = \mathbf{x}' \mid X = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}')]$$

$$= \Pr_{X \sim D}[X = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}')] \cdot \frac{\rho_s(\pi_{\mathcal{S}}(\mathbf{x}'))}{\rho_s(\mathcal{S} + \pi_{\mathcal{S}}(\mathbf{x}'_{\mathcal{C}}))}$$
$$\in \left[1, \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot \Pr_{X \sim D}[X = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}')] \cdot \frac{\rho_s(\pi_{\mathcal{S}}(\mathbf{x}'))}{\rho_s(\mathcal{S})}. \quad \text{(Lemma 2.4)}$$

In particular, if D is $D_{\mathcal{P},s}$, then we have (for all $\mathbf{x}' \in \mathcal{L}$) that $\Pr[f(X) = \mathbf{x}'] \in \left[1, \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot \frac{\rho_s(\mathbf{x}')}{\rho_s(\mathcal{P})\rho_s(\mathcal{S})}$ since $\pi_{\mathcal{S}}^{\perp}(\mathbf{x}') + \pi_{\mathcal{S}}(\mathbf{x}') = \mathbf{x}'$. Finally, summing both sides over all $\mathbf{x}' \in \mathcal{L}$ yields $1 \in \left[1, \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot \frac{\rho_s(\mathcal{L})}{\rho_s(\mathcal{P})\rho_s(\mathcal{S})}$, which proves the claim. □

We now proceed with the main proof. Suppose that the input consists of N random variables conditionally δ-similar to independent samples from $D_{\mathcal{P},s}$. By Eq. (5), we know that for all $\mathbf{x}' \in \mathcal{L}^N$ and any $I \subseteq [N]$,

$$\Pr_{(X_1,Z_1),\ldots,(X_N,Z_N)}[\forall j \in I, f(X_j) = \mathbf{x}'_j]$$
$$= \Pr_{X_1,\ldots,X_N}[\forall j \in I, X_j = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}'_j)] \cdot \Pr_{Z_1,\ldots,Z_N}[\forall j \in I, f(\pi_{\mathcal{S}}^{\perp}(\mathbf{x}'_j)) = \mathbf{x}'_j]$$
$$= \Pr_{X_1,\ldots,X_N}[\forall j \in I, X_j = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}'_j)] \cdot \prod_{j \in I} \frac{\rho_s(\pi_{\mathcal{S}}(\mathbf{x}'_j))}{\rho_s(\mathcal{S} + \pi_{\mathcal{S}}(\mathbf{x}'_{j,\mathcal{C}}))} \tag{6}$$

since the Z_j are independent when the values of the X_j are fixed. (Here, we write $\mathbf{x}'_{j,\mathcal{C}}$ for the unique $\mathbf{c} \in \mathcal{C}$ such that $\mathbf{x}'_j = \mathbf{s} + \mathbf{c}$ for $(\mathbf{s}, \mathbf{c}) \in \mathcal{S} \times \mathcal{C}$.) To conclude the proof, take any $i \in [N]$ and $\mathbf{x}' \in \mathcal{L}^N$. Then

$$\Pr_{X_1,\ldots,X_N}[f(X_i) = \mathbf{x}'_i \mid \forall j \in [N] \setminus \{i\}, f(X_j) = \mathbf{x}'_j]$$
$$= \frac{\Pr_{X_1,\ldots,X_N}[\forall j \in [N], f(X_j) = \mathbf{x}'_j]}{\Pr_{X_1,\ldots,X_N}[\forall j \in [N] \setminus \{i\}, f(X_j) = \mathbf{x}'_j]} \quad \text{(def. conditional probability)}$$
$$= \frac{\Pr_{X_1,\ldots,X_N}[\forall j \in [N], X_j = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}'_j)]}{\Pr_{X_1,\ldots,X_N}[\forall j \in [N] \setminus \{i\}, X_j = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}'_j)]} \cdot \frac{\rho_s(\pi_{\mathcal{S}}(\mathbf{x}'_i))}{\rho_s(\mathcal{S} + \pi_{\mathcal{S}}(\mathbf{x}'_{i,\mathcal{C}}))} \quad \text{(Equation (6))}$$
$$= \Pr_{X_1,\ldots,X_N}[X_i = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}'_i) \mid \forall j \in [N] \setminus \{i\}, X_j = \pi_{\mathcal{S}}^{\perp}(\mathbf{x}'_j)] \cdot \frac{\rho_s(\pi_{\mathcal{S}}(\mathbf{x}'_i))}{\rho_s(\mathcal{S} + \pi_{\mathcal{S}}(\mathbf{x}'_{i,\mathcal{C}}))}$$
$$= e^{\pm\delta} \cdot \frac{\rho_s(\pi_{\mathcal{S}}^{\perp}(\mathbf{x}'_i))}{\rho_s(\mathcal{P})} \cdot \frac{\rho_s(\pi_{\mathcal{S}}(\mathbf{x}'_i))}{\rho_s(\mathcal{S} + \pi_{\mathcal{S}}(\mathbf{x}'_{i,\mathcal{C}}))} \quad \text{(by assumption)}$$
$$= e^{\pm\delta} \cdot \frac{\rho_s(\mathbf{x}'_i)}{\rho_s(\mathcal{P})\rho_s(\mathcal{S} + \pi_{\mathcal{S}}(\mathbf{x}'_{i,\mathcal{C}}))}$$
$$\in \left[e^{-\delta}, e^{\delta} \cdot \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot \frac{\rho_s(\mathbf{x}'_i)}{\rho_s(\mathcal{P})\rho_s(\mathcal{S})} \quad \text{(by Lemma 2.4)}$$
$$\subseteq \left[e^{-\delta} \cdot \frac{1-\varepsilon}{1+\varepsilon}, e^{\delta} \cdot \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot \frac{\rho_s(\mathbf{x}'_i)}{\rho_s(\mathcal{L})}. \quad \text{(by the claim)}$$

Since $\left[\frac{1-\varepsilon}{1+\varepsilon}, \frac{1+\varepsilon}{1-\varepsilon}\right] \subseteq [e^{-3\varepsilon}, e^{3\varepsilon}]$ for all $0 < \varepsilon \le \frac{1}{2}$, the lemma follows. □

3.2 Combining to a Sublattice

We now show that, given many independent discrete Gaussian samples from a lattice $\mathcal{L}'$, we can construct many vectors in a *full-rank* sublattice $\mathcal{L} \subseteq \mathcal{L}'$ whose distributions are (conditionally) similar to a discrete Gaussian over $\mathcal{L}$.

By the convolution lemma [Pei08, MP13] (more precisely, by Lemma 2.7), the difference of two independent samples from $D_{\mathcal{L}',s}$ follows a distribution similar to $D_{\mathcal{L}',\sqrt{2}s}$. If we condition on the result being in the sublattice $\mathcal{L}$, then this distribution can in fact be shown to be similar to $D_{\mathcal{L},\sqrt{2}s}$.

Motivated by this fact, we consider an algorithm (Algorithm 3) that first buckets its input vectors in $\mathcal{L}'$ with respect to their cosets modulo the sublattice $\mathcal{L}$, and then (carefully) combines pairs of vectors in the same cosets to obtain vectors in $\mathcal{L}$.[5] If we start with at least $3|\mathcal{L}'/\mathcal{L}|$ vectors from $\mathcal{L}'$, then the number of output vectors is only a constant factor smaller, as shown by Lemma 3.2. Furthermore, if the input vectors are conditionally similar to independent samples from $D_{\mathcal{L}',s}$, the output vectors are conditionally similar to independent samples from $D_{\mathcal{L},\sqrt{2}s}$, as shown by Lemma 3.3.

Algorithm 3: BucketAndCombine($\mathcal{L}', \mathcal{L}, L$)

```
Input  : Full-rank lattices $\mathcal{L} \subseteq \mathcal{L}'$ in $\mathbb{R}^d$;
         A list $L$ with $N$ vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathcal{L}'$ for some integer $N \geq 3|\mathcal{L}'/\mathcal{L}|$
Output: A list $L_{out}$ with $\lfloor N/3 \rfloor$ vectors in $\mathcal{L}$

Initialize empty lists $B(\mathbf{c})$ for each coset $\mathbf{c} \in \mathcal{L}'/\mathcal{L}$
for $i = 1, \ldots, N$ do                                         // Bucketing
    Let $\mathbf{c}_i := \mathbf{x}_i \bmod \mathcal{L}$
    Append $\mathbf{x}_i$ to $B(\mathbf{c}_i)$
Initialize an empty list $L_{out}$
for $i = 1, \ldots, N$ do                                         // Combining
    if $B(\mathbf{c}_i)$ contains at least two elements and $|L_{out}| < \lfloor N/3 \rfloor$ then
        Let $\mathbf{x}, \mathbf{x}'$ be the first two elements in $B(\mathbf{c}_i)$
        Append $\mathbf{y} := \mathbf{x} - \mathbf{x}'$ to $L_{out}$
        Remove $\mathbf{x}$ and $\mathbf{x}'$ from $B(\mathbf{c}_i)$
return $L_{out}$
```

Lemma 3.2 (Correctness of Algorithm 3). *Algorithm 3 is correct. That is, given two full-rank lattices $\mathcal{L} \subseteq \mathcal{L}'$ in $\mathbb{R}^d$ and a list of N vectors in $\mathcal{L}'$, it returns a list of $\lfloor N/3 \rfloor$ vectors in $\mathcal{L}$ if $N \geq 3|\mathcal{L}'/\mathcal{L}|$.*

Proof. By construction, each element of L_{out} is of the form $\mathbf{y} = \mathbf{x} - \mathbf{x}'$ for $\mathbf{x} = \mathbf{x}' \bmod \mathcal{L}$, so $\mathbf{y} \in \mathcal{L}$. It thus remains to show that the output list L_{out} always consists of $\lfloor N/3 \rfloor$ elements. Suppose, for contradiction, that the algorithm

[5] Algorithm 3 is just a reformulation of [ALS21, Algorithm 2] with a different number of output vectors (and output vectors of the form $\mathbf{x} - \mathbf{x}'$ instead of $\mathbf{x} + \mathbf{x}'$).

returns a list of size ℓ for some $\ell \in \{0, \ldots, \lfloor N/3 \rfloor - 1\}$. Then the number of elements in L that are used as part of the output is 2ℓ, so there must be $N - 2\ell$ list elements that are not used. (Here, we talk about list elements instead of vectors, since two list elements may correspond to the same vector.) Their corresponding cosets must be distinct (since otherwise the algorithm would have been able to find more than ℓ output elements), so $N - 2\ell \le |\mathcal{L}'/\mathcal{L}| \le N/3$. It follows that $\lfloor N/3 \rfloor \le N/3 \le \ell$, which is a contradiction. Hence, the algorithm always succeeds to construct $\lfloor N/3 \rfloor$ output vectors. $\square$

The following is a variant of [ALS21, Lemma 4.5], suitable for our purposes.

Lemma 3.3 (Distribution of Output). *Let $\mathcal{L} \subseteq \mathcal{L}'$ be full-rank lattices in $\mathbb{R}^d$. Let $N \ge 3|\mathcal{L}'/\mathcal{L}|$ be a positive integer, and let $\delta \ge 0$, $0 < \varepsilon \le \frac{1}{2}$, and $s \ge \sqrt{2}\eta_\varepsilon(\mathcal{L}')$ be reals.*

If the input list consists of N random variables on $\mathcal{L}'$ that are conditionally δ-similar to independent samples from $D_{\mathcal{L}',s}$, then Algorithm 3 returns a list of $\lfloor N/3 \rfloor$ vectors from $\mathcal{L}$ that are conditionally $(4\delta + 3\varepsilon)$-similar to independent samples from $D_{\mathcal{L},\sqrt{2}s}$.

Our proof makes use of the following fact: given a sample X from a distribution similar to $D_{\mathcal{L}',s}$, if we condition on $X = \mathbf{c} \bmod \mathcal{L}$ for some $\mathcal{L} \subseteq \mathcal{L}'$ and $\mathbf{c} \in \mathcal{L}'$, then this distribution is similar to $D_{\mathcal{L}+\mathbf{c},s}$. More generally, conditioning on cosets preserves conditional similarity.

Lemma 3.4 (Conditioning on Cosets). *Let $\mathcal{L} \subseteq \mathcal{L}'$ be full-rank lattices in $\mathbb{R}^d$. Let N be a positive integer, and let $\delta \ge 0$ and $s > 0$ be reals. Suppose that $X_1, \ldots, X_N \in \mathcal{L}'$ are discrete random variables that are conditionally δ-similar to independent samples from $D_{\mathcal{L}',s}$. Then, for all $i \in [N]$, all $\mathbf{c} \in (\mathcal{L}'/\mathcal{L})^N$, and all $\mathbf{x} \in (\mathcal{L}')^N$ satisfying $\mathbf{x}_j = \mathbf{c}_j \bmod \mathcal{L}$ for all $j \in [N]$,*

$$\Pr[X_i = \mathbf{x}_i \mid X_{-i} = \mathbf{x}_{-i} \textit{ and } \forall j \in [N], X_j = \mathbf{c}_j \bmod \mathcal{L}] = e^{\pm 2\delta} \cdot \frac{\rho_s(\mathbf{x}_i)}{\rho_s(\mathcal{L} + \mathbf{c}_i)}.$$

Proof. Suppose that $X = (X_1, \ldots, X_N)$ consists of N random variables on $\mathcal{L}'$ that are conditionally δ-similar to independent samples from $D_{\mathcal{L}',s}$. Then, by definition, we have for all $i \in [N]$ and all $\mathbf{x} \in (\mathcal{L}')^N$ that

$$\Pr[X_i = \mathbf{x}_i \mid X_{-i} = \mathbf{x}_{-i}] = e^{\pm\delta} \cdot \frac{\rho_s(\mathbf{x}_i)}{\rho_s(\mathcal{L}')}. \tag{7}$$

Consider arbitrary $i \in [N]$, $\mathbf{c} \in (\mathcal{L}'/\mathcal{L})^N$, and $\mathbf{x} \in (\mathcal{L}')^N$ satisfying $\mathbf{x}_j = \mathbf{c}_j \bmod \mathcal{L}$ for all $j \in [N]$. Then, by definition of conditional probability,

$$\begin{aligned} &\Pr[X_i = \mathbf{x}_i \mid X_{-i} = \mathbf{x}_{-i} \text{ and } \forall j \in [N], X_j = \mathbf{c}_j \bmod \mathcal{L}] \\ &= \frac{\Pr[X_i = \mathbf{x}_i \text{ and } X_{-i} = \mathbf{x}_{-i} \text{ and } \forall j \in [N], X_j = \mathbf{c}_j \bmod \mathcal{L}]}{\Pr[X_{-i} = \mathbf{x}_{-i} \text{ and } \forall j \in [N], X_j = \mathbf{c}_j \bmod \mathcal{L}]} \\ &= \frac{\Pr[X_i = \mathbf{x}_i \text{ and } X_{-i} = \mathbf{x}_{-i}]}{\Pr[X_{-i} = \mathbf{x}_{-i} \text{ and } X_i = \mathbf{c}_i \bmod \mathcal{L}]} \qquad (\text{as } \mathbf{x}_j = \mathbf{c}_j \bmod \mathcal{L}\, \forall j \in [N]) \end{aligned}$$

$$= \frac{\Pr[X_i = \mathbf{x}_i \mid X_{-i} = \mathbf{x}_{-i}]}{\sum_{\mathbf{v}\in\mathcal{L}+\mathbf{c}_i} \Pr[X_i = \mathbf{v} \mid X_{-i} = \mathbf{x}_{-i}]}$$
$$= e^{\pm 2\delta} \frac{\rho_s(\mathbf{x}_i)/\rho_s(\mathcal{L}')}{\rho_s(\mathcal{L}+\mathbf{c}_i)/\rho_s(\mathcal{L}')}$$

where we apply Eq. (7) twice (to both the numerator and denominator) to obtain the last line. The conclusion then immediately follows. □

Proof (of Lemma 3.3). Correctness (on arbitrary input) follows from Lemma 3.2. Let $Y_1, \ldots, Y_M$ be the random variables corresponding to the vectors in the output list (in order), where $M := \lfloor N/3 \rfloor$. We want to show that, for all $j \in [M]$ and $\mathbf{y} \in \mathcal{L}^M$,

$$\Pr_{X_1,\ldots,X_N}[Y_j = \mathbf{y}_j \mid Y_{-j} = \mathbf{y}_{-j}] = e^{\pm(4\delta+3\varepsilon)} D_{\mathcal{L},\sqrt{2}s}(\mathbf{y}_j).$$

By Lemma 2.1, it suffices to show that for all $\mathbf{c}_1, \ldots, \mathbf{c}_N \in \mathcal{L}'/\mathcal{L}$ such that $\Pr_{X_1,\ldots,X_N}[\forall i \in [N], X_i = \mathbf{c}_i \bmod \mathcal{L}] > 0$, we have, for all $j \in [M]$, $\mathbf{y} \in \mathcal{L}^M$,

$$\Pr_{X_1,\ldots,X_N}[Y_j = \mathbf{y}_j \mid Y_{-j} = \mathbf{y}_{-j} \text{ and } \forall i \in [N], X_i = \mathbf{c}_i \bmod \mathcal{L}]$$
$$= e^{\pm(4\delta+3\varepsilon)} D_{\mathcal{L},\sqrt{2}s}(\mathbf{y}_j). \tag{8}$$

Consider any $\mathbf{c}_1, \ldots, \mathbf{c}_N \in \mathcal{L}'/\mathcal{L}$ such that $\Pr_{X_1,\ldots,X_N}[\forall i \in [N], X_i = \mathbf{c}_i \bmod \mathcal{L}] > 0$. Note that the output (in particular, the way the vectors are paired) is entirely determined by the cosets $(\mathbf{c}_1, \ldots, \mathbf{c}_N)$ for $\mathbf{c}_i := \mathbf{x}_i \bmod \mathcal{L}$. In particular, for any permutation $\pi \colon [N] \to [N]$ such that $(\mathbf{x}'_1, \ldots, \mathbf{x}'_N) := (\mathbf{x}_{\pi(1)}, \ldots, \mathbf{x}_{\pi(N)})$ satisfies $\mathbf{y}_j = \mathbf{x}'_{2j-1} - \mathbf{x}'_{2j}$ for all $j \in [M]$, we have that $\mathbf{x}'_1, \ldots, \mathbf{x}'_{2M}$ is entirely determined by the cosets $(\mathbf{c}_1, \ldots, \mathbf{c}_N)$. (The order of the vectors $\mathbf{x}'_{2M+1}, \ldots, \mathbf{x}'_N$ does not affect the algorithm's output.) Without loss of generality, we therefore redefine $(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ as $(\mathbf{x}'_1, \ldots, \mathbf{x}'_N)$ for such a permutation (allowing us to write $\mathbf{y}_j = \mathbf{x}_{2j-1} - \mathbf{x}_{2j}$ for all $j \in [M]$).

Let D be the distribution of the input variables $X = (X_1, \ldots, X_N)$ *conditional on* $X_i = \mathbf{c}_i \bmod \mathcal{L}$ for all $i \in [N]$. By the conditional similarity assumption and by Lemma 3.4, for all $i \in [N]$ and all $\mathbf{x} \in (\mathcal{L}')^N$ satisfying $\mathbf{x}_j = \mathbf{c}_j \bmod \mathcal{L}$ for all $j \in [N]$, we have

$$\Pr_{X\sim D}[X_i = \mathbf{x}_i \mid X_{-i} = \mathbf{x}_{-i}] = e^{\pm 2\delta} D_{\mathcal{L}+\mathbf{c}_i,s}(\mathbf{x}_i), \tag{9}$$

which we repeatedly use below.

To show that Eq. (8) holds for all $j \in [M]$, $\mathbf{y} \in \mathcal{L}^M$, and thus to finish the proof, it suffices (by another application of Lemma 2.1) to show that for all $j \in [M]$, $\mathbf{y} \in \mathcal{L}^M$, and all $\mathbf{v} \in (\mathcal{L}')^{N-2}$ satisfying $\Pr[X_{-\{2j-1,2j\}} = \mathbf{v} | Y_{-j} = \mathbf{y}_j] > 0$, we have

$$\Pr_{X\sim D}[Y_j = \mathbf{y}_j \mid Y_{-j} = \mathbf{y}_{-j} \text{ and } X_{-\{2j-1,2j\}} = \mathbf{v}] = e^{\pm(4\delta+3\varepsilon)} D_{\mathcal{L},\sqrt{2}s}(\mathbf{y}_j).$$

So take any $j \in [M]$ and $\mathbf{y} \in \mathcal{L}^M$, and any such $\mathbf{v} \in (\mathcal{L}')^{N-2}$. Since $X_{-\{2j-1,-2j\}}$ determines all the entries of Y_{-j}, we have that $\Pr[X_{-\{2j-1,2j\}} = \mathbf{v} | Y_{-j} = \mathbf{y}_j] > 0$ implies $\Pr[Y_{-j} = \mathbf{y}_{-j} \mid X_{-\{2j-1,2j\}} = \mathbf{v}] = 1$. Hence,

$$\begin{aligned}&\Pr_{X\sim D}[Y_j = \mathbf{y}_j \mid Y_{-j} = \mathbf{y}_{-j} \text{ and } X_{-\{2j-1,2j\}} = \mathbf{v}]\\ &= \Pr_{X\sim D}[Y_j = \mathbf{y}_j \mid X_{-\{2j-1,2j\}} = \mathbf{v}].\end{aligned}$$

Writing $\mathbf{c} := X_{2j-1} \bmod \mathcal{L} = X_{2j} \bmod \mathcal{L}$, we have

$$\begin{aligned}&\Pr_{X\sim D}[Y_j = \mathbf{y}_j \mid X_{-\{2j-1,2j\}} = \mathbf{v}]\\ &= \Pr_{X\sim D}[X_{2j-1} - X_{2j} = \mathbf{y}_j \mid X_{-\{2j-1,2j\}} = \mathbf{v}]\\ &= \sum_{\mathbf{x}\in\mathcal{L}+\mathbf{c}} \Pr_{X\sim D}[X_{2j-1} = \mathbf{x} \text{ and } X_{2j} = \mathbf{x} - \mathbf{y}_j \mid X_{-\{2j-1,2j\}} = \mathbf{v}]\\ &= e^{\pm 2\delta} \cdot \sum_{\mathbf{x}\in\mathcal{L}+\mathbf{c}} D_{\mathcal{L}+\mathbf{c},s}(\mathbf{x}) \cdot \Pr_{X\sim D}[X_{2j} = \mathbf{x} - \mathbf{y}_j \mid X_{-\{2j-1,2j\}} = \mathbf{v}]\end{aligned}$$

by definition of conditional probability and Eq. (9) (for $i := 2j-1$). Then,

$$\begin{aligned}&\Pr_{X\sim D}[X_{2j} = \mathbf{x} - \mathbf{y}_j \mid X_{-\{2j-1,2j\}} = \mathbf{v}]\\ &= \sum_{\mathbf{z}\in\mathcal{L}+\mathbf{c}} \Pr_{X\sim D}[X_{2j-1} = \mathbf{z} \text{ and } X_{2j} = \mathbf{x} - \mathbf{y}_j \mid X_{-\{2j-1,2j\}} = \mathbf{v}]\\ &= e^{\pm 2\delta} \cdot D_{\mathcal{L}+\mathbf{c},s}(\mathbf{x} - \mathbf{y}_j) \cdot \sum_{\mathbf{z}\in\mathcal{L}+\mathbf{c}} \Pr_{X\sim D}[X_{2j-1} = \mathbf{z} \mid X_{-\{2j-1,2j\}} = \mathbf{v}]\\ &= e^{\pm 2\delta} \cdot D_{\mathcal{L}+\mathbf{c},s}(\mathbf{x} - \mathbf{y}_j)\end{aligned}$$

using again the definition of conditional probability and Eq. (9). We complete the proof by noting that Lemma 2.7 and $\frac{1+\varepsilon}{1-\varepsilon} \le e^{3\varepsilon}$ for $\varepsilon \le \frac{1}{2}$ imply

$$\begin{aligned}\Pr_{X\sim D}[Y_j = \mathbf{y}_j \mid X_{-\{2j-1,2j\}} = \mathbf{v}] &= e^{\pm 4\delta} \cdot \sum_{\mathbf{x}\in\mathcal{L}+\mathbf{c}} D_{\mathcal{L}+\mathbf{c},s}(\mathbf{x}) \cdot D_{\mathcal{L}+\mathbf{c},s}(\mathbf{x} - \mathbf{y}_j)\\ &= e^{\pm 4\delta} \cdot \Pr_{(X_1,X_2)\sim(D_{\mathcal{L}+\mathbf{c},s})^2}[X_1 - X_2 = \mathbf{y}_j]\\ &= e^{\pm(4\delta+3\varepsilon)} \cdot D_{\mathcal{L},\sqrt{2}s}(\mathbf{y}_j).\end{aligned}$$

□

3.3 The Algorithm: Wagner as a Gaussian Sampler

We are now ready to present our Gaussian sampler, laid out as Algorithm 4.

Algorithm 4: Wagner-Style Gaussian Sampler

```
Input  : Integers $n, m, q$;
         Full-rank matrix $\mathbf{A} = [\mathbf{A}' \mid \mathbf{I}_n] \in \mathbb{Z}_q^{n \times m}$;
         Integer parameters $N, r, (p_i)_{i=1}^r, (b_i)_{i=1}^r$ with $\sum_{i=1}^r b_i = n$;
         Real parameter $s_0 > 0$
Output: List of vectors in $\Lambda_q^\perp(\mathbf{A})$

Let $\Lambda_0 := \mathbb{Z}^{m-n}$
Initialize a list $L_0$ with $3^r N$ independent samples from $D_{\Lambda_0, s_0}$
for $i = 1, \ldots, r$ do
    Let $\Lambda_i$ be defined as in Equation (3)
    Let $\Lambda_i' = \mathcal{L}(\mathbf{B}_i')$ for $\mathbf{B}_i'$ defined as in Equation (4)
    $L_{i-1}' := \emptyset$
    for $\mathbf{x} \in L_{i-1}$ do                                              // Lift
        Sample $\mathbf{x}' \sim \mathrm{DGLift}(\Lambda_{i-1}, \Lambda_i', s_{i-1}, \mathbf{x})$  ▷ Algorithm 2
        Append $\mathbf{x}'$ to $L_{i-1}'$
    $L_i = \mathrm{BucketAndCombine}(\Lambda_i', \Lambda_i, L_{i-1}')$  ▷ Algorithm 3      // Combine
    $s_i := \sqrt{2} s_{i-1}$
return $L_r$
```

Remark 3.1. In our applications of Algorithm 4, we consider input parameter s_0 satisfying $s_0 \geq \sqrt{\ln(2(m-n)+4)/\pi}$ and $\sqrt{2^{i-1}} s_0 \geq \frac{q}{p_i}\sqrt{\ln(2b_i+4)/\pi}$ for all $i \in [r]$. This allows us to use the exact sampler from Lemma 2.6 to sample from D_{Λ_0, s_0} (in the initialization) and from $D_{\frac{q}{p_i}\mathbb{Z}^{b_i}, \sqrt{2^{i-1}} s_0}$ (in iterations $i = 1, \ldots, r$).

Theorem 3.1 (Correctness of One Iteration). *Let $\delta \geq 0$ and $0 < \varepsilon \leq \frac{1}{2}$ be reals. For $i \in [r]$, consider iteration i of Algorithm 4 with well-defined parameters. If $s_{i-1} \geq \max(\eta_\varepsilon(\frac{q}{p_i}\mathbb{Z}^{b_i}), \sqrt{2}\eta_\varepsilon(\Lambda_i'), \frac{q}{p_i}\sqrt{\ln(2b_i+4)/\pi})$ and L_{i-1} consists of $|L_{i-1}| \geq 3p_i^{b_i}$ vectors in Λ_{i-1} that are conditionally δ-similar to independent samples from $D_{\Lambda_{i-1}, s_{i-1}}$, then L_i consists of $\lfloor |L_{i-1}|/3 \rfloor$ vectors in Λ_i that are conditionally $(4\delta + 15\varepsilon)$-similar to independent samples from $D_{\Lambda_i, \sqrt{2}s_{i-1}}$.*

Proof. Let $\mathcal{S}$ be the embedding of $\frac{q}{p_i}\mathbb{Z}^{b_i}$ in $\mathbb{R}^{m-n+n_i}$ and $\mathcal{P}$ the embedding of Λ_{i-1} in $\mathbb{R}^{m-n+n_i}$. Note that $\mathcal{S}$ is a primitive sublattice of Λ_i' and that $\mathcal{P} = \pi_{\mathcal{S}}^\perp(\mathcal{C})$ for some complement $\mathcal{C}$ to $\mathcal{S}$. Since $s_{i-1} \geq \eta_\varepsilon(\frac{q}{p_i}\mathbb{Z}^{b_i}) = \eta_\varepsilon(\mathcal{S})$, the application of the algorithm from Lemma 3.1 turns the $|L_{i-1}|$ random variables on Λ_{i-1} into $|L_{i-1}|$ random variables on Λ_i' that are conditionally $(\delta + 3\varepsilon)$-similar to $D_{\Lambda_i', s_{i-1}}$. Here, as oracle to sample from $D_{\frac{q}{p_i}\mathbb{Z}^{b_i}, s_{i-1}, \mathbf{c}}$ (exactly), we use Lemma 2.6 to sample from $D_{\mathbb{Z}^{b_i}, \frac{p_i}{q}s_{i-1}, \frac{p_i}{q}\mathbf{c}}$, and multiply the resulting vector by $\frac{q}{p_i}$. This is justified since we assume $\frac{p_i}{q}s_{i-1} \geq \sqrt{\ln(2b_i+4)/\pi}$.

By Lemma 3.3 (where we use that $s_{i-1} \geq \sqrt{2}\eta_\varepsilon(\Lambda_i')$), the output list L_i of BucketAndCombine consists of $\lfloor |L_{i-1}|/3 \rfloor$ vectors in Λ_i that are conditionally

δ'-similar to independent samples from $D_{\Lambda_i,\sqrt{2}s_{i-1}}$, where $\delta' = 4(\delta + 3\varepsilon) + 3\varepsilon = 4\delta + 15\varepsilon$. $\square$

Theorem 3.2 (Correctness of Algorithm 4). *Let $0 < \varepsilon \leq \frac{1}{2}$ be a real, and r an integer. If the input parameters satisfy $N \geq p_i^{b_i}$, $s_0 \geq \sqrt{\ln(2(m-n)+4)/\pi}$, and $\sqrt{2^{i-1}}s_0 \geq \max(\eta_\varepsilon(\frac{q}{p_i}\mathbb{Z}^{b_i}), \sqrt{2}\eta_\varepsilon(\Lambda_i'), \frac{q}{p_i}\sqrt{\ln(2b_i+4)/\pi})$ for all $i \in [r]$, then Algorithm 4 returns a list of size N consisting of vectors that are conditionally $4^r 5\varepsilon$-similar to independent samples from $D_{\Lambda_q^\perp(\mathbf{A}),\sqrt{2^r}s_0}$.*

Proof. To obtain the list L_0, we use the exact $D_{\mathbb{Z}^{m-n},s_0}$-sampler from Lemma 2.6, which takes time $\mathrm{poly}(m-n, \log s_0)$. This is justified since we assume $s_0 \geq \sqrt{\ln(2(m-n)+4)/\pi}$.

We show by induction that, for each iteration $i \in \{1, \ldots, r\}$, the output list L_i consists of $3^{r-i}N$ vectors in Λ_i that are conditionally δ_i-similar to independent samples from D_{Λ_i,s_i} for $\delta_i := (4^i - 1)5\varepsilon$. The theorem then immediately follows, since $(4^r - 1)5\varepsilon \leq 4^r 5\varepsilon$.

By assumption, $|L_0| = 3^r N \geq 3^r p_1^{b_1} \geq 3p_1^{b_1}$ and $s_0 \geq \max(\eta_\varepsilon(\frac{q}{p_1}\mathbb{Z}^{b_1}), \sqrt{2}\eta_\varepsilon(\Lambda_1'), \frac{q}{p_1}\sqrt{\ln(2b_1+4)/\pi})$. Therefore, Theorem 3.1 implies that the output list L_1 of iteration $i = 1$ consists of $3^{r-1}N$ vectors in Λ_1 that are conditionally 15ε-similar to independent samples from D_{Λ_1,s_1}. Since $\delta_1 = 15\varepsilon$, this proves the base case.

Consider any $i \in \{2, \ldots, r\}$, and suppose the claim holds for all $1 \leq j \leq i-1$. By the induction hypothesis and assumption, $|L_{i-1}| = 3^{r-(i-1)}N \geq 3^{r-(i-1)}p_i^{b_i} \geq 3p_i^{b_i}$ and $s_{i-1} = \sqrt{2^{i-1}}s_0 \geq \max(\eta_\varepsilon(\frac{q}{p_i}\mathbb{Z}^{b_i}), \sqrt{2}\eta_\varepsilon(\Lambda_i'), \frac{q}{p_i}\sqrt{\ln(2b_i+4)/\pi})$. Therefore, by Theorem 3.1, the output list L_i of iteration i consists of $3^{r-i}N$ vectors in Λ_i. By the induction hypothesis the vectors in L_{i-1} are conditionally δ_{i-1}-similar to independent samples from $D_{\Lambda_{i-1},s_{i-1}}$, so by Theorem 3.1 the vectors in L_i are conditionally $(4\delta_{i-1} + 15\varepsilon)$-similar to independent samples from D_{Λ_i,s_i}. Since $4\delta_{i-1} + 15\varepsilon = 4(4^{i-1} - 1)5\varepsilon + 15\varepsilon = (4^i - 1)5\varepsilon = \delta_i$, the claim follows. $\square$

Remark 3.2 (Expected Runtime). Using the exact sampler from Lemma 2.6 ensures that all vectors processed by Algorithm 4 have expected bitsize at most $\mathrm{poly}(m, r, \log s_0, \log q)$ when the parameters satisfy $p_i \leq q$ for all $i \in [r]$. Hence, the expected runtime of Algorithm 4 is at most $(3^r N + \sum_{i=1}^r |L_{i-1}|) \cdot \mathrm{poly}(m, r, \log s_0, \log q) = 3^r N \cdot \mathrm{poly}(m, r, \log s_0, \log q)$.

3.4 Putting It All Together

We now have all the ingredients to prove the existence of a subexponential-time algorithm for sampling from a distribution conditionally similar to $D_{\Lambda_q^\perp(\mathbf{A}),s}$ for random (full-rank) matrices $\mathbf{A}$. We write the width s of the desired discrete Gaussian distribution as $s = q/f$ for some $f > 1$, and remark that the difficulty of sampling with width q/f increases with f. Below, we demonstrate that subexponential complexity is feasible when q/f and m are above a certain threshold.

We note that $q^{1-n/m} \geq 2^{\Theta(\log n/\log\log n)}$ for the parameters of interest, which tends to infinity as n grows (so the assumption $q^{1-n/m} \geq 6$ is not too restrictive).

Theorem 3.3. *For $n \in \mathbb{N}$, let $m \geq n$ be an integer and $q = \mathrm{poly}(n)$ be a prime such that $q^{1-n/m} \geq 6$. Let $f > 1$ and $\varepsilon \leq \frac{1}{m}$ be positive reals such that $\frac{q}{f} \geq \sqrt{\ln(1/\varepsilon)}$. There exists a value of N satisfying*

$$\log_2(N) = \frac{n/2}{\ln(\ln(q)) - \ln(\ln(f) + \frac{1}{2}\ln\ln(1/\varepsilon)) - O(1)} \tag{10}$$

such that, if $m \geq n + 2\log_2 N$, there is a randomized algorithm that, given a uniformly random full-rank matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, with probability at least $1-2^{-\tilde{\Omega}(n)}$ returns N vectors in $\Lambda_q^\perp(\mathbf{A})$. The output vectors are conditionally $q^4\varepsilon$-similar to independent samples from $D_{\Lambda_q^\perp(\mathbf{A}),\frac{q}{f}}$. This algorithm has time and memory complexity $N \cdot \mathrm{poly}(m)$.

Proof. We first provide a choice of parameters for Algorithm 4 (instantiated with the exact sampler from Lemma 2.6) and show that the conditions of Theorem 3.2 are satisfied.

Choice of Parameters. For $\varepsilon' := \varepsilon/5$, let N be the smallest integer such that

$$\log_2(N/q) \geq \frac{n/2}{\ln(\ln(q)) - \ln(\ln(f) + \frac{1}{2}\ln(\frac{144}{\pi}\ln(3/\varepsilon')) + \frac{1}{2})}.$$

Then N satisfies Eq. (10) for sufficiently large n. Let $s_0 := \frac{q/f}{\sqrt{2^r}}$, where[6]

$$r := \lfloor 2\log_2(q/f) - \log_2(\tfrac{144}{\pi}\ln(3/\varepsilon'))\rfloor.$$

For $i \in [r]$, define $p_i := \lfloor q/\sqrt{2^i}\rfloor$. For $i \in [r-1]$, define $b_i := \lceil \frac{\log_2(N)}{\log_2(q)-i/2} - 1\rceil$, and define $b_r := n - \sum_{i=1}^{r-1} b_i$ so that $\sum_{i=1}^{r} b_i = n$. Note that p_i, b_i are integers.

To show that these parameters satisfy the conditions of Theorem 3.2, we consider sufficiently large n so that $\varepsilon' \leq \frac{1}{6}$ (i.e., $\varepsilon \leq \frac{5}{6}$) and assume $2\log_2(N) \leq m - n$.

Verifying that N is Large Enough. We claim that $N \geq p_i^{b_i}$ for all $i \in [r]$. For all $i \in [r-1]$, this claim follows immediately from the definition of b_i: $b_i \leq \frac{\log_2(N)}{\log_2(q)-i/2} \leq \frac{\log_2(N)}{\log_2(p_i)}$. So it remains to show that $b_r \leq \frac{\log_2(N)}{\log_2(p_r)}$. By definition,

$$\begin{aligned}\log_2(N/q) &\geq \frac{n/2}{\ln(\ln(q)) - \ln\left(\ln(f) + \frac{1}{2}\ln(\frac{144}{\pi}\ln(3/\varepsilon')) + \frac{1}{2}\right)}\\ &\geq \frac{n/2}{\ln(\log_2(q)) - \ln(\log_2(f) + \frac{1}{2}\log_2(\frac{144}{\pi}\ln(3/\varepsilon')) + \frac{1}{2})}\end{aligned}$$

[6] For all interesting parameters, we have $r \geq 1$. For the rare setting of parameters for which $r < 1$, we replace r by $r+10$. (This suffices since, for any valid parameters, the assumption $q/f \geq \sqrt{\ln(1/\varepsilon)}$ ensures that $2\log_2(q/f) - \log_2(144\ln(3/\varepsilon')/\pi) + 10 \geq 1$ when $n \geq 2$.) Note that increasing r by a constant additive factor does not affect the proof; in particular, Eq. (11) would still hold as it decreases with r.

$$\geq \frac{n/2}{\ln\left(\frac{2\log_2(q)}{2\log_2(q)-r}\right)} \tag{11}$$

since $r \geq 2\log_2(q/f) - \log_2(\frac{144}{\pi}\ln(3/\varepsilon')) - 1$. In particular, using known facts of integrals (recall Footnote 7 on page 6) we obtain

$$n \leq 2\log_2(N/q) \cdot \ln\left(\frac{2\log_2(q)}{2\log_2(q)-r}\right) = \int_0^r \frac{2\log_2(N/q)}{2\log_2(q)-x}dx \leq \sum_{i=1}^{r} \frac{2\log_2(N/q)}{2\log_2(q)-i}.$$

Since $\frac{2\log_2(N/q)}{2\log_2(q)-i} \leq \frac{2\log_2(N)}{2\log_2(q)-i} - 1 \leq b_i$, we obtain $n \leq \sum_{i=1}^{r-1} b_i + \frac{2\log_2(N/q)}{2\log_2(q)-r}$, and thus $b_r = n - \sum_{i=1}^{r-1} b_i \leq \frac{2\log_2(N/q)}{2\log_2(q)-r} \leq \frac{2\log_2(N)}{2\log_2(q)-r} \leq \frac{\log_2(N)}{\log_2(p_r)}$, as desired.

Verifying the Smoothing Conditions of Theorem 3.2*.* To show that the smoothing conditions in Theorem 3.2 are satisfied with probability $1-2^{-\tilde{\Omega}(n)}$, it suffices to show that the following holds (for large enough n):

(I) $s_0 \geq \sqrt{\ln(2(m-n)+4)/\pi}$ and $\sqrt{2^{i-1}}s_0 \geq \frac{q}{p_i}\sqrt{\ln(2b_i+4)/\pi}$ for all $i \in [r]$.
(II) With probability $1-2^{-\tilde{\Omega}(n)}$, $\sqrt{2^{i-1}}s_0 \geq \sqrt{2}\max(\eta_{\varepsilon'/3}(\frac{q}{p_i}\mathbb{Z}^{b_i}), \eta_{\varepsilon'/3}(\Lambda_{i-1}))$ for all $i \in [r]$.

Indeed, if $\sqrt{2^{i-1}}s_0 \geq \sqrt{2}\max(\eta_{\varepsilon'/3}(\frac{q}{p_i}\mathbb{Z}^{b_i}), \eta_{\varepsilon'/3}(\Lambda_{i-1}))$ (for any $i \in [r]$), then $\sqrt{2^{i-1}}s_0 \geq \max(\eta_{\varepsilon'}(\frac{q}{p_i}\mathbb{Z}^{b_i}), \sqrt{2}\eta_{\varepsilon'}(\Lambda'_i))$ by [EWY23, Proposition 2] (with the embedding of $\frac{q}{p_i}\mathbb{Z}^{b_i}$ in $\mathbb{R}^{m-n+n_i}$ as sublattice) and because $\eta_{\varepsilon'/3}(\frac{q}{p_i}\mathbb{Z}^{b_i}) \geq \eta_{\varepsilon'}(\frac{q}{p_i}\mathbb{Z}^{b_i})$.

To prove (I) and (II), we use that our choice of r implies $s_0 \geq \sqrt{\frac{144\ln(3/\varepsilon')}{\pi}}$. Furthermore, we emphasize that $\varepsilon \leq \frac{1}{m}$ implies $\varepsilon' \leq \frac{3}{4m}$, so $\varepsilon' \leq \min(\frac{3}{4(m-n)}, \frac{3}{4b_1})$ and $\varepsilon' \leq \min(\frac{3}{4(m-n+n_{i-1})}, \frac{3}{4b_i})$ for all $i \in \{2,\ldots,r\}$. (Hence, we can apply the results from Sect. 2.4 to bound the parameters $\eta_{\varepsilon'}(\frac{q}{p_i}\mathbb{Z}^{b_i})$ and $\eta_{\varepsilon'}(\Lambda'_i)$.)

Consider any $i \in [r]$. By Lemma 2.8 (with dimension b_i and using that $\varepsilon' \leq \frac{3}{4b_i}$), we obtain $\sqrt{2}\eta_{\varepsilon'/3}(\frac{q}{p_i}\mathbb{Z}^{b_i}) \leq \frac{q}{p_i}\sqrt{\frac{4\ln(3/\varepsilon')}{\pi}} \leq \sqrt{2^{i-1}}\sqrt{\frac{32\ln(3/\varepsilon')}{\pi}} \leq \sqrt{2^{i-1}}s_0$ (since $p_i = \lfloor q/\sqrt{2^i}\rfloor \geq (q/\sqrt{2^i})/2$ whenever $p_i \geq 2$, and thus $\frac{q}{p_i} \leq 2\sqrt{2^i}$). Furthermore, for $i=1$, we have by Lemma 2.8 that $\sqrt{2}\eta_{\varepsilon'/3}(\Lambda_0) = \sqrt{2}\eta_{\varepsilon'/3}(\mathbb{Z}^{m-n}) < \sqrt{\frac{4\ln(3/\varepsilon')}{\pi}} \leq s_0$. Since $\varepsilon' \leq \frac{3}{4m}$ we have $\frac{3}{\varepsilon'} \geq 2m \geq 2(m-n)+4$ and $\frac{3}{\varepsilon'} \geq 4m \geq 4n \geq 2n+4 \geq 2b_i+4$ for all $i \in [r]$. Hence, $s_0 \geq \sqrt{\frac{4\ln(3/\varepsilon')}{\pi}}$ implies $s_0 \geq \sqrt{\frac{\ln(3/\varepsilon')}{\pi}} \geq \sqrt{\frac{\ln(2(m-n)+4)}{\pi}}$, and $\sqrt{2^{i-1}}s_0 \geq \frac{q}{p_i}\sqrt{\frac{4\ln(3/\varepsilon')}{\pi}}$ implies $\sqrt{2^{i-1}}s_0 \geq \frac{q}{p_i}\sqrt{\frac{\ln(3/\varepsilon')}{\pi}} \geq \frac{q}{p_i}\sqrt{\frac{\ln(2b_i+4)}{\pi}}$ for all $i \in [r]$. Thus, (I) holds for $n \geq 2$.

Next, we observe that $q^{\frac{n_j}{m-n+n_j}} \leq \sqrt{2^j}$ for all $j \in [r]$. Indeed, for all $j \in [r]$, we have $n_j \leq jb_j$, so $\frac{n_j}{j}(2\log_2(q)-j) \leq b_j(2\log_2(q)-j) \leq 2\log_2(N) \leq m-$

n. Since $\frac{n_j}{j}(2\log_2(q)-j) \leq m-n$ if and only if $q^{\frac{n_j}{m-n+n_j}} \leq \sqrt{2^j}$, the claim follows. Thus, for each $i \in \{2,\ldots,r\}$, Lemma 2.11 and the previous claim imply that $\sqrt{2}\eta_{\varepsilon'/3}(\Lambda_{i-1}) < \sqrt{\frac{144\ln(3/\varepsilon')}{\pi}} q^{\frac{n_{i-1}}{m-n+n_{i-1}}} \leq \sqrt{\frac{144\ln(3/\varepsilon')}{\pi}}\sqrt{2^{i-1}} \leq \sqrt{2^{i-1}}s_0$, except with probability $< 2^{-n_{i-1}}$.

Therefore, the union bound implies that, with probability at least $1 - \sum_{i=1}^{r-1} 2^{-n_i}$, $\sqrt{2^{i-1}}s_0 \geq \sqrt{2}\max(\eta_{\varepsilon'/3}(\frac{q}{p_i}\mathbb{Z}^{b_i}), \eta_{\varepsilon'/3}(\Lambda_{i-1}))$ for all $i \in [r]$. Note that $1-\sum_{i=1}^{r-1} 2^{-n_i} \geq 1 - r2^{-b_1} = 1-2^{-\tilde{\Omega}(n)}$ since $\log_2(r) = \log_2\log_2(n) + O(1)$, $b_1 = \Omega(\frac{\log_2 N}{\log_2 n})$, and $\log_2(N) = \Omega(\frac{n}{\log_2\log_2(n)})$, so (II) holds as well.[7]

Conclusion of the Proof. Theorem 3.2 then implies that the output of this algorithm consists of at least N vectors in $\Lambda_q^\perp(\mathbf{A})$ that are conditionally $4^r 5\varepsilon'$-similar to independent samples from $D_{\Lambda_q^\perp(\mathbf{A}),\sqrt{2^r}s_0}$. Since $4^r \leq q^4$, we obtain that they are conditionally δ-similar for $\delta = q^4 5\varepsilon' = q^4\varepsilon$-similar.

Finally, the expected runtime of Algorithm 4 is at most $3^r N \cdot \mathrm{poly}(m, r, \log s_0, \log q)$ by Remark 3.2. Since $r = O(\log q)$, $s_0 \leq q$, and $q = \mathrm{poly}(m)$, it follows that the time and memory complexity are both upper bounded by $N \cdot \mathrm{poly}(m)$. $\square$

4 Asymptotic Application to Cryptographic Problems

We now apply our previous result on Gaussian sampling for SIS lattices to solving various variants of SIS in subexponential time. We also discuss why our result does not directly lead to a provable subexponential-time algorithm for LWE with narrow error distribution.

4.1 Implications for SIS$^\infty$

The following theorem instantiated with $m = n + \omega(n/\log\log n)$ such that[8] $m = \mathrm{poly}(n)$, $q = n^{\Theta(1)}$, $\beta = q/\mathrm{polylog}(n)$, and a sufficiently small $\varepsilon = 1/\mathrm{poly}(n)$ provides a subexponential-time algorithm for $\mathrm{SIS}^\infty_{n,m,q,\beta}$. In fact, our subexponential complexity holds for even smaller norm bounds: for instance, we can achieve $\beta = q/2^{\log(n)^c}$ for any constant $c < 1$.

Theorem 4.1. *For $n \in \mathbb{N}$, let $m = n + \omega(n/\ln\ln n)$ be integer and $q = n^{\Theta(1)}$ be prime such that $q^{1-n/m} \geq 6$. Let $f > 1$ and $\varepsilon \leq \frac{1}{mq^4}$ be positive reals such*

[7] It also follows that $\eta_{\varepsilon/4}(\Lambda_q^\perp(\mathbf{A})) < \frac{q}{f}$. Indeed, another application of Lemma 2.11 yields $\eta_{\varepsilon/4}(\Lambda_q^\perp(\mathbf{A})) < \sqrt{72\ln(4/\varepsilon)/\pi}q^{n/m}$, except with probability $< 2^{-n}$, but this does not affect the lower bound on the success probability. By the aforementioned fact, we have $q^{n/m} \leq \sqrt{2^r}$, so by definition of r we obtain $\eta_{\varepsilon/4}(\Lambda_q^\perp(\mathbf{A})) < \frac{q}{f}$.

[8] Note that one may always decrease m by ignoring SIS variables, hence the condition $m = \mathrm{poly}(n)$ comes with no loss of generality.

that $\ln(f\sqrt{\ln(1/\varepsilon)}) = O(\ln(n)^c)$ *for some* $c < 1$*. There exists an algorithm that solves* $\mathrm{SIS}^{\infty}_{n,m,q,\beta}$ *for* $\beta := \frac{q}{f}\sqrt{\ln m}$ *in expected time*

$$T = 2^{\frac{n/2}{\ln(\ln(q)) - \ln\left(\ln(f) + \frac{1}{2}\ln\ln(1/\varepsilon)\right) - O(1)}} \cdot \mathrm{poly}(m)$$

with success probability $1 - \frac{1}{\Omega(n)}$*.*

Proof. Apply Theorem 3.3 with input n, m, q, f, ε to the $\mathrm{SIS}^{\infty}_{n,m,q,\beta}$ instance $\mathbf{A}$. With probability at least $1 - 2^{-\tilde{\Omega}(n)}$, it returns a list of vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$ in $\Lambda_q^{\perp}(\mathbf{A})$ (so they are solutions to $\mathbf{A}\mathbf{x} = \mathbf{0} \bmod q$) that are conditionally $q^4\varepsilon$-similar to independent samples from $D_{\Lambda_q^{\perp}(\mathbf{A}),s}$ for $s := \frac{q}{f}$. In particular, it follows that the first vector $\mathbf{x}_1$ follows a distribution D that is $q^4\varepsilon$-similar to $D_{\Lambda_q^{\perp}(\mathbf{A}),s}$ (recall Remark 2.2). By Footnote 10 we have $\frac{q}{f} > \eta_{\varepsilon/4}(\Lambda_q^{\perp}(\mathbf{A}))$. We note that we may, without loss of generality, assume $\frac{q}{f} > 2\eta_{\varepsilon/4}(\Lambda_q^{\perp}(\mathbf{A}))$ by replacing the role of the constant 144 in the proof of Theorem 3.3 by a larger constant.

Since D is $q^4\varepsilon$-similar to $D_{\Lambda_q^{\perp}(\mathbf{A}),s}$, Lemma 2.2 (together with the fact that $e^{-x} \geq 1 - x$ for all $x \in \mathbb{R}$) implies

$$\begin{aligned}
\Pr[\|\mathbf{x}_1\|_\infty \leq \beta \wedge \mathbf{x}_1 \neq \mathbf{0}] &\geq e^{-q^4\varepsilon} \Pr_{X \sim D_{\Lambda_q^{\perp}(\mathbf{A}),s}}[\|X\|_\infty \leq \beta \wedge X \neq \mathbf{0}] \\
&\geq \Pr_{X \sim D_{\Lambda_q^{\perp}(\mathbf{A}),s}}[\|X\|_\infty \leq \beta \wedge X \neq \mathbf{0}] - q^4\varepsilon \\
&\geq \Pr_{X \sim D_{\Lambda_q^{\perp}(\mathbf{A}),s}}[\|X\|_\infty \leq \beta \wedge X \neq \mathbf{0}] - \frac{1}{m}.
\end{aligned}$$

We now show that $\Pr_{X \sim D_{\Lambda_q^{\perp}(\mathbf{A}),s}}[\|X\|_\infty \leq \beta \wedge X \neq \mathbf{0}] \geq 1 - \frac{2}{m^2} - \frac{1}{2^{m-1}}$, from which it follows that $\mathbf{x}_1$ is a solution to SIS^{∞} with probability at least $1 - \frac{2}{m^2} - \frac{1}{2^{m-1}} - \frac{1}{m} \geq 1 - \frac{3}{m} \geq 1 - \frac{3}{n}$ when $n \geq 2$ (where we use that $m \geq n$), thereby proving the theorem.

It remains to prove our claim. We have

$$\Pr_{X \sim D_{\Lambda_q^{\perp}(\mathbf{A}),s}}[\|X\|_\infty \leq \beta \wedge X \neq \mathbf{0}] \geq \Pr_{X \sim D_{\Lambda_q^{\perp}(\mathbf{A}),s}}[\|X\|_\infty \leq \beta] - \Pr_{X \sim D_{\Lambda_q^{\perp}(\mathbf{A}),s}}[X = \mathbf{0}].$$

Since $s = \frac{q}{f} > 2\eta_{\varepsilon/4}(\Lambda_q^{\perp}(\mathbf{A}))$, Lemma 2.5 implies $\Pr_{X \sim D_{\Lambda_q^{\perp}(\mathbf{A}),s}}[X = \mathbf{0}] \leq \frac{1+\varepsilon/4}{1-\varepsilon/4} \cdot 2^{-m} \leq \frac{1}{2^{m-1}}$ (as $\frac{1+x/4}{1-x/4} \leq 2$ for all $x \in [0,1]$). Also, Lemma 2.3 yields $\Pr_{X \sim D_{\Lambda_q^{\perp}(\mathbf{A}),s}}[\|X\|_\infty \leq \beta] > 1 - 2me^{-\pi(\frac{\beta f}{q})^2} \geq 1 - \frac{2}{m^2}$, where we use that $\beta = \frac{q}{f}\sqrt{\ln m}$. Our claim follows. $\square$

One may remark that our application of Lemma 2.3 makes us lose a $\sqrt{\ln m}$ factor on the norm bound to reach a constant success probability per sample. It would be tempting to only aim for a success probability barely greater than $1/N$ instead, however, the proof would then require $\varepsilon \approx 1/N$, which would make the algorithm exponential. This is admittedly a counterintuitive situation, and plausibly a proof artifact.

4.2 Implications for SIS$^{\times}$ and ISIS in ℓ_2 Norm

Regarding the ℓ_2-norm, for the same parameters as above, our Gaussian sampler outputs vectors of length less than $\beta = \sqrt{m} \cdot q/f$ for any $f = \text{polylog}(n)$ in subexponential time. However, SIS in ℓ_2-norm is trivial for such a bound; for example, $(q, 0, \ldots, 0)$ is a valid solution.

Yet, some schemes [ETWY22] have used the inhomogeneous version of SIS (ISIS) for bounds $\beta > q$, which was shown [DEP23] to be equivalent to solving SIS$^{\times}$, a variant of SIS where the solution must be nonzero modulo q. The work of [DEP23] notes that the problem becomes trivial when $\beta \geq q\sqrt{n/12}$ and proposes a heuristic attack that is better than pure lattice reduction when $\beta > q$; however, it appears to run in exponential time in n for $\beta = q\sqrt{n}/\text{polylog}(n)$. Our Gaussian sampler directly yields a provably subexponential-time algorithm in that regime.

Theorem 4.2. *For $n \in \mathbb{N}$, let $m = n + \omega(n/\ln \ln n)$ be integer and $q = n^{\Theta(1)}$ be prime such that $q^{1-n/m} \geq 6$. Let $f > 1$ and $\varepsilon \leq \frac{1}{mq^4}$ be positive reals such that $\ln(f\sqrt{\ln(1/\varepsilon)}) = O(\ln(n)^c)$ for some $c < 1$. There exists an algorithm that solves $\text{SIS}^{\times}_{n,m,q,\beta}$ and $\text{ISIS}_{n,m,q,\beta}$ for $\beta := \frac{q}{f}\sqrt{m}$ in expected time*

$$T = 2^{\frac{n/2}{\ln(\ln(q)) - \ln\left(\ln(f) + \frac{1}{2}\ln\ln(1/\varepsilon)\right) - O(1)}} \cdot \text{poly}(m)$$

with success probability $1 - \frac{1}{\Omega(n)}$.

The proof is essentially equivalent to that of Theorem 4.1, up to the invocation of Lemma 2.3 used to tail-bound the norm of a discrete Gaussian, which should be replaced by a similar tail-bound for the ℓ_2-norm [Ban93, Lemma 1.5].

Note again here that one may always choose $m = n(1 + o(1))$ without loss of generality even when the given m is much larger, simply by ignoring some SIS variables. Hence, reaching the norm bound $\beta = q\sqrt{m}/\text{polylog}(n)$ also permits to reach $\beta = q\sqrt{n}/\text{polylog}(n)$.

4.3 Potential Implications for LWE

Having obtained a discrete Gaussian sampler for SIS lattices, one may be tempted to apply the dual distinguisher of [AR05] and directly obtain a subexponential-time algorithm for LWE with narrow secrets. This is in fact problematic as such a distinguisher needs to consider subexponentially many samples simultaneously, so some naive reasoning using the data-processing inequality would force one to instantiate our Gaussian sampler with $\varepsilon = 2^{-\tilde{\Omega}(n)}$ rather than $\varepsilon = n^{-\Theta(1)}$. Unfortunately, for such small ε our Gaussian sampler has exponential complexity.

This issue resonates with the one raised in [HKM18] regarding the proof of [KF15] when the parameter m is linear in n; namely, their issue is about reaching exponentially small statistical distance. However, it is technically different:

in our case, it cannot be fixed by increasing m to $\Theta(n \log n)$. Tracking down the limiting factor leads to blaming the poor smoothing parameters of $\frac{q}{p_i}\mathbb{Z}^{b_i}$. Thanks to the generality of our framework, it is plausible that this superlattice of $q\mathbb{Z}^{b_i}$ can be replaced by one of similar index with a much better smoothing parameter. We hope that future work will finally be able to provably fix the claim of [KF15] for m linear in n.

Acknowledgments. The authors are grateful to Ronald de Wolf and the anonymous reviewers for their valuable comments on the manuscript. LD and JL were supported by the ERC Starting Grant 947821 (ARTICULATE). LE was supported by the Dutch National Growth Fund (NGF), as part of the Quantum Delta NL program.

References

ACKS21. Aggarwal, D., Chen, Y., Kumar, R., Shen, Y.: Improved (provable) algorithms for the shortest vector problem via bounded distance decoding. In: 38th International Symposium on Theoretical Aspects of Computer Science (STACS 2021), volume 187 of *LIPIcs*, pp. 4:1–4:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021)

ADRS15. 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, pp. 733–742. ACM (2015)

ADS15. Aggarwal, D., Dadush, D., Stephens-Davidowitz, N.: Solving the closest vector problem in 2^n time - the discrete Gaussian strikes again! In: 2015 IEEE 56th Annual Symposium on Foundations of Computer Science, pp. 563–582. IEEE Computer Society (2015)

AFFP14. Albrecht, M.R., Faugère, J.-C., Fitzpatrick, R., Perret, L.: Lazy modulus switching for the BKW algorithm on LWE. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 429–445. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54631-0_25

Ajt96. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, STOC 1996, pp. 99–108. Association for Computing Machinery (1996)

ALS21. Aggarwal, D., Li, Z., Stephens-Davidowitz, N.: A $2^{n/2}$-time algorithm for $\sqrt{n}$-SVP and $\sqrt{n}$-hermite SVP, and an improved time-approximation tradeoff for (H)SVP. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12696, pp. 467–497. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77870-5_17

AR05. Aharonov, D., Regev, O.: Lattice problems in NP $\cap$ coNP. J. ACM (JACM) **52**(5), 749–765 (2005)

AS18. Aggarwal, D., Stephens-Davidowitz, N.: Just take the average! An embarrassingly simple 2^n-time algorithm for SVP (and CVP). In: 1st Symposium on Simplicity in Algorithms (SOSA 2018), volume 61 of *Open Access Series in Informatics*, pp. 12:1–12:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2018)

Ban93. Banaszczyk, W.: New bounds in some transference theorems in the geometry of numbers. Math. Ann. **296**, 625–635 (1993)

Ban95. Banaszczyk, W.: Inequalites for convex bodies and polar reciprocal lattices in $\mathbb{R}^n$. Discrete Comput. Geom. **13**, 217–231 (1995)

BKW03. Blum, A., Kalai, A., Wasserman, H.: Noise-tolerant learning, the parity problem, and the statistical query model. J. ACM (JACM) **50**(4), 506–519 (2003)

BLP+13. Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing, STOC 2013, pp. 575–584. ACM (2013)

DEL25. Ducas, L., Engelberts, L., Loyer, J.: Wagner's algorithm provably runs in subexponential time for SIS^∞. Cryptology ePrint Archive, Paper 2025/575 (2025). Full version

DEP23. Ducas, L., Espitau, T., Postlethwaite, E.W.: Finding short integer solutions when the modulus is small. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023. LNCS, vol. 14083, pp. 150–176. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-38548-3_6

DKL+18. Ducas, L., et al.: CRYSTALS-Dilithium: a lattice-based digital signature scheme. IACR Trans. Cryptogr. Hardw. Embed. Syst., 238–268 (2018)

ETWY22. Espitau, T., Tibouchi, M., Wallet, A., Yu, Y.: Shorter hash-and-sign lattice-based signatures. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022. LNCS, vol. 13508, pp. 245–275. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15979-4_9

EWY23. Espitau, T., Wallet, A., Yang, Yu.: On Gaussian sampling, smoothing parameter and application to signatures. In: Guo, J., Steinfeld, R. (eds.) ASIACRYPT 2023. LNCS, vol. 14444, pp. 65–97. Springer, Singapore (2023). https://doi.org/10.1007/978-981-99-8739-9_3

GJMS17. Guo, Q., Johansson, T., Mårtensson, E., Stankovski, P.: Coded-BKW with sieving. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017. LNCS, vol. 10624, pp. 323–346. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70694-8_12

GPV08. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, STOC 2008, pp. 197–206. ACM (2008)

HKM18. Herold, G., Kirshanova, E., May, A.: On the asymptotic complexity of solving LWE. Des. Codes Crypt. **86**(1), 55–83 (2018)

KF15. Kirchner, P., Fouque, P.-A.: An improved BKW algorithm for LWE with applications to cryptography and lattices. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 43–62. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47989-6_3

Lyu05. Lyubashevsky, V.: The parity problem in the presence of noise, decoding random linear codes, and the subset sum problem. In: Chekuri, C., Jansen, K., Rolim, J.D.P., Trevisan, L. (eds.) APPROX/RANDOM 2005. LNCS, vol. 3624, pp. 378–389. Springer, Heidelberg (2005). https://doi.org/10.1007/11538462_32

MP13. Micciancio, D., Peikert, C.: Hardness of SIS and LWE with small parameters. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 21–39. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_2

MR07. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on Gaussian measures. SIAM J. Comput. **37**(1), 267–302 (2007)

Pei08. Peikert, C.: Limits on the hardness of lattice problems in l_p norms. Comput. Complex. **17**(2), 300–351 (2008)

Pei10. Peikert, C.: An efficient and parallel gaussian sampler for lattices. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 80–97. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14623-7_5

PR06. Peikert, C., Rosen, A.: Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 145–166. Springer, Heidelberg (2006). https://doi.org/10.1007/11681878_8

Reg05. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing, STOC 2005, pp. 84–93. ACM (2005)

Wag02. Wagner, D.: A generalized birthday problem. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 288–304. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_19