On Gaussian Sampling for q-ary Lattices and Linear Codes with Lee Weight

Maiara F. Bollauf[1], Maja Lie[2], and Cong Ling[2]

[1] Simula UiB, N-5006 Bergen, Norway
maiarabollauf@gmail.com
[2] Imperial College London, London SW7 2AZ, England
{m.lie22, c.ling}@imperial.ac.uk

Abstract. We show that discrete Gaussian sampling for a q-ary lattice is equivalent to codeword sampling for a linear code over $\mathbb{Z}_q$ with the Lee weight. This insight allows us to derive the theta series of a q-ary lattice from the Lee weight distribution of the associated code.We design a novel Gaussian sampler for q-ary lattices assuming an oracle that computes the symmetrized weight enumerator of the associated code. We apply this sampler to well-known lattices, such as the E_8, Barnes-Wall, and Leech lattice, highlighting both its advantages and limitations, which depend on the underlying code properties. For certain root lattices, we show that the sampler is indeed efficient, forgoing the need to assume an oracle. We also discuss applications of our results in digital signature schemes and the Lattice Isomorphism Problem. In many cases, our sampler achieves a significant speed-up compared to state-of-the-art sampling algorithms in cryptographic applications.

Keywords: Lattice Gaussian sampling · Lee weight · q-ary lattice · Schur product · Theta series.

1 Introduction

Lattice Gaussian sampling refers to the process of sampling vectors from a discrete Gaussian distribution with a specified width s, defined over a lattice Λ. This technique plays a fundamental role in lattice-based cryptography. Namely, Gaussian sampling over lattices with the appropriate choice of width s enables the solution of the *closest vector problem* (CVP) and the *shortest vector problem* (SVP) [2, 3] and it is also one of the main tools used in worst-to-average case reductions for lattice problems [26]. Additionally, it is being used in the construction of several lattice-based cryptographic protocols, such as signature schemes, as it can be used to hide information about the lattice basis [16].

The width s of a Gaussian sampler over a lattice determines how wide the distribution we sample from is. It is challenging for most lattices to sample over discrete Gaussian distributions when s is not so large. In practical terms, the security of the SVP or approximate SVP relies on the fact that it should be

difficult for an adversary to obtain a lattice point within a radius $s\sqrt{n}$, where n refers to the lattice dimension [30].

Micciancio and Regev [26] define the *smoothing parameter* of a lattice, which is the minimum amount of Gaussian noise that, when added to a lattice, produces a distribution close to uniform over $\mathbb{R}^n/\Lambda$. Klein's algorithm can be used to sample from an arbitrary lattice for s sufficiently large, but s may be well above the smoothing parameter [7]. Recently, Ducas and van Woerden [12] introduced a hash-then-sign signature scheme based on the *Lattice Isomorphism Problem* (LIP), which can be implemented with any lattice that has an efficient Gaussian sampler with small sampling widths. Tight sampling for lattices would enhance the security of the LIP-based signature scheme, as it would be able to withstand attacks down to small approximation factors [12].

Lattice-based cryptographic protocols typically require sampling at or above the smoothing parameter. The smoothing parameter is defined in terms of the *theta series*, a lattice geometric invariant. Roughly, the theta series of a lattice Λ characterizes the number of points in Λ with a given (Euclidean) norm. Unfortunately, for a general lattice, we may not know its theta series and, therefore, do not have a tight estimation for the smoothing parameter. As a consequence, when sampling from a general lattice, the width s may significantly exceed the smoothing parameter when, ideally, it should be as close to the smoothing parameter as possible to achieve improved security.

Gaussian sampling over $\mathbb{Z}$ for a fixed width s is usually efficient since it can use precomputed data [28,30], while for general lattices, the complexity increases. Therefore, one can sample efficiently over the $\mathbb{Z}^n$ lattice utilizing one-dimensional samplers of $\mathbb{Z}$ [7, Sec. 5.1]. One-dimensional samplers of $\mathbb{Z}$ and its shifts are often used as subroutines in samplers for other lattices as well.

Lattice-based cryptography typically uses *q-ary lattices*, which only involves integer arithmetic modulo q. Λ is called a q-ary lattice if $q\mathbb{Z}^n \subseteq \Lambda \subseteq \mathbb{Z}^n$. Such a lattice can be expressed in the form $\mathcal{C} + q\mathbb{Z}^n$, where $\mathcal{C}$ is a linear code in $\mathbb{Z}_q^n$. In lattice theory, lattices obtained via error-correcting codes are denoted as Constructions A, B, C, D, and E, where each letter indicates the linear codes that are being employed (see [11, Ch. 5]). We will extensively work with Construction A obtained from a single linear q-ary code $\mathcal{C} \subseteq \mathbb{Z}_q^n$, Construction B from two nested binary codes, and Construction D from several nested binary codes.

Ling *et al.* proposed a lattice Gaussian sampler using coset decomposition for large enough s [21, 22]. In particular, [21] presented a sampler for Construction A in which codewords are sampled uniformly at random and proved that the resultant distribution is close to a lattice Gaussian if s exceeds the smoothing parameter of $q\mathbb{Z}$ [3]; it exemplified the method by considering the checkerboard lattice D_n and Gosset lattice E_8 and remarked that the method can be extended to Construction D. Campello and Belfiore [8] proposed an efficient method for sampling from the 2-ary Construction A and 4-ary Construction B lattices for any width s. These constructions include the D_n, and the Barnes-Wall lattices

[3] In fact, [21] addressed a general version of Construction A where the quotient $\mathbb{Z}/q\mathbb{Z}$ is replaced with Λ_1/Λ_2 for a pair of lattices $\Lambda_2 \subseteq \Lambda_1$.

E_8 and BW_{16} (they also considered sampling from A_2 and the Leech lattice using coset decomposition). Their method relies on the relationship between the theta series of the lattice and the Hamming weight enumerator of the underlying binary linear code. In [14], Espitau *et al.* gave a general framework for efficient Gaussian sampling over lattices using extensions of lattices and new bounds on the smoothing parameter from lattice filtrations, assuming s is large enough. The authors used their framework to build efficient samplers for the $\mathrm{A}_n, \mathrm{D}_n$, and E_n lattices as base cases. Their method can also be used to sample from the Barnes-Wall lattices BW_{2^n} again assuming s is large enough.

1.1 Our Contribution

A smaller width s leads to tighter security arguments, so ideally we want to sample as close as possible to the smoothing parameter of Λ. This paper proposes a new unrestricted sampler for q-ary lattices. By using the code formula of the q-ary lattice, we simplify the theta series calculation for some families of q-ary lattices and express the theta series in terms of the Lee weight enumerator of the underlying q-ary codes. We extend these former results to q-ary lattices obtained via Construction A for $q \geq 2$ and Construction D. Our method requires computing the Lee weight enumerator of some code, and this computation does not depend on the choice of sampling width s. As a consequence, we improve the efficiency of current state-of-the-art sampling for root lattices. For Construction D, we build upon [8, 14, 21] by leveraging lattice filtrations in such a way that we can apply the samplers recursively. More specifically, our contributions are three-fold:

1. On the reduction front, we present polynomial-time reductions between Gaussian sampling on q-ary lattices and sampling codewords of a linear code with respect to the Lee weight profile, thereby establishing equivalence of the two problems. Recently, linear codes in the Lee metric have received attention in code-based cryptography. It is known that (the decision version of) the problem of finding codewords with a given Lee weight is NP-complete [37]. In practice, this Lee weight is chosen to be on the Lee-metric GV bound [33]. Since finding codewords with a specified Lee weight profile is at least as hard, this implies that Gaussian sampling for q-ary lattices is, in general, a computationally hard problem.
2. On the algorithmic front, we use the aforementioned reduction to design a new sampler in which one can sample with an arbitrary width s. For the q-ary ($q = 2$ and $q = 4$, respectively) lattices obtained via Construction A and B from binary codes, our sampler coincides with the method of coset decomposition explored in [8]. Then, we expand the sampling via cosets technique to another q-ary lattice family, Construction D, and show that it depends on the Lee weight distribution of the underlying codes in different levels of the filtration.
3. Our proposed sampler, when applied to root lattices and small dimensional q-ary lattices, represents an improvement compared to [14, 21], which restricts

s according to the maximum smoothing parameter of the lattice filtration. Comparison between our sampling procedure and the one presented in [14] is performed for known families of lattices (selected results are given in Table 1). The complexity, limitations, and advantages in the cryptographic context are also addressed.

	Complexity	**Sampling Width s**	
Lattice	Speed-up upper bound	[14]	This work
A_2	18×	$\approx \eta_\epsilon(\mathrm{A}_2)$	$= \eta_\epsilon(\mathrm{A}_2)$
E_8	22×	$\approx \eta_\epsilon(\mathrm{E}_8)$	$= \eta_\epsilon(\mathrm{E}_8)$
D_n	2×	$\approx \eta_\epsilon(\mathrm{D}_n)$	$= \eta_\epsilon(\mathrm{D}_n)$
Λ_{24}	2×	$\approx \eta_\epsilon(\mathrm{E}_8)$	$= \eta_\epsilon(\Lambda_{24})$
BW_n	22×	$> \eta_\epsilon(\mathrm{BW}_{n/2})$	$= \eta_\epsilon(\mathrm{BW}_n)$

Table 1: Comparison of complexity and sampling width. The upper bound on speed-up is obtained by counting the number of calls to a $\mathbb{Z}$ sampler, ignoring the overhead of codeword sampling. Note that upper bounds on speed-up for BW_n and Λ_{24} are rather conservative since the sampling widths are different. See Section 7 for details.

In summary, our Gaussian sampler can output a lattice vector for arbitrary sampling width s for any q-ary lattice Λ, since any such lattice has a code formula $\Lambda = q\mathbb{Z}^n + \mathcal{C}$, where $\mathcal{C}$ is a linear code over $\mathbb{Z}_q^n$ [27]. We note that our sampling method is limited by the need to compute the Lee weight profiles of a potentially very large code (or its cosets). For random codes, this may not be known or efficient to compute. The *Schur* (or element-wise) product can be used to compute Lee weight profiles offline.

In this paper, we start with a theoretical and general q-ary sampler and apply our technique to known families of lattices to illustrate the improvements. Section 2 provides some preliminaries on codes and lattices; in Section 3, we prove the equivalence of lattice and code sampling, and in Section 4 we derive the theta series of a q-ary lattice from the symmetrized weight enumerator of the associated code(s). In Section 5, we present our samplers for Constructions A, B, and D lattices and employ the technique to remarkable lattices in Section 6. Comparisons and improvements with respect to the state-of-the-art are in Section 7.

1.2 Techniques

Lee weight. Here we give an intuition why the somewhat mysterious Lee weight comes into play, using the trivial example of Gaussian sampling from the integers

$\mathbb{Z}$. Sampling an integer $y \in \mathbb{Z}$ may be realized in two steps: firstly, we sample $c = y \mod q$ from $\mathbb{Z}_q = \{0, 1, \ldots, q-1\}$ for $q \in \mathbb{N}$; secondly, we sample an integer from the coset $q\mathbb{Z} + c$. Now we focus on the first step, assuming the second step is efficient. The probability of sampling a coset $q\mathbb{Z} + c$ is given by the theta function $\Theta_{q\mathbb{Z}+c}(z)$ for some z defined properly.

The Lee weight of $c \in \mathbb{Z}_q$ is defined as $w_{\text{Lee}}(c) = \min\{c, q-c\}$. It resembles the Euclidean distance (which is relevant for lattices) more than the conventional Hamming distance. The Gaussian function is even symmetric so that $\Theta_{q\mathbb{Z}+c}(z) = \Theta_{q\mathbb{Z}+q-c}(z)$. This coincides with the fact that c and $-c = q - c$ in $\mathbb{Z}_q$ have the same Lee weight. Therefore, we only need the knowledge of the Lee weight, i.e., we sample coset $q\mathbb{Z} + c$ with probability $\Theta_{q\mathbb{Z}+w_{\text{Lee}}(c)}(z)$.

Our Gaussian sampling method for q-ary lattices generalizes the above idea, taking into account the underlying codes. For Construction D with multiple levels, we run recursion on the second step.

The use of the Lee norm in [34] is relevant to lattices but in a different context.

Theta series. Some families of lattices have well-documented theta series in the literature, such as the Constructions A and B lattices from binary codes. We extend it to a general Construction A lattice $q\mathbb{Z}^n + \mathcal{C}$ with $q \geq 2$ and connect with the symmetrized weight enumerator of the respective code over $\mathbb{Z}_q$. Moving forward, we consider a L-level Construction D lattice with code formula [15,20][4]:

$$\Lambda_{\text{D}} = 2^L\mathbb{Z}^n + 2^{L-1}\mathcal{C}_L + \ldots + 2\mathcal{C}_2 + \mathcal{C}_1, \tag{1}$$

where $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_L$ are binary linear codes that are *closed under Schur product* (more details follow below). In general, for $L \geq 3$, the theta series of Λ_{D} is unknown. Nonetheless, by performing case analysis, we can obtain a general form for the theta series of a 3-level Construction D lattice. Thus, the probability of a coset of such a lattice depends on the Lee weight of a linear code over $\mathbb{Z}_8$ defined by $4\mathcal{C}_3 + 2\mathcal{C}_2 + \mathcal{C}_1$. The same analysis shows that the probability of a coset of an L-level Construction D lattice depends on the Lee weight of a code over $\mathbb{Z}_{2^L}$. This is a generalization of the fact that the coset of Constructions A and B lattices depend only on the Hamming weight of a code over $\mathbb{F}_2$ [8].

The Lee weight distribution of a code allows us to sample a coset representative of the lattice. In (1), we consider a filtration $\Lambda_1 \subseteq \Lambda_2 \subseteq \ldots \subseteq \Lambda_L$ where $\Lambda_1 = 2^L\mathbb{Z}^n + 2^{L-1}\mathcal{C}_L$ is a scaled Construction A lattice, $\Lambda_2 = 2^L\mathbb{Z}^n + 2^{L-1}\mathcal{C}_L + 2^{L-2}\mathcal{C}_{L-1}$, and we proceed recursively such that in each subsequent lattice we add the next (scaled) code until we get Λ_L as the whole lattice Λ_{D}. Let $\mathbf{c}$ be a coset representative. In the end, the sampler outputs a vector from $\mathcal{D}_{2^L\mathbb{Z}^n+\mathbf{c},s}$ using one-dimensional samplers of $\mathbb{Z}$ and its shifts. Recursive application of sampling via coset decomposition using Lee weight enables exact sampling of a Construction D lattice for any width $s > 0$.

[4] Note that this is a subclass of Construction D since not all Construction D lattices admit a code formula. See [11, p. 232] for the general definition.

1.3 Open Questions

Lattice and code-based cryptography are closely related. Lately, techniques from lattice-based cryptography were used to break FuLeeca [18], the first digital signature scheme based on the Lee metric. Meanwhile, the new connection discovered in this work shows that code sampling in the Lee metric and lattice Gaussian sampling are equally hard. To the best of our knowledge, no such relation has been shown before. It is an interesting open question whether code sampling in the Lee metric can be used to construct provably secure cryptosystems[5].

Our sampling algorithms rely on knowledge of the Lee weight profile or symmetrized weight enumerator of a code or its coset, but existing methods to compute these are computationally expensive. Enhancing their efficiency would result in faster lattice Gaussian samplers. While we have made some progress in this paper using the Schur product, further exploration is left for future work.

In [12], Ducas and van Woerden hypothesize that instantiating the LIP signature scheme with remarkably decodable lattices may lead to resistance against attacks down to smaller approximation factors. The Barnes-Wall lattices are remarkable, which makes it desirable to have a sampler for these lattices with a small sampling width s. With our sampler, we can set s just above the smoothing parameter, but in practice, sampling requires us to sample codewords with respect to Lee weight (a hard problem in general). As a consequence, high dimensional Barnes-Wall lattices remain a challenge for tight sampling with our method, and in this case, [14] provide wider and more efficient samplers. We leave it to future work to optimise our samplers for Barnes-Wall lattices and to analyse the security gains of instantiating LIP with these remarkable lattices.

2 Preliminaries

2.1 Notation

We denote by $\mathbb{N}$, $\mathbb{Z}$, and $\mathbb{R}$ the set of naturals, integers, and reals, respectively. $[a : b] \triangleq \{a, a+1, \ldots, b\}$ for $a, b \in \mathbb{Z}$, $a \leq b$. The ring of integers modulo q for $q \in \mathbb{N}$ is $\mathbb{Z}_q = \{0, 1, \ldots, q-1\}$. Vectors are boldfaced, e.g., $\mathbf{x}$ and matrices are represented by capital sans serif letters, e.g. X. The symbol $+$ represents the element-wise addition over $\mathbb{R}$ and $\star$ denotes the Schur product between two elements in $\mathbb{F}_2^n$, i.e., $\mathbf{x} \star \mathbf{y} = (x_1 y_1, \ldots, x_n y_n)$, for $\mathbf{x}, \mathbf{y} \in \mathbb{F}_2^n$.

2.2 Linear Codes

The definitions and properties presented here are based on [23, 36].

A (q-ary) *linear code* $\mathcal{C}$ of length n over a finite field $\mathbb{F}_q$ (q is a prime power) is a linear subspace $\mathcal{C} \subseteq \mathbb{F}_q^n$. When q is not prime, we can also define linear codes over rings simply as additive subgroups of $\mathbb{Z}_q^n$. Throughout this paper, q will be

[5] In fact, as noted in [18], an open question remains on how to adapt the GPV framework [16] to the Lee metric.

a prime or a power of prime. We call a linear code $\mathcal{C}$ an $(n,k)_q$ code, where n is the length of the code and k is the dimension.

We consider two different notions of weight: Hamming weight and Lee weight. The *Hamming weight* of a codeword $\mathbf{c} \in \mathcal{C}$, denoted by $w_{\mathrm{H}}(\mathbf{c})$, is defined as the number of its nonzero coordinates. We will initially restrict our attention to binary codes. The Hamming weight enumerator of a code is a function that describes the weight profile of a binary code $\mathcal{C}$ in terms of the Hamming weight.

Definition 1 (Hamming Weight Enumerator). *Let $\mathcal{C}$ be a $(n,k)_2$ code. The Hamming weight enumerator of $\mathcal{C}$ is given by*

$$W_{\mathcal{C}}(x,y) = \sum_{\mathbf{c}\in\mathcal{C}} x^{n-w_{\mathrm{H}}(\mathbf{c})} y^{w_{\mathrm{H}}(\mathbf{c})} = \sum_{w=0}^{n} A_w x^{n-w} y^w$$

where $A_w(\mathcal{C}) = \#\{\mathbf{c} \in \mathcal{C} \colon w_{\mathrm{H}}(\mathbf{c}) = w\}$, with $0 \le w \le n$, is the number of codewords in $\mathcal{C}$ that have Hamming weight w_{H}.

For codes over $\mathbb{Z}_q$, we define the Lee weight, which provides a more granular description of the nonzero coordinates of a codeword.

Definition 2 (Lee Weight). *The Lee weight of $c \in \mathbb{Z}_q$ is $w_{\mathrm{Lee}}(c) = \min\{c, q-c\}$. This can be naturally extended to a vector, i.e., given $\mathbf{c} = (c_1, c_2, \ldots, c_n) \in \mathbb{Z}_q^n$, its Lee weight is $w_{\mathrm{Lee}}(\mathbf{c}) = \sum_{j=1}^{n} w_{\mathrm{Lee}}(c_j)$.*

When $q = 2, 3$, Lee weight coincides exactly with Hamming weight. The notion of Lee weight allows us to define additional weight enumerators.

Definition 3 (Symmetrized Weight Enumerator). *The symmetrized weight enumerator of a code $\mathcal{C}$ over $\mathbb{Z}_q$ is given by*

$$\mathrm{swe}_{\mathcal{C}}(x_0, x_1, \ldots, x_\ell) = \sum_{\mathbf{c}\in\mathcal{C}} x_0^{n_0(\mathbf{c})} x_1^{n_1(\mathbf{c})} \ldots x_{\ell-1}^{n_{\ell-1}(\mathbf{c})} x_\ell^{n_\ell(\mathbf{c})}$$

where $n_w(\mathbf{c}) = \#\{i \colon w_{\mathrm{Lee}}(c_i) = w\}$, with $0 \le w \le \lceil q/2 \rceil$, i.e., it refers to the number of coordinates of $\mathbf{c}$ that are $\pm w$ and $\ell = \lceil q/2 \rceil$.

Definition 4 (Lee Weight Profile). *Let $\mathcal{C}$ be a linear code over $\mathbb{Z}_q$. The Lee weight profile of $\mathbf{c} \in \mathcal{C}$ is the tuple*

$$[n_0(\mathbf{c}), n_1(\mathbf{c}), \ldots, n_\ell(\mathbf{c})], \quad \ell = \lceil q/2 \rceil .$$

Consider $\boldsymbol{\alpha} \in \mathbb{F}_2^n$. Then $\omega_{\boldsymbol{\alpha}}(\mathbf{c}_1, \ldots, \mathbf{c}_r)$ as the number of occurrences of $\boldsymbol{\alpha}$ as a row in the matrix of column vectors $\mathbf{c}_1, \ldots, \mathbf{c}_r$. For example, for the matrix with column vectors given by $\mathbf{c}_1 = (1,0,0), \mathbf{c}_2 = (1,1,0)$, and $\mathbf{c}_3 = (1,1,1)$, we have that $\omega_{1,1,1}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3) = 1$ and $\omega_{1,0,0}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3) = 0$.

Over multiple binary codes, we can define a generalized version of the *joint weight enumerator* or originally, *j-fold weight enumerator* as follows.

Definition 5 (Joint Weight Enumerator [13]**).** *Consider j binary codes $\mathcal{C}_1, \ldots, \mathcal{C}_j \subseteq \mathbb{F}_2^n$. We define their joint weight enumerator to be*

$$\mathrm{jwe}_{\mathcal{C}_1,\ldots,\mathcal{C}_j}(x) = \sum_{\mathbf{c}_1\in\mathcal{C}_1} \cdots \sum_{\mathbf{c}_j\in\mathcal{C}_j} \prod_{\boldsymbol{\alpha}\in\mathbb{F}_2^j} x_{\boldsymbol{\alpha}}^{\omega_{\boldsymbol{\alpha}}(\mathbf{c}_1,\ldots,\mathbf{c}_j)} = \sum_{\substack{(\mathbf{c}_1,\ldots,\mathbf{c}_j)\\ \in\mathcal{C}_1\times\ldots\times\mathcal{C}_j}} \prod_{\boldsymbol{\alpha}\in\mathbb{F}_2^j} x_{\boldsymbol{\alpha}}^{\omega_{\boldsymbol{\alpha}}(\mathbf{c}_1,\ldots,\mathbf{c}_j)},$$

where $x = (x_{\boldsymbol{\alpha}})$ is a 2^j-tuple of variables with $\boldsymbol{\alpha} \in \mathbb{F}_2^j$.

Notice that $\mathrm{jwe}_{\mathcal{C}_1}(x_0, x_1)$ is the ordinary Hamming weight enumerator of a binary code $\mathcal{C}_1$ and $\mathrm{jwe}_{\mathcal{C}_1,\mathcal{C}_2}(x) = \mathrm{jwe}_{\mathcal{C}_1,\mathcal{C}_2}(x_{00}, x_{01}, x_{10}, x_{11})$ is the biweight enumerator of two binary codes $\mathcal{C}_1$ and $\mathcal{C}_2$ [23, pp. 147-148].

For codes $\mathcal{C} \subseteq \mathbb{Z}_4^n$ constructed via $\mathcal{C} = \mathcal{C}_1 + 2\mathcal{C}_2$, where $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathbb{Z}_2^n$ are linear codes, there is a natural relationship between the joint and the symmetrized weight enumerators, namely $\mathrm{jwe}_{\mathcal{C}_1,\mathcal{C}_2}(x_0, x_2, x_1, x_1) = \mathrm{swe}_{\mathcal{C}}(x_0, x_1, x_2)$. Indeed,

$$\omega_{0,0}(\mathbf{c}_1, \mathbf{c}_2) = n_0(\mathbf{c}), \quad \omega_{1,0}(\mathbf{c}_1, \mathbf{c}_2) + \omega_{1,1}(\mathbf{c}_1, \mathbf{c}_2) = n_1(\mathbf{c}), \quad \omega_{0,1}(\mathbf{c}_1, \mathbf{c}_2) = n_2(\mathbf{c}),$$

where we use the fact that the Lee weight of 1 and 3 in $\mathbb{Z}_4$ is the same (i.e., $3 \equiv -1 \bmod 4$), so we get

$$\begin{aligned}\mathrm{swe}_{\mathcal{C}}(x_0, x_1, x_2) &= \sum_{\mathbf{c}\in\mathcal{C}} x_0^{n_0(\mathbf{c})} x_1^{n_1(\mathbf{c})} x_2^{n_2(\mathbf{c})} \\ &= \sum_{\mathbf{c}_1\in\mathcal{C}_1} \sum_{\mathbf{c}_2\in\mathcal{C}_2} x_0^{\omega_{0,0}(\mathbf{c}_1,\mathbf{c}_2)} x_1^{\omega_{1,0}(\mathbf{c}_1,\mathbf{c}_2)+\omega_{1,1}(\mathbf{c}_1,\mathbf{c}_2)} x_2^{\omega_{0,1}(\mathbf{c}_1,\mathbf{c}_2)} \\ &= \mathrm{jwe}_{\mathcal{C}_1,\mathcal{C}_2}(x_0, x_2, x_1, x_1). \end{aligned} \tag{2}$$

From now on we will denote $\omega_{\alpha_1,\ldots,\alpha_j}(\mathbf{c}_1, \ldots, \mathbf{c}_j)$ simply by $\omega_{\alpha_1,\ldots,\alpha_j}$, when the vectors $\mathbf{c}_1, \ldots, \mathbf{c}_j$ are clear from the context.

Next, we present the Reed-Muller codes, which are crucial to the construction of Barnes-Wall lattices.

Definition 6 (Reed-Muller Code). *A Reed-Muller code $\mathrm{RM}(r, m)$ of order r and length 2^m is a binary code defined as*

$$\mathrm{RM}(r, m) = \begin{cases} \mathbb{F}_2^{2^r}, & m = r \\ \{(\mathbf{u}, \mathbf{u}+\mathbf{v}) : \mathbf{u} \in \mathrm{RM}(r, m-1), \mathbf{v} \in \mathrm{RM}(r-1, m-1)\}, & m > r. \end{cases}$$

The $(\mathbf{u}, \mathbf{u}+\mathbf{v})$-construction where $\mathbf{u} \in \mathcal{C}, \mathbf{v} \in \mathcal{C}'$ is called the *Plotkin construction*. A Reed-Muller code $\mathrm{RM}(r, m)$ has dimension $k = \sum_{j\le r} \binom{m}{j}$ and cardinality 2^k. Examples of Reed-Muller codes are: the universe codes $\mathrm{RM}(m, m)$, the repetition codes $\mathrm{RM}(0, m)$, and the parity-check codes $\mathrm{RM}(1, m)$.

2.3 Lattices

A *real lattice* Λ is a discrete additive subgroup of $\mathbb{R}^n$ [11]. The *dual lattice* is $\Lambda^* = \{\mathbf{x} \in \mathbb{R}^n : \forall \mathbf{y} \in \Lambda, \langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z}\}$. The *theta series* of a lattice is a series whose coefficients are equal to the number of lattice points of a given norm. It is the lattice analog of the weight enumerator of a linear code.

Definition 7 (Theta Series). *For any lattice Λ, its theta series is given by*

$$\Theta_{\Lambda}(z) = \sum_{\mathbf{x}\in\Lambda} e^{\pi i z\|\mathbf{x}\|^2},$$

where $\mathrm{Im}(z) > 0$.

In particular, when z is purely imaginary, i.e., $z = i\tau$ and $\tau > 0$

$$\Theta_{\Lambda}(i\tau) = \sum_{\mathbf{x}\in\Lambda} e^{-\pi\tau\|\mathbf{x}\|^2}. \tag{3}$$

The theta series $\Theta_{\Lambda+\mathbf{t}}(z)$ for a coset $\Lambda + \mathbf{t}$ where $\mathbf{t} \in \mathbb{R}^n$ is defined analogously.

It is often relevant to express the theta series of a lattice in terms of the Jacobi theta functions [11, pp. 102-105]:

$$\vartheta_3(\xi|z) = \sum_{m\in\mathbb{Z}} e^{2mi\xi+\pi i z m^2},\ \vartheta_2(z) = \sum_{m\in\mathbb{Z}} e^{\pi i z(m+1/2)^2},$$
$$\vartheta_3(z) = \vartheta_3(0|z) = \sum_{m\in\mathbb{Z}} e^{\pi i z m^2},\ \vartheta_4(z) = \sum_{m\in\mathbb{Z}} (-1)^m e^{\pi i z m^2},$$

where $\mathrm{Im}(z) > 0$ and $\xi \in \mathbb{C}$. Observe that $\vartheta_3(z)$ coincides with the theta series of the one-dimensional lattice $\mathbb{Z}$ and $\vartheta_2(z)$ refers to the shift $\mathbb{Z} + 1/2$.

The function $\psi_k(z) = \Theta_{\mathbb{Z}+1/k}(z) = \sum_{m\in\mathbb{Z}} e^{\pi i z(m+1/k)^2}$ for $k \in \mathbb{Z}\setminus\{0\}$ is also useful in our context. Note that

$$\Theta_{\mathbb{Z}+1/k}(z) = \Theta_{\mathbb{Z}+(k-1)/k}(z) \tag{4}$$

because $\Theta_{\mathbb{Z}+(k-1)/k}(z) = \Theta_{\mathbb{Z}+1-1/k}(z) = \Theta_{\mathbb{Z}-1/k}(z)$ and we have the identity that $\psi_k(z) = \psi_{-k}(z)$ [11, p. 105]. For a general shift $t \in \mathbb{R}$ and considering z to be pure imaginary (3), we have the one dimensional theta series [6, Eq. (2.2.5)]

$$\Theta_{\mathbb{Z}+t}(i\tau) = \sum_{m\in\mathbb{Z}} e^{-\pi\tau(m+t)^2} = \tau^{-1/2} \sum_{k=-\infty}^{\infty} e^{2\pi i k t - \pi k^2/\tau} = \tau^{-1/2}\vartheta_3(\pi t|i\tau^{-1}).$$

In this paper, we focus on real lattices, but the lattice constructions can be extended to lattices over complex numbers. Several families of lattices can be constructed from linear codes, and they are of particular interest since the underlying code structure allows a simpler characterization of some of the lattice properties, like the theta series. These lattices are exactly the q-ary lattices [27].

Particularly, we consider lattices that can be expressed in terms of a code formula due to the connection with partition chains (or lattice filtrations [15,32]).

Definition 8 (Partition Chain/Filtration). *A partition chain*

$$\Lambda_L/\Lambda_{L-1}/\ldots/\Lambda_1$$

is a sequence of lattices such that each is a sublattice of the previous one, that is, $\Lambda_1 \subseteq \ldots \subseteq \Lambda_L$. This is also called a filtration of the lattice Λ_L.

A partition chain induces a coset decomposition such that every element of a lattice Λ_L can be written as a sum of an element from Λ_1 and a coset representative from each partition $[\Lambda_j/\Lambda_{j-1}]$ [15, p. 1127]. The partition chain $\mathbb{Z}/2\mathbb{Z}/4\mathbb{Z}/\ldots$ induces the binary decomposition of an integer, so we can write any $x \in \mathbb{Z}$ as a sum $x_0 + 2x_1 + 4x_2 + \ldots$. A 2^L-*ary lattice* is an integer lattice that has $2^\ell\mathbb{Z}^n$ as a sublattice for some ℓ (clearly, this is true for $\ell = L$), and the smallest such ℓ is called the *depth* or *level* of the lattice. From now on, for $q \geq 2$, we define ϕ as the natural embedding of $\mathbb{Z}_q^n$ into $\mathbb{Z}^n$, which simply maps a congruence class to its corresponding integer.

Definition 9 (Construction A). *Let $q \geq 2$ and $\mathcal{C} \subseteq \mathbb{Z}_q^n$ be a linear code, then*

$$\Lambda_{\mathrm{A}}(\mathcal{C}) = q\mathbb{Z}^n + \phi(\mathcal{C})$$

defines a lattice, referred to as (q-ary) Construction A.

Definition 10 (Construction B). *Let $\mathcal{P}_n$ be the $(n, n-1)_2$ parity-check code and $\overline{\mathcal{C}} \subset \mathbb{F}_2^n$ be doubly even code, i.e., the weight of every codeword in $\overline{\mathcal{C}}$ is divisible by 4. Notice that $\overline{\mathcal{C}} \subset \mathcal{P}_n$. Thus, the Construction B lattice is such that*

$$\Lambda_{\mathrm{B}}(\overline{\mathcal{C}}) = 4\mathbb{Z}^n + 2\phi(\mathcal{P}_n) + \phi(\overline{\mathcal{C}}).$$

Using partitions on binary lattices modulo 2 and 4, we can express Construction A and B lattices in terms of a code formula since the lattice cosets $[\Lambda_j/\Lambda_{j-1}]$ form binary linear codes [15, Lemma 3, pp. 1132-1133].

The code formula in Definition 10 can be extended to L binary linear codes [15], which leads to the following definition of Construction D, provided that the underlying linear codes satisfy some multiplicative conditions [20, Th. 1].

Definition 11 (Construction D). *The code formula*

$$2^L\mathbb{Z}^n + 2^{L-1}\phi(\mathcal{C}_L) + \ldots + 2\phi(\mathcal{C}_2) + \phi(\mathcal{C}_1)$$

defines a lattice Λ_{D} if and only if every code $\mathcal{C}_j \subseteq \mathbb{F}_2^n$ is pairwise closed under the Schur product, i.e. $\mathbf{c}_j \star \mathbf{c}'_j \triangleq (c_{j,1}c_{j',1}, \ldots, c_{j,n}c_{j',n}) \in \mathcal{C}_{j+1}$ for all $\mathbf{c}_j, \mathbf{c}'_j \in \mathcal{C}_j$. Particularly, this implies that $\mathcal{C}_1 \subseteq \mathcal{C}_2 \subseteq \ldots \subseteq \mathcal{C}_L$.

From now on, we drop the notation of ϕ for simplicity, but the map is implied. Observe that both Constructions B and D can be seen as a particular case of the q-ary Construction A if we consider $\mathcal{C} = \overline{\mathcal{C}} + 2\mathcal{P}_n$ and $q = 4$ for Construction B, and $\mathcal{C} = \mathcal{C}_1 + 2\mathcal{C}_2 + \cdots + 2^{L-1}\mathcal{C}_L$ and $q = 2^L$ for Construction D. In both cases, the set of codes satisfy the Schur product condition, which implies that the code $\mathcal{C}$ is linear over the respective $\mathbb{Z}_q$.

By the code formula, we can write any lattice vector $\mathbf{x} \in \Lambda_{\mathrm{D}}$ as $\mathbf{x} = 2^L\mathbf{z} + 2^{L-1}\mathbf{c}_L + \ldots + 2\mathbf{c}_2 + \mathbf{c}_1$ where $\mathbf{z} \in \mathbb{Z}^n$ and $\mathbf{c}_j$ is a coset representative of $\mathcal{C}_j$, for all $i = 1, \ldots, L$. Therefore, we can express Λ_{D} as a disjoint union of its coset representatives as follows:

$$\Lambda_{\mathrm{D}} = \bigcup_{\mathbf{c}_L \in \mathcal{C}_L} \cdots \bigcup_{\mathbf{c}_1 \in \mathcal{C}_1} 2^L\mathbb{Z}^n + 2^{L-1}\mathbf{c}_L + \ldots + 2\mathbf{c}_2 + \mathbf{c}_1 = \bigcup_{\mathbf{c}\in\mathcal{C}} 2^L\mathbb{Z}^n + \mathbf{c} \tag{5}$$

where $\mathcal{C} = \mathcal{C}_1 + 2\mathcal{C}_2 + \ldots + 2^{L-1}\mathcal{C}_L$ is a linear code over $\mathbb{Z}_{2^L}$.

The Barnes-Wall lattices are a remarkable example of Construction D lattices, known to have efficient bounded-distance decoding algorithms [25]. Since some families of Reed-Muller codes are nested and closed under the Schur product, the following code formulas define a lattice by Theorem 11. These lattices are exactly the real Barnes-Wall lattices of dimension $n = 2^{m+1}$ [19]:

$$\mathrm{BW}_n = \Lambda(0,m) = \begin{cases} 2^{m/2}\mathbb{Z}^{2^{m+1}} + \sum\limits_{\substack{1\le r\le m \\ m-r \text{ odd}}} \mathrm{RM}(r,m+1)2^{\frac{r-1}{2}}, & \text{if m even} \\ 2^{(m+1)/2}\mathbb{Z}^{2^{m+1}} + \sum\limits_{\substack{1\le r\le m \\ m-r \text{ even}}} \mathrm{RM}(r,m+1)2^{\frac{r-1}{2}}, & \text{if m odd.} \end{cases}$$

2.4 Discrete Gaussian Distributions

The theta series is closely related to the discrete Gaussian distribution over a lattice. The Gaussian function is given by $\rho_s(\mathbf{x}) = e^{-\pi\|\mathbf{x}\|^2/s^2}$ for $\mathbf{x} \in \mathbb{R}^n$. Further, we can define the Gaussian function over a discrete set S as $\rho_s(S) \triangleq \sum_{\mathbf{x}\in S}\rho_s(\mathbf{x})$. The *discrete Gaussian distribution* over a lattice coset $\Lambda + \mathbf{t}$ for $\mathbf{t} \in \mathbb{R}^n$ is the discrete distribution with support over the coset. The probability of choosing a vector $\mathbf{y} \in \Lambda + \mathbf{t}$ according to this distribution is [6]

$$\mathcal{D}_{\Lambda+\mathbf{t},s}(\mathbf{y}) \triangleq \frac{\rho_s(\mathbf{y})}{\rho_s(\Lambda+\mathbf{t})} = \frac{\rho_s(\mathbf{y})}{\Theta_{\Lambda+\mathbf{t}}(i/s^2)}, \tag{6}$$

where $\rho_s(\Lambda+\mathbf{t})$ is a normalisation factor. We now recall some properties of the lattice Gaussian distribution.

Proposition 1. *[8, Prop. 1] For the lattice Gaussian distribution, it holds that*

1. $\mathcal{D}_{\alpha(\Lambda+\mathbf{t}),s}(\alpha\mathbf{y}) = \mathcal{D}_{\Lambda+\mathbf{t},s/\alpha}(\mathbf{y})$,
2. $\mathcal{D}_{(\Lambda_1+\mathbf{t}_1)\oplus(\Lambda_2+\mathbf{t}_2)}(\mathbf{y}_1,\mathbf{y}_2) = \mathcal{D}_{\Lambda_1+\mathbf{t}_1}(\mathbf{y}_1)\mathcal{D}_{\Lambda_2+\mathbf{t}_2}(\mathbf{y}_2)$.

The smoothing parameter is a lattice measure based on the Gaussian distribution [26].

Definition 12 (Smoothing Parameter). *For an n-dimensional lattice Λ, and a positive real $\varepsilon > 0$, the smoothing parameter $\eta_\varepsilon(\Lambda)$ is the smallest s such that $\rho_{1/s}(\Lambda^* \setminus \{\mathbf{0}\}) \le \varepsilon$.*

In the sampling context, if a noise vector is sampled from a Gaussian distribution with a width at least as large as the smoothing parameter and then reduced modulo the fundamental parallelepiped of the lattice, the resulting distribution will be nearly uniform.

[6] In literature, there is a slightly different definition $\mathcal{D}_{\Lambda,s,\mathbf{t}}(\mathbf{x})$ with a center $\mathbf{t}$, where $\mathbf{x} \in \Lambda$. It is easy to see $\mathcal{D}_{\Lambda,s,\mathbf{t}}(\mathbf{x}) = \mathcal{D}_{\Lambda-\mathbf{t},s}(\mathbf{x}-\mathbf{t})$, namely, they are a shifted version of each other. In this paper, we follow the definition (6) since it is used in the GPV framework [16].

3 Reductions

We show that the problem of lattice sampling for q-ary lattices can be reduced to that of code sampling. Combined with a straightforward reduction in the other direction, this implies the two problems are equivalent.

For clarity, let us fix the lattice $\Lambda_{\mathrm{A}}(\mathcal{C}) = q\mathbb{Z}^n + \mathcal{C}$ with $\mathcal{C} \subseteq \mathbb{Z}_q^n$ a linear code and $q \geq 2$, and give definitions of the two problems. We also let the shift $\mathbf{t} \in \mathbb{Z}_q^n$ without loss of generality.

Definition 13 (Gaussian Sampling for q-ary Lattices). *Given some lattice $\Lambda_{\mathrm{A}} = q\mathbb{Z}^n + \mathcal{C}$, a shift $\boldsymbol{t} \in \mathbb{Z}_q^n$ and sampling width $s > 0$, the Gaussian sampling problem asks to sample a vector $\mathbf{x} \in \Lambda_{\mathrm{A}} + \boldsymbol{t}$ from the distribution $\mathcal{D}_{\Lambda_{\mathrm{A}}+\boldsymbol{t},s}$.*

Definition 14 (Sampling for q-ary Linear Codes). *Given a linear code $\mathcal{C} \subseteq \mathbb{Z}_q^n$, a shift $\boldsymbol{t} \in \mathbb{Z}_q^n$ and a parameter s, the code sampling problem with respect to the Lee weight profile asks to sample a vector $\boldsymbol{\nu} \in \mathcal{C} + \boldsymbol{t}$ with probability given by:*

$$\mathcal{L}_{\mathcal{C}+\boldsymbol{t},s}(\boldsymbol{\nu}) \triangleq \frac{\Theta_{q\mathbb{Z}}(z)^{n_0(\boldsymbol{\nu})}\Theta_{q\mathbb{Z}+1}(z)^{n_1(\boldsymbol{\nu})}\dots\Theta_{q\mathbb{Z}+\ell}(z)^{n_\ell(\boldsymbol{\nu})}}{\sum_{\boldsymbol{\nu}\in\mathcal{C}+\boldsymbol{t}}\Theta_{q\mathbb{Z}}(z)^{n_0(\boldsymbol{\nu})}\Theta_{q\mathbb{Z}+1}(z)^{n_1(\boldsymbol{\nu})}\dots\Theta_{q\mathbb{Z}+\ell}(z)^{n_\ell(\boldsymbol{\nu})}}$$

where $z = {}^{i}/_{s^2}$, $n_j(\boldsymbol{\nu})$ are the entries of the Lee weight profile of $\boldsymbol{\nu}$ and $\ell = \lceil {}^{q}/_{2} \rceil$. We denote the induced distribution as $\mathcal{L}_{\mathcal{C}+\boldsymbol{t},s}$.

Define $\mathcal{D}_{\Lambda_{\mathrm{A}}+\mathbf{t},s}(S) \triangleq \sum_{\mathbf{x}\in S}\mathcal{D}_{\Lambda_{\mathrm{A}}+\mathbf{t},s}(\mathbf{x})$ over a discrete set S.

Lemma 1 (Key Lemma). *Given a linear q-ary code $\mathcal{C} \subseteq \mathbb{Z}_q^n$, a vector $\mathbf{t} \in \mathbb{Z}_q^n$ and $\boldsymbol{\nu} \in \mathcal{C} + \mathbf{t}$, we have* [7]

$$\mathcal{D}_{\Lambda_{\mathrm{A}}+\boldsymbol{t},s}(q\mathbb{Z}^n + \boldsymbol{\nu}) = \mathcal{L}_{\mathcal{C}+\boldsymbol{t},s}(\boldsymbol{\nu}).$$

Proof. Consider $\boldsymbol{\nu} \in \mathcal{C} + \mathbf{t}$, then the j-th coordinate of $q\mathbf{z} + \boldsymbol{\nu}$ where $\mathbf{z} \in \mathbb{Z}^n$ is

$$qz_j + \nu_j = \begin{cases} qz_j & \text{if } \nu_j = 0 \\ qz_j + 1 & \text{if } \nu_j = 1 \\ \vdots & \\ qz_j + q - 1 & \text{if } \nu_j = q-1. \end{cases}$$

For each $qz_j + \nu_j$, we derive a corresponding theta series for the one-dimensional lattice coset $q\mathbb{Z} + \nu_j$. We have that

$$\begin{aligned} &\Theta_{q\mathbb{Z}}(z) = \vartheta_3(q^2 z), \\ &\Theta_{q\mathbb{Z}+j}(z) = \Theta_{q\mathbb{Z}+q-j}(z), \quad j = 1, 2, \dots, q-1 \end{aligned}$$

and the second equality follows from an analogous argument to the proof of (4).

[7] Formally, $\mathbf{t} \in \mathbb{Z}^n$, but we can assume $\mathbf{t} \in \mathbb{Z}_q^n$ without loss of generality [30].

For a fixed $\boldsymbol{\nu}$, we can write

$$\Theta_{q\mathbb{Z}^n+\boldsymbol{\nu}}(z) = \sum_{\mathbf{z}\in\mathbb{Z}^n} e^{\pi i z\|q\mathbf{z}+\boldsymbol{\nu}\|^2}. \tag{7}$$

We have that $q\mathbb{Z}^n + \boldsymbol{\nu}$ is isometric to the decomposition given by [8]

$$q\mathbb{Z}^{n_0(\boldsymbol{\nu})} \oplus \left(q\mathbb{Z}^{n_1(\boldsymbol{\nu})} + \left(1^{n_1(\boldsymbol{\nu})}\right)\right) \oplus \cdots \oplus \left(q\mathbb{Z}^{n_\ell(\boldsymbol{\nu})} + \left(\ell^{n_\ell(\boldsymbol{\nu})}\right)\right),$$

where $\ell = \lceil q/2 \rceil$ and we interpret $j^{n_j(\boldsymbol{\nu})}$ to be a vector with $n_j(\boldsymbol{\nu})$ entries of $\pm j$. It follows that

$$\begin{aligned}\Theta_{q\mathbb{Z}^n+\boldsymbol{\nu}}(z) &= \Theta_{q\mathbb{Z}^{n_0(\nu)}}(z)\Theta_{q\mathbb{Z}^{n_1(\nu)}+\left(1^{n_1(\nu)}\right)}(z) \times \cdots \times \Theta_{q\mathbb{Z}^{n_\ell(\nu)}+\left(\ell^{n_\ell(\nu)}\right)}(z) \\ &= \Theta_{q\mathbb{Z}}(z)^{n_0(\boldsymbol{\nu})}\Theta_{q\mathbb{Z}+1}(z)^{n_1(\boldsymbol{\nu})} \times \cdots \times \Theta_{q\mathbb{Z}+\ell}(z)^{n_\ell(\boldsymbol{\nu})} \end{aligned} \tag{8}$$

where the second equality follows by direct sum decomposition and using the fact that $\Theta_{\Lambda_1\oplus\Lambda_2}(z) = \Theta_{\Lambda_1}(z)\Theta_{\Lambda_2}(z)$.

The proof is completed by observing that for $z = i/s^2$,

$$\mathcal{D}_{\Lambda_{\mathrm{A}}+\mathbf{t},s}(q\mathbb{Z}^n + \boldsymbol{\nu}) = \frac{\Theta_{q\mathbb{Z}^n+\boldsymbol{\nu}}(z)}{\sum_{\boldsymbol{\nu}\in\mathcal{C}+\mathbf{t}} \Theta_{q\mathbb{Z}^n+\boldsymbol{\nu}}(z)} = \mathcal{L}_{\mathcal{C}+\mathbf{t},s}(\boldsymbol{\nu}). \qquad \square$$

Theorem 1 (Equivalence of Lattice and Code Sampling). *The Gaussian sampling problem for a Construction A lattice $\Lambda_{\mathrm{A}}(\mathcal{C}) + \mathbf{t}$ with $\mathcal{C} \subseteq \mathbb{Z}_q^n$ a linear code and $q \geq 2$ is equivalent to the problem of sampling codewords over $\mathcal{C} + \mathbf{t}$ with respect to their Lee weight profiles.*

Proof. We show the equivalence between lattice sampling (LS) and code sampling (CS) by establishing two reductions with polynomial running time. We use A $\rightarrow$ B to denote that problem A is reduced to problem B, i.e., an oracle for solving problem B can also be used as a subroutine to solve problem A efficiently.

(CS $\rightarrow$ LS): This reduction is simple, we just sample $\mathbf{x} \in \Lambda + \mathbf{t}$ with probability $\mathcal{D}_{\Lambda+\mathbf{t},s}(\mathbf{x})$, then return a vector $\boldsymbol{\nu} = \mathbf{x} \mod q$. The probability of obtaining $\boldsymbol{\nu}$ is $\mathcal{D}_{\Lambda_{\mathrm{A}}+\mathbf{t},s}(q\mathbb{Z}^n + \boldsymbol{\nu})$, since this is the probability that a point drawn from the discrete Gaussian in $\Lambda + \mathbf{t}$ lies in the coset $q\mathbb{Z}^n + \boldsymbol{\nu}$. As proven in Lemma 1, $\mathcal{D}_{\Lambda_{\mathrm{A}}+\mathbf{t},s}(q\mathbb{Z}^n + \boldsymbol{\nu}) \propto \Theta_{q\mathbb{Z}}(z)^{n_0(\boldsymbol{\nu})}\Theta_{q\mathbb{Z}+1}(z)^{n_1(\boldsymbol{\nu})} \ldots \Theta_{q\mathbb{Z}+\ell}(z)^{n_\ell(\boldsymbol{\nu})}$ where the $n_j(\boldsymbol{\nu})$ are the entries of the Lee weight profile of $\boldsymbol{\nu}$. This probability depends on the Lee weight profile of $\boldsymbol{\nu}$ only.

(LS $\rightarrow$ CS): We can decompose a Construction A lattice as a disjoint union of cosets $\Lambda_{\mathrm{A}}(\mathcal{C}) = \bigcup_{\mathbf{c}\in\mathcal{C}} q\mathbb{Z}^n + \mathbf{c}$. We sample from $\Lambda_{\mathrm{A}}(\mathcal{C}) + \mathbf{t}$ via coset decomposition:

1. Sample a coset representative $\boldsymbol{\nu} \in \mathcal{C} + \mathbf{t}$ with probability

$$\mathcal{L}_{\mathcal{C}+\mathbf{t},s}(\boldsymbol{\nu}) = \mathcal{D}_{\Lambda_{\mathrm{A}}+\mathbf{t},s}(q\mathbb{Z}^n + \boldsymbol{\nu}).$$

[8] Direct sum decomposition does not preserve order, but this is not relevant for the computation of the theta series.

2. Sample a vector $\mathbf{x}$ from $\mathcal{D}_{q\mathbb{Z}^n+\boldsymbol{\nu},s}$ of the form $\mathbf{x}=\mathbf{z}+\mathbf{v}$.

This outputs a lattice vector $\mathbf{x}=\mathbf{z}+\boldsymbol{\nu}\in q\mathbb{Z}^n+\boldsymbol{\nu}$ with probability $\mathcal{D}_{\Lambda_{\mathrm{A}}+\mathbf{t},s}(q\mathbb{Z}^n+\boldsymbol{\nu})\cdot\mathcal{D}_{q\mathbb{Z}^n+\boldsymbol{\nu},s}(\mathbf{z}+\boldsymbol{\nu})=\mathcal{D}_{\Lambda_{\mathrm{A}}+\mathbf{t},s}(\mathbf{z}+\boldsymbol{\nu})=\mathcal{D}_{\Lambda_{\mathrm{A}}+\mathbf{t},s}(\mathbf{x})$ as desired. □

If we view any multilevel lattice construction (B or D) as a Construction A lattice with $q=2^L$, then from Theorem 1 it follows that sampling a Construction D lattice is also equivalent to sampling codewords with respect to their Lee weight profiles. We omit the details.

4 Theta Series of q-ary Lattices

The theta series is the key lattice measure underlying our sampling method. In general, it is challenging to calculate the theta series of an arbitrary lattice Λ, but a closed formula is known for some well-studied lattices, which can lead to a better estimation of the smoothing parameter and improved samplers.

We now expand on the theta series of some families of q-ary lattices obtained from linear codes. We start with showing that the theta series of a Construction A lattice with $q\geq 2$ can be written in terms of the symmetrized weight enumerator of the associated linear code over $\mathbb{Z}_q$, based on [24].

Theorem 2 (Theta Series of a Construction A Lattice for $q\geq 2$). *The theta series of a Construction A lattice $\Lambda_{\mathrm{A}}(\mathcal{C})=q\mathbb{Z}^n+\mathcal{C}$ is given by*

$$\Theta_{\Lambda_{\mathrm{A}}}(z)=\mathrm{swe}_{\mathcal{C}}(\Theta_{\mathbb{Z}}(q^2z),\Theta_{\mathbb{Z}+1/q}(q^2z),\ldots,\Theta_{\mathbb{Z}+\ell/q}(q^2z)),$$

where $\ell=\lceil q/2\rceil$.

Proof. We can write

$$\Theta_{\Lambda_{\mathrm{A}}}(z)=\sum_{\mathbf{c}\in\mathcal{C}}\sum_{\mathbf{z}\in\mathbb{Z}^n}e^{\pi iz\|q\mathbf{z}+\mathbf{c}\|^2}=\sum_{\mathbf{c}\in\mathcal{C}}\Theta_{q\mathbb{Z}^n+\mathbf{c}}(z). \tag{9}$$

Using (8), we have

$$\Theta_{\Lambda_{\mathrm{A}}}(z)=\sum_{\mathbf{c}\in\mathcal{C}}\Theta_{q\mathbb{Z}}(z)^{n_0(\mathbf{c})}\Theta_{q\mathbb{Z}+1}(z)^{n_1(\mathbf{c})}\cdots\Theta_{q\mathbb{Z}+\ell}(z)^{n_\ell(\mathbf{c})}.$$

Notice that this is the symmetrized weight enumerator of the linear code $\mathcal{C}\subseteq\mathbb{Z}_q^n$ evaluated at the corresponding shifts of theta series. □

The two main challenges in calculating the theta series of any q-ary lattice obtained via Construction A are i) characterizing the symmetrized weight enumerator of a q-ary code $\mathcal{C}$ involves going through all the codewords of $\mathcal{C}$, and ii) the theta series of shifts of $q\mathbb{Z}$, i.e., $\Theta_{q\mathbb{Z}+c}(z)$ have some simplifications via the well-known Jacobi-theta functions [11, pp. 102-105], but not always. We present next some cases where this calculation is simplified due to these properties.

For the binary Construction A ($q=2$), the symmetrized weight enumerator is simply the Hamming weight enumerator of the binary code $\mathcal{C}$. Moreover, $\Theta_{\mathbb{Z}}(q^2z)=\vartheta_3(4z)$ and $\Theta_{\mathbb{Z}+1/q}(q^2z)=\vartheta_2(4z)$, and from [11, Th. 3, Ch. 7], we get the following expression for the theta series of $\Lambda_{\mathrm{A}}(\mathcal{C})$.

Theorem 3 (Theta Series of binary Construction A Lattice). *Consider an $(n,k)_2$ code $\mathcal{C}$, then the theta series of $\Lambda_{\mathrm{A}}(\mathcal{C})$ is given by*

$$\Theta_{\Lambda_{\mathrm{A}}(\mathcal{C})}(z) = W_{\mathcal{C}}(\vartheta_3(4z), \vartheta_2(4z)).$$

Given that the parity-check code $\mathcal{P}_n$ defines the Construction B lattice, its theta series can be written only in terms of the Hamming weight enumerator of the doubly-even code $\overline{\mathcal{C}} \subseteq \mathcal{P}_n$, according to [11, Th. 15, Ch. 7].

Theorem 4 (Theta Series of a Construction B Lattice). *Consider an $(n,k)_2$ doubly even code $\overline{\mathcal{C}}$. The theta series of $\Lambda_{\mathrm{B}}(\overline{\mathcal{C}})$ is*

$$\Theta_{\Lambda_{\mathrm{B}}(\overline{\mathcal{C}})}(z) = \tfrac{1}{2} W_{\overline{\mathcal{C}}}(\vartheta_3(4z), \vartheta_2(4z)) + \tfrac{1}{2}\vartheta_4(4z)^n.$$

Naturally, the same procedure can be applied to Construction D lattices. We show that its theta series can also be written as symmetrized weight enumerator of the liner code $\mathcal{C} \subseteq \mathbb{Z}_{2^L}^n$, provided that $\mathcal{C} = 2^{L-1}\mathcal{C}_L + \ldots + 2\mathcal{C}_2 + \mathcal{C}_1$ and $\mathcal{C}_j$ are binary linear codes closed under the Schur product, for $1 \leq j \leq L-1$. If we consider the j-th coordinate of a lattice vector $2^L\mathbf{z} + \mathbf{c}$ where $\mathbf{c} \in \mathcal{C}$, we get

$$2^L z_j + c_j = \begin{cases} 2^L z_j & \text{if } (c_{1,j}, ..., c_{2^L,j}) = (0,0,\ldots,0) \\ 2^L z_j + 1 & \text{if } (c_{1,j}, ..., c_{2^L,j}) = (1,0,\ldots,0) \\ \vdots & \\ 2^L z_j + 2^L - 1 & \text{if } (c_{1,j}, ..., c_{2^L,j}) = (1,1,\ldots,1). \end{cases}$$

Let $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_L) \in \mathbb{F}_2^L$ and recall that $\omega_{\boldsymbol{\alpha}}(\mathbf{c}_1, \ldots, \mathbf{c}_L)$ is the number of occurrences of $\boldsymbol{\alpha}$ as a row in the matrix of column vectors $\mathbf{c}_1, \ldots, \mathbf{c}_L$. Also, since $n_w(\mathbf{c})$ is the number of coordinates that have Lee weight w, it follows that

$$n_w(\mathbf{c}) = \omega_{\alpha_1,\ldots,\alpha_L}(\mathbf{c}_1, \ldots, \mathbf{c}_L) + \omega_{\tilde{\alpha}_1,\ldots,\tilde{\alpha}_L}(\mathbf{c}_1, \ldots, \mathbf{c}_L),$$

where $i = \alpha_1 + 2\alpha_2 + \ldots + 2^{L-1}\alpha_L$ and $2^L - i = \tilde{\alpha}_1 + 2\tilde{\alpha}_2 + \ldots + 2^{L-1}\tilde{\alpha}_L$. Therefore, we have that for the $\mathbb{Z}_{2^L}$-linear code $\mathcal{C} = \mathcal{C}_1 + 2\mathcal{C}_2 + \ldots + 2^{L-1}\mathcal{C}_L$ with each $\mathcal{C}_j$ a binary linear code, we can write the Lee weight profile of $\mathbf{c} \in \mathcal{C}$ either as $[n_0(\mathbf{c}), n_1(\mathbf{c}), \ldots, n_\ell(\mathbf{c})]$ as defined in 4 or equivalently, as

$$[\omega_{0,\ldots,0}(\mathbf{c}_1, \ldots, \mathbf{c}_L), \omega_{1,0,\ldots,0}(\mathbf{c}_1, \ldots, \mathbf{c}_L) + \omega_{0,1,\ldots,1}(\mathbf{c}_1, \ldots, \mathbf{c}_L), \ldots \\ \ldots, \omega_{1,\ldots,1}(\mathbf{c}_1, \ldots, \mathbf{c}_L)].$$

We can also rewrite (7) as a joint weight enumerator over L binary codes.

Theorem 5 (Theta Series of a Construction D Lattice). *The theta series of a Construction D lattice $\Lambda_{\mathrm{D}} = 2^L\mathbb{Z}^n + 2^{L-1}\mathcal{C}_L + \ldots + 2\mathcal{C}_2 + \mathcal{C}_1$ is given by*

$$\Theta_{\Lambda_{\mathrm{D}}}(z) = \mathrm{jwe}_{\mathcal{C}_1,\ldots,\mathcal{C}_L}(\Theta_{\mathbb{Z}+{}^t\boldsymbol{\alpha}/2^L})$$

where $\boldsymbol{\alpha} \in \mathbb{F}_2^L$ and $t_{\boldsymbol{\alpha}} = \alpha_1 + 2\alpha_2 + \ldots + 2^{L-1}\alpha_L$.

Proof. We have that

$$\begin{aligned}
\Theta_{\Lambda_{\mathrm{D}}}(z) &= \sum_{\mathbf{c}\in\mathcal{C}} \Theta_{2^L\mathbb{Z}^n+\mathbf{c}}(z) \\
&= \sum_{\mathbf{c}\in\mathcal{C}} \Theta_{2^L\mathbb{Z}}(z)^{n_0(\mathbf{c})} \cdots \Theta_{2^L\mathbb{Z}+2^L-1}(z)^{n_{2^L-1}(\mathbf{c})} \\
&= \sum_{\mathbf{c}_1\in\mathcal{C}_1} \cdots \sum_{\mathbf{c}_L\in\mathcal{C}_L} \Theta_{2^L\mathbb{Z}}(z)^{\omega_{0,\ldots,0}(\mathbf{c}_1,\ldots,\mathbf{c}_L)} \cdots \Theta_{2^L\mathbb{Z}+2^L-1}(z)^{\omega_{1,\ldots,1}(\mathbf{c}_1,\ldots,\mathbf{c}_L)} \\
&= \sum_{\mathbf{c}_1\in\mathcal{C}_1} \cdots \sum_{\mathbf{c}_L\in\mathcal{C}_L} \prod_{\boldsymbol{\alpha}\in\mathbb{F}_2^L} \Theta_{\mathbb{Z}+t_{\boldsymbol{\alpha}}/2^L}(z)^{\omega_{\boldsymbol{\alpha}}(\mathbf{c}_1,\ldots,\mathbf{c}_L)} \\
&= \mathrm{jwe}_{\mathcal{C}_1,\ldots,\mathcal{C}_L}(\Theta_{\mathbb{Z}+t_{\boldsymbol{\alpha}}/2^L}).
\end{aligned}$$

□

Theorem 5 can be also expressed in terms of the symmetrized weight enumerator over the code $\mathcal{C} = \mathcal{C}_1 + 2\mathcal{C}_2 + \ldots + 2^{L-1}\mathcal{C}_L \subseteq \mathbb{Z}_{2^L}^n$ as proposed in Theorem 2. The main difference between the code formula Construction D and Construction A for $q \geq 2$ is that we have an identity between the symmetrized weight enumerator of a single linear code over $\mathbb{Z}_{2^L}^n$ and the joint weight enumerator of multiple binary codes $\mathcal{C}_j$, for $1 \leq j \leq L$. This does not hold in general for Construction A with $q \geq 2$ because we cannot always decompose a q-ary code $\mathcal{C} \subseteq \mathbb{Z}_q^n$ into a sum of binary codes. Such a particular property will further allow a simplification of the Gaussian sampling.

For the particular case of Theorem 5 for $L = 2$, we recover the theta series for a 2-level Construction D lattice [5, Th. 22]. For instance, the theta series of $\frac{1}{2}(4\mathbb{Z}^n + 2\mathcal{C}_2 + \mathcal{C}_1)$ is given by

$$\begin{aligned}
\Theta_{\frac{1}{2}(4\mathbb{Z}^n+2\mathcal{C}_2+\mathcal{C}_1)}(z) &= \sum_{\mathbf{c}_1\in\mathcal{C}_1} \sum_{\mathbf{c}_2\in\mathcal{C}_2} \vartheta_3(4z)^{\omega_{0,0}} \left(\frac{\vartheta_2(z)}{2}\right)^{\omega_{1,0}+\omega_{1,1}} \vartheta_2(4z)^{\omega_{0,1}} \\
&= \mathrm{swe}_{\mathcal{C}}(\vartheta_3(4z), {}^{\vartheta_2(z)}\!/_2, \vartheta_2(4z)), \\
&\overset{(a)}{=} \mathrm{jwe}_{\mathcal{C}_1,\mathcal{C}_2}(\vartheta_3(4z), \vartheta_2(4z), {}^{\vartheta_2(z)}\!/_2, {}^{\vartheta_2(z)}\!/_2), \qquad (10)
\end{aligned}$$

for $\mathcal{C} = {}^1\!/_2(\mathcal{C}_1 + 2\mathcal{C}_2)$. In (a) we have used the identity from (2). Notice that $\Theta_{2\mathbb{Z}+1/2}(z) = \Theta_{\mathbb{Z}+1/4}(4z) = \Theta_{\mathbb{Z}+3/4}(4z) = \psi_4(4z)$, and then we apply the identity $\psi_4(z) = \frac{\vartheta_2(z/4)}{2}$ [11, p. 105]. In particular, considering $\mathcal{C}_1 = \overline{\mathcal{C}}$ and $\mathcal{C}_2 = \mathcal{P}_n$, where $\overline{\mathcal{C}}$ is a doubly even code, we recover the theta series of Construction B presented in Theorem 4.

Next, we use Theorem 5 to express the theta series of the Barnes-Wall lattice BW_{128}, which is a 3-level Construction D.

Example 1. Consider the Barnes-Wall lattice $\mathrm{BW}_{128} = 8\mathbb{Z}^{128} + 4\,\mathrm{RM}(5,7) + 2\,\mathrm{RM}(3,7) + \mathrm{RM}(1,7)$ where $\mathrm{RM}(r,7)$ is a Reed-Muller code of length 2^7 of order r for $r = 1, 3, 5$ [17]. Hence, via coset decomposition,

$$\mathrm{BW}_{128} = \bigcup_{\mathbf{c}_3\in\mathcal{C}_3} \bigcup_{\mathbf{c}_2\in\mathcal{C}_2} \bigcup_{\mathbf{c}_1\in\mathcal{C}_1} 8\mathbb{Z}^{128} + 4\mathbf{c}_3 + 2\mathbf{c}_2 + \mathbf{c}_1$$

and we define $\mathbf{c} \triangleq 4\mathbf{c}_3 + 2\mathbf{c}_2 + \mathbf{c}_1$. The theta series for each of the one-dimensional cosets is given by

$$\begin{aligned}
&\Theta_{8\mathbb{Z}}(z) = \vartheta_3(64z), \quad \Theta_{8(\mathbb{Z}+1/8)}(z) = \Theta_{8(\mathbb{Z}+7/8)}(z) = \psi_8(64z),\\
&\Theta_{8(\mathbb{Z}+1/4)}(z) = \Theta_{8(\mathbb{Z}+3/4)}(z) = \psi_4(64z) = {}^{\vartheta_2(16z)}/_2,\\
&\Theta_{8(\mathbb{Z}+3/8)}(z) = \Theta_{8(\mathbb{Z}+5/8)}(z) = ({}^{z}/_{i})^{-1/2}\, \vartheta_3\left({}^{3\pi}/_{8}|z\right), \quad \Theta_{8(\mathbb{Z}+1/2)}(z) = \vartheta_2(64z).
\end{aligned}$$

Hence, the theta series of a fixed coset of BW_{128} is

$$\begin{aligned}
\Theta_{8\mathbb{Z}^{128}+\mathbf{c}}(z) &= \sum_{\mathbf{z}\in\mathbb{Z}^{128}} e^{\pi i z \|8\mathbf{z}+\mathbf{c}\|^2}\\
&= \vartheta_3(64z)^{\omega_{0,0,0}} \vartheta_2(64z)^{\omega_{0,0,1}} \left({}^{\vartheta_2(16z)}/_2\right)^{\omega_{0,1,0}+\omega_{0,1,1}} \times\\
&\quad \times \psi_8(64z)^{\omega_{1,0,0}(\mathbf{c})+\omega_{1,1,1}} \Theta_{\mathbb{Z}+\frac{3}{8}}(64z)^{\omega_{1,1,0}+\omega_{1,0,1}}. \qquad (11)
\end{aligned}$$

The overall theta series is obtained by summing over all codewords $\mathbf{c} \in \mathcal{C}$.

$$\begin{aligned}
\Theta_{\mathrm{BW}_{128}}(z) &= \sum_{\mathbf{c}\in\mathcal{C}} \sum_{\mathbf{z}\in\mathbb{Z}^{128}} e^{\pi i z ||8\mathbf{z}+\mathbf{c}||^2}\\
&= \mathrm{swe}_{\mathcal{C}}\left(\vartheta_3(64z), \vartheta_2(64z), {}^{\vartheta_2(16z)}/_2, \psi_8(64z), \Theta_{\mathbb{Z}+3/8}(64z)\right),
\end{aligned}$$

where $\mathcal{C} = 4\mathcal{C}_3 + 2\mathcal{C}_2 + \mathcal{C}_1$. ◊

For Gaussian sampling over a q-ary lattice Λ, the theta series we are interested in is $\Theta_{\Lambda+\mathbf{t}}(z)$, where $\mathbf{t} \in \mathbb{Z}_q^n$. The results of this section then correspond to $\mathbf{t} = \mathbf{0}$. For $\mathbf{t} \neq \mathbf{0}$ and a binary Construction A, one can calculate the weight enumerator of the coset $\mathcal{C} + \mathbf{t}$ of a code $\mathcal{C} \subseteq \mathbb{Z}_2^n$ with $\mathbf{t} \in \mathbb{Z}_2^n$ according to Appendix A, and then obtain the theta function of $\Lambda_{\mathrm{A}}(\mathcal{C}) + \mathbf{t}$ via Theorem 3. For larger alphabets $q > 2$, expressing the symmetrized weight enumerator of a coset $\mathcal{C} + \mathbf{t}$ of a q-ary code $\mathcal{C}$ is more challenging and might not have a closed form. In this case, we assume an oracle for computing the symmetrized weight enumerator of a coset.

4.1 Smoothing Parameter

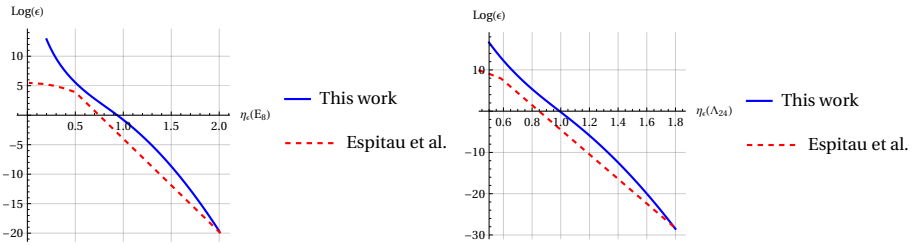
The smoothing parameter $\eta_\epsilon(\Lambda)$ is computed by finding s such that $\Theta_{\Lambda^*}(is^2) - 1 = \epsilon$ where $\Theta_{\Lambda^*}(is^2) = \sum_{\mathbf{x}\in\Lambda^*} e^{-\pi\tau\|\mathbf{x}\|^2}$ with $\tau = s^2$ in (3). For a general Construction A lattice, its dual is also a Construction A lattice up to a scaling of ${}^1/_{\sqrt{q}}$ (more details in Appendix B). That is,

$$\Lambda_{\mathrm{A}}(\mathcal{C})^* = (q\mathbb{Z}^n + \mathcal{C})^* = q\mathbb{Z}^n + \mathcal{C}^\perp = \Lambda_{\mathrm{A}}(\mathcal{C}^\perp).$$

The theta series of the dual is then given by

$$\Theta_{\Lambda_{\mathrm{A}}^*}(z) = \mathrm{swe}_{\mathcal{C}^\perp}(\Theta_{\mathbb{Z}}(q^2 z), \Theta_{\mathbb{Z}+1/q}(q^2 z), \ldots, \Theta_{\mathbb{Z}+\ell/q}(q^2 z))$$

where $\ell = \lceil {}^{q}/_{2} \rceil$. We can obtain the symmetrized weight enumerator of $\mathcal{C}^\perp$ from the symmetrized weight enumerator of $\mathcal{C}$ using MacWilliams identities [23,



(a) E_8 lattice (b) Λ_{24} lattice

Fig. 1: Smoothing parameters. The log scale refers to the natural logarithm.

Th. 12] for $q < 5$. Now we set $z = is^2$ and solve for s such that $\Theta_{\Lambda^*}(is^2) - 1 = \epsilon$. For $q \geq 5$, we cannot apply the MacWilliams identities [1], but we can solve for s using a formulation involving the primal lattice instead [22]. Typically in cryptography, the modulus q is chosen such that $q \geq 5$.

Fig. 1 represents the smoothing parameters of two lattices, E_8 and the Leech Λ_{24}. The approximations presented in [14] are usually fair for small values of ϵ.

5 Sampling Algorithms for q-ary Lattices

The reduction in Section 3 suggests an algorithm for lattice Gaussian sampling. Let $\Lambda = \bigcup_{\mathbf{c}\in\mathcal{C}} \Lambda' + \mathbf{c}$ be a disjoint union of cosets. To sample from $\Lambda + \mathbf{t}$, we now consider the decomposition with respect to the coset $\mathcal{C} + \mathbf{t}$. Sample first a vector $\boldsymbol{\nu} \in \mathcal{C} + \mathbf{t}$ with probability $\mathcal{D}_{\Lambda+\mathbf{t},s}(\Lambda' + \boldsymbol{\nu})$, then sample a lattice vector $\mathbf{x}' \in \Lambda'$ with probability $\mathcal{D}_{\Lambda'+\boldsymbol{\nu},s}(\mathbf{x}' + \boldsymbol{\nu})$ to output a vector $\mathbf{x} = \mathbf{x}' + \boldsymbol{\nu}$, with target probability $\mathcal{D}_{\Lambda+\mathbf{t},s}(\Lambda' + \boldsymbol{\nu}) \cdot \mathcal{D}_{\Lambda'+\boldsymbol{\nu},s}(\mathbf{x}' + \boldsymbol{\nu}) = \mathcal{D}_{\Lambda+\mathbf{t},s}(\mathbf{x})$.

5.1 Sampling General Construction A with $q \geq 2$

Algorithm 1 describes the coset decomposition sampling over a shift of a general Construction A lattice with code formula $q\mathbb{Z}^n + \mathcal{C}$. When $\mathbf{t} = \mathbf{0}$, the probability of sampling a coset corresponding to a codeword $\mathbf{c} \in \mathcal{C}$ depends on its Lee weight profile (see Section 3). We can compute the Lee weight profiles of the code offline using the Schur product, according to Appendix C, and store this information.

For $\mathbf{t} \neq \mathbf{0}$, we can use the symmetrized weight enumerator of the coset $\mathcal{C} + \mathbf{t}$. When $q = 2$, it is simply the Hamming weight enumerator of the coset of a code (see Appendix A)[9]. When $q < 5$, we can use a similar approach as in the binary case to compute the symmetrized weight enumerator of $\mathcal{C} + \mathbf{t}$, but for $q \geq 5$ we assume we have access to an oracle that computes performs this calculation offline, since we cannot rely on the relationship between a code and its dual [1].

In Algorithm 1, we let $\tilde{\boldsymbol{\nu}}$ be *any* codeword in $\mathcal{C} + \mathbf{t}$ that has Lee weight profile $\boldsymbol{\ell}$ and $\mathcal{D}_{q\mathbb{Z}^n+(\mathcal{C}+\mathbf{t}),s}(q\mathbb{Z}^n + \tilde{\boldsymbol{\nu}})$ be the probability that we sample a lattice

[9] When $q = 2$ and $\mathbf{t} = \mathbf{0}$, we recover the Construction A sampler from [8].

Algorithm 1: Sampling for General Construction A

Require: $\Lambda_{\mathrm{A}} = q\mathbb{Z}^n + \mathcal{C}, \mathcal{C} \subseteq \mathbb{Z}_q^n$ and $\mathrm{swe}_{\mathcal{C}+\mathbf{t}}(\cdot)$ for $\mathbf{t} \in \mathbb{Z}_q^n$ and sampling width s
Ensure: $\mathbf{x} \leftarrow \mathcal{D}_{\Lambda_{\mathrm{A}}+\mathbf{t},s}(\mathbf{x})$
1: Select Lee weight profile $\boldsymbol{\ell}$ with probability $p_{\boldsymbol{\ell}} = A_{\boldsymbol{\ell}} \cdot \mathcal{D}_{q\mathbb{Z}^n+(\mathcal{C}+\mathbf{t}),s}(q\mathbb{Z}^n + \tilde{\boldsymbol{\nu}})$ where $A_{\boldsymbol{\ell}}$ is a coefficient of $\mathrm{swe}_{\mathcal{C}+\mathbf{t}}(\cdot)$
2: Select a codeword $\boldsymbol{\nu} \in \mathcal{C} + \mathbf{t}$ with Lee weight profile $\boldsymbol{\ell}$ uniformly at random
3: **for** $j = 1, \ldots, n$ **do**
  $x_j \leftarrow q \cdot \mathrm{Sampler}_{\mathbb{Z}+\nu_j/q}(s/q)$
  end
4: **return** $\mathbf{x} = (x_1, \ldots, x_n)$

vector in any coset whose representative has Lee weight profile $\boldsymbol{\ell}$. By Lemma 1, $\mathcal{D}_{q\mathbb{Z}^n+(\mathcal{C}+\mathbf{t}),s}(q\mathbb{Z}^n + \tilde{\boldsymbol{\nu}})$ is proportional to a term in the symmetrized weight enumerator of $\mathcal{C} + \mathbf{t}$ with exponents defined by the coefficients of $\boldsymbol{\ell}$. We select $\boldsymbol{\ell}$ with probability $\mathcal{D}_{q\mathbb{Z}^n+(\mathcal{C}+\mathbf{t}),s}(q\mathbb{Z}^n + \tilde{\boldsymbol{\nu}})$ weighted by the number of codewords $A_{\boldsymbol{\ell}}$ with Lee weight profile $\boldsymbol{\ell}$. We then select a single codeword $\boldsymbol{\nu}$ associated to $\boldsymbol{\ell}$ uniformly over all such $A_{\boldsymbol{\ell}}$ codewords. We output a final lattice vector $\mathbf{x} = \mathbf{x}' + \boldsymbol{\nu}$ by sampling $q\mathbb{Z}^n + (\mathcal{C} + \mathbf{t})$ using one dimensional $\mathbb{Z}$-samplers. The correctness of Algorithm 1 follows by applying the reduction in Theorem 1 with the general method of coset decomposition sampling.

Example 2. Consider the D_n lattice, obtained via binary Construction A as $\mathrm{D}_n = 2\mathbb{Z}^n + \mathcal{P}_n$, where $\mathcal{P}_n$ is the parity-check code and a shift $\mathbf{t} = \mathbf{0}$. There are $\binom{n}{2m}$ vectors of weight $2m$ in $\mathcal{P}_n$. The probability of selecting such coset is

$$p_{2m} = \binom{n}{2m} \frac{\Theta_{\mathbb{Z}+1/2}(4z)^{2m} \Theta_{\mathbb{Z}}(4z)^{n-2m}}{\Theta_{\mathrm{D}_n}(z)}$$

as given in [8, Equation 3]. Hence, to sample in D_n, we proceed as in Algorithm 2, according to [8]. ◊

Sampling in $\mathrm{D}_n + \mathbf{t}$, where $\mathbf{t} \neq \mathbf{0}$ will be discussed next.

5.2 Sampling Construction B Lattices

The Construction B lattice can be written as $\Lambda_{\mathrm{B}}(\overline{\mathcal{C}}) = 2\,\mathrm{D}_n + \overline{\mathcal{C}}$. Thus, sampling from Λ_{B} requires base samplers of the D_n lattice [8, Alg. 7]. Campello and Belfiore provide a sampler for $\Lambda_{\mathrm{B}}(\overline{\mathcal{C}})$ when $\mathbf{t} = \mathbf{0}$, which we summarize below.

Note that any permutation of coordinates is an automorphism of D_n [11, p. 118] and the addition by a lattice vector is an affine automorphism [11, p. 91], so it follows that $2\,\mathrm{D}_n + \mathbf{c} \simeq 2\,\mathrm{D}_n + (1^w, 0^{n-w})$ as lattices since $\overline{\mathcal{C}}$ is doubly even, so the sum of the coordinates of $\mathbf{c}$ is even and therefore $\mathbf{c} \in \mathrm{D}_n$. It follows that the theta series of a coset $2\,\mathrm{D}_n + \mathbf{c}$ depends only on the Hamming weight $w = w_{\mathrm{H}}(\mathbf{c})$. The theta series of $2(\mathrm{D}_n + \mathbf{c}/2)$ [8, Sec. VI] is

Algorithm 2: Sampling over D_n

Require: $\mathrm{D}_n = 2\mathbb{Z}^n + \mathcal{P}_n$ and sampling width s
Ensure: $\mathbf{x} \leftarrow \mathcal{D}_{\mathrm{D}_n,s}(\mathbf{x})$
1: Select $m \in \{1, \ldots, \lfloor n/2 \rfloor\}$ with probability p_{2m}
2: Select a subset $\mathcal{J} \subset \{1, \ldots, n\}$ with size $2m$
3: **for** $j \in \mathcal{J}$ **do**
| $x_j \leftarrow 2 \cdot \mathrm{Sampler}_{\mathbb{Z}+1/2}(s/2)$
end
4: **for** $j \in \{1, \ldots, n\} \setminus \mathcal{J}$ **do**
| $x_j \leftarrow 2 \cdot \mathrm{Sampler}_{\mathbb{Z}}(s/2)$
end
5: **return** $\mathbf{x} = (x_1, \ldots, x_n)$

$$\Theta_{\mathrm{D}_n + \mathbf{c}/2}(z) = \Theta_{\mathrm{D}_w + \left(\frac{1}{2}^w\right)}(4z)\Theta_{\mathrm{D}_{n-w}}(4z) + \Theta_{\mathrm{D}_w + \left(\frac{3}{2}^1, \frac{1}{2}^{w-1}\right)}(4z)\Theta_{\mathrm{D}_{n-w} + (1^1, 0^{n-w-1})}(4z).$$

Let $\mathcal{D}_{\Lambda_\mathrm{B},s}(2\,\mathrm{D}_n(\mathbf{c}) + (1^w, 0^{n-w})) = {}^{\Theta_{2\,\mathrm{D}_n + (1^w, 0^{n-w})}(z)}\!/_{\Theta_{\Lambda_\mathrm{B}}(z)}$ be the probability that a lattice vector sampled from the distribution on Λ_B lies in the coset $2\,\mathrm{D}_n + (1^w, 0^{n-w})$. The sampling procedure works by sampling a Hamming weight w with probability $p_w = A_w \mathcal{D}_{\Lambda_\mathrm{B},s}(2\,\mathrm{D}_n + (1^w, 0^{n-w}))$, similar to the procedure for Construction A. The sampler selects a codeword $\mathbf{c} \in \overline{\mathcal{C}}$ uniformly from the set of all codewords of Hamming weight w. Next, we decompose $\mathrm{D}_n = (\mathrm{D}_w \oplus \mathrm{D}_{n-w}) \bigcup (\overline{\mathrm{D}}_w \oplus \overline{\mathrm{D}}_{n-w})$ where $\overline{\mathrm{D}}_n = \mathrm{D}_n + (1^1, 0^{n-1})$ by [8, Eq. (9)]. We then sample from $\mathrm{D}_w \oplus \mathrm{D}_{n-w}$ with probability $p_{\mathrm{even},w} \triangleq \mathcal{D}_{2\,\mathrm{D}_n + \mathbf{c},s}((2\,\mathrm{D}_w + (1^w)) \oplus 2\,\mathrm{D}_{n-w})$. We sample from the complementary part of the decomposition with probability $1 - p_{\mathrm{even},w}$. This allows us to sample from a shift of a D_n lattice.

We now apply this idea to sampling a shift of $\mathrm{D}_n + \mathbf{t}$ where $\mathbf{t} \in \mathbb{Z}_2^n$. By replacing $\mathbf{c}/2$ with $\mathbf{t}$, we get that the theta series $\mathrm{D}_n + \mathbf{t}$ is

$$\Theta_{\mathrm{D}_n + \mathbf{t}}(z) = \Theta_{\mathrm{D}_w + \mathbf{1}}(4z)\Theta_{\mathrm{D}_{n-w}}(4z) + \Theta_{\mathrm{D}_w + (2^1, 1^{w-1})}(4z)\Theta_{\mathrm{D}_{n-w} + (1^1, 0^{n-w-1})}(4z)$$

where $w = w_\mathrm{H}(\mathbf{t})$. Note that since $\mathbf{t}$ is not necessarily a codeword in $\overline{\mathcal{C}}$, we may have that w is odd. We decompose D_n as before. After that, we sample from $\mathrm{D}_w \oplus \mathrm{D}_{n-w}$ with probability $p_{\mathrm{even},w} \triangleq \mathcal{D}_{\mathrm{D}_n + \mathbf{t},s}((\mathrm{D}_w + (1^w)) \oplus \mathrm{D}_{n-w})$, and from the complementary part of the decomposition with probability $1 - p_{\mathrm{even},w}$. Notice that if w is even, sampling $\mathrm{D}_n + \mathbf{t}$ is equivalent to sampling D_n when $\mathbf{t} = \mathbf{0}$.

5.3 Sampling Construction D Lattices for $L \geq 2$

Decomposing a linear q-ary code using a code formula offers significant advantages in the sampling process when $\mathbf{t} = \mathbf{0}$, as was the case for Construction B, and now we extend to a multilevel Construction D lattice. Consider a filtration of sublattices of a Construction D lattice of the form

$$2^L\mathbb{Z}^n + 2^{L-1}\mathcal{C}_L \subseteq 2^L\mathbb{Z}^n + 2^{L-1}\mathcal{C}_L + 2^{L-2}\mathcal{C}_{L-2} \subseteq \ldots \subseteq 2^L\mathbb{Z}^n + 2^{L-1}\mathcal{C}_L + \ldots + 2\mathcal{C}_2 + \mathcal{C}_1 \triangleq \Lambda_\mathrm{D}$$

where we notice that the first lattice in the filtration is a shifted Construction A lattice scaled by 2^{L-1} for $L \geq 1$.

Our sampler begins at the first level of the filtration and samples $\mathbf{c}_L \in \mathcal{C}_L$ with respect to a scaled Construction A lattice. The sampler proceeds to sample the next codeword by considering the next level of the filtration, and so on. After we have sampled $\mathbf{c}_j$ for $j = 1, \ldots, L$, we sample a lattice point from the overall lattice using base samplers of $\mathbb{Z}^n$ or D_n.

To illustrate, we present a summary of the method for $L = 2$ in Algorithm 3.

Algorithm 3: Sampling for $L = 2$

Require: $\Lambda_{\mathrm{D}} = 4\mathbb{Z}^n + 2\mathcal{C}_2 + \mathcal{C}_1$, $\mathcal{C}_1 \subseteq \mathcal{C}_2$ closed under Schur product, $W_{\mathcal{C}_2}(\cdot)$, $\mathrm{swe}_{2\mathcal{C}_2+\mathcal{C}_1}(\cdot)$ and sampling width s

Ensure: $\mathbf{x} \leftarrow \mathcal{D}_{\Lambda_{\mathrm{D}},s}(\mathbf{x})$

$\Lambda_1 \leftarrow 4\mathbb{Z}^n + 2\mathcal{C}_2$

$A_w \leftarrow \#\{\mathbf{c}_2 \in \mathcal{C}_2 : w_{\mathrm{H}}(\mathbf{c}_2) = w\}$

$\mathcal{S}_w \leftarrow \{\mathbf{c}_2 \in \mathcal{C}_2 : w_{\mathrm{H}}(\mathbf{c}_2) = w\}$

Sample a weight w with probability $p_w = A_w \cdot \mathcal{D}_{\Lambda_1,s}(4\mathbb{Z}^n + 2\tilde{\mathbf{c}}_2)$

Sample $\mathbf{c}_2 \leftarrow \mathcal{S}_w$ uniformly at random

$\Lambda_2 \leftarrow 4\mathbb{Z}^n + 2\mathcal{C}_2 + \mathcal{C}_1 = \Lambda_{\mathrm{D}}$

$A_{\boldsymbol{\ell}} \leftarrow \#\{\mathbf{c}_1 \in \mathcal{C}_1 : [\omega_{0,0}, w_{\mathrm{H}}(\mathbf{c}_1), \omega_{0,1}] = \ell\}$

$\mathcal{S}_{\boldsymbol{\ell}} \leftarrow \{\mathbf{c}_1 \in \mathcal{C}_1 : [\omega_{0,0}, w_{\mathrm{H}}(\mathbf{c}_1), \omega_{0,1}] = \ell\}$

Sample Lee weight profile $\boldsymbol{\ell}$ with probability $p_{\boldsymbol{\ell}} = A_{\boldsymbol{\ell}} \cdot \frac{\mathcal{D}_{\Lambda_{\mathrm{D}},s}(4\mathbb{Z}^n + 2\mathbf{c}_2 + \tilde{\mathbf{c}}_1)}{\mathcal{D}_{\Lambda_1,s}(4\mathbb{Z}^n + 2\tilde{\mathbf{c}}_2)}$

Sample $\mathbf{c}_1 \leftarrow \mathcal{S}_{\boldsymbol{\ell}}$ uniformly at random

$\mathbf{c} \leftarrow 2\mathbf{c}_2 + \mathbf{c}_1$

for $j = 1, \ldots, n$ **do**

  $x_j \leftarrow 4 \cdot \mathrm{Sampler}_{\mathbb{Z}+c_j/4}(s/4)$

end

return $\mathbf{x} = (x_1, \ldots, x_n)$

Alternatively, we can directly use $\mathcal{C} = 2\mathcal{C}_2 + \mathcal{C}_1$ as a code over $\mathbb{Z}_4$ and sample in a single step without defining conditional probabilities. In this case, we can sample the 2-level lattice $\Lambda_{\mathrm{D}} = 4\mathbb{Z}^n + 2\mathcal{C}_2 + \mathcal{C}_1$ by using Algorithm 1 with $q = 4$ and $\mathcal{C} = 2\mathcal{C}_2 + \mathcal{C}_1$. The correctness of Algorithm 3 is analogous to the correctness of Algorithm 4, by restricting it to two levels. We next provide the description and correctness proof for Algorithm 4.

For a 3-level Construction D lattice $\Lambda_{\mathrm{D}} = 8\mathbb{Z}^n + 4\mathcal{C}_3 + 2\mathcal{C}_2 + \mathcal{C}_1$, we repeat what was done for $L = 2$ to sample with respect to the first two levels of the filtration. Next, we sample with respect to the third and final filtration, which is the entire lattice Λ_{D}, under the condition that $\mathbf{c}_3$ and $\mathbf{c}_2$ were previously sampled in the first two steps. The theta series of a coset in this lattice is, according to (11)

$$\Theta_{8\mathbb{Z}^n+4\boldsymbol{\nu}_3+2\boldsymbol{\nu}_2+\boldsymbol{\nu}_1}(z) = \vartheta_3(64z)^{\omega_{0,0,0}}\vartheta_2(64z)^{\omega_{0,0,1}}\left(\frac{\vartheta_2(16z)}{2}\right)^{\omega_{0,1,0}+\omega_{0,1,1},}$$
$$\psi_8(64z)^{\omega_{1,0,0}+\omega_{1,1,1}}\Theta_{\mathbb{Z}+\frac{3}{8}}(64z)^{\omega_{1,1,0}+\omega_{1,0,1}}$$

where $\mathbf{c}_1 \in \mathcal{C}_1$. The probability of a coset depends on the frequency values $\omega_{\boldsymbol{\alpha}}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$, where $\boldsymbol{\alpha} \in \mathbb{F}_2^3$. Now, let $A_{\mathbf{k}}$ be the number of Lee weight profiles $\mathbf{k}$ defined by these frequency values and let $\tilde{\mathbf{c}}_1 \in \mathcal{C}_1$ be any codeword having weight profile $\mathbf{k}$ with respect to fixed $\mathbf{c}_3, \mathbf{c}_2$. Define the conditional probability that we select a vector from a discrete Gaussian on $8\mathbb{Z}^n + 4\mathcal{C}_3 + 2\mathcal{C}_2 + \mathcal{C}_1$ which lies in a coset of the form $8\mathbb{Z}^n + 4\mathbf{c}_3 + 2\mathbf{c}_2 + \tilde{\mathbf{c}}_1$ given that we have previously sampled $\mathbf{c}_3, \mathbf{c}_2$. The probability that we have already sampled $\mathbf{c}_3$ and $\mathbf{c}_2$ is the probability that step 1 outputs $\mathbf{c}_3$ and step 2 outputs $\mathbf{c}_2$. Overall, it follows that

$$p_{\mathrm{cond}_2} \triangleq \frac{\mathcal{D}_{8\mathbb{Z}^n+4\mathcal{C}_3+2\mathcal{C}_2+\mathcal{C}_1,s}(8\mathbb{Z}^n + 4\mathbf{c}_3 + 2\mathbf{c}_2 + \tilde{\mathbf{c}}_1)}{\mathcal{D}_{8\mathbb{Z}^n+4\mathcal{C}_3+2\mathcal{C}_2,s}(8\mathbb{Z}^n + 4\mathbf{c}_3 + 2\tilde{\mathbf{c}}_2)}.$$

We sample $\mathbf{c}_1$ in two steps: first, we choose a Lee weight profile $\mathbf{k}$ with probability $p_{\mathbf{k}} = A_{\mathbf{k}} \cdot p_{\mathrm{cond}_2}$, then we sample a codeword $\mathbf{c}_1$ uniformly from all codewords in $\mathcal{C}_1$ that have Lee weight profile k with respect to $\mathbf{c}_3, \mathbf{c}_2$. This selects a codeword $\mathbf{c}_1 \in \mathcal{C}_1$ with probability $p_{\mathbf{k}}/A_{\mathbf{k}}$.

The final step is to sample a vector from $\mathcal{D}_{8\mathbb{Z}^n+4\mathbf{c}_3+2\mathbf{c}_2+\mathbf{c}_1,s}$ by recursively applying one-dimensional samplers of $\mathbb{Z}$ and its shifts. The sampler outputs a lattice vector total probability

$$\begin{aligned}
&\mathcal{D}_{\Lambda_1,s}(8\mathbb{Z}^n + 4\mathbf{c}_3) \cdot \tfrac{\mathcal{D}_{\Lambda_2,s}(8\mathbb{Z}^n+4\mathbf{c}_3+2\mathbf{c}_2)}{\mathcal{D}_{\Lambda_1,s}(8\mathbb{Z}^n+4\mathbf{c}_3)} \cdot \tfrac{\mathcal{D}_{\Lambda_{\mathrm{D}},s}(8\mathbb{Z}^n+\mathbf{c})}{\mathcal{D}_{\Lambda_2,s}(8\mathbb{Z}^n+4\mathbf{c}_3+2\mathbf{c}_2)}.\\
&\mathcal{D}_{8\mathbb{Z}^n+4\mathbf{c}_3+2\mathbf{c}_2+\mathbf{c}_1,s}(8\mathbf{z}+\mathbf{c}) = \mathcal{D}_{\Lambda_{\mathrm{D}},s}(8\mathbb{Z}^n + \mathbf{c})\mathcal{D}_{8\mathbb{Z}^n+\mathbf{c},s}(8\mathbf{z}+\mathbf{c})\\
&\qquad\qquad = \mathcal{D}_{\Lambda_{\mathrm{D}},s}(8\mathbf{z}+\mathbf{c}),
\end{aligned}$$

where $\Lambda_1 \subseteq \Lambda_2 \subseteq \Lambda_{\mathrm{D}} = 8\mathbb{Z}^n + 4\mathcal{C}_3 + 2\mathcal{C}_2 + \mathcal{C}_1$ is the lattice filtration that we consider and $\mathbf{c} = 4\mathbf{c}_3 + 2\mathbf{c}_2 + \mathbf{c}_1$. This yields a sampling method that does not restrict the sampling width s since it samples exactly from the true distribution of the lattice. We can extend this sampling technique to higher levels, as in Algorithm 4. The computation of Lee weight profiles can be performed offline, and the frequency of each Lee weight profile is contained in the symmetrized weight enumerator. This information is used in the online phase of sampling to define the relevant probabilities. We also sample codewords corresponding to a chosen Lee weight profile in the online phase.

5.4 Algorithm Complexity

For a general L-level Construction D lattice, we sample by recursively selecting codewords to obtain a coset representative $\mathbf{c} = 2^L\mathbf{c}_L + \ldots + 2\mathbf{c}_2 + \mathbf{c}_1$, then we sample a final lattice vector via one-dimensional samplers over the integers. Sampling these codewords depends on Lee weight profiles. In general, we may not have this information. Considering the Schur product of codewords might be an alternative; see Appendix C.

In terms of complexity, Algorithm 4 first samples $\mathbf{c}_L$ with respect to its Hamming weight. We can compute the Hamming weight enumerator of $\mathcal{C}_L$ offline. To sample a codeword of a given weight, we can use a lookup table of the codewords sorted by weight. The online lookup time is $O(\log|\mathcal{C}_L|) = O(k_L)$

Algorithm 4: Sampling for $L \geq 3$

Require: $\Lambda_{\mathrm{D}} = 2^L\mathbb{Z}^n + 2^{L-1}\mathcal{C}_L + \ldots + 2\mathcal{C}_2 + \mathcal{C}_1$, $\mathcal{C}_j$ closed under Schur product for $1 \leq j \leq L$, $W_{\mathcal{C}_L}(\cdot)$, symmetrized weight enumerators of linear combinations of $\mathcal{C}_L, \ldots, \mathcal{C}_1$, sampling width s

Ensure: $\mathbf{x} \leftarrow \mathcal{D}_{\Lambda_{\mathrm{D}},s}(\mathbf{x})$

1: Sample $\mathbf{c}_L$ and $\mathbf{c}_{L-1}$ as in Algorithm 3

2: $\Lambda_3 \leftarrow 2^L\mathbb{Z}^n + 2^{L-1}\mathcal{C}_L + 2^{L-2}\mathcal{C}_{L-1} + 2^{L-3}\mathcal{C}_{L-2}$

3: $A_{\mathbf{k}} \leftarrow \#\{\mathbf{c}_{L-2} \in \mathcal{C}_{L-2} : [\omega_{0,0,0}, \omega_{1,0,0} + \omega_{1,1,1}, \omega_{0,1,0} + \omega_{0,1,1}, \omega_{1,1,0} + \omega_{1,0,1}, \omega_{0,0,1}] = \mathbf{k}\}$

4: $\mathcal{S}_{\mathbf{k}} \leftarrow \{\mathbf{c}_{L-2} \in \mathcal{C}_{L-2} : [\omega_{0,0,0}, \omega_{1,0,0} + \omega_{1,1,1}, \omega_{0,1,0} + \omega_{0,1,1}, \omega_{1,1,0} + \omega_{1,0,1}, \omega_{0,0,1}] = \mathbf{k}\}$

5: Sample a Lee weight profile k with probability $p_{\mathbf{k}} = A_{\mathbf{k}} \cdot \frac{\mathcal{D}_{\Lambda_3,s}(2^L\mathbb{Z}^n + 2^{L-1}\mathbf{c}_L + 2^{L-2}\mathbf{c}_{L-1} + 2^{L-3}\tilde{\mathbf{c}}_{L-2})}{\mathcal{D}_{\Lambda_2,s}(2^L\mathbb{Z}^n + 2^{L-1}\mathbf{c}_L + 2^{L-2}\tilde{\mathbf{c}}_{L-1})}$

6: Sample $\mathbf{c}_{L-2} \leftarrow \mathcal{S}_{\mathbf{k}}$ uniformly at random

7: Repeat lines 4-8 for $\Lambda_4 \subseteq \ldots \subseteq \Lambda_{L-1} \subseteq \Lambda_L = \Lambda_{\mathrm{D}}$

8: $\mathbf{c} \leftarrow 2^L\mathbf{c}_L + \ldots + 2\mathbf{c}_2 + \mathbf{c}_1$

9: **for** $j = 1, \ldots, n$ **do**
    $x_j \leftarrow 2^L \cdot \mathrm{Sampler}_{\mathbb{Z}+c_j/2^L}(s/2^L)$
    end

10: **return** $\mathbf{x} = (x_1, \ldots, x_n)$

where k_L is the dimension of $\mathcal{C}_L$, but the storage complexity of the table is at least the size of the code, which is exponential in k_L. Clearly, this is not polynomial, but we present cases where we can avoid using such lookup tables and still sample with respect to Hamming weight efficiently in Section 6.

Algorithm 4 proceeds to recursively sample from cosets of $\mathcal{C}_{L-1}, \ldots, \mathcal{C}_1$ with respect to Lee weight profiles. We can compute Lee weight profiles offline via the Schur product over $\mathcal{C}_{L-1}, \ldots, \mathcal{C}_1$. We need to compute the Lee weight profiles of $\mathcal{C}_L$ with $\mathcal{C}_{L-1}$ in the second step, then repeat with $\mathcal{C}_L, \mathcal{C}_{L-1}$ and $\mathcal{C}_{L-2}$ in the third step, and so on. However, notice that we can recover the Lee weight profiles in each step by simply performing the computation over all of the codes $\mathcal{C}_L, \ldots, \mathcal{C}_1$. For example, to extract the Lee weight profiles of $\mathcal{C}_L$ with $\mathcal{C}_{L-1}$ from this larger computation, we find when $\mathbf{c}_{L-2} = \ldots = \mathbf{c}_1 = \mathbf{1}$. The complexity of this larger computation is $O(n \cdot |\mathcal{C}_L| \cdot \ldots \cdot |\mathcal{C}_1|) = O(n2^{k_L + \ldots + k_1})$ where k_j is the dimension of $\mathcal{C}_j$. This is much larger than $O(k_L 2^{k_L})$, so the overall offline complexity is given by $O(n2^{k_L+\ldots+k_1})$.

The online complexity is harder to estimate since, in general, it is not efficient to sample codes of a given Hamming weight nor of a given Lee weight profile. If we store a lookup table for codewords associated to given Lee weight profiles, then this cost would be $O(\log(|\mathcal{C}_L| \cdot \ldots \cdot |\mathcal{C}_1|)) = O(k_L \ldots k_1) = O(n^L)$ since $k_j \leq n$ for all j. Added to this, we would have the complexity of applying n one-dimensional samplers over a shift of the integers which has complexity $O(n)$, so the overall online complexity is given by $O(n^L)$. However, it should be noted that the storage for such a lookup table would be exponential.

We point out that computing Lee weight profiles do not depend on the sampling width s, so the choice of s does not impact the offline complexity of the sampler. On the other hand, we can simplify the overall sampling procedure if we allow s to be sufficiently above smoothing. For sufficiently large values of s, we can sample by choosing codewords uniformly at random, since the sampled lattice vectors now appear uniformly distributed over $\mathbb{R}^n/\Lambda$. This comes at a loss of increasing the sampling width.

Limitations. As a consequence, the sampling algorithm for Construction D with arbitrary q and level L is not usually efficient. This is expected since the major challenge of our sampling method relies on sampling codewords with respect to Lee weight profiles, which is a hard problem. Even for the simplest case when $L = 1$, we may not be able to sample a codeword of specific Hamming weight w efficiently. However, given that the codewords can be sampled efficiently, the sampling method is consequently efficient (and in particular, we can make the sampling width s close to the smoothing parameter of the associated lattice). Sampling codewords can be done efficiently for codes of reasonably small dimensions by enumeration or codes with some special structure [8, p. 167]. For example, the $(n, n-1)_2$ code consists of all n-tuples with an even number of non-zero coordinates. Due to its special structure, we can sample such a code by randomly sampling a subset of coordinates of even size. It is unclear how to optimize the computation of Lee weight profiles using the properties of the Schur product for certain structured codes. We provide some details for this in Appendix C.2.

6 Sampling Remarkable Lattices

We provide applications of our sampler to some remarkable lattices. Recall that our method only applies to lattices that can be expressed via a code formula as in (1).

6.1 Sampling Root Lattices

A D_n lattice is constructed from the $(n, n-1)_2$ even weight code $\mathcal{C}_1$. We can sample D_n using Algorithm 2.

We can efficiently sample the exceptional root lattices E_7 and E_8. The Barnes-Wall lattice in dimension 8 will cover the case of E_8. We can build E_7 from Construction A using the $(7, 4)_2$ Hamming code.

Alternatively, we can use the fact that $\mathrm{E}_n \triangleq \mathrm{D}_n^+$ from [11,14] when n is even. All cosets of $\mathrm{D}_n + (1/2, \ldots, 1/2)$ have equal theta series by [8, Prop. 3]. Denote the probability that a lattice point in D_n^+ lies in the D_n part of the decomposition as $\mathcal{D}_{\mathrm{D}_n^+,s}(\mathrm{D}_n) = \Theta_{\mathrm{D}_n}(z)/\Theta_{\mathrm{D}_n^+}(z)$. With probability $\mathcal{D}_{\mathrm{D}_n^+,s}(\mathrm{D}_n)$, sample from D_n. Otherwise, sample from $\mathrm{D}_n + (1/2, \ldots, 1/2)$. This is done by sampling $\mathbf{c} \in \mathcal{P}_n$ uniformly at random since all cosets of $\mathrm{D}_n + (1/2, \ldots, 1/2)$ have equal theta series.

To sample from $\mathrm{E}_8 + \mathbf{t}$, we have that $\mathrm{E}_8 + \mathbf{t} = \mathrm{D}_8^+ + \mathbf{t}$. With probability $\mathcal{D}_{\mathrm{E}_8+\mathbf{t},s}(\mathrm{D}_8+\mathbf{t}) = \Theta_{\mathrm{D}_8+\mathbf{t}}(z)/\Theta_{\mathrm{E}_8+\mathbf{t}}(z)$, we use the sampler for $\mathrm{D}_8 + \mathbf{t}$ as in Section 5.2. Otherwise we modify the sampler from Section 5.2 to sample from $\mathrm{D}_8 + (1/2, \ldots, 1/2) + \mathbf{t}$.

6.2 Sampling Leech Lattice

We propose an alternative way to [8, Sec. VII] of sampling the Leech lattice Λ_{24}. We can write a scaled version of the Leech lattice as

$$\Lambda_{24} = (\mathbf{0} + 2\mathcal{G}_{24} + 4\mathcal{P}_{24} + 8\mathbb{Z}^{24}) \cup (\mathbf{1} + 2\mathcal{G}_{24} + 4\mathcal{O}_{24} + 8\mathbb{Z}^{24}), \tag{12}$$

where $\mathcal{G}_{24}$ is the $(24, 12)_2$ Golay code, $\mathcal{P}_{24}$ is the $(24, 23)_2$ parity-check code and $\mathcal{O}_{24}$ is a coset of $\mathcal{P}_{24}$ containing all codewords in $\mathbb{F}_2^{24}$ with odd weights, i.e., $\mathcal{O}_{24} = \{\mathbf{x} \in \mathbb{F}_2^{24} : \sum_{j=1}^{24} x_j \equiv 1 \bmod 2\} = (1, 0, \ldots, 0) + \mathcal{P}_{24}$. Note that $\mathcal{O}_{24} = \mathcal{P}_{24} + (1, 0^{23})$ so we can rewrite this as

$$\begin{aligned}
\Lambda_{24} &= (\mathbf{0} + 2\mathcal{G}_{24} + 4\mathcal{P}_{24} + 8\mathbb{Z}^{24}) \cup (\mathbf{1} + 2\mathcal{G}_{24} + 4\mathcal{O}_{24} + 8\mathbb{Z}^{24}) \\
&= (\mathbf{0} + 2\mathcal{G}_{24} + 4\mathcal{P}_{24} + 8\mathbb{Z}^{24}) \cup (\mathbf{1} + 2\mathcal{G}_{24} + 4\mathcal{P}_{24} + (4, 0^{23}) + 8\mathbb{Z}^{24}) \\
&= (\mathbf{0} + 2\mathcal{G}_{24} + 4\mathcal{P}_{24} + 8\mathbb{Z}^{24}) \cup (\mathbf{1} + 2\mathcal{G}_{24} + 4(2\mathbb{Z}^{24} + \mathcal{P}_{24} + (1, 0^{23}))) \\
&= 2\Lambda_B(\mathcal{G}_{24}) \cup (\mathbf{1} + 2\mathcal{G}_{24} + 4\overline{D}_{24}),
\end{aligned}$$

where $\overline{\mathrm{D}}_n = \mathrm{D}_n + (1, 0^{n-1})$. By the decomposition given by $\mathrm{D}_n = (\mathrm{D}_w \oplus \mathrm{D}_{n-w}) \cup (\overline{\mathrm{D}}_w \oplus \overline{\mathrm{D}}_{n-w})$ in [8], we have that

$$\begin{aligned}
\overline{\mathrm{D}}_n &= \mathrm{D}_n + (1, 0^{n-1}) \\
&= (\mathrm{D}_w + (1, 0^{w-1}) \oplus \mathrm{D}_{n-w}) \cup (\overline{\mathrm{D}}_w + (1, 0^{w-1}) \oplus \overline{D}_{n-w}) \\
&= (\overline{D}_w \oplus \mathrm{D}_{n-w}) \cup (\mathrm{D}_w + (2, 0^{w-1}) \oplus \overline{\mathrm{D}}_{n-w}) \\
&\cong (\overline{\mathrm{D}}_w \oplus \mathrm{D}_{n-w}) \cup (\mathrm{D}_w \oplus \overline{\mathrm{D}}_{n-w}),
\end{aligned}$$

since $(2, 0^{w-1}) \in \mathrm{D}_w$. Consider a coset of the form $4\overline{D}_{24} + 2\mathbf{c} + \mathbf{1}$. By [8, Eq. 10], we have that

$$\begin{aligned}
\Theta_{\overline{\mathrm{D}}_{24}+\mathbf{c}/2+(1/4^{24})}(16z) &= \Theta_{\mathrm{D}_w+(3/2,1/2^{w-1})+(1/4^w)}(16z)\Theta_{\mathrm{D}_{n-w}+(1/4^{n-w})}(16z) \\
&\quad + \Theta_{\mathrm{D}_w+(1/2^w)+(1/4^w)}(16z)\Theta_{\mathrm{D}_{n-w}+(1,0^{n-w-1})+(1/4^{n-w})}(16z) \\
&= \Theta_{\mathrm{D}_w+(7/4,3/4^{w-1})}(16z)\Theta_{\mathrm{D}_{n-w}+(1/4^{n-w})}(16z) \\
&\quad + \Theta_{\mathrm{D}_w+(3/4^w)}(16z)\Theta_{\mathrm{D}_{n-w}+(5/4,1/4^{n-w-1})}(16z),
\end{aligned}$$

so the theta series depends on the Hamming weight w of $\mathbf{c} \in \mathcal{G}_{24}$. The Hamming weight distribution of the Golay code is well known, which can be read off of the Hamming weight enumerator $W_{\mathcal{G}_{24}}(x, y) = x^{24} + 759x^8y^{16} + 2576x^{12}y^{16} + 759x^{16}y^8 + y^{24}$ [35].

6.3 Sampling Barnes-Wall Lattices

Dimensions 4 and 8. The Barnes-Wall lattices BW_4 and BW_8 are simply Construction A lattices from $\mathrm{RM}(1,2)$ and $\mathrm{RM}(1,3)$, respectively. Reed-Muller codes of order 1 have codewords with weight either $0, n/2$ or n. When $w = 0, n$ we have either $\mathbf{0}$ or $\mathbf{1}$. When $w = n/2$, there are $2^{m+1}-2$ codewords of weight w. These are all of the codewords in $\mathrm{RM}(1,m)$ except the all ones or all zeros codeword. We can sample from this weight class using rejection sampling with $(2^{m+1}-2)/2^{m+1}$ iterations. As m increases, this goes to 1.

Dimension 16. The Barnes-Wall lattice BW_{16} is a Construction B lattice given by the code formula $4\mathbb{Z}^{16} + 2\,\mathrm{RM}(3,4) + \mathrm{RM}(1,4)$, since $\mathrm{RM}(1,4)$ is a doubly even code. Therefore, we can apply the sampler proposed in Section 5.2 and need only to sample a codeword of fixed weight w from $\mathrm{RM}(1,4)$. To do this, we apply rejection sampling with $= 1.067$ iterations.

Dimension 32. The Barnes-Wall lattice BW_{32} has the code formula $4\mathbb{Z}^{32} + 2\,\mathrm{RM}(3,5) + \mathrm{RM}(1,5)$ where $\mathrm{RM}(3,5)$ is not the parity-check code. Therefore, we are in the case of Section 5.3, which requires us to sample a codeword $\mathbf{c}_2 \in \mathrm{RM}(3,5)$ first by Hamming weight and $\mathbf{c}_1 \in \mathrm{RM}(1,5)$ by computing Lee weight profiles. For sampling $\mathbf{c}_2$ by Hamming weight, we can enumerate the codewords of weight w using design theory which is detailed in the Appendix D. We can compute Lee weight profiles for the second step offline or even move this step online since the size of $\mathrm{RM}(1,5)$ is only twice the dimension of the lattice.

Dimensions $\geq$ 64. As the dimensions of the Barnes-Wall lattices grow, so do the dimensions of the Reed-Muller codes used to construct them. The online complexity will increase in response since we need to sample with respect to a chosen Lee weight profile over large codes.

Similarly to Barnes-Wall lattices, we briefly mention a family of Construction D lattices not addressed in the work of [14], denoted as B_n with $n = 2^m$, which are built from a tower of BCH codes. When the dimensions of the associated BCH codes are reasonably small, our sampler can be applied efficiently. These lattices are considered good packings in the sense that they are dense and have efficient decoding algorithms [4, 29].

7 Improvements to State-of-the-Art Sampling

We discuss the improvements of our sampling method in some families of lattices, focusing on efficiency and restrictions on the sampling width. A summary is presented in Table 1 and supporting Sage [35] code can be found here https://anonymous.4open.science/r/q-ary-sampling-3337.

Given that the samplers in [14] mostly rely on $\mathbb{Z}$ samplers and do not require much additional computation whereas we require sampling codewords, we estimate a lower bound for the running time of our sampling algorithms by counting

the number of calls to a $\mathbb{Z}$ sampler, which is used in a black-box manner. We assume the weight enumerators of all cosets of a linear code are known and ignore the costs of sampling codewords (which is cheap for small lattices), so this will give a reasonable estimate of the speed-up. Instead of explicitly stating these costs, we give some explanation on how one might sample codewords efficiently in several cases.

Root lattices. In [14], the authors provide base samplers for the root lattices D_n ($n \geq 1$) and E_n ($n = 6, 7, 8$). We can sample these lattices with the method proposed in Section 5.2 using n one-dimensional samplers of $\mathbb{Z}$ or $\mathbb{Z} + 1/2$. This is compared to the sampler in [14], which performs n one-dimensional samplers of $\mathbb{Z}$ twice on average. The width of the sampler in [14] is approximately the smoothing parameter $s \approx \eta_\epsilon(\mathrm{D}_n)$ whereas the sampler proposed in Section 5.2 allows us to choose $s = \eta_\epsilon(\mathrm{D}_n)$ exactly. In terms of the number of calls to a $\mathbb{Z}$ sampler, our sampler is approximately twice as fast as [14]. This is supported by simulations performed in [35] for $n = 8, 16, 24$.

We can sample from E_7 as a Construction A lattice with the $(7, 4)_2$ Hamming code. The codewords of weight 3 form a 1-(7,3,3) design, so every codeword of weight 3 uniquely decodes to a block in the design (see Appendix D). We can sample a codeword of weight 3 by sampling a block in the design via a choosing procedure as in [8, Sec. VII], then decoding. To sample a codeword of weight 4, we simply add the all-ones vector to a codeword of weight 3 due to the code's symmetry. Overall, we require 7 one-dimensional samplers of shifts of $\mathbb{Z}$ whereas [14] requires repeating the E_8 base sampler 4 times on average, which results in 64 one-dimensional $\mathbb{Z}$ samplers, so our sampler is roughly 9 times faster. Our simulation compares by sampling 100,000 samples 8.64 times faster.

Regarding the E_8 (or BW_8) lattice, an improved gadget framework was proposed in [14, Sec. 10]. It was shown a gain of 9 bits in security and 113 bytes of the signature size for Eagle-1024 while maintaining tight sampling. We can replace their E_8 sampler with our sampler, which is more efficient and still samples at the smoothing parameter, preserving the security gains.

Recall that E_8 is a Construction A lattice given by the code formula $2\mathbb{Z}^8 + \mathrm{RM}(1, 3)$. Using the Construction A sampler, we first sample a weight $w \in \{0, 4, 8\}$, and then we sample a codeword with the corresponding weight. If $w = 0, 8$, we obtain either all-zero or all-one codeword. Otherwise, all remaining codewords have weight 4. We apply rejection sampling to obtain a codeword of weight 4 with 1.143 iterations on average. The sampler then applies 8 one-dimensional samplers of $\mathbb{Z}$ and $\mathbb{Z} + 1/2$. Without the small cost of sampling codewords via rejection sampling, the complexity of our E_8 sampler is dominated by sampling over the integers 8 times. In contrast, the sampling method from [14] requires sampling a shift of D_8 with rejection sampling repeated 11 times to optimize the sampling width, which itself uses 8 samplers over $\mathbb{Z}$ twice on average. As a result, we expect our algorithm to be around 22 times faster. In fact, our simulations showed that our algorithm obtains 100,000 samples ≈ 25 times faster than [14].

Barnes-Wall lattices up to Dimension 64. For the Barnes-Wall lattices of dimension 4,8, and 16, our sampler coincides with the Construction A and B samplers. In the state-of-the-art sampling of [14], the authors apply their idea of domain extension to sample a Barnes-Wall lattice BW_n using $\mathrm{BW}_{n/2}$ twice. They can sample a distribution of statistical distance at most 6ϵ to $\mathcal{D}_{\mathrm{BW}_n,s}$ given that they have a discrete Gaussian sampler of $\mathrm{BW}_{n/2}$ with $s > \eta_\epsilon(\mathrm{BW}_{n/2})$ [14, Cor. 1].

Compared to the state-of-the-art sampling in [14], we can improve the sampling for BW_4 and BW_8 using Construction A sampling. For BW_{16}, our sampler is 22 times faster and samples at a width equal to $\eta_\epsilon(\mathrm{BW}_{16})$. This is because to sample BW_{16} in [14], we sample BW_8 which requires, on average, 16 one-dimensional $\mathbb{Z}$ samplers due to rejection sampling and gets repeated 11 times on average to optimize the sampling width $s \approx \eta_\epsilon(\mathrm{E}_8)$. Compared with our sampler for BW_{16}, we sample a shift of D_{16} since this is a Construction B lattice, which translates to 16 one-dimensional samplers of shifts of $\mathbb{Z}$. In fact, we find that for general BW_n, if we can sample with respect to Lee weight profiles efficiently, we estimate our sampler to be 11 times faster than in [14]. For our simulation with BW_{16}, we obtain 100,000 samples 9.41 times faster than [14], which is reasonably close to our expectation.

Leech lattice. The running time of sampling the Leech lattice for Espitau et. al [14] is dominated by the complexity of a E_8 sampler by their choice of filtration $2\,\mathrm{E}_8 \subset T_\theta \subset T$ where $T_\theta \cong T \cong E_8$ [14, Sec. 8.2.3]. This requires 3 calls to an E_8 sampler and samples at a width that is approximately the smoothing parameter of E_8 [14, Alg. 12], which is above $\eta_\epsilon(\Lambda_{24})$.

Recall $\Lambda_{24} = \Lambda_B(\mathcal{G}_{24}) \cup (\mathbf{1} + 2\mathcal{G}_{24} + 4\overline{\mathrm{D}}_{24})$. We sample from the Construction B part, as detailed in [8, p. 169], or from the $\mathbf{1} + 2\mathcal{G}_{24} + 4\overline{D}_{24}$ part using the Hamming weight of codewords in $\mathcal{G}_{24}$ following the procedure in [8, Sec. 6]. This samples twice as fast with $s = \eta_\epsilon(\Lambda_{24})$ since we require 24 samplers of a shift of D_n, which in turn requires 24 one-dimensional samplers of shifts of $\mathbb{Z}$ [8, Alg. 7]. In comparison, the E_8 sampler in [14] requires, on average, 16 one-dimensional $\mathbb{Z}$ samplers, which get called upon three times to sample the Leech lattice.

Hexagonal lattice. Even if A_2 is not a Construction A lattice, we include it here for completeness since we use a technique based on coset decomposition [8, Sec. V].

Espitau et al. [14] propose an efficient sampler for the A_2 root lattice, which the authors apply to sample in $\mathcal{R}_m = \mathbb{Z}[\zeta_m]$ where $m = 2^\ell 3^k$ for Mitaka. Using the fact that $\mathcal{R}_m \cong \bigoplus_{i=1}^{m/6} \mathcal{R}_3$, they sample in $\mathcal{R}_m$ by independently sampling A_2 $m/6$ times [14, Sec. 9]. The width of the proposed A_2 sampler is at the smoothing parameter $\eta_\epsilon(\mathrm{A}_2)$ for sufficiently small ϵ [14, p. 40] and, on average, requires 9 repetitions of the E_8 sampler to acquire four samples in A_2.

Following [8, Sec. V], we have the coset decomposition $\mathrm{A}_2 = \bigcup_{\mathbf{t}\in\mathcal{T}} \Lambda' + \mathbf{t}$ where $\Lambda' = \mathbb{Z} \oplus \sqrt{3}\mathbb{Z}$ and $\mathcal{T} = \{(0,0), (1/2, \sqrt{3}/2)\}$. Sample $\mathbf{t} = (0,0)$ with probability $\mathcal{D}_{\mathrm{A}_2,s}(\mathbb{Z} \oplus \sqrt{3}\mathbb{Z})$, then sample $\mathbf{x} \in \mathbb{Z} \oplus \sqrt{3}\mathbb{Z}$ from $\mathcal{D}_{\mathbb{Z}\oplus\sqrt{3}\mathbb{Z},s}(\mathbf{x})$ using two (scaled) one-dimensional samplers of $\mathbb{Z}$. Otherwise, sample $\mathbf{t} = (1/2, \sqrt{3}/2)$

with remaining probability $\mathcal{D}_{\mathrm{A}_2,s}(\mathbb{Z} \oplus \sqrt{3}\mathbb{Z} + \mathbf{t})$ and a lattice vector $\mathbf{x} + \mathbf{t}$ with probability $\mathcal{D}_{\mathbb{Z}\oplus\sqrt{3}\mathbb{Z}+\mathbf{t},s}(\mathbf{x} + \mathbf{t})$. We can sample $\mathbf{x} + \mathbf{t}$ with two (scaled) one-dimensional samplers of $\mathbb{Z} + 1/2$. This outputs four samples in A_2 approximately 18 times faster. This comes from the fact that to obtain four samples of A_2, the coset decomposition sampler requires 8 samplers over a shift of $\mathbb{Z}$ whereas the sampler in [14] repeats an E_8 sampler 9 times on average which itself requires 2 repetitions of 8 samplers over $\mathbb{Z}$. In terms of the number of samplings over the integers, we expect our sampler to be 18 times faster. We compared our algorithm with the running time of the E_8 sampler in [14] repeated the expected 9 times, which is the main work behind their A_2 sampler. Our simulation showed that it performed better than estimated and was ≈ 32 times faster over 100,000 samples.

References

1. Abdelghany, N., Wood, J.A.: Failure of the MacWilliams identities for the Lee weight enumerator over $\mathbb{Z}_m$, $m \geq 5$. Discrete Mathematics **343**(11), 112036 (2020)
2. 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 '15, Association for Computing Machinery, New York, NY, USA (2015)
3. 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 (2015)
4. Barnes, E.S., Sloane, N.J.A.: New lattice packings of spheres. Canadian Journal of Mathematics **35**(1), 117–130 (1983)
5. Bollauf, M.F., Lin, H.Y., Ytrehus, Ø.: Secrecy gain of formally unimodular lattices from codes over the integers modulo 4. IEEE Transactions on Information Theory **70**(5), 3309–3329 (2023)
6. Borwein, J., Borwein, P.: Pi and the AGM: a study in analytic number theory and computational complexity. Wiley-Interscience and Canadian Mathematics Series of Monographs and Texts, Wiley (1987)
7. 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. pp. 575–584. Association for Computing Machinery, New York, NY, USA (2013)
8. Campello, A., Belfiore, J.: Sampling algorithms for lattice Gaussian codes. In: 24th International Zurich Seminar on Communications. pp. 165–169 (2016)
9. Cascudo, I., Gundersen, J.S., Ruano, D.: Squares of matrix-product codes. Finite Fields and Their Applications **62**(2) (2020)
10. Charpin, P.: Weight distributions of cosets of two-error-correcting binary BCH codes, extended or not. IEEE Transactions on Information Theory **40**(5), 1425–1442 (1994)
11. Conway, J., Sloane, N.: Sphere packings, lattices and groups, A Series of Comprehensive Studies in Mathematics, vol. 290. Springer, 3rd edn. (1999)
12. Ducas, L., van Woerden, W.: On the lattice isomorphism problem, quadratic forms, remarkable lattices, and cryptography. In: Advances in Cryptology – EUROCRYPT 2022. pp. 643–673. Springer International Publishing, Cham (2022)

13. Duke, W.: On codes and Siegel modular forms. International Mathematics Research Notices **1993**(5), 125–136 (1993)
14. Espitau, T., Wallet, A., Yu, Y.: On Gaussian sampling, smoothing parameter and application to lattice signatures. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology – ASIACRYPT 2023. pp. 65–97. Springer Nature Singapore, Singapore (2023)
15. Forney, D.: Coset codes I: introduction and geometrical classication. IEEE Transactions on Information Theory **34**(5), 1123–1151 (1989)
16. 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. p. 197–206. STOC '08, Association for Computing Machinery, New York, NY, USA (2008)
17. Harshan, J., Viterbo, E., Belfiore, J.: Construction of Barnes-Wall lattices from linear codes over rings. In: 2012 IEEE International Symposium on Information Theory. pp. 3110–3114. IEEE (2012)
18. Hörmann, F., van Woerden, W.: FuLeakage: Breaking FuLeeca by learning attacks. In: Advances in Cryptology – CRYPTO 2024. Lecture Notes in Computer Science, vol. 14925, pp. 253–286. Springer (2024)
19. Hu, S., Nebe, G.: Strongly perfect lattices sandwiched between Barnes–Wall lattices. Journal of the London Mathematical Society **101** (2018)
20. Kositwattanarerk, W., Oggier, F.E.: Connections between Construction D and related constructions of lattices. Designs, Codes and Cryptography **73**, 441–455 (2013)
21. Ling, C., Belfiore, J.C.: Achieiving AWGN channel capacity with lattice Gaussian coding. IEEE Transactions on Information Theory **60**(10), 5918–5929 (2014)
22. Ling, C., Luzzi, L., Belfiore, J.C., Stehlé, D.: Semantically secure lattice codes for the Gaussian wiretap channel. IEEE Transactions on Information Theory **60**(10), 6399–6416 (2014)
23. MacWilliams, F.J., Sloane, N.J.A.: The theory of error-correcting codes. North-Holland, Amsterdam, The Netherlands (1977)
24. Maher, D.P.: Lee polynomials of codes and theta functions of lattices. Canadian Journal of Mathematics **30**(4), 738–747 (1978)
25. Micciancio, D., Nicolosi, A.: Efficient bounded distance decoders for Barnes-Wall lattices. In: 2008 IEEE International Symposium on Information Theory. pp. 2484–2488 (2008)
26. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on Gaussian measures. SIAM Journal on Computing **37**(1), 267–302 (2007)
27. Micciancio, D., Regev, O.: Lattice-based cryptography, pp. 147–191. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
28. Micciancio, D., Walter, M.: Gaussian sampling over the integers: efficient, generic, constant-time. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10402, pp. 455–485. Springer (2017)
29. Mook, E., Peikert, C.: Lattice (list) decoding near Minkowski's inequality. IEEE Transactions on Information Theory **68**, 863–870 (2020), https://api.semanticscholar.org/CorpusID:221994997
30. Peikert, C.: An efficient and parallel Gaussian sampler for lattices. In: Rabin, T. (ed.) Advances in Cryptology – CRYPTO 2010. pp. 80–97. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)

31. Randriambololona, H.: On products and powers of linear codes under componentwise multiplication. Contemporary Mathematics **637**, 3–78 (2015)
32. Regev, O., Stephens-Davidowitz, N.: A reverse Minkowski theorem. In: Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing. p. 941–953. Association for Computing Machinery, New York, NY, USA (2017)
33. Ritterhoff, S., Maringer, G., Bitzer, S., Weger, V., Karl, P., Schamberger, T., Schupp, J., Wachter-Zeh, A.: FuLeeca: A Lee-based signature scheme. In: Esser, A., Santini, P. (eds.) Code-Based Cryptography. pp. 56–83. Springer Nature Switzerland, Cham (2023)
34. Shparlinski, I.E., Steinfeld, R.: Noisy Chinese remaindering in the Lee norm. Journal of Complexity **20**(2), 423–437 (2004)
35. The Sage Developers: SageMath, the Sage Mathematics Software System (2022), https://www.sagemath.org, dOI 10.5281/zenodo.6259615
36. Wan, Z.X.: Quaternary codes. World Scientific (1997)
37. Weger, V., Khathuria, K., Horlemann, A.L., Battaglioni, M., Santini, P., Persichetti, E.: On the hardness of the Lee syndrome decoding problem. Advances in Mathematics of Communications **18**(1), 233–266 (2024)

A Weight Distribution of the Coset of a Code

We start with a binary code. Consider a $(n,k)_2$ code $\mathcal{C}$. Then, it is possible to describe the weight distribution (resp. the weight enumerator) of a coset $\mathcal{C}+\mathbf{t}$, where $\mathbf{t}\in\mathbb{F}_2^n$. In particular, we generalize [10, Lemma 1], originally presented for BCH codes, to any binary linear code $\mathcal{C}$.

Consider the weight enumerator of the coset $\mathcal{C}+\mathbf{t}$, i.e.,

$$W_{\mathcal{C}+\mathbf{t}}(x,y)=\sum_{\boldsymbol{\nu}\in\mathcal{C}+\mathbf{t}} x^{n-w_{\mathrm{H}}(\boldsymbol{\nu})}y^{w_{\mathrm{H}}(\boldsymbol{\nu})}=\sum_{j=0}^{n}A_j x^{n-j}y^j$$

where $A_j(\mathcal{C}+\mathbf{t})=\#\{\boldsymbol{\nu}\in\mathcal{C}+\mathbf{t}\colon w_{\mathrm{H}}(\boldsymbol{\nu})=j\}$, with $0\le j\le n$, is the number of codewords in $\mathcal{C}+\mathbf{t}$ that have Hamming weight w_{H}. Also, $\mathcal{C}^{\perp}=\{\mathbf{y}\in\mathbb{F}_2^n : \langle\mathbf{x},\mathbf{y}\rangle\equiv 0 \bmod 2,\ \forall\mathbf{x}\in\mathcal{C}\}$ refers to the *dual code* of $\mathcal{C}$.

Lemma 2. *Consider a $(n,k)_2$ code $\mathcal{C}$, $\mathbf{t}\in\mathbb{F}_2^n$ and set $\mathcal{C}_{\mathbf{t}}=\mathcal{C}\cup(\mathcal{C}+\mathbf{t})$. Then*

$$W_{\mathcal{C}+\mathbf{t}}(x,y)=\frac{1}{2^{n-k}}\left(2W_{\mathcal{C}_{\mathbf{t}}^{\perp}}(x+y,x-y)-W_{\mathcal{C}^{\perp}}(x+y,x-y)\right).$$

Proof. If $\mathbf{t}\in\mathcal{C}$, then $W_{\mathcal{C}+\mathbf{t}}(x,y)=W_{\mathcal{C}}(x,y)=\frac{1}{2^{n-k}}W_{\mathcal{C}^{\perp}}(x+y,x-y)$, which coincides with the MacWilliams identity [23, Th. 1, p. 127].

Otherwise, if $\mathbf{t}\notin\mathcal{C}$, $\mathcal{C}_{\mathbf{t}}$ is a binary linear code with 2^{k+1} codewords. Then, applying the MacWilliams identity to $\mathcal{C}_{\mathbf{t}}$, we get

$$W_{\mathcal{C}_{\mathbf{t}}}(x,y)=\frac{1}{2^{n-(k+1)}}W_{\mathcal{C}_{\mathbf{t}}^{\perp}}(x+y,x-y),$$

but $W_{\mathcal{C}_{\mathbf{t}}}(x,y)=W_{\mathcal{C}}(x,y)+W_{\mathcal{C}+\mathbf{t}}(x,y)$. Thus,

$$\begin{aligned}W_{\mathcal{C}+\mathbf{t}}(x,y)&=\frac{1}{2^{n-(k+1)}}W_{\mathcal{C}_{\mathbf{t}}^{\perp}}(x+y,x-y)-\frac{1}{2^{n-k}}W_{\mathcal{C}^{\perp}}(x+y,x-y)\\&=\frac{1}{2^{n-k}}\left(2W_{\mathcal{C}_{\mathbf{t}}^{\perp}}(x+y,x-y)-W_{\mathcal{C}^{\perp}}(x+y,x-y)\right).\end{aligned}$$

□

Therefore, the relationship between the theta function of a binary Construction A lattice and the weight enumerator of the code (see Theorem 3) together with Lemma 2 allows us to calculate $\sum_{\boldsymbol{\nu}\in\mathcal{C}+\mathbf{t}} \Theta_{2\mathbb{Z}^n+\boldsymbol{\nu}}(z)$ and proceed with our sampling technique.

For larger alphabets, $q > 2$, although a similar identity to the one presented in Lemma 2 holds for the Hamming weight enumerator of a code over $\mathbb{Z}_q$, provided that also the MacWilliams identity is satisfied [23, Ch. 6, §6], the symmetrized weight enumerator requires more efforts. Moreover, it is known that for $q \geq 5$, it does not exist a version of the MacWilliams identity for the Lee weight enumerator [1]. It means that expressing the symmetrized weight enumerator of a coset $\mathcal{C} + \mathbf{t}$ of a q-ary code $\mathcal{C}$ is more challenging and might not have a formulation in terms of its dual.

B Dual of a Construction A Lattice

The generator matrix of the scaled q-ary Construction A lattice is

$$\mathsf{B} = \frac{1}{\sqrt{q}} \begin{pmatrix} \mathsf{I} & \mathsf{A} \\ 0 & q\mathsf{I} \end{pmatrix},$$

where $\mathsf{G} = [\mathsf{I}\ \mathsf{A}]$ is a generator matrix of the q-ary linear code $\mathcal{C}$. This is a square matrix, so B^{-T} is the basis of the dual lattice. We will now demonstrate that it generates $q\mathbb{Z}^n + \mathcal{C}^\perp$. Indeed,

$$\mathsf{B}^{-T} = \frac{1}{\sqrt{q}} \begin{pmatrix} q\mathsf{I} & 0 \\ -\mathsf{A}^T & \mathsf{I} \end{pmatrix}.$$

Permuting rows does not change the lattice generated by this basis, hence

$$\mathsf{B}^* = \frac{1}{\sqrt{q}} \begin{pmatrix} -\mathsf{A}^T & \mathsf{I} \\ 0 & q\mathsf{I}, \end{pmatrix}$$

which is the same format as B. We have that $\mathsf{H} = [-\mathsf{A}^T\ \mathsf{I}]$ is the parity check matrix of G, so it is a generator for $\mathcal{C}^\perp$. It follows that B^* is a basis for the q-ary lattice $q\mathbb{Z}^n + \mathcal{C}^\perp$ as desired.

C Improving the Complexity via Schur Products

C.1 Calculating Lee Weight Profiles

We show that the Schur product allows us to obtain a solvable system of equations that gives the exact Lee weight profile. Furthermore, we show that the positions of the nonzero elements in a codeword completely determine the frequency of each Lee weight profile. For instance, we consider a 3-level construction with code formula $8\mathbb{Z}^n + 4\mathcal{C}_3 + 2\mathcal{C}_2 + \mathcal{C}_1$.

The second stage of the sampling process involves calculating the Lee weight profile $[\omega_{0,0}(\mathbf{c}_2, \mathbf{c}_3), w_{\mathrm{H}}(\mathbf{c}_2), \omega_{0,1}(\mathbf{c}_2, \mathbf{c}_3)]$ for each codeword $\mathbf{c}_2 \in \mathcal{C}_2$, once $\mathbf{c}_3 \in \mathcal{C}_3$ has been fixed. We proceed in the following way:

1. Compute the component-wise or Schur product of $\mathbf{c}_2 \star \mathbf{c}_3$ for all $\mathbf{c}_2 \in \mathcal{C}_2$. As we perform each iteration for all $\mathbf{c}_2$, take note of $w_{\mathrm{H}}(\mathbf{c}_2) = \omega_{1,0}(\mathbf{c}_2, \mathbf{c}_3) + \omega_{1,1}(\mathbf{c}_2, \mathbf{c}_3)$.
2. Compute the binary Hamming weight of each product $\mathbf{c}_2 \star \mathbf{c}_3$, which corresponds to $\omega_{1,1}(\mathbf{c}_2, \mathbf{c}_3)$.
3. Since we $\mathbf{c}_3$ is fixed, we know its binary Hamming weight $w_{\mathrm{H}}(\mathbf{c}_3)$. It follows that $w_{\mathrm{H}}(\mathbf{c}_3) = \omega_{0,1}(\mathbf{c}_2, \mathbf{c}_3) + \omega_{1,1}(\mathbf{c}_2, \mathbf{c}_3)$. We solve for $\omega_{0,1}$.
4. From $w_{\mathrm{H}}(\mathbf{c}_2)$, we can solve for $\omega_{1,0}$. It remains to compute $\omega_{0,0} = n - \omega_{1,0} - \omega_{1,1} - \omega_{0,1}$. Record the Lee weight profile $t \triangleq [\omega_{0,0}, w_{\mathrm{H}}(\mathbf{c}_2), \omega_{0,1}]$.
5. Count the number of recurring tuples. For a given tuple $\mathbf{t}$, denote the number of codewords associated with it by $A_{\mathbf{t}}$.

Performing the Schur product for all $\mathbf{c}_2 \in \mathcal{C}_2$ takes $O(n|\mathcal{C}_2|)$ operations, which is intractable for codes with high dimension. Algorithm 4 samples $\mathbf{c}_1$ with respect to $8\mathbb{Z}^n + 4\mathbf{c}_3 + 2\mathbf{c}_2 + \mathcal{C}_1$. The probability of a coset of this lattice depends on the Lee weight tuple of a codeword defined by $4\mathbf{c}_3 + 2\mathbf{c}_2 + \mathbf{c}_1$ over $\mathbb{Z}_8$. We similarly calculate such a tuple as before. In steps 1 and 2, we fix $\mathbf{c}_3$ and $\mathbf{c}_4$. Therefore, we know the binary Hamming weights $w_{\mathrm{H}}(\mathbf{c}_3), w_{\mathrm{H}}(\mathbf{c}_2)$ and $d_{1,1}(\mathbf{c}_2, \mathbf{c}_3)$. We compute $\mathbf{c}_1 \star \mathbf{c}_2 \star \mathbf{c}_3$ for all $\mathbf{c}_1 \in \mathcal{C}_1$. The Hamming weight of this product corresponds to $\omega_{1,1,1}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$. We have $\omega_{1,1}(\mathbf{c}_2, \mathbf{c}_3) = \omega_{0,1,1}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3) + \omega_{1,1,1}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$ and solve for $\omega_{0,1,1}$. We can also compute $\mathbf{c}_1 \star \mathbf{c}_2$ and $\mathbf{c}_1 \star \mathbf{c}_3$ for all $\mathbf{c}_1 \in \mathcal{C}_1$ to obtain $\omega_{1,1}(\mathbf{c}_1, \mathbf{c}_2) = \omega_{1,1,0}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3) + \omega_{1,1,1}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$ and $\omega_{1,1}(\mathbf{c}_1, \mathbf{c}_3) = \omega_{1,0,1}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3) + \omega_{1,1,1}(\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)$. We can therefore solve for $\omega_{0,1,1}$ and $\omega_{1,0,1}$. Finally, from

$$
\begin{aligned}
w_{\mathrm{H}}(\mathbf{c}_3) &= \omega_{1,1,1} + \omega_{0,1,1} + \omega_{1,0,1} + \omega_{0,0,1} \\
w_{\mathrm{H}}(\mathbf{c}_2) &= \omega_{1,1,1} + \omega_{0,1,1} + \omega_{1,1,0} + \omega_{0,1,0} \\
w_{\mathrm{H}}(\mathbf{c}_1) &= \omega_{1,1,1} + \omega_{1,0,1} + \omega_{1,1,0} + \omega_{1,0,0}
\end{aligned}
$$

we solve for $\omega_{1,0,0}, \omega_{0,1,0}, \omega_{0,0,1}$ and $\omega_{0,0,0}$. However, it is redundant to perform the Schur product three times. We can perform all of the necessary multiplications by performing the technique twice. This is because the element-wise product $\star$ is commutative and associative [31], so we only need to compute $\mathbf{c}_1 \star \mathbf{c}_2$ and $\mathbf{c}_1 \star \mathbf{c}_3$ for all $\mathbf{c}_1 \in \mathcal{C}_1$. It follows that $\mathbf{c}_1 \star \mathbf{c}_2 \star \mathbf{c}_1 \star \mathbf{c}_3 = \mathbf{c}_1 \star \mathbf{c}_2 \star \mathbf{c}_3$ for all $\mathbf{c}_1 \in \mathcal{C}_1$. Therefore, the complexity of this operation is $2 \cdot O(n|\mathcal{C}_1|) = O(n|\mathcal{C}_1|)$.

Overall, computing the Lee weight profiles for a 3-level construction has complexity $O(n|\mathcal{C}_2|) + O(n|\mathcal{C}_1|) = O(n|\mathcal{C}_2|)$ where the equality comes from the fact that $\mathcal{C}_1 \subseteq \mathcal{C}_2$. Equivalently, we can say the complexity is $O(n2^{k_2})$ where k_2 is the dimension of $\mathcal{C}_2$. It is straightforward to extend this technique for higher-level constructions.

C.2 Schur Product of Reed-Muller Codes

Given that the Reed-Muller codes are well-behaved under the Schur product, we state some properties that may be useful to optimize this operation in future work. First, we recall the definition Schur product of sets and of linear codes.

Definition 15. *[31, Def. 1.2, 1.3] Let $\mathbb{F}$ be a field. If $\mathcal{S}, \mathcal{S}' \subseteq \mathbb{F}^n$ are two subsets, we define*

$$\mathcal{S} \dot{\star} \mathcal{S}' = \{\mathbf{c} \star \mathbf{c}' : (\mathbf{c}, \mathbf{c}') \in \mathcal{S} \times \mathcal{S}'\}$$

where $\star$ is the element-wise product. If $\mathcal{C}, \mathcal{C}' \subseteq \mathbb{F}^n$ are two linear subspaces, i.e. two linear codes of the same length n, we define

$$\mathcal{C} \star \mathcal{C}' = \langle \mathcal{C} \dot{\star} \mathcal{C}' \rangle$$

as the linear span of $\mathcal{C} \dot{\star} \mathcal{C}'$, which we call the Schur product of two codes.

The operation $\star$ is commutative, associative and distributive. The Schur product of Reed-Muller codes is particularly special since $\mathrm{RM}(r,m) \star \mathrm{RM}(r',m) = \mathrm{RM}(r+r',m)$ as long as $r + r' \leq m$ [31]. It follows that $\mathrm{RM}(r,m) = \mathrm{RM}(1,m)^{\star^r}$ where we perform the Schur product r times.

Consider when the sampler of a Barnes-Wall lattice needs to sample a coset representative with respect to a scalar version of the sublattice $4\mathbb{Z}^n + 2\,\mathrm{RM}(r,m) + \mathrm{RM}(r',m)$. Following our procedure, we will have already sampled a codeword $\mathbf{c} \in \mathrm{RM}(r,m)$. The probability of selecting a coset in the sublattice $4\mathbb{Z}^n + 2\mathbf{c} + \mathrm{RM}(r',m)$ depends on frequency tuples of the form $[\omega_{0,0}(\mathbf{c},\mathbf{c}'), w_{\mathrm{H}}(\mathbf{c}'), \omega_{0,1}(\mathbf{c},\mathbf{c}')]$ where $\mathbf{c}'$ is any codeword in $\mathrm{RM}(r',m)$. As shown in the previous section, we can use the element-wise product to compute $\omega_{1,1}(\mathbf{c},\mathbf{c}')$ and with the knowledge of $w_{\mathrm{H}}(\mathbf{c})$ and $w_{\mathrm{H}}(\mathbf{c}')$ we can compute the number of each tuple. To avoid having to perform the Schur product $\mathbf{c} \star \mathbf{c}'$ for each $\mathbf{c}' \in \mathrm{RM}(r',m)$, we use the fact that $\mathrm{RM}(r',m) = \mathrm{RM}(1,m)^{\star^{r'}}$. Since for a single codeword $\mathbf{c} \in \mathcal{C}$, $\{\mathbf{c}\}$ is not linear, we consider it as a singleton set for formality and use the $\dot{\star}$ operation.

$$\begin{aligned} c \dot{\star} \mathrm{RM}(r',m) &= c \dot{\star} \underbrace{\langle \mathrm{RM}(1,m) \dot{\star} \ldots \dot{\star} \mathrm{RM}(1,m) \rangle}_{r' \text{ times}} \\ &= \langle c \dot{\star} \mathrm{RM}(1,m) \dot{\star} \ldots \dot{\star} \mathrm{RM}(1,m) \rangle \\ &= \langle (c \dot{\star} \mathrm{RM}(1,m)) \dot{\star} \ldots \dot{\star} \mathrm{RM}(1,m) \rangle, \end{aligned}$$

where we have considered the linearity and associativity of the element-wise product.

If we get lucky, we may recover a large portion of the code using the Schur product, so we may be able to minimize the number of linear combinations of codewords that we need to take to recover the entire code. Unfortunately, it is not clear when that is the case. In general, taking the linear span increases the complexity significantly.

Example 3. To demonstrate the method for a small example, consider $\mathbf{c} \dot{\star} \mathrm{RM}(2,2)$. For a general codeword $\mathbf{c} = (c_1, c_2, c_3, c_4)$, we compute

$$\begin{aligned} (c_1,c_2,c_3,c_4) \dot{\star} \mathrm{RM}(1,2) = \{&(0,0,0,0), (c_1,c_2,c_3,c_4), (c_1,c_2,0,0), (0,0,c_3,c_4), \\ &(0,c_2,c_3,0), (c_1,0,0,c_4), (c_1,0,c_3,0), (0,c_2,0,c_4)\}. \end{aligned}$$

For each element $\mathbf{x}$ in this set, we compute $x \dot{\star} \mathrm{RM}(1,2)$. We obtain the set

$$\begin{aligned}\{&(0,0,0,0),(c_1,c_2,c_3,c_4),(c_1,c_2,0,0),(0,0,c_3,c_4),\\ &(0,c_2,c_3,0),(c_1,0,0,c_4),(c_1,0,c_3,0),(0,c_2,0,c_4),\\ &(c_1,0,0,0),(0,c_2,0,0),(0,0,c_3,0),(0,0,0,c_4)\}.\end{aligned}$$

Note that this is nearly equal to the Schur product with the entire code given by

$$\begin{aligned}(c_1,c_2,c_3,c_4)\dot{\star}\,\mathrm{RM}(2,2) =\{&(0,0,0,0),(c_1,c_2,c_3,c_4),(c_1,c_2,0,0),(0,0,c_3,c_4),\\ &(0,c_2,c_3,0),(c_1,0,0,c_4),(c_1,0,c_3,0),(0,c_2,0,c_4),\\ &(c_1,0,0,0),(0,c_2,0,0),(0,0,c_3,0),(0,0,0,c_4),\\ &(c_1,c_2,c_3,0),(c_1,c_2,0,c_4),(c_1,0,c_3,c_4),(0,c_2,c_3,c_4)\}.\end{aligned}$$

We only need to do 4 linear combinations to recover the rest of the set. From this example, it is clear that the positions of the nonzero coordinates of $\mathbf{c}$ completely determine the number of $\mathbf{c} \star \mathbf{c}'$ associated with a fixed value of $\omega_{1,1}(\mathbf{c}', \mathbf{c})$ for $\mathbf{c}' \in \mathrm{RM}(2,2)$. In particular, if $\mathbf{c} = (1,0,0,0)$ then the number of codeword products with $\omega_{1,1}(\mathbf{c}', \mathbf{c}) = 1$ is equal to 8, which is the number of codewords in $\mathbf{c} \dot{\star}\, \mathrm{RM}(2,2)$ in which the c_1 position appears. $\Diamond$

It is not clear how we can further optimize the Schur product, particularly for special codes like the Reed-Muller codes, to make the sampling procedure more efficient. It may be of interest that Reed-Muller codes are also Plotkin construction codes which have some nice properties under the Schur product [9]. We leave this as an interesting open question.

D Enumerating Codewords from RM(3, 5)

We provide the details for enumerating codewords of a given Hamming weight w for the Reed-Muller code $\mathrm{RM}(3,5)$ using design theory.

Definition 16 (t-design [23, p. 58]**).** *Let X be a set of v elements. A t-design is a collection of distinct k-subsets (blocks) of X with the property that any t-subset of X is contained in exactly λ blocks. We denote this as a $t-(v,k,\lambda)$ design.*

A t-design is specified by its incidence matrix $\mathsf{A} = (a_{i,j})$ where $a_{i,j} = 1$ if an element p_j is contained in the block B_i and 0 otherwise. When $t = 1$, we call the design a Steiner system. For every block in a Steiner system, we can associate a row of the incidence matrix, which we can interpret as a codeword. This forms a constant weight code or a code that consists of codewords of the same weight w. Consequently, a single block in a Steiner system can be uniquely decoded to a codeword of some weight w. Furthermore, the following result gives conditions for when a linear code produces a t-design for more general t.

Theorem 6 (Assmus-Mattson Theorem). *Let $\mathcal{C}$ be a $(n,k)_2$ linear code with minimum Hamming weight d and $\mathcal{C}^\perp$ be the dual code with minimum weight $d^\perp$. Let $t < d$. If $\mathcal{C}$ has at most $d^\perp - t$ non zero weights less than or equal to $n-t$, then for each weight w with $d \leq w \leq n-t$, then $B = \{\mathrm{supp}(c) : c \in \mathcal{C}, w_{\mathrm{H}}(c) = w\}$ are blocks forming a t-design. That is, the set of codewords of weight w form a t-design. Furthermore, the dual code also forms t-designs.*

The dual of $\mathrm{RM}(r,m)$ is $\mathrm{RM}(m-r-1,m)$. Thus, $\mathrm{RM}(3,5)^\perp = \mathrm{RM}(2,5)$ which has minimum distance 8. Applying Assmus-Mattson, we get that for a weight w such that $4 \leq w \leq 29$, the support design in $\mathrm{RM}(3,5)$ is a 3-design. The codewords of minimum weight form a Steiner system, so every vector of weight 3 in $\mathbb{F}_2^{32}$ uniquely decodes to a codeword of weight 4, so to sample a codeword of minimum weight from $\mathrm{RM}(3,5)$ we can perform a 3-out-of-32 choosing procedure and use state-of-the-art decoding for Reed-Muller codes. Unfortunately, for the 3-designs which are not Steiner systems, we cannot uniquely decode blocks to codewords. However, we can notice that the blocks of the design still correspond to the supports of the codewords of weight $w > 4$ in $\mathrm{RM}(3,5)$, so we can precompute the blocks of each design for each weight $w \leq 16$. To sample a codeword of weight $w \leq 16$, we can uniformly sample from the blocks of the corresponding design. We can sample codewords of weight $w > 16$ by adding all ones vector $\mathbf{1}$ to a codeword of weight $32 - w$.