Neural Lattice Reduction: A Self-Supervised Geometric Deep Learning Approach

Giovanni Luca Marchetti* GLMA@KTH.SE
Gabriele Cesa GCESA@QTI.QUALCOMM.COM
Kumar Pratik KPRATIK@QTI.QUALCOMM.COM
Arash Behboodi BEHBOODI@QTI.QUALCOMM.COM
Qualcomm AI Research, Amsterdam†

Editors: Sophia Sanborn, Christian Shewmake, Simone Azeglio, Nina Miolane

Abstract

Lattice reduction is a combinatorial optimization problem aimed at finding the most orthogonal basis in a given lattice. In this work, we address lattice reduction via deep learning methods. We design a deep neural model outputting factorized unimodular matrices and train it in a self-supervised manner by penalizing non-orthogonal lattice bases. We incorporate the symmetries of lattice reduction into the model by making it invariant and equivariant with respect to appropriate continuous and discrete groups.

Keywords: Lattice Reduction, Geometric Deep Learning, Equivariance, Self-Supervised

1. Introduction

Lattices are discrete geometric objects representing 'high-dimensional grids' that are ubiquitous in mathematics and computer science. In particular, two fundamental computational problems are based on lattices: the shortest and the closest vector problem respectively. These arise in several areas, among which asymmetric post-quantum cryptography (Hoffstein et al., 1998; Regev, 2009) and multi-input multi-output digital signal decoding (Worrall et al., 2022; Hassibi and Vikalo, 2005). Both are known to be computationally hard (Goldreich et al., 1999; Ajtai, 1996) and no efficient algorithms exist to address them exactly. However, these problems are easier to solve on lattices with highly-orthogonal bases (Nguyen, 2009). This has motivated the introduction of *lattice reduction* – another computational problem consisting of finding the most orthogonal basis for a given lattice. While the problem is still NP-hard, it can be solved approximately in polynomial time. The Lenstra–Lenstra–Lovász (LLL) algorithm (Lenstra et al., 1982) is a celebrated algorithm for this purpose, and can be thought as a discrete analogue of the Gram-Schmidt orthogonalization procedure.

In this work, we propose a deep learning method to address lattice reduction. The hope is that data-driven techniques are not only highly-parallelizable and therefore computationally efficient, but might find patterns leading to better solutions compared to approximate classical algorithms. Motivated by this, we develop a neural network model that outputs a base-change unimodular matrix given a lattice basis as input. The training objective is self-supervised, in the sense that the model minimizes a measure of orthogonality for the

* Work done during the internship in Qualcomm AI research.
† Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

bases found. From the perspective of model design, we propose an architecture incorporating the symmetries of lattice reduction into the neural network. Specifically, we design a deep model which is simultaneously invariant and equivariant to the orthogonal group and to a finite subgroup of the unimodular group, respectively.

2. Background

An n-dimensional *lattice* is a discrete subgroup of $\mathbb{R}^n$ of maximal rank. If $\Lambda \subseteq \mathbb{R}^n$ is a lattice, then there is an isomorphism of groups $\Lambda \simeq \mathbb{Z}^n$. Such an isomorphism determines a *basis* of the lattice i.e., a set of linearly independent vectors $b_1, \cdots, b_n \in \mathbb{R}^n$ such that $\Lambda = \mathbb{Z}b_1 \oplus \cdots \oplus \mathbb{Z}b_n$. A basis is succinctly described by an invertible matrix $B \in \mathrm{GL}_n(\mathbb{R})$ whose columns are the basis vectors. Given a lattice Λ, any two bases $B, B' \in \mathrm{GL}_n(\mathbb{R})$ are related via (right) multiplication by an integral invertible matrix i.e., $B' = BQ$ for some $Q \in \mathrm{GL}_n(\mathbb{Z}) = \{Q \in \mathbb{Z}^{n\times n} \mid \det(Q) = \pm 1\}$. Matrices belonging to $\mathrm{GL}_n(\mathbb{Z})$ are deemed *unimodular*.

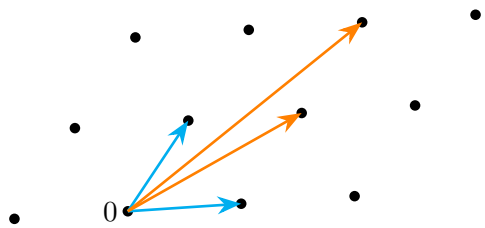


Figure 1: Two bases of a two-dimensional lattice.

In order to solve computational problems over lattices, it is convenient that the basis vectors are as orthogonal as possible. This follows from the Minkowski's theorem for lattices – see Nguyen (2009) for a detailed discussion. The amount by which a basis is not orthogonal is measured by the *orthogonality defect*:

$$\delta(B) = \frac{\prod_i \|b_i\|}{|\det(B)|} \tag{1}$$

Indeed, $\delta(B) \geq 1$ by Hadamard's inequality and $\delta(B) = 1$ iff the b_i's are mutually orthogonal. Given a basis $B \in \mathrm{GL}_n(\mathbb{R})$, *lattice reduction* is a computational problem consisting of finding $Q \in \mathrm{GL}_n(\mathbb{Z})$ minimizing $\delta(BQ)$. However, lattice reduction is NP-hard and, therefore, unfeasible to solve directly. The celebrated *Lenstra–Lenstra–Lovász (LLL) algorithm* (Lenstra et al., 1982) is an approximate algorithm for lattice reduction that finds a basis B' with orthogonality defect bounded by $\delta(B') \leq 2^{n(n-1)/4}$. The algorithm runs in $\mathcal{O}(n^6 \log^3 \beta)$ time (without fast arithmetics), where $\beta = \max_i \|b_i\|$, and therefore is polynomial in complexity. We review the LLL algorithm in the Appendix (Sec. A).

3. Method

Our goal is deploying (geometric) deep learning in order to approximate lattice reduction. To this end, we design a model of the form $\varphi : \mathrm{GL}_n(\mathbb{R}) \to \mathrm{GL}_n(\mathbb{Z})$, where the input

represents a basis B of a lattice, while the output represents the base-change unimodular matrix Q. The objective of the model on a datapoint B is minimizing the (logarithmic) orthogonality defect of the reduced basis $B' = B\varphi(B)$ i.e., the loss is

$$\mathcal{L}(B, \theta) = \log \delta(B') = \sum_i \log \|b'_i\| - \log |\det(B')|, \tag{2}$$

where θ represents the parameters of φ. Note that $\det(B') = \det(BQ) = \det(B)$ is constant and, therefore, is not optimized. In the following, we discuss two challenges related to to the design of the model: outputting unimodular matrices and incorporating the symmetries of lattice reduction.

3.1. Unimodular Outputs

A first challenge is designing φ in order to output unimodular matrices. To this end, we design φ to output a matrix with 1 on the diagonal and 0 everywhere except for one column and one row, i.e.:

$$\begin{pmatrix} 1 & 0 & \cdots & & m_{1,j} & \cdots & 0 \\ 0 & 1 & & & & & \\ \vdots & & \ddots & & \vdots & & \vdots \\ m_{i,1} & & \cdots & 1 \cdots & m_{i,j} & \cdots & m_{i,n} \\ \vdots & & \ddots & & \vdots & & \vdots \\ 0 & & & \cdots & m_{n,j} & \cdots & 1 \end{pmatrix} \tag{3}$$

We refer to the matrix above as an *extended Gauss move*. The model is applied recursively k times to obtain $Q = T_1 \cdots T_k$, where $T_i = \varphi(BT_1 \cdots T_{i-1})$. The motivation is the following fact on $\mathrm{SL}_n(\mathbb{Z}) = \{Q \in \mathbb{Z}^{n\times n} \mid \det(Q) = 1\} \subseteq \mathrm{GL}_n(\mathbb{Z})$ – see the Appendix (Sec. B) for a proof.

Proposition 3.1:
For $n \geq 3$, every matrix in $\mathrm{SL}_n(\mathbb{Z})$ is a product of at most $4n + 51$ extended Gauss moves.

Note that the distinction between $\mathrm{SL}_n(\mathbb{Z})$ and $\mathrm{GL}_n(\mathbb{Z})$ is irrelevant for the purpose of lattice reduction since the orthogonality defect is invariant to multiplying a basis vector by -1. We remark that the above bound is not strict, especially for the additive constant, but ideally the number of steps k should be chosen in the order of $\mathcal{O}(n)$.

In order to produce an extended Gauss move, φ first outputs a matrix $M \in \mathbb{R}^{n\times n}$. The absolute values of the $n(n-1)$ entries of M away from the diagonal are normalized to a probability distribution over indices (i, j) of M with $i \neq j$. An index (i, j) is then sampled via the Gumbel-Softmax trick (Jang et al., 2016), which is necessary to differentiate through sampling. The corresponding entry $m_{i,j}$ of M is used to build an extended Gauss move with non-trivial i-th row and j-th column. Lastly, the latter are discretized in order to lie in $\mathbb{Z}$. To this end, we adopt the procedure of *stochastic rounding* (Gupta et al., 2015; Louizos et al., 2018), consisting of rounding $m_{i,j}$ to one of the two closest integers by sampling (via the Gumbel-Softmax trick) from a Bernoulli distribution with probability equal to (one minus) the rounding error.

3.2. Invariance and Equivariance

Another challenge is designing the model in order to preserve the symmetries of lattice reduction. To this end, note that if $B' = BQ$ is a basis minimizing the orthogonality defect (Equation 1) for some $B \in \mathrm{GL}_n(\mathbb{Z})$, then the following symmetries hold:

- If $\widetilde{B} = BH$ for some $H \in \mathrm{GL}_n(\mathbb{Z})$, then $B' = \widetilde{B}\widetilde{Q}$ minimizes the orthogonality defect for $\widetilde{B}$, where $\widetilde{Q} = H^{-1}Q$.
- If $\widetilde{B} = UB$ for some $U \in \mathrm{O}_n(\mathbb{R})$, then $\widetilde{B}Q$ minimizes the orthogonality defect for $\widetilde{B}$.

Intuitively, the first property above reflects 'internal' symmetries due to base change within the lattice, while the second one reflects 'external' symmetries due to rigid transformations of the lattice in Euclidean space. Based on the properties above, we aim to design a model φ satisfying:

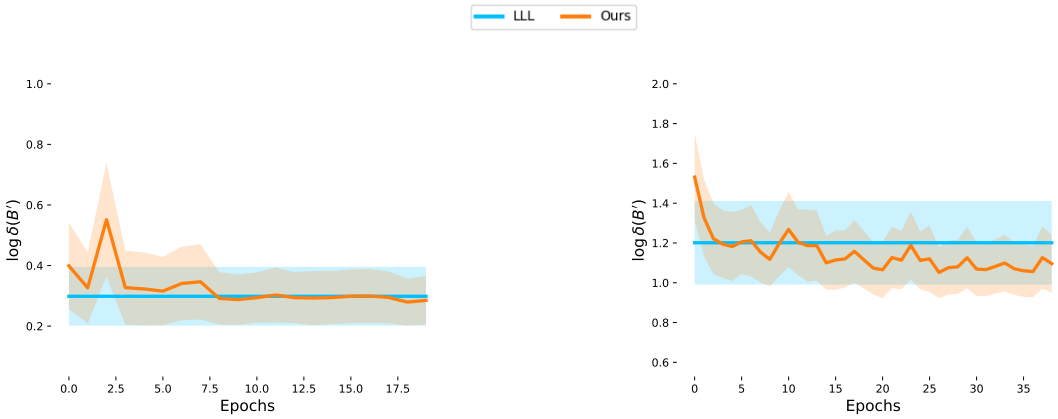
- *Right unimodular equivariance*: $\varphi(BH) = H^{-1}\varphi(B)$ for $H \in \mathrm{GL}_n(\mathbb{Z})$.
- *Left orthogonal invariance*: $\varphi(UB) = \varphi(B)$ for $U \in \mathrm{O}_n(\mathbb{R})$.

In order to address left orthogonal invariance, we simply input the *Gram matrix* $G = B^T B$ to the model. In what follows, we will abuse the notation for φ and write both $\varphi(B)$ and $\varphi(G)$, depending on the input considered. In order to address right unimodular equivariance, note first that G transforms as $BH \mapsto H^T GH$. Therefore, the model needs to satisfy $\varphi(H^T GH) = H^{-1}\varphi(G)$ for $H \in \mathrm{GL}_n(\mathbb{Z})$. However, the latter equivariance property turns our to be too challenging to be achieved. Not only it involves different transformations between input and output, but the unimodular group $\mathrm{GL}_n(\mathbb{Z})$ is algebraically subtle and understood only partially. Due to this, we focus on a relevant finite subgroup of $\mathrm{GL}_n(\mathbb{Z})$: the *hyperoctahedral group* H_n. The latter consists of the $2^n n!$ signed permutation matrices and can be thought as the group of isometries of both the hypercube and the cross-polytope. Considering the hyperoctahedral subgroup simplifies the challenge of designing an equivariant φ for a number of reasons. First, $\mathrm{H}_n \subseteq \mathrm{O}_n(\mathbb{R})$ and therefore $H^{-1} = H^T$ for $H \in \mathrm{H}_n$. Moreover, the orthogonality defect is invariant to the hyperoctahedral group i.e., $\delta(BH) = \delta(B)$. Therefore, right unimodular equivariance reduces equivalently to:

$$\varphi(H^T GH) = H^T \varphi(G) H. \tag{4}$$

The latter not only is a natural tensorial transformation law, but is already satisfied (in expectation) by the (stochastic) procedure described in Sec. 3.1 to obtain extended Gauss moves. Moreover, it is compatible with the recursion implemented to obtain a product of multiple moves.

In order to design a model that is equivariant as above, we implement ideas from Jo et al. (2021) and deploy a message-passing Graph Neural Network (GNN) on a graph whose nodes are pairs of indices (i, j) and that has edge between (i, j) and (i', j') if $i = i'$ or $j = j'$ or $i = j'$ or $j = i'$. This leads to a neural architecture that is equivariant for permutation matrices H. In order to extend equivariance to diagonal H with ± 1 on the diagonal, we adapt the methods from Lim et al. (2023) and deploy activation functions that are even or odd appropriately.



(a) $n = 4$ (b) $n = 8$

Figure 2: Performance of our model and the LLL algorithm in $n \in \{4, 8\}$ dimensions.

4. Experiments

We implement our model and train it on a randomly-generated dataset. To this end, we procedurally generate 1000 random matrices at each epoch by sampling entries independently from a uniform distribution in $[0, 1]$ and applying the matrix exponential in order to obtain an invertible matrix $B \in \mathrm{GL}_n(\mathbb{R})$ representing a basis of a lattice. The test set contains 4000 random lattices from the same distribution and is fixed. We set the number of Gauss moves produced by the model equal to the dimension of the lattice i.e., $k = n$, and compute the loss in Eq. 2 at each step.

Fig. 2 reports the mean and standard deviation over test matrices B for our model and the LLL algorithm. The plots display results as training progresses in dimensions $n = 4, 8$. As can be seen, our model not only performs similarly to LLL at convergence, but even outperforms it, especially in dimension $n = 8$. In order to highlight the performance difference, in the Appendix (Sec. C) we include an additional analysis where the two methods are evaluated on a portion of data where they perform respectively the worst.

5. Conclusions and Future Work

In this work, we have addressed lattice reduction via deep learning methods. We have designed a model to output unimodular matrices and to respect the symmetries of lattice reduction. As a limitation, our model is not equivariant (on the right) to the whole unimodular group (but only to the hyperoctahedral subgroup) and therefore does not leverage upon the complete spectrum of symmetries of the lattice reduction problem. The challenge of designing a model equivariant to $\mathrm{GL}_n(\mathbb{Z})$ remains open, and represents an interesting line for future investigation.

Acknowledgements

We wish to thank Daniel Worral, Roberto Bondesan and Markus Peschl for their help and the insightful discussions.

References

Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108, 1996.

David Carter and Gordon Keller. Bounded elementary generation of sln (o). *American Journal of Mathematics*, pages 673–687, 1983.

Oded Goldreich, Daniele Micciancio, Shmuel Safra, and J-P Seifert. Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. *Information Processing Letters*, 71(2):55–61, 1999.

Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. Deep learning with limited numerical precision. In *International conference on machine learning*, pages 1737–1746. PMLR, 2015.

Babak Hassibi and Haris Vikalo. On the sphere-decoding algorithm i. expected complexity. *IEEE transactions on signal processing*, 53(8):2806–2818, 2005.

Jeffrey Hoffstein, Jill Pipher, and Joseph H Silverman. Ntru: A ring-based public key cryptosystem. In *International algorithmic number theory symposium*, pages 267–288. Springer, 1998.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Jaehyeong Jo, Jinheon Baek, Seul Lee, Dongki Kim, Minki Kang, and Sung Ju Hwang. Edge representation learning with hypergraphs. *Advances in Neural Information Processing Systems*, 34:7534–7546, 2021.

Arjen K Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische annalen*, 261:515–534, 1982.

Derek Lim, Joshua Robinson, Stefanie Jegelka, Yaron Lipman, and Haggai Maron. Expressive sign equivariant networks for spectral geometric learning. In *ICLR 2023 Workshop on Physics for Machine Learning*, 2023.

Christos Louizos, Matthias Reisser, Tijmen Blankevoort, Efstratios Gavves, and Max Welling. Relaxed quantization for discretized neural networks. *arXiv preprint arXiv:1810.01875*, 2018.

Phong Q Nguyen. Hermite's constant and lattice algorithms. In *The LLL Algorithm: Survey and Applications*, pages 19–69. Springer, 2009.

Phong Q. Nguyen and Damien Stehlé. An lll algorithm with quadratic complexity. *SIAM J. Comput.*, 39(3):874–903, aug 2009. ISSN 0097-5397. doi: 10.1137/070705702. URL https://doi.org/10.1137/070705702.

Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):1–40, 2009.

Daniel E Worrall, Markus Peschl, Arash Behboodi, and Roberto Bondesan. Learning perturbations for soft-output linear mimo demappers. In *GLOBECOM 2022-2022 IEEE Global Communications Conference*, pages 5213–5218. IEEE, 2022.

Appendix A. The LLL Algorithm

In this section, we describe the Lenstra–Lenstra–Lovász (LLL) algorithm for lattice reduction (Lenstra et al., 1982) in details. Let $B \in \mathrm{GL}_n(\mathbb{R})$ be a basis of a lattice and denote by B^* the orthogonal basis of $\mathbb{R}^n$ obtained via the Gram-Schmidt algorithm applied to B. For each i, j consider moreover

$$\mu_{i,j} = \frac{b_i \cdot b_j^*}{\|b_j^*\|^2}. \tag{5}$$

Definition 1:
A basis $B \in \mathrm{GL}_n(\mathbb{R})$ is *Siegel-reduced* if for all $i > j$:

$$|\mu_{i,j}| \leq \tfrac{1}{2} \qquad\qquad \frac{\|b_i^*\|^2}{\|b_{i-1}^*\|^2} \geq \left(\tfrac{3}{4} - \mu_{i,i-1}^2\right)$$

Size Condition *Lovász Condition*

The following is a bound on the orthogonality defect for Siegel-reduced bases.

Lemma A.1 (Lenstra et al. (1982)):
If B is Siegel-reduced then:

$$\delta(B) \leq 2^{\frac{n(n-1)}{4}}. \tag{6}$$

The LLL algorithm finds a Siegel-reduced basis of an integral lattice $\Lambda \subseteq \mathbb{Z}^n \subseteq \mathbb{R}^n$ starting from an arbitrary basis $B \in \mathrm{GL}_n(\mathbb{R}) \cap \mathbb{Z}^{n\times n}$ (see Algorithm 1). It can be seen that the algorithm runs in $\mathcal{O}(n^6 \log^3 \beta)$ time (without fast arithmetics) (Nguyen and Stehlé, 2009), where $\beta = \max_i \|b_i\|$, and therefore is polynomial in complexity.

Algorithm 1: LLL Algorithm

Input: Basis of an (integral) lattice $B \in \mathrm{GL}_n(\mathbb{R}) \cap \mathbb{Z}^{n\times n}$
Output: Siegel-reduced basis B
$B^* \leftarrow$ Gram-Schmidt(B)
$k \leftarrow 2$
while $k \leq n$ **do**
  for $j = k-1$ **to** 1 **do**
    if $|\mu_{k,j}| > \frac{1}{2}$ **then**
      $b_k \leftarrow b_k - \lceil \mu_{k,j} \rfloor b_j$
      $B^* \leftarrow$ Gram-Schmidt(B)
    end
  end
  if $\frac{\|b_k^*\|^2}{\|b_{k-1}^*\|^2} \geq \left(\frac{3}{4} - \mu_{k,k-1}^2\right)$ **then**
    $k \leftarrow k+1$
  end
  else
    Swap b_k and b_{k-1}
    $k \leftarrow \max\{k-1, 2\}$
    $B^* \leftarrow$ Gram-Schmidt(B)
  end
end

Appendix B. Theoretical Result

We will call *Gauss move* a $Q \in \mathrm{SL}_n(\mathbb{Z})$ with 1 on the diagonal ($Q_{i,i} = 1$ for all i) and at most one non-zero element $Q_{i,j}$ away from the diagonal ($i \neq j$). The following is a deep algebraic property of Gauss moves.

Theorem B.1 (Carter and Keller (1983)):
For $n \geq 3$, $\mathrm{SL}_n(\mathbb{Z})$ is boundedly generated by Gauss moves. Specifically, every matrix in $\mathrm{SL}_n(\mathbb{Z})$ is a product of at most $\frac{1}{2}(3n^2 - n) + 51$ Gauss moves.

Our result is derived from the above one as follows.

Proposition B.2:
For $n \geq 3$, every matrix in $\mathrm{SL}_n(\mathbb{Z})$ is a product of at most $4n + 51$ extended Gauss moves.

Proof Pick $Q \in \mathrm{SL}_n(\mathbb{Z})$. Note first that the non-zero entries of each row and each column of Q are coprime since they can be arranged in an integral linear combination equal to $\det(Q) = 1$. Next, consider the last row of Q, denoted by $u_1 = Q_{n,1}, \cdots, u_n = Q_{n,n}$. We would like to show that the non-zero entries among $u_1, \cdots, u_{n-1}$ can be made coprime by multiplying Q on the right by at most one (non-extended) Gauss move. If $u_n = 0$, this is true already for Q by the above observation. Suppose that $u_n \neq 0$. If there are less than two indices $1 \leq i < n$ such that $u_i \neq 0$, we can replace one vanishing u_i with u_n by multiplying Q on the right by a (non-extended) Gauss move and obtain the desired result. Otherwise, assume without loss of generality that $u_1 \neq 0$ and consider some $t \in \mathbb{Z}$ such that:

- $t \equiv 1$ mod all the primes dividing all the non-zero u_i for $i \in \{1, \cdots, n-1\}$,
- $t \equiv 0$ mod all the primes dividing all the non-zero u_i for $i \in \{2, \cdots, n-1\}$ but not dividing u_1.

Then the non-zero integers among $u_1 + tu_n, u_2 \cdots, u_{n-1}$ are coprime, as desired.

We therefore assume that $u_1, \cdots, u_{n-1}$ are coprime. By the Bézout's identity, there exist integers $a_1, \cdots, a_{n-1}$ such that $\sum_{1 \leq i < n} a_i u_i = 1 - u_n$. By multiplying Q on the right by the extended Gauss move

$$\begin{pmatrix} 1 & 0 & \cdots & a_1 \\ 0 & 1 & & \vdots \\ \vdots & & \ddots & a_{n-1} \\ 0 & \cdots & 0 & 1 \end{pmatrix} \tag{7}$$

we reduce to the case $u_n = 1$. By further multiplying on the right by the extended Gauss move

$$\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & & \vdots \\ \vdots & & \ddots & 0 \\ -u_1 & \cdots & -u_{n-1} & 1 \end{pmatrix} \tag{8}$$

we reduce to the case where the last row is $(0, \cdots, 0, 1)$. Similarly, by multiplying by another move on the left, the last column reduces to $(0, \cdots 0, 1)$, obtaining a matrix of the form:

$$\begin{pmatrix} & & & 0 \\ & A & & \vdots \\ & & & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix} \tag{9}$$

where $A \in \mathrm{SL}_{n-1}(\mathbb{Z})$.

Putting everything together, via induction with $4n-12$ extended Gauss moves we arrive at a matrix of the form $A' \oplus I$, where $A' \in \mathrm{SL}_3(\mathbb{Z})$ and I is the identity matrix of dimension $n-3$. Since by Carter and Keller (1983) any matrix in $\mathrm{SL}_3(\mathbb{Z})$ can be written as a product of 63 (non-extended) Gauss moves, this concludes the proof.

■

Appendix C. Additional Experimental Results

In order to highlight the performance difference between our model and LLL, we design another empirical comparison as follows. We evaluate both the models on a portion $p \in [0, 1]$ of test data on which the two methods perform respectively the worst. This provides empirical insights into the extent by which one method can compensate for the deficiencies of the other. Fig. 3 reports the results in dimensions 4 and 8 for $p = 20\%$. As can be seen, our model outperforms LLL by a large extent when both are evaluated on the matrices on which LLL performs the worst. On the contrary, the performance of the two methods are more comparable when evaluated on the matrices on which our method performs the worst. Note that in this case the performance of LLL is unstable since it is evaluated on a distibution that is changing over time. We conclude that our model can partially amend for the eventual poor performance of LLL, while the opposite is less likely.

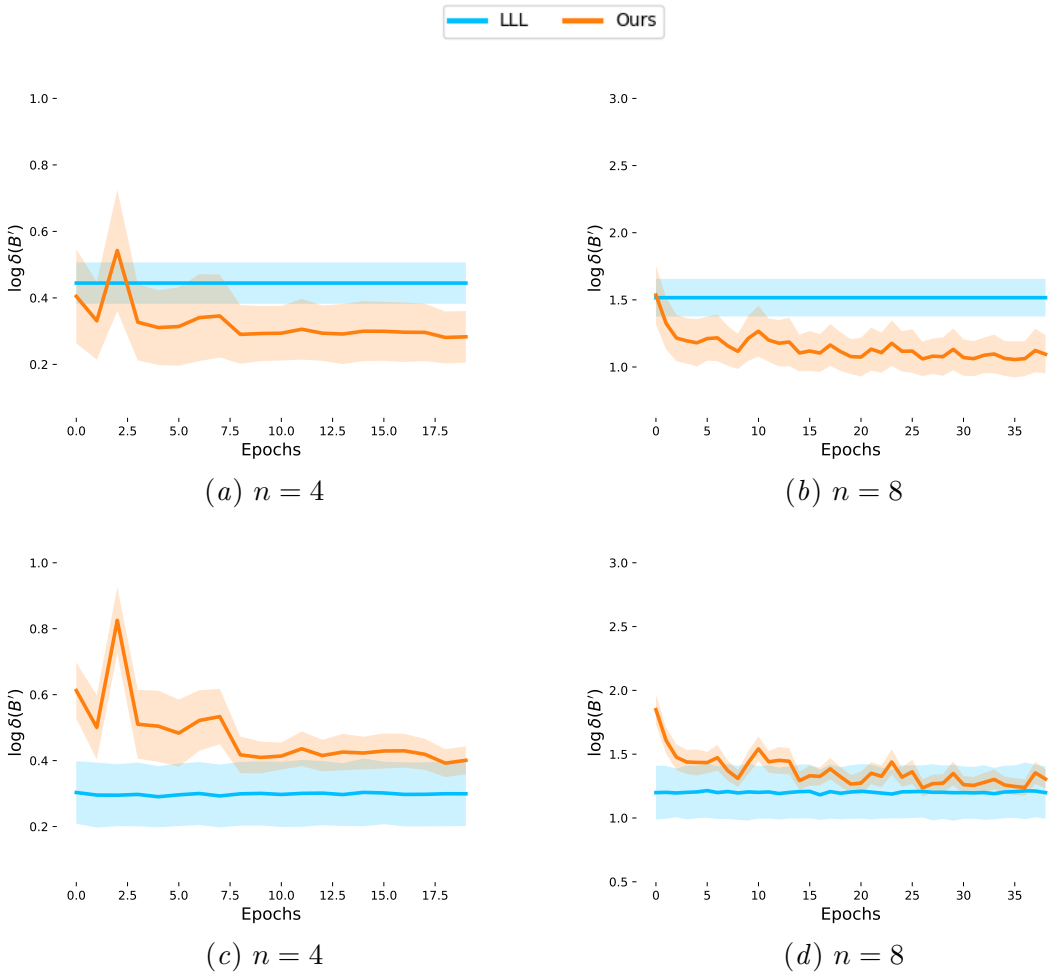


Figure 3: Performance on the 20% of test data on which LLL (**top**) and our model (**bottom**) perform respectively the worst.