Predicting Lattice Reduction on Ideal Lattices (PeRIL)

Bryan Ek* Emily Nystrom Peter Curry Jamie Lyle Scott Batson
Bryan Williams

Abstract
The shortest vector problem (SVP) is at the core of many applications of lattices. Lattice reduction algorithms output a lattice basis containing a short vector and are a common approach to the SVP. The research conducted by the PeRIL team supports an investigation of the performance of lattice reduction algorithms when applied to lattices from the ideal case versus lattices from the general case. The foundational work of Gama and Nguyen provided empirical results for the case of general lattices. Since their 2008 publication, experimental results have been published for various aspects of lattice reduction algorithms, including some results for the case of ideal lattices (only). Although the implications of understanding the impact of lattice class on lattice reduction algorithms have been referenced in the literature, to our knowledge, a systematic, empirical investigation of the impact of the class of lattices (e.g., ideal vs. general), in relation to its impact on lattice reduction algorithm performance, has not been previously published. We address that gap in this paper.

Key Words: ideal, lattice, reduction, root hermite factor, runtime, experimental

1. Background and Definitions

Lattice-based cryptography is widely conjectured to be a candidate for post-quantum cryptography [GN08, PS13, Bat15]. In general, lattices take significant space and time to maintain and manipulate. Using ideal lattices instead of general lattices allows for storage and computation savings [Bat15, PS13]. Many sources have posed the question of whether ideal lattices are as secure as general lattices [PS13, dP12, SB10]. To our knowledge, we are the first paper to provide a comparison of the performance of lattice reduction algorithms when applied to ideal versus general lattices.

Our experiments focus on the performance of lattice reduction algorithms, particularly with respect to the root Hermite factor and the algorithm runtime. Section 1 provides definitions and descriptions pertinent to our later discussion of experimental setup (Section 3) and results (Section 4). Section 2 provides a short summary of results from the literature. Our main results are summarized in our conclusion (Section 5) and supported by the data in Appendix B. The background descriptions in this report are intended to provide a brief description to facilitate a working understanding of the purposes and performance quantifiers for lattice reduction algorithms. A detailed account, which includes extensive references as well as discussion of related algebraic theory, is provided in [Bat15].

*Naval Information Warfare Center Atlantic (NIWC)

1.1 Definitions

Definition 1 (Lattice)**.** A *lattice* is the integer span of independent vectors

$$\mathcal{L} = \left\{ \sum_{1 \le i \le n} \alpha_i \mathbf{b}_i : \alpha_i \in \mathbb{Z} \right\}.$$

The set of vectors $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is a basis. The volume of the fundamental parallelepiped of $\mathcal{L}$ is given by $Vol(\mathcal{L}) = \det(\mathcal{L}) = \sqrt{\det(BB^T)}$ and is invariant of basis choice [SBL09].

If the basis (B) of linearly independent columns is full rank (i.e., $n = d$, making B square and nonsingular; that is, if $\mathcal{L}$ is of full-dimension), then $\det(\mathcal{L}) = |\det(B)|$. In this paper, we represent vectors as columns and only full-rank lattices will be considered. Thus, we use n exclusively as the dimension of a lattice. See Figure 1 for a depiction of a 2D lattice ($n = 2$).

Definition 2 (Ideal Lattice)**.** An ideal in a ring, $I \lhd R$, is a subgroup of R with the additional restriction that $\forall i \in I, r \in R$, we have $ir, ri \in I$. If R has an additive group isomorphic to $\mathbb{Z}^n$, then an ideal, I, can be embedded as a lattice $\mathcal{L} \subseteq \mathbb{Z}^n$ with the mapping

$$(\alpha_0, \ldots, \alpha_{n-1}) \in \mathcal{L} \leftrightarrow \alpha_0 + \cdots + \alpha_{n-1}x^{n-1} \in I,$$

where n is the dimension of the lattice. A lattice is *ideal* if it corresponds to some ideal of a ring.

We consider ideals in cyclotomic quotient rings: $I \lhd \mathbb{Z}[x]/\Phi_m(x)$; m is the *cyclotomic index*. In this scheme, the dimension of the embedded lattice is $n = \phi(m)$ (the Euler-Totient function). Ideal lattices can be constructed from more than solely cyclotomic polynomials or even other quotient rings. Of note, the lattice in Figure 1 corresponds to an ideal in a non-cyclotomic ring. Our restriction is because ideals of cyclotomic quotient rings are known to be full rank. See [Bat15] for more discussion on ideal lattices.

General lattices exhibit a group structure, whereas ideal lattices exhibit a ring structure. To describe a general lattice, a full matrix of n^2 data points is required; to describe an ideal lattice, only the n values of a vector and the $n+1$ values of a divisor polynomial are needed. Computation is also faster in an ideal lattice since manipulating the structure involves polynomial multiplication as opposed to matrix multiplication in a general lattice. The $O(n^2) \to O(n)$ memory and computation savings are the driving force behind using ideal lattices for cryptography [Bat15, PS13]. The driving force behind this paper is to empirically verify that these storage and computation savings do not compromise security.

Definition 3 (Lattice Minima)**.** The i^{th} *lattice minima*, $\lambda_i(\mathcal{L})$, is defined as the radius of the smallest zero-centered ball containing at least i linearly independent lattice vectors. I.e. $\lambda_i(\mathcal{L}) = \min\{r \mid \dim(\text{span}(\mathbf{B}_n(r) \cap \mathcal{L})) \ge i\}$, where $\mathbf{B}_n(r)$ is the n-dimensional hyperball of radius r.

The lattice reduction algorithms analyzed in this paper have the goal of finding the first lattice minima: $\lambda_1(\mathcal{L})$.

The root Hermite factor (RHF) measures the quality of a lattice reduction algorithm. Reduction algorithms are designed with the goal of producing the vector with the shortest possible length given a basis. See Figure 1 for an example of bases with different RHF. The RHF measures the "shortness" in such a way that allows for comparison of the algorithm's success between different lattices.

Definition 4 (Root Hermite Factor)**.** The *root Hermite factor* (RHF) for a given basis B of a lattice $\mathcal{L}$ is defined as

$$\gamma_{\text{Hermite}} = \sqrt[n]{\frac{\|\mathbf{b}_1\|}{[\det(\mathcal{L})]^{1/n}}}, \tag{1}$$

where n denotes the dimension of $\mathcal{L}$, and $\mathbf{b}_1$ denotes the shortest vector of B (see [SBL09], [GN08]).

It is important to note the difference between "Hermite factor" and RHF. The Hermite factor facilitates comparison of lattice reduction performance on lattices (or simply different lattice bases) across the same dimension. Because current algorithms, in particular LLL and BKZ, are only able to produce exponential Hermite factors in polynomial time [FSW14], it is fitting to consider the RHF, which allows for comparisons across different dimensions.

Original basis (red - - -). RHF= 1.899.

$$\begin{pmatrix} 9 & 15 \\ 6 & 11 \end{pmatrix}$$

Reduced basis (blue —). RHF= 1.

$$\begin{pmatrix} 3 & 0 \\ 1 & 3 \end{pmatrix}$$

Both bases span the same set of points in $\mathbb{Z}^2$. In addition, this lattice corresponds to the ideal generated by $(3+x)$ in $\mathbb{Z}[x]/(x^2-x-3)$.

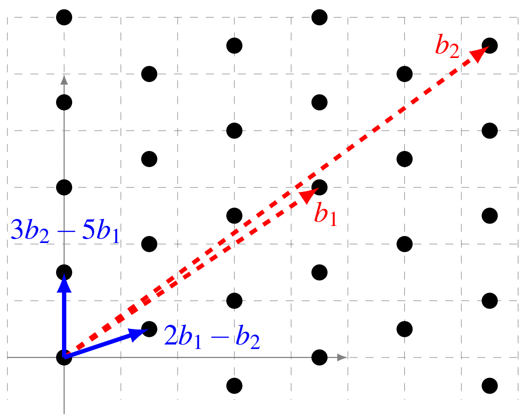

Figure 1: A lattice with two matrix representations, an ideal representation, and a visual representation.

1.2 Lattice Reduction Algorithms

Problem 5 (SVP)**.** Given an arbitrary basis for a lattice $\mathcal{L}$, the shortest vector problem is the task of finding the vector of shortest length. This problem is NP-hard in the random lattice case [Ajt96, Ajt98].
The approximate short vector problem (α-SVP), requires finding a vector $\mathbf{v}$ such that $\|\mathbf{v}\| \leq \alpha \|\lambda_1(\mathcal{L})\|$. There are polynomial time algorithms for solving the approximate case with $\alpha = \left(\sqrt{\frac{4}{3}}\right)^n$ [SB10, GN08, SBL09].

Lattice reduction algorithms are typically concerned with finding a highly orthogonal basis and a short vector. A first estimate of the length of the shortest vector is the Gaussian heuristic: the length of the shortest vector is roughly the radius of an n-dimensional ball of volume $Vol(\mathcal{L})$ [Che16]. This gives us the estimate

$$\frac{\lambda_1(\mathcal{L})}{Vol(\mathcal{L})^{1/n}} \approx \frac{1}{Vol(\mathbf{B}_n(1))^{1/n}} = \frac{\Gamma(1+n/2)^{1/n}}{\sqrt{\pi}} \approx \sqrt{\frac{n}{2\pi e}}.$$

Thus, a reasonable goal to expect is to find an RHF with value $\widehat{g} := \left(\frac{n}{2\pi e}\right)^{1/(2n)}$. For $n \leq 17$, $\widehat{g} < 1$; we have a goal *below* 1 in lower dimension. The heuristic is maximized at $n = 46$: $\widehat{g} = 1.0108$.

Whereas some lattice reduction algorithms perform an exhaustive search and identify an exact solution to the SVP, other algorithms return an approximate solution. In this report, we consider two common approximation lattice reduction algorithms: LLL and BKZ.

1.2.1 LLL

Created in 1982 by Lenstra, Lenstra, and Lovasz [LLL82], the LLL algorithm is the "first polynomial-time basis reduction algorithm" [dP14]. The LLL algorithm produces an LLL-reduced basis (see [LLL82], [FSW14], and [SBL09]).

Definition 6. Let $\delta \in (1/4, 1]$. A basis $B = [\mathbf{b}_1, ..., \mathbf{b}_n]$ of a lattice $\mathcal{L}(B)$ is *δ-LLL reduced* $\Leftrightarrow$

- It is size reduced (i.e., $|\mu_{i,j}| \leq \frac{1}{2}$ for all $1 \leq j < i \leq n$), and
- it satisfies the Lovasz condition (i.e., $\delta \|\mathbf{b}_k^*\|^2 \leq \|\mathbf{b}_{k+1}^*\|^2 + \mu_{k+1,k}\|\mathbf{b}_k^*\|^2$ for $1 \leq k < n$),

where $\mathbf{b}_k^*$ denotes the Gram-Schmidt orthogonalization of vector $\mathbf{b}_k$ and $\mu_{i,j} = \frac{\mathbf{b}_i^T \mathbf{b}_j^*}{\|\mathbf{b}_j^*\|}$.

$\delta = 1$ corresponds to solving the exact SVP. We use the standard of $\delta = 0.99$ for an accurate comparison to others' work [GN08, SB10, PSZ13, FSW14]. For a detailed description of the algorithm, see [LLL82, PSZ13, SE94].

LLL provably achieves [GN08, Ngu17]

$$\|\mathbf{b}_1\| \leq \left(\frac{4}{3}\right)^{(n-1)/4} Vol(\mathcal{L})^{1/n}, \text{ where } n \text{ denotes dimension.} \tag{2}$$

This is an approximation factor of $\lessapprox (4/3)^{(n-1)/2} \approx 1.155^n$ and RHF $\lessapprox \sqrt[n]{(4/3)^{(n-1)/4}} \approx 1.075$. As seen in Table 1, LLL typically does much better than this upper bound. However, "the approx factor $(4/3+\varepsilon)^{(n-1)/4}$ [based on Eq. 2] is tight ... only for worst-case bases of certain lattices" ([Ngu17] slide 43). Schneider et al. [SBL09] produced a probabilistic analysis that gave a good estimate (see Table 1) of what to expect in the random case.

1.2.2 Blockwise Korkine-Zolotarev (BKZ)

(Schnorr and Euchner, 1994; [SE94]): The BKZ algorithm is a blockwise generalization of LLL. "Each of the local blocks defines a lattice. Each iteration looks at one block and ensures that the first vector of the block is the shortest vector inside the lattice spanned by this block" (p.8, [Ham13]). For a blocksize of 2, BKZ is equivalent to LLL. For a blocksize of n, BKZ is equivalent to HKZ (see below). For a detailed description of the algorithm, see [SE94, CN11].

The best known bound for the shortest basis vector in a BKZ-β reduced basis is given by

$$\|\mathbf{b}_1\| \leq \sqrt{\gamma_\beta}^{(n-1)/(\beta-1)} \cdot \lambda_1(L), \tag{3}$$

where γ_β is the Hermite constant in dimension β [SB10, SBL09, GN08, HPS11, Ngu17].[1] From this, and bounds on γ_β, an RHF bound (for $\beta = 20$) of 1.1461 is obtained, which is worse than the already applicable LLL bound. However, the algorithm performs much better empirically than theoretically. The probabilistic analysis of Schneider et al. [SBL09] provides a good estimate (see Table 1) of what to expect in the random case of BKZ-20 reduction.

1.2.3 Additional Variants

We chose to focus on LLL and BKZ as they are the most common reduction algorithms in use. Other lattice reduction methods are listed below.

1. Hermite-Korkine-Zolotarev reduction (HKZ) [Ham13]: Essentially BKZ with blocksize equal to dimension.
2. Floating Point LLL (fplll) [SE94]: There is an heuristic version (FP) referenced in [PSZ13] as well as a standard version (L2).
3. Adaptive Precision Floating Point LLL (ap-fplll): (Plantard, Susilo, and Zhang; 2013; [PSZ13])
4. Pot-LLL [FSW14]: potential LLL. Adds a "potential energy" function that the algorithm attempts to minimize.
5. Deep LLL [FSW14]: "expected to improve the Hermite factor and the approximation factor of LLL, but no provable upper bound is known (except essentially that of LLL)" ([GN08] p.5).
6. BKZ 2.0: (developed by Chen & Nguyen; see [dP14] for summary or [CN11] for more detail)

[1] [SB10, SBL09] do not include the $\sqrt{}$ while [GN08, HPS11, Ngu17] do. [HPS11] includes a proof.

7. BKZ Simulation Algorithm: (developed by Chen & Nguyen; see [Ham13] for summary or [CN11] for more detail). Among other things, this algorithm can be used to simulate the runtime of the BKZ algorithm.

2. Experimental Results in the Literature

2.1 RHF

Table 1 lists experimental results for RHF with LLL or BKZ from the literature. Some of the included sources list results for additional algorithms other than those considered in this paper (LLL/BKZ). See additional variants in Section 1.2.

	RHF Source				
Algorithm	[SB10] p.6	[GN08] p.8	[FSW14] p.10	[SBL09] p.8/11	[PS13] p.12
LLL	1.0162*	1.0219	1.0212	1.0193	1.022†
BKZ-5	1.0156*	–	1.0164	–	–
BKZ-10	1.0146*	–	1.0145	–	1.014†
BKZ-20	1.0126	1.0128	–	1.0157	1.013†
BKZ-28	1.0111	1.0109	–	–	–
BKZ-30	1.0107*	–	–	–	1.011†

Table 1: *extrapolated results based on Eq. (4). †rough values pulled from Figure 4 in [PS13] p.12. All algorithms were run with $\delta = 0.99$, as is standard. [FSW14] includes confidence intervals in their graph of RHF, which appear to have widths < 0.0001. [SBL09] are probabilistic results based on experimentation. [SB10, GN08, FSW14, SBL09] contain data for general lattices and [PS13] contains data for ideal lattices.

A Hermite factor of 1.01^n in high (around 500) lattice dimension is within reach (as of 2008), but "a Hermite factor of 1.005^n in dimension 500 looks totally out of reach" ([GN08] p.9). Gama and Nguyen based their estimates on using BKZ-25. A sharp increase in runtime occurs at this blocksize; see discussion in Section 2.2 or [GN08] p.16.

Results have been published for considering the performance of lattice reduction algorithms for general lattices (e.g., [GN08, SB10, SBL09, FSW14]) as well as for ideal lattices [PS13]. The work of the PeRIL project team allows for a comparison of the performances of lattice reduction algorithms using ideal lattices versus using general lattices.

Based on their experimental results, [SB10] (p.246) reported the following relationship between the predicted RHF ($\widehat{RHF}$)[2] and BKZ blocksize (Eq. (4)):

$$\widehat{RHF} = 1.01655 - 0.000196185 \text{ Blocksize}. \quad (4)$$

Unfortunately, no detail about the methods used by [SB10] to derive this estimated equation (Eq. (4)) accompanied the paper. We were able to reproduce *almost* the same result in Eq. (27).

[2]The definition of Hermite Factor given in [SB10] (p.245) appears to have a typo. We assume that (correcting for typos) [SB10]'s definition agrees with their definition of Hermite Factor given in [SBL09] (p.3).

2.2 Runtime

Exact algorithms for solving SVP utilize polynomial space and super exponential time or exponential space and exponential time [Ngu11] (with respect to dimension). The approximation algorithms that we are testing run in polynomial space. For $\delta < 1$, LLL is proven to run in polynomial time [Ngu11, SB10, PSZ13], whereas the best theoretical upper bound for BKZ is $(n\beta)^n$ [GN08, HPS11]. However, experiments have shown that BKZ runtime is polynomial in dimension and exponential in blocksize [GN08,SB10,CN11, BLR08, PS13, FSW14].

- [GN08] (p.14): "[Using LLL, exact] SVP can be solved in dimension 60 within an hour, but ... a 100-dimensional lattice [is estimated to take] at least 35,000 years."
- [GN08] (p.15): "No good upper bound on the complexity of BKZ and DEEP [-LLL] is known. ... The best upper bound is $(n\beta)^n$ ..., but this upper bound does not seem tight: it only takes a few seconds to reduce a 100-dimensional lattice with blocksize 20."
- [SB10] (p.248): "[The runtime for] NTL's BKZ-50 for a 108-dimensional lattice to be around 1 year and for a 150-dimensional lattice around 1300 years."

It would be interesting to use the BKZ 2.0 simulation algorithm [CN11] to confirm or disprove the above statement on BKZ-50 runtime. We leave that for the next group.

Whereas most papers have stated polynomial runtime for fixed blocksize, all have used a log-scale when plotting time against dimension rather than a log-log scale. Their graphs look less than linear [GN08, FSW14, PS13, BLR08], which is an indication that runtime is at most sub-exponential in dimension. For higher blocksize, the polynomial degree is much higher, so some plots look exponential over the small number of data points that they contain. [GN08] also provides a graph indicating a relationship between time and blocksize as at least exponential. See our models in Section 4.2 and estimations in Section 4.3.

The possible exception to this is for higher blocksize. Around blocksize 23, there is a sharp increase in runtime [GN08, HPS11]; we found a large jump in runtime from BKZ-20 to BKZ-25 as well . The explanation given by [GN08] is that the number of calls to the enumeration subroutine suddenly increases. [SB10] reiterates this with a claim that the amount of time spent enumerating is more than 99% of total reduction time for blocksize ≥ 40.

As enumeration techniques improve, we believe the blocksize of sharp increase should be pushed higher. [SB10] states that high blocksize reductions would benefit from the improvements in enumeration techniques over the years. [CN11] implements improvements in their BKZ 2.0 algorithm.

3. Our Experiment

Our goal is to determine whether lattice class has an impact on the RHF or runtime. Thus, we run LLL and BKZ on comparable matrices.

Step 1: Generate these matrices.

1. The Lattice Challenge [BLR08, LRBN10] provides a framework for constructing general lattices. The scheme guarantees short vectors [Ajt96].

2. To construct ideal lattices, use the scheme of Plantard and Schneider [PS13, LRBN10].
3. In both cases, the matrices are generated using a random seed which allows for exact duplication of the matrix in question.
4. To determine if the starting basis has an effect (on runtime and RHF), construct unimodular matrices so as to change the starting basis, but preserve the lattice's form.

Step 2: Run lattice reduction algorithms.

1. For each of the lattices, run LLL and BKZ-5, 10, 15, 20, and 25 and record the runtime for the reduction and root Hermite factor from the reduced basis.
2. Transform the lattice basis into a "random" [Ngu17] basis (for the same lattice) by multiplying by a set of 5 unimodular matrices. Then rerun the reduction algorithms for each.
3. The same unimodular matrices are used for ideal and general lattices of the same dimension.

Step 3: Compare results.

1. Use least squares estimation (LSE) to fit regression models for RHF and time.
2. The experimental factors to consider are blocksize, dimension, lattice class, and cyclotomic index (when lattice class is restricted to ideal).
3. Determine whether or not the lattice class (and/or cyclotomic index) variable is statistically significant.

Additional detail about statistical models and analysis is provided in Section 4. Additional notes about our experimental process are provided in Appendix A.

4. Statistical Analysis of PeRIL's Experiments

In this section, we describe the statistical methodology and resulting analysis performed on the lattice reduction performance results collected by the PeRIL team.

4.1 Experimental Variables

- Two response variables were considered: Root Hermite Factor (RHF) and experimental runtime. Both response variables quantify algorithm performance.
- Five predictor variables were considered based on the following experimental factors: BKZ blocksize, dimension, lattice class, and cyclotomic index (for ideal lattices).[3]
- Factors for future consideration include computational precision, operating system, background programs (e.g., anti-virus software), software (e.g., package used), methods for generating unimodular matrices, hyperparameters (e.g., bounds for matrix entries). These factors may impact runtime and/or RHF.

[3]The LLL algorithm is a specialization of BKZ with blocksize 2. 'Algorithm' could have been used as a categorical representation of blocksize.

4.2 Methods

Linear regression models were used to inform our investigation of lattice reduction algorithm performance (response variables) in relation to characteristics of interest (predictor variables). As a matter of notation, we use “Model (i)” to reference the model expressed in Equation (i). Since our research focuses on comparing the performance of general lattices with that of ideal lattices, our formulations extend the models presented in the literature [SB10, GN08, FSW14, PSZ13] with the addition of lattice class. In our investigation of the usefulness of lattice class for predicting RHF or runtime, we also posited the question of whether cyclotomic index (for ideal lattices only), influences the RHF or runtime.

We chose to test multiple models to ascertain that lattice class was not significant in more than one specific model. Showing the non-significance in multiple models, covering different combinations of what may reasonably affect RHF and runtime, helps provide evidence toward our conclusion that lattice class is not a useful predictor of our response variables.

4.2.1 Root Hermite Factor Models

The following model expresses the RHF as a function of our experimental factors

$$RHF_i = \beta_0 + \beta_1 X_{1,i} + \beta_2 X_{2,i} + \beta_3 X_{3,i} + \varepsilon_i, \tag{5}$$

where X_1 represents blocksize, X_2 represents dimension, and X_3 represents lattice class (0: general, 1: ideal). ε denotes the random error term, which was assumed (for convenience) to follow the traditional assumptions of a linear model (e.g., independent and identically distributed following a normal distribution with mean zero and constant variance), and i denotes the observation index.

For analyses on solely ideal lattices, we augment Model (5) by adding cyclotomic index (X_4), and X_3 is set to 1 since the lattice class is fixed:

$$\textbf{Ideal} \qquad \begin{aligned} RHF_i &= \beta_0 + \beta_1 X_{1,i} + \beta_2 X_{2,i} + \beta_3\ (1) + \beta_4 X_{4,i} + \varepsilon_i \\ &= \underbrace{(\beta_0 + \beta_3)}_{\text{intercept}} + \beta_1 X_{1,i} + \beta_2 X_{2,i} + \beta_4 X_{4,i} + \varepsilon_i. \end{aligned} \tag{6}$$

Note that the β_i’s in Eq. (5) are not the same as the β_i’s in Eq. (6), nor are the error terms ε_i. They are simply placeholder coefficients. This is the case for all models presented.

Since cyclotomic index is largely dependent on dimension[4], it may be more appropriate to use $\frac{X_4}{X_2}$ to differentiate its impact on RHF:[5]

$$\textbf{Ideal} \qquad RHF_i = \beta_0 + \beta_1 X_{1,i} + \beta_2 X_{2,i} + \beta_{4,2} \frac{X_{4,i}}{X_{2,i}} + \varepsilon_i. \tag{7}$$

Models (6) and (7) differ in that they consider different data transformations for index and dimension.[6]

[4]Note that we have the bounds $\frac{C}{\log\log(C)} < \phi(C) < C$, where $\phi(C)$, the Euler-totient function, is the dimension and C is the cyclotomic index [HW+79]. Thus $1 < \frac{C}{\phi(C)} < \log\log(C)$.

[5]The ratio $\frac{X_4}{X_2}$ could be represented as a“new” predictor (R) to maintain the linearity of the model. Note that the use of dependent predictors R and X_2 still violates a traditional assumption of independence between predictors in a linear model.

[6]When comparing Model (6) to Model (7), note that the X_3 is set to 1 in both models.

According to the literature [SB10, SBL09, GN08, FSW14, PS13, ACD+18, Ngu17] and our experimental data, RHF converges to a fixed value at higher dimensions. Thus, we reduce the previous Models (5) and (7) to

$$RHF_i = \beta_0 + \beta_1 X_{1,i} + \beta_3 X_{3,i} + \varepsilon_i, \tag{8}$$

Ideal
$$RHF_i = \beta_0 + \beta_1 X_{1,i} + \beta_{4,2} \frac{X_{4,i}}{X_{2,i}} + \varepsilon_i, \tag{9}$$

respectively.

The following simple linear model (with predictor X_1) is also considered to allow comparison with [SB10]'s results (show in our Eq. (4)):

$$RHF_i = \beta_0 + \beta_1 X_{1,i} + \varepsilon_i. \tag{10}$$

These models have a limited range of applicability. We know that extrapolation beyond our sample range is dangerous. But even more so, since RHF and blocksize are negatively correlated (see Figure 2), for large enough blocksize we would obtain an estimated RHF well below 1 (and for extremely large blocksize, > 6000, a negative estimated RHF).[7] A more likely relationship between RHF and blocksize is $RHF = D \cdot X_1^{-c}$ for some positive constants D, c as supported by evidence in [BLR08, GN08, CN11]. An alternative but similar relationship is $RHF = D \cdot e^{-c \cdot X_1}$.

We considered additional models in which a transformation is applied to response and/or predictor variables, where the transformation's functional form of is inspired by the theoretical background underlying our particular application. For example, we considered transformations of RHF that account for it being strictly positive. Exponential and inverse polynomial are candidates with the positivity property.[8]

The transformed variables are shown in the following models:

$$\log RHF_i = \beta_0 + \beta_1 \log(X_{1,i}) + \beta_3 X_{3,i} + \varepsilon_i, \tag{11}$$

$$\log RHF_i = \beta_0 + \beta_1 X_{1,i} + \beta_3 X_{3,i} + \varepsilon_i, \tag{12}$$

Ideal
$$\log RHF_i = \beta_0 + \beta_1 \log(X_{1,i}) + \beta_{4,2} \frac{X_{4,i}}{X_{2,i}} + \varepsilon_i, \tag{13}$$

Ideal
$$\log RHF_i = \beta_0 + \beta_1 X_{1,i} + \beta_{4,2} \frac{X_{4,i}}{X_{2,i}} + \varepsilon_i. \tag{14}$$

Models (11) and (13) transforms the response and blocksize from Models (8) and (9), respectively. Models (12) and (14) transforms only the response from Models (8) and (9), respectively. These may give better results over a larger range of blocksizes, but eventually the estimation would approach arbitrarily close to 0 (in violation of the Gaussian heuristic which approaches 1 from above).[9]

[7] There is the caveat that larger blocksizes require a larger dimension in which to be run. But β_1 (see Section 4.3), is much greater in absolute value than β_2 and β_4.

[8] RHF very likely also has a positive lower bound in asymptotic averages. The addition of an additive constant to a model here, or another form, is left to the next group.

[9] For data reproducibility, our use of log is with respect to base $e = 2.71828\ldots$.

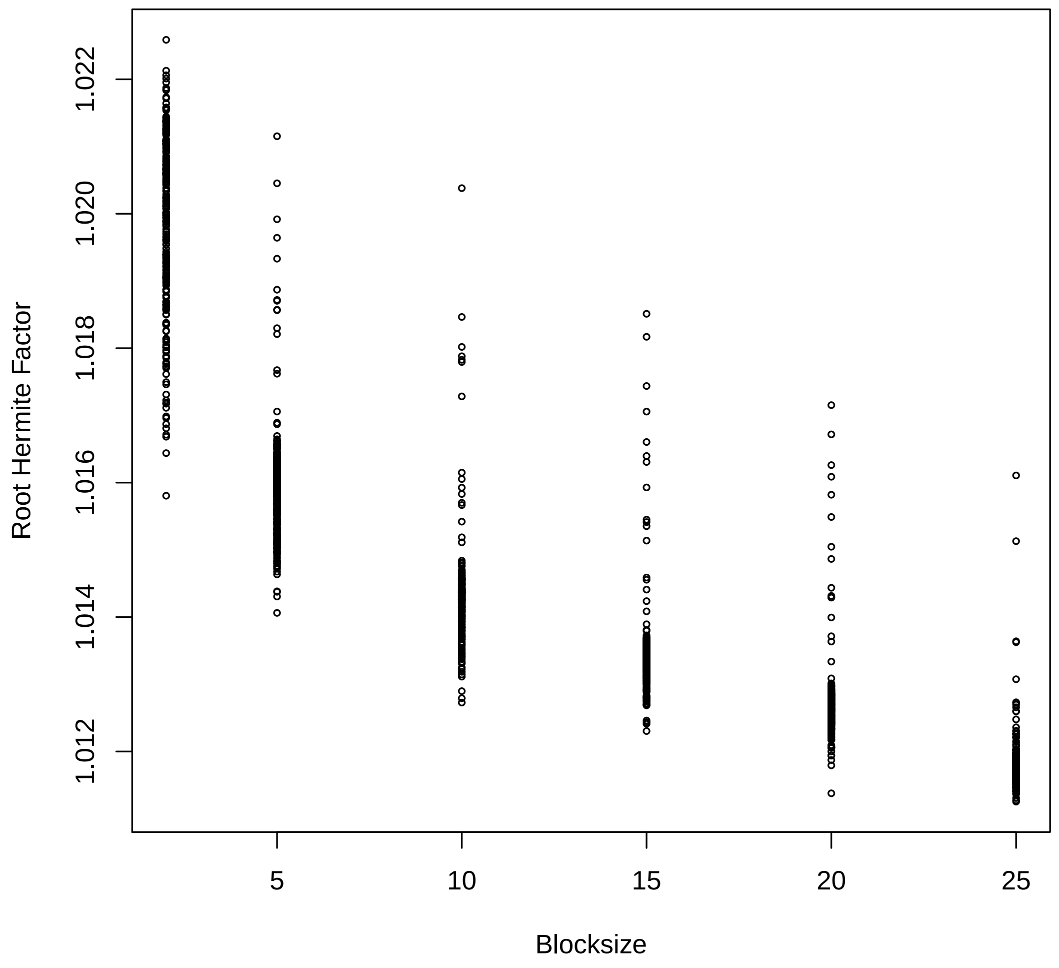

Figure 2: A demonstration of the negative correlation between RHF obtained and blocksize. Data was pulled from the subset of datawith dimension ≥ 50 (to remove noise at lower dimensions).

4.2.2 Runtime Models

The following model expresses the log-transformed algorithm runtime (T) as a function of our predictors:

$$\log T_i = \beta_0 + \beta_1 X_{1,i} + \beta_2 X_{2,i} + \beta_3 X_{3,i} + \varepsilon_i, \tag{15}$$

$$\textbf{Ideal} \qquad \log T_i = \beta_0 + \beta_1 X_{1,i} + \beta_2 X_{2,i} + \beta_4 X_{4,i} + \varepsilon_i. \tag{16}$$

Models (15) and (16) use the same linear predictors as Models (5) and (6), respectively.

We are interested in determining whether there is a statistically significant difference between the runtime of lattice reduction algorithms for the ideal lattices compared to the general lattices. The following hypotheses identify the regression coefficients from Model (15) that quantify the impact of lattice class in a model that already includes blocksize and dimension (X_1 and X_2, respectively).

$$H_0\colon \beta_3 = 0 \text{ vs. } H_1\colon \beta_3 \neq 0.$$

For the ideal lattice case, we are also interested in the impact of cyclotomic index (X_4; in Model (16)) on the runtime of lattice reduction algorithms. The corresponding hypotheses for the relevance of cyclotomic index are given by

$$H_0\colon \beta_4 = 0 \text{ vs. } H_1\colon \beta_4 \neq 0.$$

As with RHF Model (7), Model (16) replaces the raw cyclotomic index (X_4) with the cyclotomic index-dimension ratio (X_4/X_2):

$$\textbf{Ideal} \qquad \log T_i = \beta_0 + \beta_1 X_{1,i} + \beta_2 X_{2,i} + \beta_{4,2} \frac{X_{4,i}}{X_{2,i}} + \varepsilon_i. \tag{17}$$

It is known that the runtime of LLL is polynomial with respect to dimension, and it is strongly conjectured that BKZ is as well [Ngu11, SB10]. Many resources have shown a less than exponential relationship between time and dimension [GN08, FSW14, PS13, BLR08]. Thus, we transform Models (15) and (17) as follows:

$$\log T_i = \beta_0 + \beta_1 X_{1,i} + \beta_2 \log(X_{2,i}) + \beta_3 X_{3,i} + \varepsilon_i \text{ and} \tag{18}$$

$$\textbf{Ideal} \qquad \log T_i = \beta_0 + \beta_1 X_{1,i} + \beta_2 \log(X_{2,i}) + \beta_{4,2} \frac{X_{4,i}}{X_{2,i}} + \varepsilon_i, \tag{19}$$

respectively. For a direct comparison to the literature, we also produce the further reduced models:

$$\log T_i = \beta_0 + \beta_1 X_{1,i} + \beta_2 X_{2,i} + \varepsilon_i, \text{ and} \tag{20}$$

$$\log T_i = \beta_0 + \beta_1 X_{1,i} + \beta_2 \log(X_{2,i}) + \varepsilon_i, \tag{21}$$

from which X_3 and $\frac{X_4}{X_2}$ were removed. Model (20) is a reduced version of Models (15), (16), and (17). Model (21) is a reduced version of Models (18) and (19).

Further support for the nonlinear relationship between RHF and blocksize is that the relation between time and (-RHF) appears to be superexponential [RS10]. Since time is exponential with respect to blocksize, X_1, a superlinear relationship between RHF and blocksize would yield a superexponential relationship between time and RHF as RHF decreases.

4.3 Results

In this section, we discuss results from analysis of our lattice reduction experiments using the methods discussed in Section 4.2. Regression coefficients were calculated using traditional least squares estimation (LSE) and rounded to 6 decimal places. See Appendix B for standard errors of model coefficients.

4.3.1 Root Hermite Factor

Eq. (22) through (29) were calculated using our experimental results. Our results were run including LLL data (setting blocksize=2). Models (5) through (10) are estimated in Eq. (22) through (27), respectively.

$$\widehat{RHF} = 1.013116 - 0.000181\,X_1 + 0.000025\,X_2 + 0.000843\,X_3. \tag{22}$$

Ideal $$\widehat{RHF} = 1.013736 - 0.000187\,X_1 + 0.000013\,X_2 + 0.000009\,X_4. \tag{23}$$

Ideal $$\widehat{RHF} = 1.014328 - 0.000182\,X_1 + 0.000021\,X_2 - 0.000016\,\frac{X_4}{X_2}. \tag{24}$$

$$\widehat{RHF} = 1.016087 - 0.000177\,X_1 + 0.000044\,X_3. \tag{25}$$

Ideal $$\widehat{RHF} = 1.017707 - 0.000176\,X_1 - 0.000825\,\frac{X_4}{X_2}. \tag{26}$$

$$\widehat{RHF} = 1.016118 - 0.000177\,X_1. \tag{27}$$

General $$\widehat{RHF} = 1.016118 - 0.000179\,X_1. \tag{28}$$

Ideal $$\widehat{RHF} = 1.016118 - 0.000175\,X_1. \tag{29}$$

In Eq. (22) to (29), **Ideal** indicates that results were calculated from data of ideal lattices only, whereas **General** indicates that results were calculated from data of general lattices only. Otherwise, all lattices (both general lattices and ideal lattices) were used to estimate a given model.

The difference between the regression estimates of Eq. (4) and (27) cannot be directly compared for several reasons. Experimental factors varied between our experiments and [SBL09]'s experiments (e.g., different blocksizes used). Additionally, the standard errors for Eq. (4)'s regression coefficients are not provided in [SBL09], limiting our ability to conduct formal hypothesis tests.

See Appendix B.1 for regression coefficients, including standard errors.

Based on our experimental results (Tables 2, 3, and 4), dimension was identified as a statistically useful[10] predictor of RHF. However, according to the literature (theoretically [SBL09] and experimentally [SB10, GN08]), dimension does not affect RHF, apart from allowing possibly shorter vectors at low dimension. In lower dimensions, BKZ actually solves the SVP giving much lower RHFs than expected asymptotically. This leads to the possibly erroneous conclusion that dimension is a useful *linear* predictor. If dimension is *asymptotically* relevant as a linear predictor, then β_2 should remain constant as dimension changes. In contrast to that expectation, when we restricted our data to lattices with dimensions above a given value (0, 50, 100, 150, and 200), we obtained varied estimates for β_2 ($\widehat{\beta}_2 \cdot 10^6 = 25, 9, 6, 0,$ and 3, respectively) in our estimation of Model (5).

[10]This determination holds at any reasonable individual-test significance level. P-values for each of the tests are given in Tables 2, 3, and 4. Note that our results do not account for potential dependence between predictors.

The estimates for β_2 similarly become smaller in magnitude in the estimations of Models (6) and (7) with higher dimension sampling.[11] This provides evidence backing our choice to remove lone dimension (X_2) as a useful predictor of RHF. [12]

Considering that the true functional relation between RHF and blocksize is unlikely to be linear over large enough ranges, Models (11) through (14) are estimated by

$$\log\widehat{(RHF)} = 0.017495 - 0.001673\ \log(X_1) + 0.000054\, X_3, \tag{30}$$

$$\log\widehat{(RHF)} = 0.015924 - 0.000173\, X_1 + 0.000059\, X_3, \tag{31}$$

$$\textbf{Ideal} \qquad \log\widehat{(RHF)} = 0.019148 - 0.001684\ \log(X_1) - 0.000822\ \frac{X_4}{X_2},\ \text{and} \tag{32}$$

$$\textbf{Ideal} \qquad \log\widehat{(RHF)} = 0.017549 - 0.000173\, X_1 - 0.000815\ \frac{X_4}{X_2}, \tag{33}$$

respectively. These give relationships of

$$RHF \approx X_1^{-0.001673},\ e^{-0.000173\ X_1},\ X_1^{-0.001684},\ e^{-0.000173\ X_1},$$

respectively. For blocksize $X_1 = 100$ (extrapolating far beyond the initial sampling space), the RHF estimated by Eq. (4) is 0.996932 and Eq. (27) estimates 0.998418, both of which seem optimistic. For general lattices ($X_3 = 0$), Eq. (30) estimates 1.009839 (pessimistic based on recent accomplishments [RS10, CN11]), and Eq. (31) estimates 0.998625. Assuming a cyclotomic index of $X_4 = 1232$, thus dimension $X_2 = \phi(1232) = 480$, Eq. (32) and (33) estimate similar values (1.009326 and 0.998159 respectively).

Regarding the reasonability of estimated models, particularly in light of suggested variable transformations, we found that the estimated model with untransformed variables (e.g., Eq. (27)) would not produce obviously erroneous predictions for RHF until very high dimension, so it is not a surprise that estimates from "improved" Models (11)-(14) do not produce very different results over our limited range. Thus, models with untransformed variables provides a reasonable approximate when we restrict the domain. We are presenting the alternatives here as an idea for a reader to use as future study.

Based on a 0.05 significance level, lattice class is a useful predictor to add to a model for RHF that already includes predictors X_1 and X_2 (p-value: 0.000016; Table 2). However, the opposite result was seen regarding the usefulness of lattice class in models with fewer predictors (e.g., Tables 5, 10, and 11). This may be a result of model misspecification. As discussed earlier, dimension does not asymptotically affect the RHF [SBL09, SB10, GN08]. Thus, in light of the theory for our particular application, we are more inclined to favor the results from models without dimension.

Similarly, cyclotomic index is seen as a useful predictor in a model with X_1 and X_2; Table 3. But recall that cyclotomic index and dimension (X_2) are heavily correlated[13]. In

[11]Note that blocksize was dependent on dimension to some extent (e.g., can't run BKZ-β for dimension $n < \beta$). Since our range was not restricted to dimensions above 25, the representation by algorithm could be an issue. Areas for potential improvement and future consideration include handling missing values that accounts for the known relationship between dimension and blocksize.

[12]Additional methods for appropriately handling dimension and accounting for the variation observed may be considered for future research.

[13]Recall the dimension of a lattice with cyclotomic index C is $\phi(C)$.

fact, the coefficient estimate of β_2 in Model (5), Eq. (22) is roughly the sum of β_2 and β_4 from Eq. (23)[14]. When $\frac{X_4}{X_2}$ was used instead of X_4, we obtain mixed results for its significance (Tables 4, 6, 12, and 13). In fact, this ratio was a useful addition to models that did not already include dimension, but when dimension was already included in the model that ratio was not a useful predictor (at any reasonable significance level, based on results from an individual t-test with hypotheses H_0: $\beta_{4,2} = 0$ vs. H_1: $\beta_{4,2} \neq 0$). However, X_4 was always determined to be a useful additional predictor for every model considered, regardless of whether dimension was or was not already included in those models.

4.3.2 Runtime

Estimates regression equations for runtime, corresponding to Models (15), (16), and (17) are given by

$$\widehat{\log T} = -4.844043 + 0.109764\,X_1 + 0.059562\,X_2 - 0.054894\,X_3. \tag{34}$$

Ideal $$\widehat{\log T} = -5.735850 + 0.095660\,X_1 + 0.051618\,X_2 + 0.011606\,X_4. \tag{35}$$

Ideal $$\widehat{\log T} = -5.407435 + 0.101272\,X_1 + 0.063402\,X_2 + 0.140254\,\frac{X_4}{X_2}. \tag{36}$$

We suspect, and are supported by the literature [GN08,FSW14,PS13,BLR08], that runtime is only polynomial (not exponential) with respect to dimension. Corresponding estimated regression equations, which are based on Models (18) and (19), are given by

$$\widehat{\log T} = -19.699629 + 0.068977\,X_1 + 4.927231\,\log(X_2) - 0.297858\,X_3, \tag{37}$$

Ideal $$\widehat{\log T} = -19.024828 + 0.060018\,X_1 + 4.823624\,\log(X_2) - 0.221438\,\frac{X_4}{X_2}. \tag{38}$$

The estimations in Eq. (37) and (38) have values of $R^2 \approx 0.93$ versus $R^2 \approx 0.87$ from Eq. (34) through (36). That is, more variability in the response was explained by models that used the log-transformed runtime as the response variable, compared to models for the untransformed runtime. This empirical evidence aligns with our expectation that runtime is polynomial in dimension.

For comparisons to other papers, the most reduced Models (20) and (21) are estimated as

$$\widehat{\log T} = -4.891670 + 0.109753\,X_1 + 0.059656\,X_2, \text{ and} \tag{39}$$

$$\widehat{\log T} = -20.054940 + 0.068648\,X_1 + 4.961946\,\log(X_2), \tag{40}$$

respectively.

At the 0.05 significance level, lattice class is not a useful predictor to add to a model for log runtime that already includes predictors X_1 and X_2 (p-value = 0.45; Table 14). However the opposite conclusion is seen for models that additionally include X_3. That is, lattice class

[14]Directions for future improvement to this model include considering methods and implications for handling dependence among predictors.

is a useful predictor to add to a model that already includes X_1, X_2, and X_3 ($p = 0.000001$; Table 17).[15]

On average, there is an increase in runtime from the unmodified lattice to any multipled lattice (original basis multiplied by a unimodular matrix). The average factor of increase is 2.18 for general lattices and 1.94 for ideal lattices. The increase can be explained in one of two ways. Either the original lattice (and the special form it has) is that much easier to reduce or the adjusted bases have entries of order higher than 10^n. The reason for the increase is not addressed in this paper. Since some transformations actually reduced the runtime (almost exclusively for BKZ-25), it may be that the form was more of a factor than bitsize.

However, the difference in increases between general and ideal lattices cannot be explained by the above. If a difference exists, it may have potential cryptographic ramifications. Since we showed that random bases for general and ideal lattices have roughly the same runtime, this would mean the special form of the constructed general lattice is actually reduced *faster* than the constructed form of the ideal lattice by current algorithms. We encourage anyone to test if there is a difference.

5. Conclusion

In this paper, we described an experiment that we conducted in order to evaluate the impact of lattice class on result quality and computational runtime for using lattice reduction algorithms. Our research is novel in its direct comparison of experimental results between the class of general lattices and the class of ideal lattices. In this paper, we described our methods for collecting algorithm performance (e.g., quality and runtime) data. Then, we developed statistical models to analyze this performance data. Finally, results from statistical models were described in light of our overarching questions. A summary of key results is provided below.

Comparing statistical results from various regression models estimated to our experimental data, we did not find consistent evidence to determine whether lattice class has a significant impact on RHF or runtime. In models for RHF including blocksize and dimension as predictors, lattice class appears to be a useful predictor. But once we remove dimension, as we argued that it shouldn't be asymptotically relevant to RHF prediction, we found that lattice class lost its usefulness for predicting RHF. In the opposite direction, for runtime we found that lattice class was not a useful predictor when combined with blocksize and dimension. Once we adjusted the predictors so that runtime was modeled as polynomial in dimension, as referenced in the literature, we found that lattice class became a useful predictor of runtime.

Neither did we find consistent evidence to determine whether cyclotomic index has a significant impact on RHF or runtime. The conclusions depended on the overall model being run. We provided an explanation for why cyclotomic index and dimension shouldn't both be used as linear predictors in the same model due to their high correlation. This

[15]As previously indicated, application-specific theory is relevant in considering whether or not to include dimension in the model. If instead, the inclusion of dimension was based solely on its predictive merit as observed in statistical models applied to our data, dimension would be included as a predictor because it is a statistically useful predictor to add to a model that already includes blocksize and class (at any reasonable signficance level, based on an individual t-test for β_2; Table 14).

left us with models that included index divided by dimension as a fifth “linear” predictor. When dimension was also included, this fifth predictor was not useful, but when dimension was left out, it appeared significant to predicting RHF.

Work building on this would further investigate the functional specification of our model. Specifically, the addition of interactions between the predictors may be considered. Using a common precision across all dimensions may help solidify conclusions by removing this source of variability which was not addressed in our paper.

Acknowledgements

This research was supported by the DoD section 219 Naval Innovative Science and Engineering Program.

References

[ACD$^+$18] Martin R. Albrecht, Benjamin R. Curtis, Amit Deo, Alex Davidson, Rachel Player, Eamonn W. Postlethwaite, Fernando Virdia, and Thomas Wunderer. Estimate all the {LWE, NTRU1} schemes! Cryptology ePrint Archive, Report 2018/331, 2018.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108. ACM, 1996.

[Ajt98] Miklós Ajtai. The shortest vector problem in L_2 is np-hard for randomized reductions. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 10–19. ACM, 1998.

[Bat15] S. Batson. *On the Relationship Between Two Embeddings of Ideals into Geometric Space and the Shortest Vector Problem in Principal Ideal Lattices*. PhD dissertation, NC State University, 2015.

[BLR08] Johannes Buchmann, Richard Lindner, and Markus Rückert. Explicit hard instances of the shortest vector problem. In *International Workshop on Post-Quantum Cryptography*, pages 79–94. Springer, 2008.

[Che16] Hao Chen. A measure version of gaussian heuristic. IACR Cryptology ePrint Archive, Report 2016/439, 2016.

[CN11] Yuanmi Chen and Phong Q Nguyen. BKZ 2.0: Better lattice security estimates. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 1–20. Springer, 2011.

[dP12] J. Van de Pool. Quantifying the security of lattice-based crytosystems in practice, Jan 2012.

[dP14] J. Van de Pool. The BKZ algorithm, May 2014.

[FSW14] Felix Fontein, Michael Schneider, and Urs Wagner. PotLLL: a polynomial time version of LLL with deep insertions. *Designs, codes and cryptography*, 73(2):355–368, 2014.

[GN08] N. Gama and P. Nguyen. Predicting lattice reduction. *Advances in Cryptology – EuroCrypt 2008*, 4965:31–51, 2008. [proceedings].

[Ham13] T. Hamann. The BKZ simulation algorithm. Thesis, Technische Universität Darmstadt, July 2013.

[HPS11] Guillaume Hanrot, Xavier Pujol, and Damien Stehlé. Terminating BKZ. *IACR Cryptology ePrint Archive*, 2011:198, 2011.

[HW$^+$79] Godfrey Harold Hardy, Edward Maitland Wright, et al. *An introduction to the theory of numbers*. Oxford university press, 1979.

[LLL82] Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[LRBN10] R. Lindner, M. Rüeckert, P. Baumann, and L. Nobach. TU Darmstadt Lattice Challenge, 2010.

[Ngu11] Phong Q Nguyen. Lattice reduction algorithms: Theory and practice. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 2–6. Springer, 2011.

[Ngu17] P. Nguyen. Lattice algorithms: Design, analysis and experiments, March 2017.

[PS13] Thomas Plantard and Michael Schneider. Creating a challenge for ideal lattices. *IACR Cryptology EPrint Archive*, 2013:39, 2013.

[PSZ13] T. Plantard, W. Susilo, and Z. Zhang. Adaptive precision floating point lll. *Australasian Conference on Information Security and Privacy (ACISP 2013)*, 7959:104–117, 2013.

[RS10] Markus Rückert and Michael Schneider. Estimating the security of lattice-based cryptosystems. Cryptology ePrint Archive, Report 2010/137, 2010.

[SB10] Michael Schneider and Johannes A Buchmann. Extended lattice reduction experiments using the BKZ algorithm. In *Sicherheit*, volume 10, pages 241–252, 2010.

[SBL09] M. Schneider, J. Buchman, and R. Lindner. Probabilistic analysis of LLL reduced bases. *Algorithms and Number Theory. Dagstuhl Seminar Proceedings*, 2009.

[SE94] C. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical Programming*, 66:181–199, 1994.

Appendix

A. Notes about our experimental process

- Software used:
  - NTL version 10.1.0 and C++11 for lattice reductions and Sage version 5.10 for lattice generation.
  - For the statistical analysis, we used the linear model function in R version 3.5.1.
  - Our "runtime" is actually the walltime. We presume that the cputime is quite similar. Reductions were run on a dedicated computer.
- Dimensions considered:
  - BKZ-n can only be run on dimension $\geq n$. So BKZ-5 was only run on dimension ≥ 5 lattices, BKZ-10 was only run on dimension ≥ 10 lattices, etc. BKZ-25 was not run on lattices of dimension > 196 (or some lattices of dimension 190,192,194) due to precision errors and time constraints.
  - We only used ideal lattices corresponding to ideals in cyclotomic fields of index ≤ 275. This meant we had repeats of some dimensions and missed other dimensions.
  - For general lattices, we used every even dimension < 200 as well as those dimensions covered by ideal lattices above.
  - We had to change precision at different stages: floating point for $n \leq 50$, extended double for $52 \leq n \leq 194$, $n = 198$, and LLL and BKZ-5,10,15 for $n = 200$, and arbitrary precision for the rest. Floating point and extended double are hardware defined as compared to the software defined arbitrary precision. This causes a statistical artifact of much higher runtimes for reduction when arbitrary precision is used.
  - There are a few outliers in our data that we think were caused by errors in precision. We did not rerun these instances because of time constraints.
- Lattice basis creation:
  - We created matrices that had entries on the order of 10^n.
  - "There is no natural probability space over the infinite set of bases" [Ngu17]. We decided to address this issue by multiplying the constructed lattice (with very rigid form, see [LRBN10, PS13]) by a random unimodular matrix.
  - For each dimension, we created 5 unimodular matrices to change the basis. Initially, the unimodular matrices were constructed using the Sage command **random_matrix(ZZ,dim,dim,algorithm='unimodular')**.
    This proved to take too long in higher dimensions ($>$ 1 month for the highest dimensions), so we changed to the following technique for dimensions > 180.

1. Construct a general random matrix using the Sage command **A=random_matrix(ZZ,dim,dim)**.
2. Compute the Hermite form using **H,U=Hermite_form(transformation=True)**.
3. The secondary output of this function is the unimodular matrix U such that $UA = H$.
   We assume this U is a sufficiently random unimodular matrix and use it for the change of bases.

– Because we wanted to consider performance of lattice reduction algorithms on random bases of random lattices, we only used the transformed data in our analysis.

B. Regression Data

If there is any discrepancy between coefficients listed here and coefficients listed in Section 4.3, the data in this section should be presumed to be correct.

In our first attempt of analysis we found that the estimated regression equation based on Model (10) was given by.

$$\widehat{RHF} = 1.025402 - 0.000684 \text{ Blocksize}.$$

For ease of comparison, we repeat Eq. (4) from the literature:

$$\widehat{RHF} = 1.01655 - 0.000196185 \text{ Blocksize}.$$

The large disparity with Eq. (4) gave us pause. After reviewing our data, we found that our RHF from LLL tended to approach 1.04 compared to the 1.02 in literature [SB10, SBL09, GN08, FSW14, PS13]. We believe, though were unable to confirm, that our LLL data was erroneously squared or missing a square root.[16] The regression equations were built with the "corrected" data. We also ran the regression equations with the "uncorrected" data and arrived at the same conclusions.

B.1 RHF

Tables 2 through 9 are estimating RHF. Tables 10 through 13 are estimating $\log(\text{RHF})$.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.013116	0.000269	3771	0.000000
Blocksize	-0.000181	0.000011	-16.45	0.000000
Dimension	0.000025	0.000001	17.44	0.000000
Class	0.000843	0.000195	4.318	0.000016

Table 2: Estimation of Model (5) using both ideal and general data sets combined.

[16]The LLL data was produced using a separate function in the NTL library, hence the BKZ data was as expected. Potentially in the version of NTL we used, it was built to output the approximation factor as opposed to the Hermite factor. We checked in an updated version of NTL and found it output 1.02 consistently, as expected.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.013736	0.000185	5468	0.000000
Blocksize	-0.000187	0.000009	-20.88	0.000000
Dimension	0.000013	0.000002	6.995	0.000000
Index	0.000009	0.000001	6.605	0.000000

Table 3: Estimation of Model (6) using only data from ideal lattices.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.014328	0.000316	3208	0.000000
Blocksize	-0.000182	0.000009	-20.12	0.000000
Dimension	0.000021	0.000001	14.86	0.000000
$\frac{\text{Index}}{\text{Dimension}}$	-0.000016	0.000107	-0.1459	0.883999

Table 4: Estimation of Model (7) using only data from ideal lattices.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.016087	0.000222	4575	0.000000
Blocksize	-0.000177	0.000012	-14.99	0.000000
Class	0.000044	0.000203	0.2166	0.828554

Table 5: Estimation of Model (8) using both ideal and general data sets combined.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.017707	0.000235	4323	0.000000
Blocksize	-0.000176	0.000010	-18.19	0.000000
$\frac{\text{Index}}{\text{Dimension}}$	-0.000825	0.000098	-8.394	0.000000

Table 6: Estimation of Model (9) using only data from ideal lattices.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.016118	0.000170	5963	0.000000
Blocksize	-0.000177	0.000012	-15	0.000000

Table 7: Estimation of Model (10) using both ideal and general data sets combined.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.016118	0.000143	7091	0.000000
Blocksize	-0.000175	0.000010	-17.72	0.000000

Table 8: Estimation of Model (10) using only data from ideal lattices.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.016118	0.000461	2203	0.000000
Blocksize	-0.000179	0.000032	-5.621	0.000000

Table 9: Estimation of Model (10) using only data from general lattices.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	0.017495	0.000289	60.46	0.000000
log(Blocksize)	-0.001673	0.000107	-15.69	0.000000
Class	0.000054	0.000204	0.2646	0.791363

Table 10: Estimation of Model (11) using both ideal and general data sets combined.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	0.015924	0.000224	70.96	0.000000
Blocksize	-0.000173	0.000012	-14.52	0.000000
Class	0.000059	0.000205	0.289	0.772593

Table 11: Estimation of Model (12) using both ideal and general data sets combined.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	0.019148	0.000273	70.23	0.000000
log(Blocksize)	-0.001684	0.000085	-19.82	0.000000
$\frac{\text{Index}}{\text{Dimension}}$	-0.000822	0.000096	-8.606	0.000000

Table 12: Estimation of Model (13) using only data from ideal lattices.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	0.017549	0.000233	75.3	0.000000
Blocksize	-0.000173	0.000010	-18.07	0.000000
$\frac{\text{Index}}{\text{Dimension}}$	-0.000815	0.000097	-8.379	0.000000

Table 13: Estimation of Model (14) using only data from ideal lattices.

B.2 Runtime

Tables 14 through 20 are all estimating log(Time).

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-4.844043	0.100053	-48.41	0.000000
Blocksize	0.109764	0.004103	26.75	0.000000
Dimension	0.059562	0.000533	111.7	0.000000
Class	-0.054894	0.072726	-0.7548	0.450447

Table 14: Estimation of Model (15) using both ideal and general data sets combined.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-5.735850	0.093633	-61.26	0.000000
Blocksize	0.095660	0.004513	21.2	0.000000
Dimension	0.051618	0.000909	56.77	0.000000
Index	0.011606	0.000720	16.11	0.000000

Table 15: Estimation of Model (16) using only data from ideal lattices.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-5.407435	0.170273	-31.76	0.000000
Blocksize	0.101272	0.004867	20.81	0.000000
Dimension	0.063402	0.000771	82.25	0.000000
$\frac{\text{Index}}{\text{Dimension}}$	0.140254	0.057437	2.442	0.014727

Table 16: Estimation of Model (17) using only data from ideal lattices.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-19.699629	0.145181	-135.7	0.000000
Blocksize	0.068977	0.002937	23.49	0.000000
log(Dimension)	4.927231	0.030167	163.3	0.000000
Class	-0.297858	0.051341	-5.802	0.000000

Table 17: Estimation of Model (18) using both ideal and general data sets combined.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-19.024828	0.203672	-93.41	0.000000
Blocksize	0.060018	0.003364	17.84	0.000000
log(Dimension)	4.823624	0.038044	126.8	0.000000
$\frac{\text{Index}}{\text{Dimension}}$	-0.221438	0.037511	-5.903	0.000000

Table 18: Estimation of Model (19) using only data from ideal lattices.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-4.891670	0.077640	-63	0.000000
Blocksize	0.109753	0.004103	26.75	0.000000
Dimension	0.059656	0.000518	115.1	0.000000

Table 19: Estimation of Model (20) using both ideal and general data sets combined.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	-20.054940	0.132641	-151.2	0.000000
Blocksize	0.068648	0.002959	23.2	0.000000
log(Dimension)	4.961946	0.029794	166.5	0.000000

Table 20: Estimation of Model (21) using both ideal and general data sets combined.