LLL via the Lenstra graph

Jinsu Kim, Miran Kim, Seungki Kim

Abstract. [1] In this paper, we interpret the LLL algorithm as one of the "random" walks on a certain graph introduced by H. K. Lenstra [Len01], and present a few experiments aimed at understanding LLL in such context. Thereby we answer several folklore questions on the behavior of LLL, and make some progress towards explaining the well-known gap between the worst-case and average-case output qualities of LLL. Moreover, we point out a related security issue that is imminent yet neglected, and explain how our careful study of the Lenstra graph can lead to its resolution.

Keywords. Lattices, basis reduction, the LLL algorithm, security estimates.

1 Introduction

Motivating question. Since its advent in 1982, the practitioners of the LLL algorithm [LLL82] have been well aware that it often — in fact, most of the time — outputs a much shorter vector in a much shorter time than it guarantees. This continues to pique the curiosity of many to this day, yet surprisingly it has hardly ever been investigated before. To our best knowledge, the first written work on this phenomenon is that of Nguyen and Stehlé [NS06], where its authors experimented on the performance of LLL in practice; for example, they find that the average root Hermite factor (RHF) of the (0.999,0.501)-LLL reduction is approximately 1.02, as opposed to the theoretical bound ≈ 1.074. [GN08] confirms this finding, but not much else has been written on this topic ever since.

In [Ste10], Stehlé raises the question of whether this gap reflects the property of the LLL bases, or that of the algorithm:

> One may wonder if the geometry of "average" LLL-reduced bases is due to the fact that most LLL-reduced bases are indeed of this shape, or if the LLL algorithm biases the distribution.

This question has recently been answered, to some extent. Kim and Venkatesh [KV16] provides theoretical evidence that the answer is the latter, provided one is willing to accept that the LLL reduction and the Siegel reduction — in which the Lovász condition is replaced by a simpler inequality — should not behave

[1] Part of this work overlaps with the thesis of S. K.; it has not been submitted to a peer-reviewed journal or conference.

too differently. It proves that, in all sufficiently high dimensions, almost every Siegel basis of almost any lattice has the length of its shortest vector extremely close to the theoretical worst bound. Thus, if LLL consistently yields much shorter vectors, the algorithm must be doing something extraordinary to make it happen.

A security issue. This over-performance of LLL is more than a subject of idle curiosity. It points to a serious and difficult problem concerning the security of lattice-based cryptosystems. Recall that Lenstra [Len01] has interpreted the LLL algorithm as a direction for walking on a certain oriented graph. There is no reason to assume that one could not attain an arbitrarily better RHF by giving some other contrived direction, especially if one already has 1.02 by chance. If 1.005 is attainable, say, it would obviously be a disaster to the field. Of course, this would sound absurd to many well-experienced cryptologists, but the point is that we need to develop a scientific argument, rather than some collective intuition, that there exists a uniform lower bound on the RHF of any LLL-like algorithm.

The same discussion could be extended to any other reduction algorithm which operates iteratively on blocks of the basis matrix e.g. BKZ (reference), of which LLL serves as the simplest case. We must remove the possibility of a vast improvement in either output quality or time complexity simply by tweaking the algorithm a little bit, i.e. we must establish that the strength of the local operation on blocks imposes an inevitable *lower* bound on the quality and cost. However, to the best of the authors' knowledge, all that has been done under the name of "security estimates" so far is the analyses of the popular variants of BKZ under certain contexts. The problem with this approach is that there could always be possibilities for new variants and contexts.

It would been much less of concern if the schemes were based on the shortest or closest vector problem, whose hardness is relatively well-established (references). However, for practical reasons — e.g. key sizes — most of the recently proposed schemes are based on substantially easier problems. Hence a relevant statement in defense of lattice-based cryptography should be of the form, "Any algorithm that achieves RHF $< x$ must have time complexity of at least y." Yet there has been so far no work in the literature recognizing, much less addressing, this critical point — our close-to-zero understanding of the properties of even the most fundamental device such as LLL is an indication of this.

This paper. We present the first in-depth study of the behavior of LLL, and its variants. As suggested above, we have two motivations in mind:

(i) explain the average RHF ≈ 1.02 of LLL,
(ii) establish that no LLL-like algorithm — an algorithm that operates by walking on Lenstra's graph — can achieve an RHF that is better than ≈ 1.02.

Our approach goes by establishing the following

Conjecture 1. Various statistical properties of LLL-like algorithms — e.g. RHF, complexity — are more or less determined by the structure of the underlying Lenstra's graph. They vary only marginally across the family of LLL-like algorithms.

Of course, this is an extremely ambitious goal that cannot be expected to be achieved in a single paper, even numerically. This paper presents some of our early experiments that inspires us to conceive of Conjecture 1, and gives evidence toward it. We focus on two variants of LLL which we call *random* and *potential* (see Section 3), and on the case of low dimensions, in which the size of the data is small enough to manage with limited computing resources. We demonstrate that the original, random, and potential LLL have about the same output quality and complexity (Section 3), and furthermore, various aspects of their behavior on the Lenstra graph are strikingly similar (Section 4). Experiments under more general conditions will be carried out in a series of forthcoming works.

Let us explain how Conjecture 1 relates to the problems mentioned above. It reduces (i) to the study of the distribution of LLL bases on the Lenstra graph, which makes it much more approachable. Indeed, we construct a plausible hypothesis (reference) in Section 5 that will explain why LLL yields better outputs than theoretically expected, which we will verify in a sequel to this paper.

As for (ii), it is resolved immediately by Conjecture 1, since *a fortiori* the number 1.02 comes from the graph, not from the particularities of the algorithm. In another future work, we plan to address the above-mentioned security issue by associating the average RHF of a reduction algorithm to the cost of its "local operation," e.g. the local operation of BKZ-β would be the enumeration technique in dimension β. The case of LLL serves as the simplest case for building and testing this framework.

Implications on security. Researchers often quote the numerical estimate 1.02 of [NS06] for estimating the security of their schemes against polynomial-time attacks. However, as we pointed out earlier, this only serves as an upper bound on the performance of LLL-like algorithms. Our work in this paper provides evidence that 1.02 is also a *lower bound*, which not only makes it more useful but is also important to establish, as stressed above.

It has been conjectured (references) that the output quality of a reduction algorithm is independent of the choice of the input basis (unless it has a highly unusual distribution). Its implication to cryptography is that the choice of the public key requires hardly any deliberation. In the case of LLL, our proposed explanation for the 1.02 RHF of LLL explains this conjecture as well — see the discussion in Section 5.

Other contributions. We also make a number of perhaps of lesser scale, but more solid, contributions to the understanding of LLL. The first and foremost is the exact formula for the average number of LLL bases of a random lattice (Theorem 1). This is a vast improvement from the previous folklore estimate $O(2^{n^3})$. Theorem 1 ensures the statistical credibility of our experiments, and also enables us to make several interesting findings.

We also provide in Appendix satisfactory explanations for the two special cases noted by Stehlé [Ste10] in which LLL works particularly well: i) when dimension is ≤ 6, ii) when two successive minima λ_i and λ_{i+1} differ greatly. The former is immediately explained by Theorem 1, the formula for the number of LLL bases. For the latter, the idea is that, if $\lambda_i \ll \lambda_{i+1}$, the lattice under question is almost a direct sum of two lattices of dimensions i and $n-i$, so that the bound for the i-dimensional LLL applies.

Our work also presents the first investigation of the variants of LLL in the spirit of [NS06] — especially, see Section 3. The random and potential variants are well-known and much discussed in informal contexts, but they have never been given a serious treatment before. We also make our source codes for them available on our Github site.[2]

Kim and Venkatesh [KV16] suggests that LLL is "biased," in sufficiently high dimensions. Their work does not make it clear at which dimension this phenomenon starts to occur. In this paper, Section 4, we present conspicuous numerical evidence that LLL is biased, even in dimensions as low as 10. Theorem 1, combined with our empirical observations, indicate that the bias is quite severe: in tens of thousands of trials, the majority of the LLL bases never show up.

Furthermore, the frequency that a given LLL basis appears as an output of LLL is correlated to a quantity associated to the basis that we call *energy* — see Figures 2 and 4, for example. The slope of the correlation appears to be more or less independent of the underlying lattice.

Organization. In Section 2, we recall all the basic facts used here about lattices and the LLL algorithm. In particular, we present an exposition on the bias of LLL. In Sections 3 and 4, we describe our experiments, and analyze their results. All our source codes for the experiments and the raw data will be made available at our Github page. In Section 5, we return to the questions presented here and give more detailed discussions, drawing from the numerical results obtained in the previous sections. In Appendix, we provide mathematical explanations for the two special cases in which LLL is especially effective.

2 Background

A primer on lattices. A *lattice* in $\mathbb{R}^n$ is a $\mathbb{Z}$-span of a set of linearly independent vectors in $\mathbb{R}^n$. Throughout this paper, we assume that all lattices have *full rank*, that is, we only consider lattices in $\mathbb{R}^n$ spanned by n linearly independent vectors. Any set of independent vectors that span a lattice is called a *basis* of that lattice. For technical reasons that have to do with the LLL algorithm, we take a basis to be an *ordered set*, so that, for example, $\{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$ and $\{\mathbf{b}_2, \mathbf{b}_1, \mathbf{b}_3\}$ are different bases.

[2] We do not link it in this draft as we are submitting anonymously.

Let $\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ be a basis of some lattice in $\mathbb{R}^n$. It is sometimes convenient to write this basis into a matrix form:

$$\begin{pmatrix} - \; \mathbf{b}_1 \; - \\ \vdots \\ - \; \mathbf{b}_n \; - \end{pmatrix}. \tag{1}$$

Multiplying this matrix from the left by any element of $\mathrm{GL}(n, \mathbb{Z})$, we get another basis of the same lattice; conversely, any two bases of the same lattice differ by a left multiplication by some element of $\mathrm{GL}(n, \mathbb{Z})$.

The *covolume* or *determinant* of a lattice is the determinant of any basis put into the matrix form (1); it is the same as the volume of the fundamental parallelepiped formed by the basis vectors. Multiplying by a suitable scalar matrix, we can normalize (1) to be an element of $\mathrm{SL}^{\pm}(n, \mathbb{R})$, the set of all matrices with determinant ± 1.

Random lattice. From the above discussion, we see that $\mathrm{GL}(n, \mathbb{Z}) \backslash \mathrm{SL}^{\pm}(n, \mathbb{R})$ is the set of all (normalized) lattices. It is convenient to identify this space with $\mathrm{SL}(n, \mathbb{Z}) \backslash \mathrm{SL}(n, \mathbb{R})$, which is what we obtain if we ignore the issue of the orientation (which has to do with the sign of the determinant).

Denote this space by X_n. The celebrated theorem of Siegel [Sie45] asserts that X_n has a probability measure μ_n that is invariant under the right multiplication by $\mathrm{SL}(n, \mathbb{R})$. This μ_n provides the standard notion of *random lattice*.

The well-known *Hecke equidistribution* by Goldstein and Meyer [GM03] is a statement that, for p a large prime, and $x_1, \dots, x_{n-1} \in [0, p-1]$ uniformly chosen integers, the distribution of the lattices of form

$$\begin{pmatrix} p & & & & \\ x_1 & 1 & & & \\ x_2 & & 1 & & \\ \vdots & & & \ddots & \\ x_{n-1} & & & & 1 \end{pmatrix}, \tag{2}$$

(after a suitable normalization) approaches the distribution dictated by μ_n as $p \to \infty$. This is frequently used in computer experiments to sample lattices at random. Theoretically, p must be of size about 2^{n^2} to guarantee that the distribution is really alike that of μ_n.

"Random" bases. In order to carry out experiments on LLL, often it becomes necessary to generate multiple bases of a given lattice. The problem is that there is no satisfactory mathematical notion of a "random basis"; in other words, there exists no canonical probability measure that can be defined on the space of the bases. We refer the readers to [NS06] for an excellent discussion on this point, and several attempts that have been tried to get around this difficulty. Here, we will satisfy ourselves with describing their method for generating bases of a lattice given by the form (2), which is what we adopted for our experiments.

Start with a lattice L, given by a basis of form (2), which we denote by B. Sample uniformly n vectors from $L \cap [-p/2, p/2]^n$, and denote by M the matrix whose rows consist of these sampled vectors. (To sample a vector from $L \cap [-p/2, p/2]^n$, multiply B from the left by a row vector $(y_1, \dots, y_n)$, where $y_2, \dots, y_n$ are chosen uniformly from $[-p/2, p/2]$, and then determine y_1 so that the first entry falls into $[-p/2, p/2]$. This works thanks to the special shape of (2).) It turns out that, with overwhelming probability, $\det M \neq 0$. Assuming this, we can write $M = QB$ for some integral matrix Q of nonzero determinant. Write $Q = HU$, where H is the upper Hermite normal form of Q, and $U \in \mathrm{GL}(n, \mathbb{Z})$. Finally, $B' := UB$ is the resulting "random basis." In experiments, one observes that all entries of B' have about the same number of digits.

LLL bases. Let $\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ be any basis of $\mathbb{R}^n$. Define $\mathbf{b}_i^*$ to be the component of $\mathbf{b}_i$ that is orthogonal to $\mathrm{span}(\mathbf{b}_1, \dots, \mathbf{b}_{i-1})$, and for $i > j$ define $\mu_{i,j} = \langle \mathbf{b}_i, \mathbf{b}_j^* \rangle / \|\mathbf{b}_j^*\|^2$. These are related by the equality

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

Let us call the $\|\mathbf{b}_i^*\|$'s and $\mu_{i,j}$'s the *Gram-Schmidt coefficients* of the basis. The well-known *Gram-Schmidt process* is the algorithm that computes the Gram-Schmidt coefficients of a given basis.

Choose $1/4 < \delta \leq 1$ and $1/2 \leq \eta < \sqrt{\delta}$. We say $\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ is a (δ, η)-*LLL reduced basis* if it satisfies

(i) $|\mu_{i,j}| \leq \eta$ for all $i > j$ (i.e. $\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ is size-reduced), and
(ii) $\delta \|\mathbf{b}_i^*\|^2 \leq \|\mathbf{b}_{i+1}^* + \mu_{i+1,i} \mathbf{b}_i^*\|^2$ for all $i = 1, \dots, n-1$ (Lovász condition).

This definition does not *a priori* refer to any lattice. In case an LLL basis spans a lattice L, we call it an LLL basis of L. Any lattice has a finite number of LLL bases; see [LLL82] for a proof.

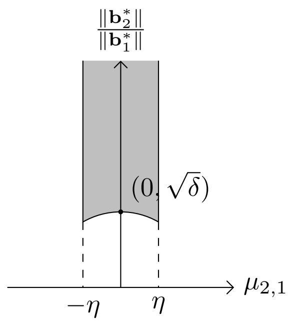

Fig. 1. The set of LLL bases in dimension 2.

Figure 1 shows a way of visualizing the set of LLL bases in dimension 2. The entire coordinate plane is essentially the set of all normalized bases of $\mathbb{R}^2$ (modulo rotation and determinant, to be precise), and the shaded region is the set of (δ, η)-reduced LLL bases.

Writing $x = \mu_{2,1}$ and $y = \|\mathbf{b}_2^*\|/\|\mathbf{b}_1^*\|$, we have the formula $\mu_2 = \frac{3}{\pi}\frac{dxdy}{y^2}$ — for a proof, see Chapter 7 of Serre [Ser73]. An important point to note is that, from this expression of μ_2, we can tell that its density gets more concentrated as the vertical coordinate in Figure 1, or the y-coordinate, approaches zero.

In dimension n, one can visualize the set of LLL bases by drawing $n-1$ copies of Figure 1, each copy with coordinates $\mu_{i,i-1}$ and $\|\mathbf{b}_i^*\|/\|\mathbf{b}_{i-1}^*\|$. Some coordinates will be missing, but nothing interesting seems to occur along those directions anyway. There also exists a general explicit formula for μ_n — see, e.g., [KV16] — from which one observes that the greater n is, the more is μ_n concentrated towards the bottom. In particular, the vast majority of the mass of the shaded region is located near the cusps. With a simple computation, one finds that the LLL bases located at the cusps are also those with the longest $\|\mathbf{b}_1\|$, whose length coincides with the well-known worst-case bound for $\|\mathbf{b}_1\|$ of an LLL basis. Thus, at least heuristically, almost all LLL bases have $\|\mathbf{b}_1\| \approx$ (worst bound).

LLL algorithm. The LLL algorithm is, in some sense, the most straightforward procedure of translating a given basis into the set of LLL bases. Below is a very rough pseudocode of the algorithm, assuming $\eta = 0.5$. The notation $\lceil x \rfloor$ in Line 3 means the closest integer to x.

Algorithm 1 The LLL algorithm

```
0. Input: a basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ of $\mathbb{R}^n$, often integer-valued
1. $k := 1$
2. while $k < n$, do:
3.     (size reduction) for $i = k$ down to 1: $\mathbf{b}_{k+1} := \mathbf{b}_{k+1} - \lceil \mu_{k+1,i} \rfloor \mathbf{b}_i$
4.     (Lovász test) if $\delta\|\mathbf{b}_k^*\|^2 \leq \|\mathbf{b}_{k+1}^* + \mu_{k+1,k}\mathbf{b}_k^*\|^2$ then $k := k+1$
5.     else swap $\mathbf{b}_k$ and $\mathbf{b}_{k+1}$, and $k := \max(k-1, 2)$
6. Output $\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$, a $\delta$-reduced LLL basis.
```

This description is meant to depict only the skeleton of the code, and ignores practical issues that arise while actually implementing it. For a discussion on such issues, we direct readers to [NS05]. But let us point out just one thing that is relevant for our experiments, which does not seem to have been mentioned elsewhere. Upon following the (floating-point) LLL algorithm step by step, one finds that the Gram-Schmidt process often computes $\|\mathbf{b}_n^*\|$ wrong (not infrequently, even negative) but somehow the algorithm successfully yields an LLL basis anyway. This happens probably because if $\|\mathbf{b}_n^*\|$ is very small it has very little impact on the Lovász condition — if it has to be swapped it will be swapped.

Still, it may cause the algorithm, especially our variants, to walk on the Lenstra graph differently than intended; as we prioritize accurately observing LLL over efficiently implementing it, in our experiments we use a higher precision than needed for a valid output.

Statistics of LLL bases, and the bias of LLL. The simplicity of the shape of the set of LLL bases, as shown in Figure 1, and the known explicit formula for μ_n allow us to derive some statistical results about LLL bases. For example,

Theorem 1. *Recall that each lattice has a finite number of LLL bases. The average number of (δ, η)-LLL bases of a random lattice in dimension n equals*

$$2 \cdot \prod_{j=2}^{n} \frac{S_j}{\zeta(j)} \cdot \frac{1}{n} \prod_{i=1}^{n-1} \frac{1}{i(n-i)} \int_{-\eta}^{\eta} \sqrt{\delta - x^2}^{-i(n-i)} dx,$$

where S_j denotes the volume of the unit sphere in $\mathbb{R}^j$. When $\delta \approx 1$ and $\eta \approx 1/2$, this is asymptotically of size $o((4/3)^{\frac{1}{12}(n^3-n)})$.[3]

Proof (sketch). Write $\alpha_k = \|\mathbf{b}_{k+1}^*\|/\|\mathbf{b}_k^*\|$ There exists a standard formula for integrating *Siegel sets*, i.e. subsets of $\mathrm{SL}(n, \mathbb{R})$ satisfying $\mu_{i,j} \in [c_{i,j}, c'_{i,j}], \alpha_k > a_i$, for some $c_{i,j}, c'_{i,j} \in \mathbb{R}$, $a_k \in \mathbb{R}_{>0}$, $1 \leq j < i \leq n$, $1 \leq k < n$ — see e.g. [KV16]. The set of LLL bases is not a Siegel set, but can be approximated by arbitrarily many Siegel sets. The formula now follows from a standard Riemann sum argument.

By the same technique, we can also estimate the rate of concentration of LLL bases at the "cusps," which proves rigorously that $\|\mathbf{b}_1\| \approx$ (worst bound) for almost all LLL bases. For example, in dimension 180, at least 99 percent of LLL bases are contained in the region $\alpha_i < 0.89$ and $|\mu_{i+1,i}| > 0.45$.

From now on, let us call an LLL basis *bad* if $\|\mathbf{b}_1\| \approx$ (worst bound), and *good* if $\|\mathbf{b}_1\|$ is much shorter. Thus the previous statement translates to: almost all LLL bases are bad.

Still, there may be a spectrum of possibilities regarding the distribution of the bad bases among lattices. Two extreme possibilities are: i) a miniscule percentage of lattices have a huge number of LLL bases most of which are bad, and the majority of lattices have much fewer LLL bases but mostly of good quality ii) almost all lattices have about the same number of LLL bases, most of which are bad. If the former were the case, the algorithm need not be biased in order to demonstrate the folklore observation of over-performance. But if the latter were true, we are forced to conclude in the opposite.

It is possible to determine between the two possibilities, if we relax the Lovász condition to $\delta\|\mathbf{b}_i^*\|^2 < \|\mathbf{b}_{i+1}^*\|^2$, and fix $\eta = 0.5$. The resulting counterpart of the LLL bases is called the *Siegel bases*. On Figure 1, this has the effect of replacing the arcs with horizontal lines connecting the cusps. The reduction to a Siegel basis, using the same LLL algorithm with only the obvious modification, consistently yields a better outcome than the theoretical bound as well, although

[3] It is interesting to compare this with Proposition 8.3 of [Len01].

the output quality is somewhat worse than the original LLL. The main theorem of Kim and Venkatesh [KV16] states that almost all lattices have about the same number of Siegel bases, for all sufficiently large n:

Theorem 2 (Kim and Venkatesh [KV16]). [4] *There exists absolute constants $A, d > 0$ such that the standard deviation of the number of Siegel bases of an n-dimensional random lattice is at most Ae^{-dn^2} times its mean.*

As discussed earlier, Theorem 2 provides compelling evidence that the Siegel version of the LLL algorithm is biased i.e. given a lattice, it outputs some of its LLL bases more often than others. We expect the same statement to hold for the actual LLL bases, although proving it would be a much more involved task.

Lenstra graph. Let $L \subseteq \mathbb{R}^n$ be a lattice. Lenstra [Len01] defines an oriented graph attached to L, which we will call the *Lenstra graph* of L, as follows: the vertices consist of the set of all size-reduced bases (i.e. $|\mu_{i,j}| \leq 1/2$) of L, and an edge of color i is formed from $\mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ to $\mathcal{B}' = \{\mathbf{b}'_1, \ldots, \mathbf{b}'_n\}$ if and only if i) $\mathbf{b}_i$ and $\mathbf{b}_{i+1}$ fail the Lovász condition and ii) swapping $\mathbf{b}_i$ and $\mathbf{b}_{i+1}$ and then size-reducing $\mathcal{B}$, one obtains $\mathcal{B}'$. It is clear from this definition that $\mathcal{B}$ is an LLL basis if and only if there exists no edge starting at $\mathcal{B}$, and that the LLL algorithm is the random walk on the graph that always walks on the edge of the lowest color. Of course, all this depends on the implicit parameter δ, and as for η we simply fix it to $1/2$.

There are two important invariants attached to the vertices and the edges of the Lenstra graph, respectively. One is the *energy* of a basis $\mathcal{B}$, defined by

$$\text{energy}(\mathcal{B}) = \prod_{i=1}^{n} \|\mathbf{b}_1 \wedge \ldots \wedge \mathbf{b}_i\| = \prod_{i=1}^{n} \|\mathbf{b}_i^*\|^{n-i+1},$$

where $\|\mathbf{b}_1 \wedge \ldots \wedge \mathbf{b}_i\|$ refers to the volume of the fundamental parallelogram made of $\mathbf{b}_1, \ldots, \mathbf{b}_i$. Another is the *potential* of the edge $\mathcal{B} \to \mathcal{B}'$, defined by

$$\text{potential}(\mathcal{B} \to \mathcal{B}') = \text{energy}(\mathcal{B}')/\text{energy}(\mathcal{B}).$$

If $\mathcal{B} \to \mathcal{B}'$ is of color i, we have

$$\text{potential}(\mathcal{B} \to \mathcal{B}') = \frac{\|\mathbf{b}_{i+1}^* + \mu_{i+1,i}\mathbf{b}_i^*\|}{\|\mathbf{b}_i^*\|} < \sqrt{\delta}.$$

Therefore, if $\delta < 1$, each time one passes through a path the energy decreases by at least a factor of $\sqrt{\delta}$. On the other hand, the energy of a basis of L is bound from below by a quantity depending on L e.g. if L is given by integral vectors, one can tell from the definition above that $\text{energy}(\mathcal{B}) \geq 1$ for any basis $\mathcal{B}$ of L. This implies immediately that it takes $O(n^2 \log \max_{1 \leq i \leq n} \|\mathbf{b}_i\|)$ swaps to reduce a basis $\mathcal{B}$, and thus LLL has polynomial-time complexity.

[4] The proof of the theorem as it appears in [KV16] assumes the Riemann hypothesis. But the authors claim that it could be removed with extra work.

Theorem 2 shows that almost all Lenstra graphs have about the same number of LLL bases, and of size-reduced bases whose $\|\mathbf{b}_i^*\|$'s fall inside a given (small) interval. This leads us to suspect that almost all Lenstra graphs have very similar shapes, as dimension goes to infinity. (Mention the widespread conjecture that reduction is independent of lattices.)

A comment on terminology: In [Len01], the terms energy and potential were called *size* and *length*, respectively. In the literature of cryptography, energy is often referred to as *potential.* Our motivation for calling it energy comes from a work of Cohn et al. [BBC$^+$09] where they experiment on energy-minimizing point configurations on n-spheres — especially, see Section 3.9 of [BBC$^+$09]. And "potential" is admittedly an inelegant naming by us intended to simply mean change in energy.

3 Experiment 1: the random and the potential variants

We begin this section by formally introducing the random and the potential variants of LLL. We do not claim that they are our original inventions; neither are hard to come up with after all, and they are in fact frequently brought up in informal discussions. There has been much curiosity as to how their performances compare to that of the original LLL, but to our best knowledge, no written work on either of them exists as of yet. We take up on this task in the present section, in the spirit of Nguyen and Stehlé [NS06].

The notion of the Lenstra graph adds to the significance of considering the random and potential variants. First, as will be explained below, they are in certain senses better suited for exploring the Lenstra graph than the original LLL. Second, comparing the behavior of different random walks on the Lenstra graph is essential for determining whether the graph's inherent structure or the particularities of the random-walking procedure plays a major role in shaping the bias of LLL.

We did not intend to write time-efficient codes for the variants. In addition, as discussed in the LLL algorithm subsection of Section 2, we had to set the precision higher than needed for the correct output, in order to ensure that potentials are computed accurately so that the algorithms move along the Lenstra graph in the intended manner. Still, our implementations are noticeably faster than LLL-RR of NTL [Sho]. At any rate, for our purposes in this paper, the number of swaps, or equivalently the length of the path taken by the algorithm on the Lenstra graph, is a much more important measure of complexity than the running time, being independent of a specific implementation of that algorithm.

Random variant. Recall that the original version of the LLL algorithm walks on the Lenstra graph by always choosing the path of the lowest color. The *random variant* of the LLL algorithm, as its name suggests, proceeds by choosing the next path randomly and uniformly among the available edges. We provide its pseudocode below.

What is special about the random variant is that it transparently reflects the shape of the Lenstra graph. So if the random variant and something else exhibit

Algorithm 2 The random variant of the LLL algorithm

```
0. Input: a basis $\mathcal{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ of $\mathbb{R}^n$
1. size-reduce $\mathcal{B}$
2. while *true*, do:
3.     let $S$ be the set of all $1 \le k \le n-1$ for which $\delta\|\mathbf{b}_k^*\|^2 > \|\mathbf{b}_{k+1}^* + \mu_{k+1,k}\mathbf{b}_k^*\|^2$
4.     if $S$ is empty, break
5.     else
6.         choose $i$ randomly and uniformly from $S$
7.         swap $\mathbf{b}_i$ and $\mathbf{b}_{i+1}$
8.         size-reduce $\mathcal{B}$
9. Output $\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$, a $\delta$-reduced LLL basis.
```

the same phenomenon, it would serve as evidence that the property comes from the Lenstra graph itself, rather than some idiosyncrasy of the other random walk.

Potential variant. The *potential variant* walks on the edge of the *least* potential, seeking to decrease the energy as quickly as possible. In this sense it is a direct analogue of gradient descent. One could reasonably guess that it might run faster than the original LLL, and that is one of the reasons we consider it. Below is a pseudocode for the potential variant:

Algorithm 3 The potential variant of the LLL algorithm

```
0. Input: a basis $\mathcal{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ of $\mathbb{R}^n$
1. size-reduce $\mathcal{B}$
2. while *true*, do:
3.     $p := \delta$, $i := n$
4.     for $1 \le k \le n-1$:
5.         $p' := \frac{\|\mathbf{b}_{k+1}^* + \mu_{k+1,k}\mathbf{b}_k^*\|^2}{\|\mathbf{b}_k^*\|^2}$
6.         if $p' \le p$, $p := p'$ and $i := k$
7.     if $i = n$, break
8.     else swap $\mathbf{b}_i$ and $\mathbf{b}_{i+1}$, and size-reduce $\mathcal{B}$
9. Output $\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$, a $\delta$-reduced LLL basis.
```

A useful property of the potential variant is that it can be used to relate Lenstra graphs for different values of δ's:

Lemma 1. *Let $1/4 < \delta_1 < \delta_2 \le 1$, and choose an input (size-reduced) basis $\mathcal{B}$. The path of the potential variant on the δ_2-Lenstra graph starting at $\mathcal{B}$ is the concatenation of the path on the δ_1-Lenstra graph which starts at $\mathcal{B}$ and ends at a δ_1-LLL basis, and the path from that basis to a δ_2-LLL basis.*

Proof. Suppose that at some point of the algorithm we are at some basis $\mathcal{B}'$. If there exists at least one edge of potential $\leq \sqrt{\delta_1}$, δ_1-LLL and δ_2-LLL make the same decision as to the next step to take. If there exists no such edge, δ_1-LLL terminates at $\mathcal{B}'$. This completes the proof.

Several recent works have proposed versions of LLL that are also based on the idea of energy minimization, but they differ from our version in significant ways. For instance, PotLLL by Fontein, Schneider, and Wagner [FSW14], a variant of deep-LLL, allows an insertion of $\mathbf{b}_i$ farther than just in front of $\mathbf{b}_{i-1}$, so it does not respect the structure of the Lenstra graph. The algorithms by Schnorr [Sch11] and Neumaier [Neu16] happen to respect the structure of the Lenstra graph, but once they detect an edge of the minimal potential, they perform the Gauss reduction on the pair of the basis vectors; in other words, they keep walking on the edge of the same color until it cannot.

Design of Experiment 1. For the inputs, we generated lattices and bases in dimensions $n = 10, 20, 30, \ldots, 100$ in the manner described in Section 2, with $p = \det \approx 2^{3n}$. In dimensions 10 through 50, we generated 10 lattices per dimension, and 3 bases for each lattice; we did this to see if there exists any difference in performance across lattices. Due to computational limitations, in dimensions 60 through 100, we generated 4 lattices per dimension, and 3 bases per lattice.

On these bases, we ran each of our own implementations of the original, random, and potential versions of LLL, and recorded the root Hermite factor $(\|\mathbf{b}_1\| / \det^{\frac{1}{n}})^{\frac{1}{n}}$, running time, and number of swaps.

Result and discussions. The average performances of the three variants are presented in Table 1. RHF stands for root Hermite factor, and time is measured in seconds. One interesting point to note is that all the statistics in Table 1 have very small standard deviations — less than one percent of the mean most of the time. In other words, the numbers in Table 1 represent not only averages but what happens in each individual trial.

From Table 1, one observes that all three variants yield more or less the same output quality. This may appear anticlimactic at first, but in view of the security question posed in the introduction, it is in fact really fortunate to see that no improvement is made by varying the method of the random walk. Recall that we wish to develop the security framework that roughly sounds like: "If we want RHF < 1.02, we must use an algorithm substantially costlier than LLL." Had it happened otherwise, we would have to go through a far more difficult course to argue that LLL-like algorithms have a lower bound on RHF, as we can no longer reduce it to a property of the Lenstra graph. And if we fail to make this point, it would jeopardize the security of lattice-based cryptosystems.

In the number of swaps and running time, there is a more noticeable difference. Table 2 gives comparisons of both complexity measures for the three variants. It appears that the ratio of the number of swaps stabilizes to 9 for the original versus random variants of LLL, and to 15 for the original versus potential. This is an interesting and mysterious phenomenon whose cause we do

	original			random			potential		
dim	RHF	#swaps	time	RHF	#swaps	time	RHF	#swaps	time
10	0.9875	2276	0.071	0.9875	735	0.032	0.9875	482	0.025
20	1.0134	42793	2.442	1.0141	7028	0.889	1.0132	4352	0.652
30	1.0157	192286	16.1	1.0148	26462	6.481	1.0146	16663	4.829
40	1.0160	521,637	63.4	1.0167	65619	28.1	1.0165	40,546	20.6
50	1.0174	1,092,704	189	1.0178	124,396	88.6	1.0176	74,721	63.1
60	1.0182	1,969,105	423	1.0176	226,054	225	1.0184	137,023	168
70	1.0194	3,216,819	843	1.0176	372,479	497	1.0188	226,534	359
80	1.0183	4,901,971	1544	1.0188	575,599	998	1.0185	351,166	723
90	1.0199	7,087,008	2759	1.0190	805,417	1851	1.0192	488,205	1339
100	1.0198	9,841,849	4733	1.0198	1,091,607	3261	1.0190	659,915	2360

Table 1. Performances of the three versions of LLL.

not understand; there is no apparent reason for them not to be some randomly fluctuating quantities. The ratios of the running times also stabilize to 1.5 and 2 for the random and potential variants, respectively. The numbers are (perhaps disappointingly) smaller than their swapping number counterparts probably because those variants take a longer time to decide on which pair of the vectors to exchange.

It may be interesting to try to improve on the implementation of, say, the potential variant, so that it will become (nearly) 15 times as fast as the original LLL. For the original LLL, we adopted the sophisticated methods of NTL [Sho] and Nguyen and Stehlé [NS05], which accelerated its running time by a factor of 4. These methods rely heavily upon the sequential nature of the original LLL, and we are yet to make an exploitation of such kind for the other variants.

We are also rather lucky, from the security perspective, that the ratios of the number of swaps is constant in n. On the other hand, our results suggest a possibility that some practical improvements can still be made in complexity aspect.

4 Experiment 2: distribution of output LLL bases

The basic idea of the present experiment is: fix a lattice, run LLL on it many times, and see what happens. We carefully analyze the patterns that appear in our data, and observe that all three variants are quite similar with respect to those patterns, thereby supporting Conjecture 1. We even find that, on a close enough neighborhood of LLL bases, they act in almost the same manner.

Furthermore, we find compelling evidence that LLL severely distorts the natural distribution of LLL bases. In other words, LLL prefers to output certain bases more than others. In fact, the majority of LLL bases are almost never chosen by the algorithm. We also find that the degree of the preference by the algorithm for a given LLL basis is correlated to its energy. Again, our three variants differ marginally at best in all quantitative aspects of the bias.

dim	#swaps		time	
	$\frac{\text{(original)}}{\text{(random)}}$	$\frac{\text{(original)}}{\text{(potential)}}$	$\frac{\text{(original)}}{\text{(random)}}$	$\frac{\text{(original)}}{\text{(potential)}}$
10	3.1	4.7	2.2	2.8
20	6.1	9.8	2.7	3.7
30	7.3	11.5	2.5	3.3
40	7.9	12.9	2.3	3.1
50	8.8	14.6	2.1	3.0
60	8.7	14.4	1.9	2.5
70	8.6	14.2	1.7	2.3
80	8.5	14.0	1.5	2.1
90	8.8	14.5	1.5	2.1
100	9.0	14.9	1.5	2.0

Table 2. Ratios of complexity measures

It is regrettable that we carried out this experiment only in dimension 10; but that is about the highest feasible dimension in which it can be conducted with a reasonable amount of resource. Here we are interested in much finer patterns than RHF or complexity, which requires that we run LLL at least several hundred times the expected number of LLL bases per lattice. Yet Theorem 1 tells us that, already in dimension 14, there are around one hundred million LLL bases per lattice on average.

Design. For a lattice L of dimension 10 and det $\approx 2^{30}$ generated in the form of (2), we generated 70,000 bases in the manner described in Section 2. For each of the three variants of LLL, we do the following: $(\delta, 0.5)$-reduce all 70,000 bases, where $\delta = 0.999, 0.994, 0.99, 0.98, 0.97$, and for each value of δ record which LLL bases are "hit" how many times. In counting this number of hits, we identify bases up to the sign of each vector e.g. $\{\pm\mathbf{b}_1, \pm\mathbf{b}_2, \pm\mathbf{b}_3\}$ is regarded as one basis, not eight.[5] We repeated this procedure on 29 different lattices.

The reason for the number 70,000 is to at least informally guarantee statistical credibility; according to Theorem 1, there are on average about 35 $(0.999, 0.5)$-LLL bases per lattice, and 646 $(0.97, 0.5)$-LLL bases per lattice.

Result and discussions. A sample data for a lattice, with respect to the potential variant, is shown in Table 3, sorted in ascending order of energy, and numbered for convenient reference. Let us call such a data table the *frequency table* of the lattice, and each number in an entry the *frequency* or *the number of hits* e.g. the frequency of Basis 9 for $\delta = 0.97$ is 1937. An empty entry indicates that the corresponding basis is not an LLL basis for that value of δ.

One may wonder whether the ratios of the frequencies in a frequency table, e.g. Table 3, stabilizes as we increase the number of trials. It does, and it does pretty fast; if we reduce the number of input bases to 700 instead of 70,000, the

[5] That LLL treats equally the bases that differ only by signs can be justified both rigorously and experimentally.

figures on the resulting frequency table would be 100 times as small as those in Table 3. Hence, one may think of the frequency table as representing some sort of a distribution, which we refer to as the *frequency distribution* or sometimes the *output distribution*.

Bases	$\delta = 0.999$	$\delta = 0.994$	$\delta = 0.99$	$\delta = 0.98$	$\delta = 0.97$	$\|\mathbf{b}_1\|$	log(energy)
1	22422	9760	9760	9760	9760	7.348	163.340
2		12662	12662	12662	12662	7.348	163.342
3	10915	10915	10915	10915	10277	7.348	163.369
4	6832	6832	6832	6832	6832	7.348	163.413
5	7836	7836	7836	7836	7836	7.348	163.452
6	4534	4111	4111	4111	2892	7.348	163.493
7	3710	3710	3710	3710	3710	7.348	163.500
8					1219	7.348	163.511
9	9681	5483	5483	5483	1937	7.348	163.521
10	4070	2872	2872	2872	2872	7.348	163.534
11		1198	1198	1198	1198	7.348	163.536
12					1817	7.348	163.537
13					1146	7.348	163.539
14		3253	3253	3253	2192	7.348	163.541
15					583	7.348	163.555
16					1061	7.348	163.557
17					638	8.426	163.827
18		423	423	423	423	8.426	164.068
19		945	945	580	220	8.367	164.196
20					188	8.367	164.212
21					116	8.367	164.216
22				365	253	8.426	164.226
23					56	8.367	164.232
24					112	8.426	164.242

Table 3. Frequency table for a lattice generated with seed = 36365, potential variant.

Some more general comments are in order before we extract patterns from Table 3. First, the example we present here is representative of the typical situation, regardless of the lattices or the variants. In other words, even if we displayed a frequency table for a different lattice and/or variant, we would have made the same observations as we have below — skeptical readers may examine the raw data at our github page. However, those observations are meant to point out tendencies, not absolute facts. For example, it is true that bases with smaller energies normally have smaller $\|\mathbf{b}_1\|$'s, but one already finds some exceptions in Table 3. The lattice we chose to present here actually exhibits more idiosyncrasies than most others, but the patterns we are about to point out are so pervasive and conspicuous that we can still make our points with it.

Now onto the analysis. What is immediately noticeable from Table 3 is that the distribution of the frequencies of the output LLL bases is anything but uniform — this is what we meant by the phrase "LLL is biased." Considering that a 10-dimensional random lattice has on average 41 $(0.999, 0.501)$-reduced bases, and that we have tested on 70,000 input bases, it is unlikely that this is a coincidence. Furthermore, observe that we have found only 8 0.999-reduced bases, which is much less than the average 35. This suggests that there may exist many more 0.999-bases that the algorithm missed. Indeed, see Table 4, where we recorded the average of the number of LLL bases over all lattices we experimented on, and compared it with the true average computed with Theorem 1. It is apparent that a sizable portion of the LLL bases have never been visited by the algorithm, despite the relatively huge number of iterations. The presence of these *dark LLL bases* could also be inferred from Theorem 2 and the observed size of the average root Hermite factor; they also predict that the ratio of the dark bases will tend to 1 as dimension goes to infinity.

Also readily visible from Table 3 is that the number of "hits" of an LLL basis is correlated to its energy. Figure 2 clarifies this correlation by plotting the 0.97-LLL bases from Table 3; each point represents an LLL basis, whose x-coordinate is its log(energy) and y-coordinate is log(frequency). Figure 2 and our data for other lattices — see also Figure 4, which shows a clearer picture — suggest that log(frequency) is a decreasing affine linear function of log(energy). To be more precise, the plots often form a triangular shape as in Figure 4: popular bases tend to have low energy, but not necessarily the other way around — but in any case, the pattern is clear and consistent.

The slope of the correlation depends on the choice of a lattice, but some relevant statistics are heavily concentrated: for most lattices, the maximum of log(frequency) lies in 13 ± 1, and log(energy) of all its LLL bases are contained in $[160, 165]$. In higher dimensions, we expect the slope to be overwhelmingly dependent on dimension only, as various geometric properties of high-dimensional lattices tend to have small variances — *cf.* [Ajt02]. Unfortunately, it is infeasible to check this numerically, due to the overwhelmingly huge number of LLL bases in those dimensions.

Table 3 suggests that energy of an output basis is also correlated with its $\|\mathbf{b}_1\|$ as well. In particular, LLL is biased in terms of energy too — in fact we can directly compute the average energy and observe that it differs from practice, exactly as we have worked with $\|\mathbf{b}_1\|$. In dimensions as low as 10, one may complain that there are not enough variations of $\|\mathbf{b}_1\|$ to decide on the existence of a correlation. However, it can be quickly verified numerically in any higher dimension. Moreover, it seems to us that the proportionality constant may be explained theoretically; this will be done in a forthcoming paper.

There is still another interesting phenomenon we can find in Table 3: observe that the frequencies of Basis 1 and Basis 2 for $\delta = 0.994$ add up exactly to the frequency of Basis 1 for $\delta = 0.999$ $(9760+12662 = 22422)$. It happens everywhere on the table e.g. Bases 9, 14, 19 — the reader may enjoy finding other relations of this form. Indeed, that the numbers must add up like this is a simple corollary

δ	0.999	0.994	0.99	0.98	0.97
theory	35	58	86	234	646
experiment	22	30	37	74	133

Table 4. A comparison of the theoretical and experimental averages of the number of LLL bases. The experimental averages here are found with the potential variant, but they are almost the same for other variants.

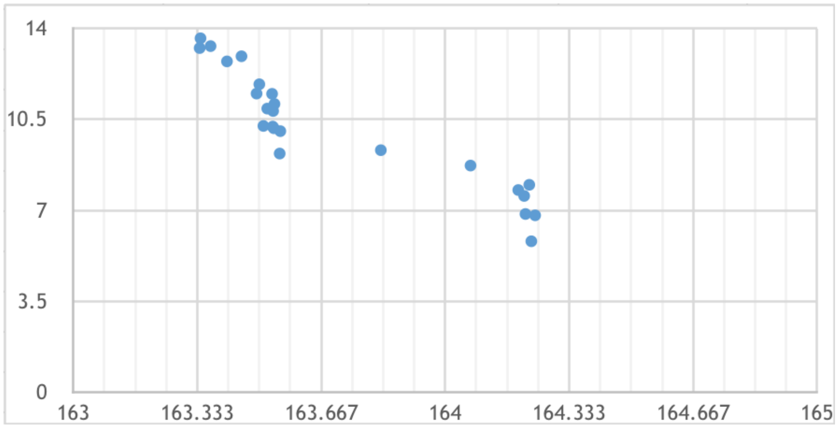

Fig. 2. A plot of the 0.97-LLL bases in Table 3. The horizontal and vertical axes represent log (energy) and log(frequency), respectively.

of Lemma 1; also note that the basis with the lowest energy must be the one that is "absorbing." This allows us to take a peek at some of the dynamics that is happening on the Lenstra graph this way: for example, we can see that 12662 of the paths that eventually reached Basis 1 passed by Basis 2.

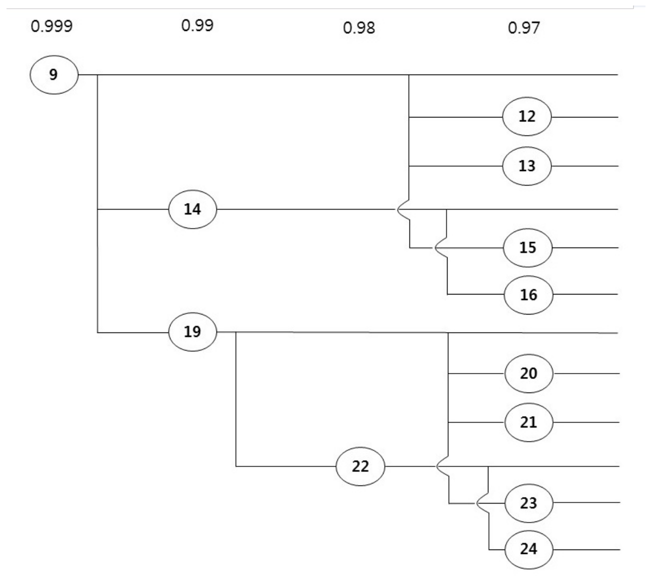

Fig. 3. Part of the boundary structure for seed = 36365 involving Basis 9.

Collecting all these summation relations, we can draw some portion of the Lenstra graph, which is partially done in Figure 3. One may think of Figure 3 as showing (a part of) the *boundary structure* of the Lenstra graph. The motivation for this terminology is that it shows how the bases at the "boundary" of Table 3 are connected together. Figure 3 reveals all the paths that the algorithm have taken in our experiment. For instance, a particular run could pass by Basis 24, then 22, 19, and finally arrive at 9; but a run from Basis 24 directly to Basis 9 has never occurred, and is impossible (because the potential variant is deterministic).

So far, we have discussed what we can find from a single table. In summary:

- Fixing a lattice L, LLL outputs different LLL bases of L with different frequencies. log(frequency) is a decreasing linear function of log(energy), where its slope may depend on L but not too much.
- Energy is also correlated to $\|\mathbf{b}_1\|$.
- (For the potential variant, for now) Through the boundary structure, we can observe how the algorithm moves on the Lenstra graph.

A natural follow-up question is how the data from the same lattice but different variants compare. To this end, we present in Table 5 parts of the frequency table for the same lattice as in Table 3 with respect to the original and random LLL.

Bases	$\delta = 0.999$	$\delta = 0.994$	$\delta = 0.99$	Bases	$\delta = 0.999$	$\delta = 0.994$	$\delta = 0.99$
1	13569	6740	4844	1	15124	6619	5816
2		6819	8698	2		8663	9225
3	10605	12820	13319	3	12374	12841	13039
4	4445	4405	4414	4	6760	6848	6812
5	6139	6148	6233	5	7090	7028	6879
6	2447	2230	2238	6	4451	3704	3879
7	6140	6138	6135	7	5267	5346	5238
⋮				⋮			
9	21492	12831	12120	9	14369	9261	9340
10	5163	3803	3404	10	4565	3091	3085
11		1372	1497	11		1216	1336
⋮				⋮			
14		4433	5066	14		3965	3924
⋮				⋮			
18		212	425	18		396	453
19		2049	1607	19		1022	974
⋮				⋮			

Table 5. Part of the frequency table for seed = 36365, original and random.

The numbers appearing in Tables 3 and 5 are not exactly identical; especially standing out is Basis 9, which is especially favored by the original LLL. Still, it is not to say that they are totally random. For every frequency data we have obtained in this experiment, log(frequency) of a basis varies at most by ± 1, often much less, across the variants. In particular, all three variants discover nearly the same set of bases, and an LLL basis "popular" to one variant is never obscure to any other variant, and vice versa.

Furthermore, all three variants observe more or less the same boundary structure. For example, in Table 5, just as in Table 3, the frequencies of Bases 1 and 2 with $\delta = 0.994$ add up to nearly that of Basis 1 with $\delta = 0.999(6740 + 6819 = 13559 \approx 13569$, $6619 + 8663 = 15282 \approx 15124)$; the reader may enjoy verifying the relation for the rest of the boundary structure.

Varying δ opens and closes some paths, which does not affect the potential variant but does for the others. This allows us to find some additional structure of the Lenstra graph: for example, (i) between $\delta = 0.99$ and $\delta = 0.994$, it is as though Bases 1 and 2 are exchanging some of their hits, (ii) for the original version, the sum of the frequencies of Bases 9, 14, 19 with $\delta = 0.994$ equals 19313, which falls a bit short of 21492 for Basis 9 with $\delta = 0.999$ — the remaining 2000 hits probably come from Basis 3.

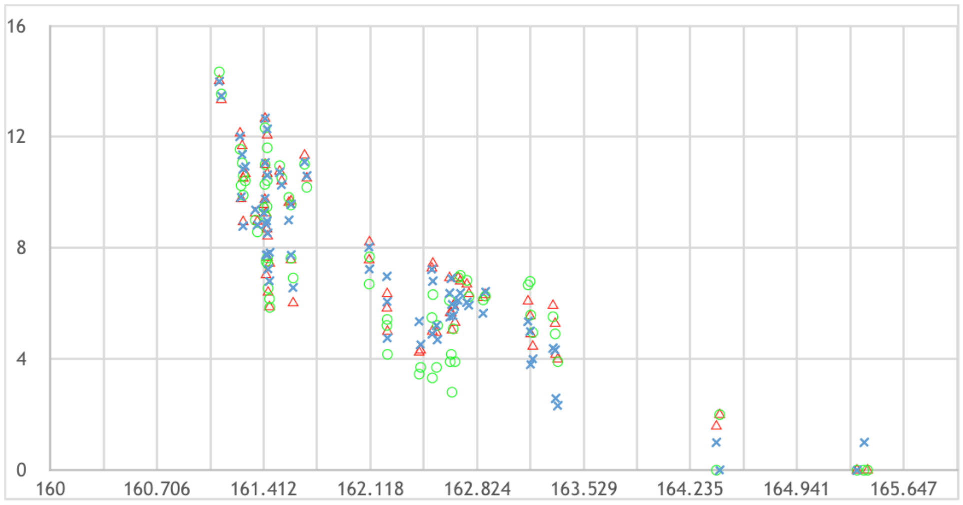

Fig. 4. The same data as Figure 2, for seed = 36368. Circle, triangle, and X shape correspond to the original, random, and potential variants, respectively.

On the shape of the Lenstra graph. Our data from Experiment 2 supports Conjecture 1 by suggesting that the Lenstra graph has a very forcing structure — in other words, put it roughly, no matter how one walks on it, somehow one ends up taking the same paths (in a statistical sense). The fact that all three variants exhibit the identical boundary structure implies that, near the terminal points of the Lenstra graph, all three variants behave in almost the same way. More precisely, our data makes it plain clear that, if we chose the input bases uniformly from the 0.97-LLL bases found in the experiment, the variants would yield indistinguishable frequency tables. As we start from farther and farther away, this tendency probably weakens, as there may exist occasional divergences off the main flow directed by the graph's structure (in near future, we plan to carry out a deeper investigation in this direction). Still, even in this case, the graph seems to play the dominant role in shaping the dynamics of any random walk on it: variation in log(frequency) is bounded by a small number, and thus so should be the "coefficient of proportionality" between log(frequency) and log(energy) — see Figure 4 for instance.

All of this is very surprising, since the odds that procedures as disparate as our three variants display such similar patterns on a family of graphs that are as vast as the Lenstra graph must be extremely tiny. By the same token, it is truly fortunate that Conjecture 1 has survived the challenge. Had it been otherwise, one would have to take into account the Lenstra graph *plus* the specific characteristics of each LLL-like algorithm in order to handle the problems concerning its behavior — e.g. those mentioned in the introduction — which is likely to make matters virtually intractable.

As $n \to \infty$, we predict that the phenomena we have discovered in this section will become more conspicuous, though it may seem counterintuitive to some. The source of our intuition is the conjectured 0-1 law for polynomial-time computable properties of random lattices due to Ajtai [Ajt02], which rigorously phrases the conventional wisdom "All lattices look the same." Unfortunately, as we commented earlier, an experiment as microscopic as what we have conducted here is computationally infeasible in higher dimensions. In a forthcoming work, we will report on a separate set of experiments that test Conjecture 1 in dimensions at least 100.

5 Summary and further discussions

The original intention of the experiments above was to explore if walking on the Lenstra graph in different ways leads to any improvement of LLL. On the contrary, we have found that those variants exhibit remarkably alike behavior, at many levels — hence Conjecture 1. This is in fact a very pleasant news for lattice-based cryptography because it reduces the study of all LLL-like algorithms to the study of the Lenstra graph. In this section, we discuss the role of Conjecture 1 in responding to the motivating problems in the introduction, which will be the subject of a series of forthcoming works.

Potential explanation for "1.02." For $x_1 < x_2$, let $D(x_1, x_2)$ be the "density" of LLL bases among the set of all size-reduced bases of energy between x_1 and x_2. Here, the word density is defined in some appropriate manner, and we do not have to specify a lattice, thanks to Theorem 2, which suggests that $D(x_1, x_2)$ should not differ by much across lattices.

Clearly $D(x_1, x_2)$ is expected to approach 1 as x_1 and x_2 goes to zero. Now imagine any walk on the Lenstra graph. Our basic idea is that, this walk is very likely to get "stuck" and terminate while passing a certain energy window (x_1, x_2) at which $D(x_1, x_2)$ is high enough.

To elaborate, observe that from $D(x_1, x_2)$ one can compute the (right cumulative) distribution function of the energy of the terminal point. The correlation between energy and $\|\mathbf{b}_1\|$ can be quickly studied, so this translates to the distribution function, say $B(x)$ of $\|\mathbf{b}_1\|$. Our goal is to show that $B(x)$ sharply decreases at $x \approx (1.02)^n \det(L)^{1/n}$, in accordance with the experiment of [NS06]. If this can be verified, it would be a convincing explanation for the over-performance of LLL. It would also explain why the input basis would not affect RHF, as conjectured in (reference): our suggested shape of $B(x)$ puts an overwhelming pressure for a walk to terminate at a certain point, and it must be extremely difficult to find the starting point of a path that could circumvent it.

The hardest part in studying $D(x_1, x_2)$ is to find the correct notion of "density." The set of the size-reduced bases of a given energy range can easily be shown to have infinite cardinality, so one has to trim it carefully so as to reflect the actual behavior of LLL. Once this is done, $D(x_1, x_2)$ would be computed by the same method as in the proof of Theorem 1.

Note that the idea outlined above would have been quickly falsified had different variants induced different values for RHF in Experiment 1. In other words, at least some part of Conjecture 1 must be true in order for the present argument to work at all.

The case of BKZ. The earlier subsection may be recapitulated as follows:

LLL-like algorithms exhibit RHF ≈ 1.02 due to the high concentration of terminal points at a certain section of the Lenstra graph.

In future works, we hope to argue something similar of BKZ that leads to a lower bound on its performance. However, a good analogue of the Lenstra graph in this case does not immediately come to mind.

We take a step back and ask, why does LLL fail to reach a basis of better quality? The reason we suggested above may be paraphrased as follows: since LLL is limited to working on local blocks of size two, it cannot "see" an improvement that is too far away (and the Lenstra graph is precisely the pictorial representation of what LLL can immediately see from where it is standing). BKZ would be capable of seeing farther, the more so if the blocksize β is greater. Our eventual aim is to find a scientific argument directly associating β to the performance of *any variant* of BKZ-β, just as in the present paper we initiated the project for $\beta = 2$, via the Lenstra graph. We believe this is a superior approach to the security estimate of lattice-based cryptosystems to re-estimating everything each time a new variant of BKZ (reference to the Progressive BKZ) arrives to the scene.

References

[Ajt02] Miklos Ajtai. Random lattices and a conjectured 0-1 law about their polynomial time computable properties. *Proc. of FOCS*, pages 13–39, 2002.

[BBC$^+$09] Brandon Ballinger, Grigoriy Blekherman, Henry Cohn, Noah Giansiracusa, Elizabeth Kelly, and Achill Schürmann. Experimental study of energy-minimizing point configurations on spheres. *Experiment. Math.*, 18(3):257–283, 2009.

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

[GM03] Daniel Goldstein and Andrew Mayer. On the equidistribution of Hecke points. *Forum Math.*, 15(2):165–189, 2003.

[GN08] Nicolas Gama and Phong Q. Nguyen. Predicting lattice reduction. In *Advances in cryptology—EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Comput. Sci.*, pages 31–51. Springer, Berlin, 2008.

[KV16] Seungki Kim and Akshay Venkatesh. The behavior of random reduced bases. Preprint, available at `http://math.stanford.edu/~akshay/research/research.html`, 2016.

[Len01] Hendrik W. Lenstra, Jr. Flags and lattice basis reduction. In *European Congress of Mathematics, Vol. I (Barcelona, 2000)*, volume 201 of *Progr. Math.*, pages 37–51. Birkhäuser, Basel, 2001.

[LLL82] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261(4):515–534, 1982.

[Neu16] Arnold Neumaier. Bounding basis reduction properties. *Des. Codes Cryptogr.*, pages 1–23, 2016.

[NS05] Phong Q. Nguyen and Damien Stehlé. Floating-point LLL revisited. In *Advances in cryptology—EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Comput. Sci.*, pages 215–233. Springer, Berlin, 2005.

[NS06] Phong Q. Nguyen and Damien Stehlé. LLL on the average. In *Algorithmic number theory*, volume 4076 of *Lecture Notes in Comput. Sci.*, pages 238–256. Springer, Berlin, 2006.

[Sch11] Peter Schnorr. Accelerated and improved slide- and LLL-reduction. *ECCC*, 11(50), 2011.

[Ser73] J.-P. Serre. *A course in arithmetic*. Springer-Verlag, New York-Heidelberg, 1973. Translated from the French, Graduate Texts in Mathematics, No. 7.

[Sho] Victor Shoup. NTL, number theory C++ library. `http://www.shoup.net/ntl/`.

[Sie45] Carl Ludwig Siegel. A mean value theorem in geometry of numbers. *Ann. of Math. (2)*, 46:340–347, 1945.

[Ste10] Damien Stehlé. Floating-point LLL: theoretical and practical aspects. In Phong Nguyen and Brigitte Vallée, editors, *The LLL Algorithm, Survey and Applications*, Informaton Security and Cryptography, chapter 5, pages 179–213. Springer-Verlag, Berlin, Heidelberg, 2010.

Appendix: special cases

In this appendix, we account for the two known cases, according to Stehlé [Ste10], in which LLL is known to perform particularly well; that is, it often finds the shortest vector of the lattice.

The case dimension ≤ 6. This is easily explained by Theorem 1, which provides an exact formula for the average number of LLL bases per lattice. Table 6 shows the average number of $(0.999, 0.501)$-LLL bases in dimensions 2 through 15, up to the sign of each basis vector. Table 6 makes it plain that in dimensions ≤ 6, an LLL basis is very much likely also a Minkowski-reduced basis — so it is very hard for LLL to not find the shortest vector in these cases.

The case $\lambda_i \ll \lambda_{i+1}$ for some i. λ_i, as usual, means the i-th successive minimum of a lattice under question, say L. $\lambda_i \ll \lambda_{i+1}$ means that there exists a constant $C > 0$, depending on the dimension n of L and $\det L$, such that $C\lambda_i < \lambda_{i+1}$. Throughout the argument below, we will be implicitly increasing C whenever needed for the desired inequality to hold.

Let $\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ be a (δ, η)-reduced basis, and write $\beta = 1/(\delta - \eta^2) > 1$. The crucial fact is that

$$\beta^{1-k}\lambda_k \leq \|\mathbf{b}_k\|^2 \leq \beta^{n-1}\lambda_k \tag{3}$$

n	average/2^n	n	average/2^n
2	1.002	9	9.036
3	1.013	10	41.09
4	1.046	11	378
5	1.132	12	8351
6	1.345	13	524517
7	1.878	14	1.096×10^8
8	3.397	15	8.830×10^{10}

Table 6. Average number of (0.999,0.501)-LLL bases in low dimensions.

for all $k = 1, \dots, n$; for a proof, see [LLL82]. Now let $\{\mathbf{m}_1, \dots, \mathbf{m}_n\}$ be the Minkowski-reduced basis of the lattice under question. Then using (3) we can show that

$$\|\mathbf{m}_i\|^2 \le \|\mathbf{b}_i\|^2 \le \beta^{n-1}\lambda_i \ll \lambda_{i+1} \le \|\mathbf{m}_{i+1}\|^2.$$

Since a Minkowski-reduced basis is also size-reduced, this implies that

$$\|\mathbf{b}_i\| \ll \|\mathbf{m}^*_{i+1}\|.$$

Furthermore, because a Minkowski-reduced basis is also $(1, 0.5)$-reduced, we have $\|\mathbf{m}^*_{i+1+j}\| \ge (3/4)^{j/2}\|\mathbf{m}^*_{i+1}\|$. Therefore,

$$\mathrm{span}_{\mathbb{Z}}(\mathbf{b}_1, \dots, \mathbf{b}_i) = \mathrm{span}_{\mathbb{Z}}(\mathbf{m}_1, \dots, \mathbf{m}_i).$$

So $\|\mathbf{b}_1\| \le \beta^{(i-1)/4}\det(\mathbf{m}_1 \dots \mathbf{m}_i)^{1/i}$. It remains to show that $\det(\mathbf{m}_1 \dots \mathbf{m}_i)^{1/i}$ is not too large. Therefore,

$$\det L \ge \prod_{j=1}^{i} \|\mathbf{m}^*_j\| \cdot \left(\frac{3}{4}\right)^{\frac{(n+i+2)(n-i+1)}{2}} \|\mathbf{m}^*_{i+1}\|^{n-i}.$$

So if $\|\mathbf{m}^*_{i+1}\|$ is sufficiently large, $\prod_{j=1}^{i}\|\mathbf{m}^*_j\| = \det(\mathbf{m}_1 \dots \mathbf{m}_i)$ must be small, as desired.

Remark. We do not claim that the above argument, though completely rigorous, describes the exact reality. However, it does convey a couple of intuitions as to why LLL should operate particularly well on lattices satisfying $\lambda_i \ll \lambda_{i+1}$ for some i: i) LLL acts on such a lattice as though it does on a direct sum of two lattices, each of dimensions i and $n - i$, ii) if λ_{i+1} is sufficiently large, $\det(\mathbf{m}_1 \dots \mathbf{m}_i)$ is small, or at least not too large.