Randomised Algorithm for Integer Signal Recovery

Yingzi Xu
Supervisor: Prof. Xiao-Wen Chang
Second Supervisor: Prof. Tim Hoheisel

November 2019

Abstract

In many applications such as multi-input and multi-output (MIMO) systems in wireless communications, it is needed to recover the signal vector, whose entries are all integers, from observations. The optimal approach is to solve an integer least squares (ILS) problem, which has been proven to be NP-hard. In this project we focus on Klein's randomised algorithm, which finds a sub-optimal solution to the ILS problem. We will investigate theoretical aspects of this algorithm, including the effect of input parameter and success probability derivation, as well as some experimental observations on the behaviour of Klein's algorithm.

1 Introduction

Consider the following linear model

$$\mathbf{y} = \mathbf{A}\mathbf{x}^* + \mathbf{v} \tag{1}$$

where $\mathbf{y} \in \mathbb{R}^m$ is a measurement vector, $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a fixed model matrix with full column rank, $\mathbf{x}^* \in \mathbb{Z}^n$ is an unknown integer parameter vector, and $\mathbf{v} \in \mathbb{R}^m$ is a noise vector, for which we assume that it follows a multivariate Gaussian distribution $\mathbf{v} \sim N(0, \sigma^2\mathbf{I})$, with σ being known. To find the parameter vector $\mathbf{x}^*$, we can solve the integer least squares (ILS) problem, given by

$$\min_{\mathbf{x}\in\mathbb{Z}^n} ||\mathbf{y} - \mathbf{A}\mathbf{x}||_2^2 \tag{2}$$

This problem is NP-hard [5]. This means the running time of all existing algorithms for solving the ILS problem is exponential in the dimension of $\mathbf{A}$ in the worst case. A typical approach for solving the ILS problem is sphere decoding, which finds the optimal solution in an ellipsoid. The most often used sphere decoding algorithm is due to Schnorr and Euchner [6].

As there is no known efficient algorithm to solve this problem, some algorithms have been developed to find sub-optimal solutions to (2), with much lower complexity compared to sphere decoding.

One commonly used algorithm which finds a sub-optimal solution to the ILS problem is the nearest plane algorithm by Babai [1]. The found sub-optimal solution is known as the Babai point. It is a deterministic point that can be computed very efficiently. The Babai point is also the first integer point found by the Schnorr-Euchner's sphere decoding algorithm.

Klein's algorithm [4] (also known as random successive interference cancellation (SIC) algorithm in communications), which is the focus of this project, is a randomised version of Babai's nearest plane algorithm. Instead of outputting a deterministic point, the algorithm samples integer points as approximations to the solution of (2) following a probability distribution. Due to this additional randomness, this algorithm can give us better results compared to the Babai point, if we are allowed to run it multiple times. Details of Klein's algorithm will be given in Section 2.

To verify the goodness of a sub-optimal solution, one measure is its success probability, the probability that the estimated integer vector $\hat{\mathbf{x}}$ equals to the true parameter vector $\mathbf{x}^*$. In the original problem (2) the randomness comes from the noise vector $\mathbf{v}$, whose distribution is known. If we use the Babai point as the estimator, the success probability of the Babai point is derived in [2]. It is also shown in [7, Corollary 2] that as $\sigma \to 0$, the success probability of the Babai point goes to 1. Therefore the Babai point can be considered as a good estimator when σ is small. Since Klein's algorithm is a randomised version of the Babai point, it is natural that we investigate its success probability, and the relationship between the randomized Babai point and the deterministic Babai point, which will be presented in the following sections of this report.

2 Brief Description of Klein's Algorithm

Assume the model matrix $\mathbf{A}$ with full column rank has the QR-factorisation

$$\mathbf{A} = \begin{bmatrix}\mathbf{Q}_1 & \mathbf{Q}_2\end{bmatrix}\begin{bmatrix}\mathbf{R}\\ \mathbf{0}\end{bmatrix} = \mathbf{Q}_1\mathbf{R} \tag{3}$$

where $\begin{bmatrix}\mathbf{Q}_1 & \mathbf{Q}_2\end{bmatrix} \in \mathbb{R}^{m\times m}$ is orthonormal, and $\mathbf{R} \in \mathbb{R}^{n\times n}$ is an upper-triangular matrix with positive diagonal entries. Denote r_{ij} the (i,j) entry of the matrix $\mathbf{R}$.

Using this decomposition, we can transform the original ILS problem (2) to

$$\min_{\mathbf{x}\in\mathbb{Z}^n} ||\tilde{\mathbf{y}} - \mathbf{R}\mathbf{x}||_2^2 \tag{4}$$

where

$$\tilde{\mathbf{y}} = \mathbf{Q}_1^T\mathbf{y} = \mathbf{R}\mathbf{x}^* + \tilde{\mathbf{v}}, \quad \tilde{\mathbf{v}} = \mathbf{Q}_1^T v \sim N(0, \sigma^2\mathbf{I}). \tag{5}$$

Note the distribution of noise vector remains the same due to the column orthogonality of $\mathbf{Q}_1$.

We first describe the Babai nearest algorithm in Algorithm 1, which takes inputs $\mathbf{R}$ and $\tilde{\mathbf{y}}$ and outputs the Babai point $\mathbf{x}^{\mathrm{B}}$.

Algorithm 1 Babai's Nearest Plane Algorithm (Babai Point)

```
Input: $\tilde{\mathbf{y}} \in \mathbb{R}^n$ and non-singular upper-triangular matrix $\mathbf{R} \in \mathbb{R}^{m\times n}$
Output: $\mathbf{x}^{\mathrm{B}} = \mathrm{babai}(\mathbf{R}, \tilde{\mathbf{y}})$
1: for $i = n : -1 : 1$ do
2:     $c_i = (\tilde{y}_i - \sum_{j=i+1}^{n} r_{ij}\hat{x}_j)/r_{ii}$
3:     $\hat{x}_i^{\mathrm{B}} = \lfloor c_i \rceil$
4: end for
```

Unlike Babai's algorithm, which rounds c_i to the nearest integer, Klein's algorithm randomly rounds it to an integer number according to a probability distribution. Specifically it has a sub-algorithm (see Algorithm 2), which takes as input a real number c and a parameter $\beta > 0$ and builds a discrete Gaussian-like probability distribution according to these two inputs, and output one integer sampled randomly from the constructed distribution. The probability that an integer is sampled depends on the distance from c. The closer an integer l is to the input c, the more likely it is to be sampled. Klein's algorithm is described in Algorithm 3. Klein's algorithm has an extra input parameter $\alpha > 0$, compared to Babai's algorithm. By calling the sub-algorithm to obtain each entry of the parameter vector $\hat{\mathbf{x}}$, Klein's algorithm constructs a multivariate Gaussian-like discrete distribution, with centre being the projected measurement vector $\tilde{\mathbf{y}}$ [3].

Algorithm 2 Randomised Rounding

```
Input: $c \in \mathbb{R}$ and parameter $\beta > 0$
Output: $\hat{x} = \mathrm{randRound}(c, \beta)$
1: $a = c - \lfloor c \rfloor$
2: $b = 1 - a$
3: $s = \sum_{j\geq 0} e^{-\beta(a+j)^2} + e^{-\beta(b+j)^2}$
4: Randomly choosing an integer $\hat{x}$ according to the distribution:
   $\Pr(\hat{x} = c - (a+l)) = \frac{e^{-\beta(a+l)^2}}{s}$, $\Pr(\hat{x} = c + (b+l)) = \frac{e^{-\beta(b+l)^2}}{s}$ for $l \geq 0$
```

Algorithm 3 Klein's Randomised Nearest Plane Algorithm

```
Input: $\tilde{\mathbf{y}} \in \mathbb{R}^n$, non-singular upper-triangular matrix $\mathbf{R} \in \mathbb{R}^{m\times n}$ and parameter $\alpha > 0$
Output: $\hat{\mathbf{x}} = \mathrm{randBabai}(\mathbf{R}, \tilde{\mathbf{y}}, \alpha)$
1: for $i = n : -1 : 1$ do
2:     $c_i = (\tilde{y}_i - \sum_{j=i+1}^{n} r_{ij}\hat{x}_j)/r_{ii}$
3:     $\hat{x}_i = \mathrm{randRound}(c_i, \alpha r_{ii}^2)$
4: end for
```

Constructing the Probability Distribution of Klein's Algorithm Efficiently

The result of this section comes from Zhilong Chen.

Constructing the probability distribution in Algorithm 2 is time-consuming. On theoretical level, all integers are potential candidates to be sampled; while on computation level, we need to find all integers that have non-zero probability after rounding to be sampled, and the number of candidates could be very large if the parameter α is small.

We can efficiently construct a discrete distribution which is similar to that in Algorithm 2 using continuous Gaussian distribution as follows. Let $\xi \sim N(0, \omega^2)$, where the value of ω to be determined later. Let

$$l = \lfloor c + \xi \rceil$$

and the probability that c is randomly rounded to l is given by

$$\Pr(\hat{x} = l) = \int_{l-0.5}^{l+0.5} \frac{1}{\sqrt{2\pi}\omega} e^{-\frac{(\zeta - c)^2}{2\omega^2}} d\zeta \tag{6}$$

We re-write the above equation (6) into a similar form to the probability in Algorithm 2 as

$$\begin{aligned} \Pr(\hat{x} = l) &= \frac{\int_{l-0.5}^{l+0.5} \frac{1}{\sqrt{2\pi}\omega} e^{-\frac{(-\zeta - c)^2}{2\omega^2}} d\zeta}{\sum_{m \in \mathbb{Z}} \int_{m-0.5}^{m+0.5} \frac{1}{\sqrt{2\pi}\omega} e^{-\frac{(\zeta - c)^2}{2\omega^2}} d\zeta} \\ &= \frac{\int_{l-0.5}^{l+0.5} e^{-\frac{(\zeta - c)^2}{2\omega^2}} d\zeta}{\sum_{m \in \mathbb{Z}} \int_{m-0.5}^{m+0.5} e^{-\frac{(\zeta - c)^2}{2\omega^2}} d\zeta} \end{aligned} \tag{7}$$

Using Mean Value Theorem, there exists $z \in [l - 0.5, l + 0.5]$ such that

$$\int_{l-0.5}^{l+0.5} e^{-\frac{(\zeta - c)^2}{2\omega^2}} d\zeta = e^{-\frac{(z-c)^2}{2\omega^2}} \tag{8}$$

Setting ω such that it satisfies $\alpha = \frac{1}{2\omega^2}$, we can treat (7) as an approximation to the probability distribution defined in Algorithm 2. Using this we only need to sample $\xi \sim N(0, \omega^2)$ to obtain $l \in \mathbb{Z}$, instead of constructing the whole discrete distribution.

In the following sections, all analysis is still conducted under the original construction of probability distribution in [4].

3 The Effect of Parameter α in Klein's Algorithm

We can see from Algorithms 2 and 3 that the parameter α controls the probability distribution of integers. It is easy to see that if we let $\alpha \to 0$, then at each level the randomised rounding algorithm samples an integer from a distribution that converges towards a uniform distribution. Therefore setting α to be a very small number does not help us much.

Klein's algorithm is a randomised version of Babai's point, therefore we would like to investigate the relationship between the sampled integer vector from Klein's algorithm and the deterministic Babai point. In particular, we would like to know when this algorithm outputs Babai point almost surely.

We start from considering only the randRound algorithm. We claim that we have the following lemma.

Lemma 3.1. *The randRound algorithm with input (c, β) outputs $\lfloor c \rceil$ almost surely as $\beta \to \infty$, provided $c - \lfloor c \rceil \neq \frac{1}{2}$.*

Proof. We have that

$$a = c - \lfloor c \rceil \qquad b = 1 - a \qquad a \neq b$$

Without loss of generality assume $a > b$, i.e. $\lfloor c \rceil = c + b$.

Claim 1. $\frac{\sum_{l \geq 1} e^{-\beta(a+l)^2} + e^{-\beta(b+l)^2}}{\sum_{j \geq 0} e^{-\beta(a+j)^2} + e^{-\beta(b+j)^2}} \to 0$ *as* $\beta \to \infty$

This is equivalent to

$$\frac{e^{-\beta a^2} + e^{-\beta b^2}}{\sum_{j \geq 0} e^{-\beta(a+j)^2} + e^{-\beta(b+j)^2}} \to 1 \quad \text{as } \beta \to \infty \tag{9}$$

Note that $\sum_{j \geq 0} e^{-\beta(a+j)^2} + e^{-\beta(b+j)^2}$ is a convergent series. Therefore there exists some $\epsilon = \epsilon(\beta) > 0$ such that $\sum_{j \geq 1} e^{-\beta(a+j)^2} + e^{-\beta(b+j)^2} < \epsilon$.

When $\beta \to \infty$ every term in this series converges to 0 and the sum is always finite. Therefore the supremum converges to 0 as $\beta \to \infty$. Hence we can send the upper bound $\epsilon \to 0$. Then we have

$$1 > \frac{e^{-\beta a_n^2} + e^{-\beta b^2}}{\sum_{j \geq 0} e^{-\beta(a+j)^2} + e^{-\beta(b+j)^2}} > \frac{e^{-\beta a^2} + e^{-\beta b^2}}{e^{-\beta a^2} + e^{-\beta b^2} + \epsilon} \to 1 \text{ as } \beta \to \infty \tag{10}$$

By Squeeze Theorem, (9) holds. This proves Claim 1.

Claim 2. $\frac{e^{-\beta b^2} - e^{-\beta a^2}}{\sum_{j \geq 0} e^{-\beta(a+j)^2} + e^{-\beta(b+j)^2}} \to 1$ *as* $\beta \to \infty$.

Similar as in Claim 1, we can show that

$$1 > \frac{e^{-\beta b^2} - e^{-\beta a^2}}{\sum_{j \geq 0} e^{-\beta(a+j)^2} + e^{-\beta(b+j)^2}} > \frac{e^{-\beta b^2} - e^{-betaa^2}}{e^{-\beta a^2} + e^{-\beta b^2} + \epsilon} \to \frac{e^{-\beta b^2} - e^{-\beta a^2}}{e^{-\beta a^2} + e^{-\beta b^2}} \tag{11}$$

as $\beta \to \infty$. Furthermore,

$$\begin{aligned} \frac{e^{-\beta b^2} - e^{-\beta a^2}}{e^{-\beta a^2} + e^{-\beta b^2}} &= \frac{1}{e^{-\beta(a^2 - b^2)} + 1} - \frac{1}{1 + e^{-\beta(b^2 - a^2)}} \\ &\to \frac{1}{0+1} - \frac{1}{1+\infty} \qquad \text{as } \beta_n \to \infty \\ &= 1 - 0 = 1 \end{aligned} \tag{12}$$

Therefore by Squeeze Theorem, Claim 2 holds.

From the above two claims we have

$$\frac{e^{-\beta b^2}}{\sum_{j\geq 0} e^{-\beta(a+j)^2} + e^{-\beta(b+j)^2}} \to 1 \text{ as } \beta \to \infty \tag{13}$$

and therefore $\Pr(\text{randRound}(c, \beta) = c + b) = 1$ as $\beta \to \infty$.

By symmetrical arguments, if $a < b$ we can show that $\Pr(\text{randRound}(c, \beta) = c - a) = 1$ as $\beta \to \infty$. □

Remark. When $a = b = \frac{1}{2}$, Claim 1 still holds, but Claim 2 fails since $e^{-\beta b^2} = e^{-\beta a^2}$.

In [4] the author suggested the parameter $\alpha = \frac{\log(n)}{\min_{1\leq i\leq n} r_{ii}^2}$. We are to demonstrate the effect of varying the parameter on Klein's algorithm by proving the following theorem.

Theorem 3.1. *Given* $\mathbf{R}$ *and* $\tilde{\mathbf{y}}$*, Klein's algorithm outputs the Babai point almost surely as* $\alpha \to \infty$*.*

Proof. We prove this theorem using Lemma 3.1. In order to use this lemma, we first need to show that at each level $i \in \{1, ..., n\}$, $c_i - \lfloor c_i \rfloor = \frac{1}{2}$ has probability 0.

From (5) we have the probability distribution $\tilde{\mathbf{y}} \sim N(\mathbf{R}\mathbf{x}^*, \sigma^2\mathbf{I})$, and at n^{th} level we have $y_n \sim N(r_{nn}x_n^*, \sigma^2)$. So $c_n = \frac{y_n}{r_{nn}} \sim N(x_n^*, \frac{\sigma^2}{r_{nn}^2})$ which is a continuous distribution. Therefore there are a countable number of points such that $c_n - \lfloor c_n \rfloor = \frac{1}{2}$, which leads to that the probability of having such points is 0. Similarly for other levels, c_i also follows a continuous Gaussian distribution, therefore the probability of having $c_i - \lfloor c_i \rfloor = \frac{1}{2}$ is 0.

Now assume it holds that at each level $i \in \{1, ..., n\}$, $c_i - \lfloor c_i \rfloor \neq \frac{1}{2}$. In Algorithm 3, at each level we call the randomised rounding algorithm. From Lemma 3.1 we can obtain that $\text{randRound}(c_n, \alpha r_{nn}^2)$ outputs $\lfloor c_n \rceil$ almost surely if $\alpha \to \infty$. Thus by induction it holds with probability 1 that, at each level i,

$$c_i = x_i^* + \frac{\tilde{v}_i}{r_{ii}}$$

and therefore $\text{randRound}(c_i, \alpha r_{ii}^2)$ outputs $\lfloor c_i \rceil$ almost surely at each level if $\alpha \to \infty$. Thus Klein's algorithm outputs Babai point almost surely if we let $\alpha \to \infty$. □

4 Success Probability of the Randomised Babai Point

In this section we will give a derivation of the Randomised Babai Point produced by Klein's algorithm.

In [4, Lemma 3.5] Klein gives a lower bound on the success probability. Given $\tilde{\mathbf{y}}$ or given $\tilde{\mathbf{v}}$ in (5), the probability that Algorithm 3 outputs $\mathbf{x}^*$ is

$$\Pr(\hat{\mathbf{x}} = \mathbf{x}^* | \tilde{\mathbf{y}}) = \Pr(\hat{\mathbf{x}} = \mathbf{x}^* | \tilde{\mathbf{v}}) \geq \prod_{i=1}^{n} \frac{e^{-\alpha ||\tilde{\mathbf{v}}||^2}}{\sum_{j\geq 0} e^{-\alpha r_{ii}^2 j^2} + e^{-\alpha r_{ii}^2 (1+j)^2}} \tag{14}$$

Numerical experiments show that this bound is not tight, and the looseness comes from the inequality given in [4, Lemma 3.1]:

$$\sum_{j\geq 0} e^{-\alpha r_{ii}^2(a_i+j)^2} + e^{-\alpha r_{ii}^2(b_i+j)^2} \leq \sum_{j\geq 0} e^{-\alpha r_{ii}^2 j^2} + e^{-\alpha r_{ii}^2(1+j)^2} \tag{15}$$

To obtain a good theoretical success probability, we do not use the inequality (15).

Note that

$$\Pr(\hat{\mathbf{x}} = \mathbf{x}^*) = \Pr(\hat{x}_n = x_n^*) \prod_{i=1}^{n-1} \Pr(\hat{x}_i = x_i^* | \hat{x}_j = x_j^*,\ j = i+1, ..., n) \tag{16}$$

At level n, we have

$$c_n = \frac{\tilde{y}_n}{r_{nn}} = x_n^* + \frac{\tilde{v}_n}{r_{nn}} \tag{17}$$

Let $a_n := \frac{\tilde{v}_n}{r_{nn}} - \lfloor \frac{\tilde{v}_n}{r_{nn}} \rfloor$ and $b_n := 1 - a_n$. Then

$$\begin{aligned} \Pr(\hat{x}_n = x_n^* | \tilde{v}_n) &= \int_{-\infty}^{\infty} \Pr(\hat{x}_n = x_n^* | \tilde{v}_n) d\tilde{v}_n \\ &= \int_{-\infty}^{\infty} \frac{e^{-\alpha v_n^2}}{\sum_{j\geq 0} e^{-\alpha r_{nn}^2(a_n+j)^2} + e^{-\alpha r_{nn}^2(b_n+j)^2}} d\tilde{v}_n \\ &= \int_{-\infty}^{\infty} \frac{e^{-\alpha v_n^2}}{\sum_{j\geq 0} e^{-\alpha r_{nn}^2(\frac{\tilde{v}_n}{r_{nn}} - \lfloor \frac{\tilde{v}_n}{r_{nn}} \rfloor + j)^2} + e^{-\alpha r_{nn}^2(1 - \frac{\tilde{v}_n}{r_{nn}} + \lfloor \frac{\tilde{v}_n}{r_{nn}} \rfloor + j)^2}} d\tilde{v}_n \end{aligned} \tag{18}$$

Let f be the probability density function of the Gaussian distribution $N(0, \sigma^2)$. The probability that $\hat{x}_n = x_n^*$ is given by

$$\begin{aligned} \Pr(\hat{x}_n = x_n^*) &= \int_{-\infty}^{\infty} \Pr(\hat{x}_n = x_n^* | \tilde{v}_n) f(\tilde{v}_n) d\tilde{v}_n \\ &= \int_{-\infty}^{\infty} \frac{e^{-\alpha v_n^2} f(v_n)}{\sum_{j\geq 0} e^{-\alpha r_{nn}^2(\frac{\tilde{v}_n}{r_{nn}} - \lfloor \frac{\tilde{v}_n}{r_{nn}} \rfloor + j)^2} + e^{-\alpha r_{nn}^2(1 - \frac{\tilde{v}_n}{r_{nn}} + \lfloor \frac{\tilde{v}_n}{r_{nn}} \rfloor + j)^2}} d\tilde{v}_n \end{aligned} \tag{19}$$

Furthermore,

$$\begin{aligned} e^{-\alpha \tilde{v}_n^2} f(\tilde{v}_n) &= e^{-\alpha \tilde{v}_n^2} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\tilde{v}_n^2}{2\sigma^2}} \\ &= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(\alpha + \frac{1}{2\sigma^2})\tilde{v}_n^2} \\ &= \frac{1}{\sqrt{2\pi\sigma^2}} \cdot \sqrt{2\pi\delta^2} \left(\frac{1}{\sqrt{2\pi\delta^2}} e^{-\frac{\tilde{v}_n^2}{2\delta^2}} \right) \quad \text{where } \delta = \frac{\sigma}{\sqrt{1 + 2\sigma^2\alpha}} \\ &= \frac{1}{\sqrt{1 + 2\sigma^2\alpha}} g(\tilde{v}_n) \end{aligned} \tag{20}$$

where g is the probability density function of the Gaussian distribution $N(0, \frac{\sigma^2}{1+2\sigma^2\alpha})$. Thus

$$\Pr(\hat{x}_n = x_n^*) = \frac{1}{\sqrt{1+2\sigma^2\alpha}} \int_{-\infty}^{\infty} \frac{g(\tilde{v}_n)}{\sum_{j\geq 0} e^{-\alpha r_{nn}^2(\frac{\tilde{v}_n}{r_{nn}} - \lfloor \frac{\tilde{v}_n}{r_{nn}} \rfloor + j)^2} + e^{-\alpha r_{nn}^2(1-\frac{\tilde{v}_n}{r_{nn}} + \lfloor \frac{\tilde{v}_n}{r_{nn}} \rfloor + j)^2}} d\tilde{v}_n \quad (21)$$

For any $i \in \{1, ..., n-1\}$, suppose that $\hat{x}_j = x_j^*$ for $j = i+1, ..., n$. Then

$$c_i = (\tilde{y}_i - \sum_{j=i+1}^{n} r_{ij}\hat{x}_j)/r_{ii} = \left(\sum_{j=i}^{n} r_{ij}x_j^* + \tilde{v}_i - \sum_{j=i+1}^{n} r_{ij}x_j^* \right) /r_{ii} = x_i^* + \frac{\tilde{v}_i}{r_{ii}} \quad (22)$$

Therefore we can obtain the expression of $\Pr(\hat{x}_i = x_i^* | \hat{x}_j = x_j^*,\ j = i+1, ..., n)$ by the expression of $\Pr(\hat{x}_n = x_n^*)$ with proper index change:

$$\Pr(\hat{x}_i = x_i^* | \hat{x}_j = x_j^* \, \forall j > i) = \frac{1}{\sqrt{1+2\sigma^2\alpha}} \int_{-\infty}^{\infty} \frac{g(\tilde{v}_i)}{\sum_{j\geq 0} e^{-\alpha r_{ii}^2(\frac{\tilde{v}_i}{r_{ii}} - \lfloor \frac{\tilde{v}_i}{r_{ii}} \rfloor + j)^2} + e^{-\alpha r_{ii}^2(1-\frac{\tilde{v}_i}{r_{ii}} + \lfloor \frac{\tilde{v}_i}{r_{ii}} \rfloor + j)^2}} d\tilde{v}_i \quad (23)$$

By (16), the success probability is given by

$$\Pr(\mathbf{\hat{x}} = \mathbf{x}^*) = (1+2\sigma^2\alpha)^{-\frac{n}{2}} \prod_{i=1}^{n} \int_{-\infty}^{\infty} \frac{g(\tilde{v}_i)}{\sum_{j\geq 0} e^{-\alpha r_{ii}^2(\frac{\tilde{v}_i}{r_{ii}} - \lfloor \frac{\tilde{v}_i}{r_{ii}} \rfloor + j)^2} + e^{-\alpha r_{ii}^2(1-\frac{\tilde{v}_i}{r_{ii}} + \lfloor \frac{\tilde{v}_i}{r_{ii}} \rfloor + j)^2}} d\tilde{v}_i \quad (24)$$

Numerical Approximation of Success Probability

Since it is difficult to compute the integral, we can use numerical approximation to get an estimated probability. Let

$$h(x, \beta) := \sum_{j\geq 0} e^{-\beta(x+j)^2} + e^{-\beta(1-x+j)^2}$$

with $x \in [0, 1)$. Let $N \in \mathbb{N}$ be a reasonably large positive integer number. In the numerical estimation we will use midpoint rule, in which N represents the equidistant partitioning of each interval $[k, k+1] \, \forall k \in \mathbb{N}_0$. The larger N is, the more accurate this estimate is, and the higher the computational cost is.

Denote $u_i := \frac{\tilde{v}_i}{r_{ii}}$, and G the cumulative distribution function of $N(0, \frac{\sigma^2}{1+2\sigma^2\alpha})$. Then (23) gives

$$
\begin{aligned}
&\Pr(\hat{x}_i = x_i^* | \hat{x}_j = x_j^* \, \forall j > i) \\
&= \frac{2}{\sqrt{1+2\sigma^2\alpha}} \int_0^\infty \frac{g(\tilde{v}_i)}{h(u_i - \lfloor u_i \rfloor, \alpha r_{ii}^2)} d\tilde{v}_i \\
&= \frac{2}{\sqrt{1+2\sigma^2\alpha}} \sum_{k=0}^{\infty} \int_k^{k+1} \frac{g(\tilde{v}_i)}{h(u_i - k, \alpha r_{ii}^2)} d\tilde{v}_i \\
&= \frac{2}{\sqrt{1+2\sigma^2\alpha}} \sum_{k=0}^{\infty} \sum_{j=1}^{N} \int_{k+\frac{j-1}{N}}^{k+\frac{j}{N}} \frac{g(\tilde{v}_i)}{h(u_i - k, \alpha r_{nn}^2)} d\tilde{v}_i \\
&\approx \frac{2}{\sqrt{1+2\sigma^2\alpha}} \sum_{k=0}^{\infty} \sum_{j=1}^{N} \frac{1}{h(\frac{2j-1}{2N}, \alpha r_{nn}^2)} \int_{k+\frac{j-1}{N}}^{k+\frac{j}{N}} g(\tilde{v}_n) d\tilde{v}_n \quad \text{by midpoint rule} \\
&= \frac{2}{\sqrt{1+2\sigma^2\alpha}} \sum_{k=0}^{\infty} \sum_{j=1}^{N} \frac{1}{h(\frac{2j-1}{2N}, \alpha r_{nn}^2)} \left[G\left(k + \frac{j}{N}\right) - G\left(k + \frac{j-1}{N}\right) \right]
\end{aligned}
\tag{25}
$$

And the expression of $\Pr(\hat{x}_i = x_i^* | \hat{x}_j = x_j^*, \ j = i+1, ..., n)$ by the expression of $\Pr(\hat{x}_n = x_n^*)$ with proper index change:

$$
\Pr(\hat{x}_i = x_i^* | \hat{x}_j = x_j^* \, \forall j > i) \approx \frac{2}{\sqrt{1+2\sigma^2\alpha}} \sum_{k=0}^{\infty} \sum_{j=1}^{N} \frac{1}{h(\frac{2j-1}{2N}, \alpha r_{ii}^2)} \left[G\left(k + \frac{j}{N}\right) - G\left(k + \frac{j-1}{N}\right) \right] \tag{26}
$$

By (16), the success probability is given by

$$
\Pr(\hat{\mathbf{x}} = \mathbf{x}^*) = \left(\frac{2}{\sqrt{1+2\sigma^2\alpha}} \right)^n \prod_{i=1}^{n} \sum_{k=0}^{\infty} \sum_{j=1}^{N} \frac{1}{h(\frac{2j-1}{2N}, \alpha r_{ii}^2)} \left[G\left(k + \frac{j}{N}\right) - G\left(k + \frac{j-1}{N}\right) \right] \tag{27}
$$

5 Extension to Box-Constrained ILS Problems

In some applications, the integer parameter vector $\mathbf{x}^*$ in the linear model (1) is random and is uniformly distributed over in box $\mathcal{B} = \mathcal{B}_1 \times \mathcal{B}_2 \times ... \times \mathcal{B}_n$, where $\mathcal{B}_i = [l_i, u_i]$ for all $i = 1, ..., n$. Specifically, the box-constrained integer least squares (BILS) problem is given by

$$
\min_{\mathbf{x} \in \mathcal{B}} ||\mathbf{y} - \mathbf{A}\mathbf{x}||_2^2 \tag{28}
$$

in which we generally assume that $\mathbf{x}^*$ follows a multivariate uniform distribution inside the box $\mathcal{B}$.

Like the ordinary case, the BILS problem we can do the same transformation on the problem to

$$
\min_{\mathbf{x} \in \mathcal{B}} ||\tilde{\mathbf{y}} - \mathbf{R}\mathbf{x}||_2^2 \tag{29}
$$

where $\tilde{\mathbf{y}}, \mathbf{R}$ the same as before. The box $\mathcal{B}$ remains unchanged.

Therefore we can do some slight modifications on the original Klein's algorithm such that it works for box-constrained ILS problems as well (See Algorithm 4 and 5).

Algorithm 4 Randomised Rounding with Box Constraint

```
Input: $c \in \mathbb{R}$, parameter $\beta > 0$, lower bound $l \in \mathbb{Z}$ and upper bound $u \in \mathbb{Z}$ (with $l \leq u$)
Output: $\hat{x} = \text{randRoundBox}(c, \beta, l, u)$
1: $s = \sum_{j=l}^{u} e^{-\beta(c-j)^2}$
2: for $k = l : u$ do
3:     $\Pr(\hat{x} = k) = \frac{e^{-\beta(c-k)^2}}{s}$
4: end for
5: output one integer $\hat{x}$ randomly according to the distribution:
   $\Pr(\hat{x} = k) = \frac{e^{-\beta(c-k)^2}}{s}$
```

Algorithm 5 Klein's Randomised Nearest Plane Algorithm with Box Constraint

```
Input: $\tilde{\mathbf{y}} \in \mathbb{R}^n$, non-singular upper-triangular matrix $\mathbf{R} \in \mathbb{R}^{m \times n}$, parameter $\alpha > 0$, lower bound vector $\mathbf{l} = [l_1, ..., l_n]^T$ and upper bound vector $\mathbf{u} = [u_1, ..., u_n]^T$
Output: $\hat{\mathbf{x}} = \text{randBabaiBox}(\mathbf{R}, \tilde{\mathbf{y}}, \alpha, \mathbf{l}, \mathbf{u})$
1: for $i = n : -1 : 1$ do
2:     $c_i = (\tilde{y}_i - \sum_{j=i+1}^{n} r_{ij}\hat{x}_j)/r_{ii}$
3:     $\hat{x}_i = \text{randRoundBox}(c_i, \alpha r_{ii}^2, l_i, u_i)$
4: end for
```

As in the unconstrained case, let g be the probability density function of Gaussian distribution $N(0, \frac{\sigma^2}{1+2\sigma^2\alpha})$.

Using similar procedure we can derive the success probability of this box-constrained Klein's algorithm at the i^{th} level, given by

$$\Pr(\hat{x}_i = x_i^* | (x_i^*, \hat{x}_j = x_j^*,\ j = i+1, \ldots, n)) = \frac{1}{\sqrt{1+2\sigma^2\alpha}} \int_{-\infty}^{\infty} \frac{g(\tilde{v}_i)}{\sum_{k=l_i}^{u_i} e^{-\alpha r_{ii}^2 (x_i^* + \frac{\tilde{v}_i}{r_{ii}} - k)^2}} d\tilde{v}_i \quad (30)$$

Since the true parameter vector follows uniform distribution over the box, we have $\Pr(x_i^*) = \frac{1}{u_i - l_i + 1}$. Then

$$\begin{aligned} &\Pr(\hat{x}_i = x_i^* | \hat{x}_j = x_j^*, j = i+1, \ldots, n) \\ &= \sum_{x_i^* = l_i}^{u_i} \Pr(\hat{x}_i = x_i^* | (x_i^*, \hat{x}_j = x_j^*,\ j = i+1, \ldots, n)) \Pr(x_i^*) \\ &= \frac{1}{u_i - l_i + 1} \cdot \frac{1}{\sqrt{1+2\sigma^2\alpha}} \sum_{x_i^* = l_i}^{u_i} \int_{-\infty}^{\infty} \frac{g(\tilde{v}_i)}{\sum_{k=l_i}^{u_i} e^{-\alpha r_{ii}^2 (x_i^* + \frac{\tilde{v}_i}{r_{ii}} - k)^2}} d\tilde{v}_i \end{aligned} \quad (31)$$

Therefore the success probability is given by

$$\Pr(\hat{\mathbf{x}} = \mathbf{x}^*) = (1 + 2\sigma^2\alpha)^{-\frac{n}{2}} \prod_{i=1}^{n} \frac{1}{u_i - l_i + 1} \sum_{x_i^*=l_i}^{u_i} \int_{-\infty}^{\infty} \frac{g(\tilde{v}_i)}{\sum_{k=l_i}^{u_i} e^{-\alpha r_{ii}^2 (x_i^* + \frac{\tilde{v}_i}{r_{ii}} - k)^2}} d\tilde{v}_i \tag{32}$$

The integration can be computed by a numerical integration method.

6 Numerical Experiments

In this section we will present some numerical experimental results to investigate the performance and behaviour of Klein's algorithm, as well as to verify some of the theoretical results we presented in the previous sections.

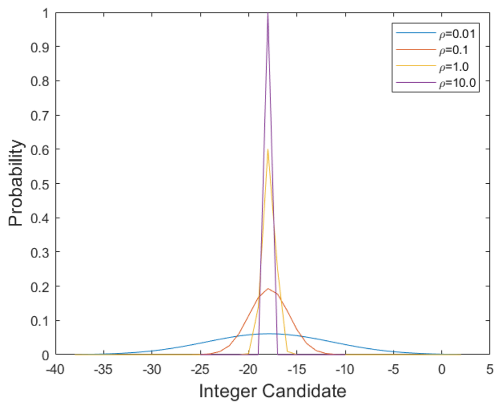
If there is no specific mentioning, experiments are conducted using MATLAB with model matrix $\mathbf{A}$ = randn(50,40), where randn(m, n) is a MATLAB built-in function which generates a random $m \times n$ matrix, whose entries follow the standard normal distribution. For each experiment we generate a new model matrix A and fix it, then we give 2000 runs, in each run we generate a new integer parameter vector $\mathbf{x}^* \in \mathbb{Z}^{40}$ and a new noise vector $\mathbf{v} \in \mathbb{R}^{50}$. Each parameter vector is generated using MATLAB randsample(100,40,true) which generates a integer vector with 40 entries, each entry sampled from [100] with discrete uniform distribution, with replacement. Each noise vector is generated using MATLAB normrnd(0,σ,[40,1]) which generates a 40×1 vector with mean 0 and standard deviation σ. Success likelihood represents the ratio of number of success runs and 2000, in which if the output integer vector is exactly the same as the true parameter vector, it counts as a success run.

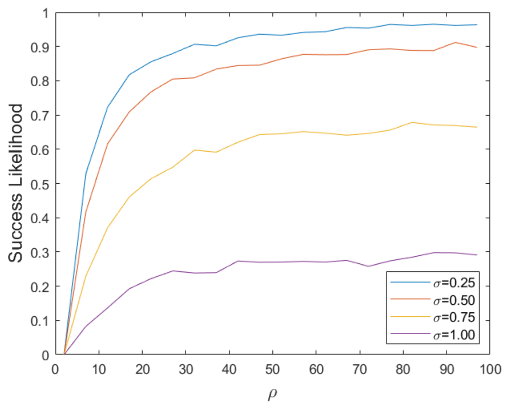
6.1 Varying the Parameter α

In Section 3 we proved that as the parameter $\alpha \to \infty$, Klein's algorithm outputs Babai point with probability 1; while when $\alpha \to 0$, the randomised rounding algorithm samples integer following uniform distribution in $\mathbb{Z}$. Intuitively, we can see this as that increasing α lowers the variance of integer vector sampling.

Figure 1 (a) shows the probability distribution in 1-dimensional case. As the value of α increases, the distribution becomes denser around the input, therefore the algorithm is more likely to output integers that are near to the input value. In the experiment we choose $\rho \in \{0.01, 0.1, 1, 10\}$, ranging from very small value (close to 0), whose distribution has a large variance, to large value, whose distribution is very dense around the centre.

When the noise vector $\mathbf{v}$ has small variance, i.e. when σ is small, Babai point has decent success probability. Therefore when knowing σ is small, increasing α can increase the success probability of Klein's algorithm. Figure 1 (b) shows the effect of α on the experimental success likelihood of Klein's algorithm.





(a) Probability distribution for 1D sampling with $\alpha = \frac{\rho}{\min_{1\le i\le n} r_{ii}^2}$, with input value $c = -17.8746$

(b) Success likelihood with different ρ for $\alpha = \frac{\log(\rho)}{\min_{1\le i\le n} r_{ii}^2}$.

Figure 1: Effect of varying the parameter α

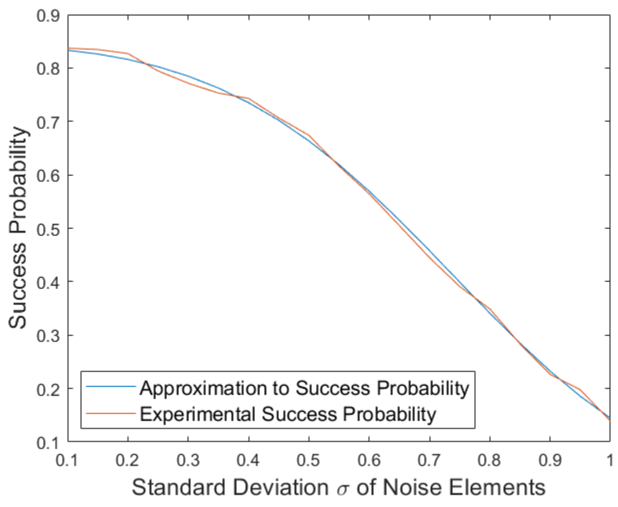


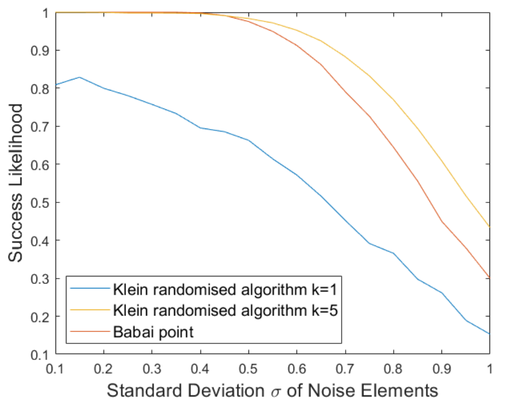
Figure 2: Experimental success probability and theoretical success probability estimate with $N = 1000$.

6.2 Success Probability of the Randomised Babai Point

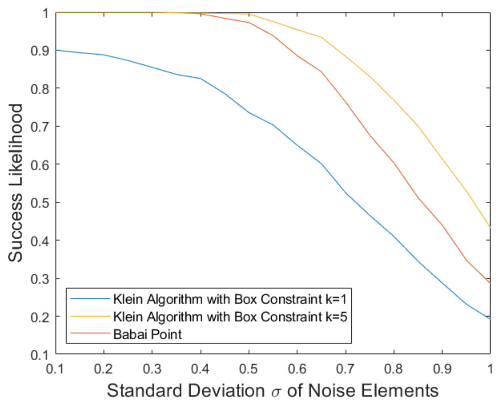
In Section 4 we derived a formula for the success probability of the randomized Babai point obtained Klein's algorithm as well as its numerical estimation. To verify how good this estimation is, we have conducted numerical experiment, whose result is shown in Figure 2. Setting large N makes the estimate which uses midpoint rule more accurate, while it increases the computational complexity. Here we set $N = 1000$.

6.3 Repeating Klein's Algorithm

The reason why we choose Klein's algorithm for ILS problems instead of Babai's point is because of the extra degree of randomness on the rounding process. Because of this we can run Klein's algorithm multiple times and take the "best" output as our final output. In this experiment we let the best point be the sampled integer vector whose resultant lattice point is closest to the measurement vector among all sampled ones.



(a) Without box constraint



(b) With box constraint $\mathcal{B} = [1,10]^{40}$

Figure 3: Success likelihood of running Klein's algorithm k times compared with that of Babai point.

Figure 3(a) shows the success likelihood improvement on running Klein's algorithm 5 times compared to running it once, along with Babai point success likelihood shown as well. Although running Klein's algorithm once has lower success probability than Babai point, repeating the algorithm 5 times the best point has more chance to be the true parameter vector.

However it is also worth noting that, repeating Klein's algorithm also takes more running time, which could even be longer than that of the sphere decoding algorithm when σ is small, while sphere decoding tends to have better success probability.

For box-constrained case, we set the box $\mathcal{B}$, and generate real parameter vector $\mathbf{x}^* \in \mathbb{Z}^{40}$ randomly with each entry sampled within the box following discrete uniform distribution. Figure 3(b) shows the success likelihood comparisons. Similar to unconstrained case, running Klein's algorithm 5 times makes the output more likely to succeed compared to Babai point.

7 Conclusion

In this project we investigated how Klein's randomised algorithm behaves under different parameters and noise variances. We have shown some theoretical results, including the effect of parameter changing and success probability derivation, as well as some experimental observations on the algorithm.

For future investigation we could focus more on the theoretical aspects of box-constrained Klein's algorithm, such as success probability compared to unconstrained Klein's algorithm. Since Klein's algorithm introduces one more degree of randomness compared to Babai point, to make the best use of it we would like to conduct more theoretical study on the repeated Klein's algorithm, including the derivation of the success probability with k repetitions.

References

[1] László Babai. On lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.

[2] X. Chang, J. Wen, and X. Xie. Effects of the lll reduction on the success probability of the babai point and on the complexity of sphere decoding. *IEEE Transactions on Information Theory*, 59(8):4915–4926, Aug 2013.

[3] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, pages 197–206, New York, NY, USA, 2008. ACM.

[4] Philip Klein. Finding the closest lattice vector when it's unusually close. In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '00, pages 937–941, Philadelphia, PA, USA, 2000. Society for Industrial and Applied Mathematics.

[5] D. Micciancio. The hardness of the closest vector problem with preprocessing. *IEEE Transactions on Information Theory*, 47(3):1212–1215, March 2001.

[6] C. P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical Programming*, 66(1):181–199, Aug 1994.

[7] J. Wen and X. Chang. Success probability of the babai estimators for box-constrained integer linear models. *IEEE Transactions on Information Theory*, 63(1):631–648, Jan 2017.