A Hybrid Method for Lattice Basis Reduction

by

Zhaofei Tian and Sanzheng Qiao

CAS Report Series **CAS-14-01-SQ**

Department of Computing and Software **Jan. 2014**
Information Technology Building
McMaster University
1280 Main Street West, Hamilton, Ontario, Canada L8S 4K1

A Hybrid Method for Lattice Basis Reduction

Zhaofei Tian and Sanzheng Qiao
Department of Computing and Software, McMaster University
Hamilton, Ontario, L8S 4K1, Canada

Jan. 2014

Abstract

Lattice reduction has a wide range of applications. In this paper, we first present a polynomial time Jacobi method for lattice basis reduction by modifying the condition for the Lagrange reduction and integrating the size reduction into the algorithm. We show that the complexity of the modified Jacobi algorithm is $O(n^5 \log B)$, where n is the dimension of the lattice and B is the maximum length of the input lattice basis vectors. To improve the quality of the computed bases, we then enhance our method by including a postprocessing without compromising the complexity. Our experiments show that our hybrid algorithm computes better reduced bases than the well-known LLL algorithm in terms of both the orthogonality defect and the condition number of the basis matrix. Moreover, although our algorithm has higher complexity than the LLL algorithm, it runs faster for problems of sizes under 90.

Keywords LLL, Gaussian reduction, lattice reduction, shortest vector problem, integer least squares

1 Introduction

Lattice basis reduction has been applied to a wide range of applications, such as cryptography, signal processing, integer linear programming, and number theory [2, 9, 14, 17, 22, 27]. In signal processing applications, lattice reduction is used for solving problems including global positioning systems (GPS), frequency estimation, multi-input multi-output systems, and data decoding systems [7, 24, 28, 29]. In cryptography, lattice reduction is used in RSA, GGH, NTRUEncrypt and many other public-key cryptography systems [1, 4, 6].

There are different notions of lattice basis reduction, such as Hermite-reduction, the Hermite-Korkine-Zolotareff (HKZ) reduction, the Minkowski-reduction, and the Lenstra-Lenstra-Lovász (LLL) reduction [12, 11, 18]. The algorithms for the HKZ-reduction or Minkowski-reduction are nonpolynomial [3, 10, 8, 11, 30]. In 1982, A. Lenstra, W. Lenstra, and L. Lovász presented a lattice reduction algorithm of complexity $O(n^4 \log B)$, where n is the dimension of the lattice and B is the maximum length of the basis vectors to be reduced [13]. This algorithm has been widely used and known as the LLL algorithm, because it has low complexity and produces good results in practice. All of the above notions of lattice reduction are based on reducing the lengths of lattice basis vectors. In 2012, S. Qiao proposed a generic Jacobi method for lattice basis reduction, which is based on improving the orthogonality between the basis vectors. Although its complexity is still unknown, experimentally it runs faster than the LLL algorithm.

Then in 2012, an $O(n^4 \log B)$ quasi-Jacobi method was proposed [23]. Its complexity is the same as that of the LLL algorithm, however, the condition number of the basis matrix produced by the algorithm is not as small as those produced by the LLL algorithm. Recently, an enhanced Jacobi method was presented [24]. It computes better reduced basis than the LLL algorithm in terms of both orthogonality defect and condition number. However, its convergence is unproven.

In this paper, we present an $O(n^5 \log B)$ hybrid method for lattice basis reduction. To guarantee the convergence, we modify the condition for the Lagrange reduction used in the generic Jacobi method. To improve the condition number of the computed lattice basis matrix, we integrate the size reduction into the algorithm. To further enhance our algorithm, especially the condition number of the computed basis matrix, we include a postprocessing. Our experiments have shown that our hybrid method produces better reduced bases than the well-known LLL algorithm, measured by both the orthogonality defect and the condition number of the basis matrices. Moreover, our algorithm requires less cpu time than the LLL algorithm for lattices of dimensions under 90.

The rest of the paper is organized as follows. In the next section, after reviewing some basic concepts in lattice theory, we describe the Lagrange reduction algorithm and the generic Jacobi reduction algorithm. In Section 3, we give a modified Jacobi method by modifying the condition for the Lagrange reduction and integrating the size reduction. Also, we give a complexity analysis. In Section 4, we present a hybrid method along with its complexity analysis. Our experimental results are demonstrated in Section 5. Finally, Section 6 concludes the paper.

Notations: Matrices and column vectors are denoted respectively by uppercase and lowercase boldface letters, the determinant and transpose of a square matrix $\mathbf{A}$ are denoted by $\det(\mathbf{A})$ and $\mathbf{A}^T$, respectively. The length of a vector $\mathbf{v}$ is measured by the Euclidean norm $\|\mathbf{v}\|_2$. We use $\mathbf{A}(a:b, c:d)$ to denote a submatrix of $\mathbf{A}$ with elements from rows a to b and from columns c to d. The symbol ":" in subscript denotes a complete row or column of a matrix. $\mathbf{I}_n$ denotes the identity matrix of order n.

2 Preliminaries

In this section, we briefly describe some basic concepts of lattice theory, and review the Lagrange reduction algorithm and the generic Jacobi method for lattice basis reduction.

2.1 Basic Concepts of Lattice Theory

Suppose that $\mathbf{A}$ is an $m \times n$ $(m \geq n)$ real matrix of full column rank, then the *lattice* generated by $\mathbf{A}$ is defined by the set:

$$L(\mathbf{A}) = \{\mathbf{Az} \mid \mathbf{z} \in \mathbb{Z}^n\},$$

where $\mathbb{Z}^n$ denotes the set of integer n-vectors. The columns of $\mathbf{A}$ form a *basis* for the lattice $L(\mathbf{A})$, and n is called the *dimension* of $L(\mathbf{A})$. The matrix $\mathbf{A}$ is called the generator matrix or basis matrix.

There are infinitely many bases for a lattice of dimension at least two [9]. If $\mathbf{A}'$ and $\mathbf{A}$ are two basis matrices for the same lattice, then they are related by $\mathbf{A}' = \mathbf{AZ}$, where $\mathbf{Z}$ is an integer matrix whose determinant $\det(\mathbf{Z}) = \pm 1$, called a *unimodular matrix*. The lattice reduction problem is to find a unimodular matrix

$\mathbf{Z}$ for a given lattice basis matrix $\mathbf{A}$ such that $\mathbf{AZ}$ is reduced depending on the notion of lattice reduction.

2.2 Lagrange Reduction Algorithm

We call a two dimensional lattice basis matrix $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2]$ *Lagrange-reduced*, if

$$|\mathbf{a}_1^T \mathbf{a}_2| \leq \frac{1}{2}\|\mathbf{a}_1\|_2^2 \quad \text{and} \quad \|\mathbf{a}_1\|_2 \leq \|\mathbf{a}_2\|_2. \tag{1}$$

Denoting the angle between $\mathbf{a}_1$ and $\mathbf{a}_2$ by θ. Then, we have $|\cos(\theta)| = |\mathbf{a}_1^T\mathbf{a}_2|/(\|\mathbf{a}_1\|_2\|\mathbf{a}_2\|_2) \leq |\mathbf{a}_1^T\mathbf{a}_2|/\|\mathbf{a}_1\|_2^2 \leq 1/2$, implying that $\theta \in [\pi/3,\ 2\pi/3]$. Thus, we may say that $\mathbf{a}_1$ and $\mathbf{a}_2$ are close to being orthogonal to each other. In dimension two, a Lagrange-reduced basis is Minkowski-reduced [4, 19].

Algorithm 1, the Lagrange reduction algorithm [25, 26], computes a Lagrange-reduced basis [19, 20, 23]. The notation $\lfloor\cdot\rceil$ in line 4 denotes the nearest integer rounding.

Algorithm 1: Lagrange Reduction Algorithm `Lagrange`$(\mathbf{a}_1, \mathbf{a}_2)$

```
Input  : Two basis vectors $\mathbf{a}_1, \mathbf{a}_2$
Output: Updated $\mathbf{a}_1, \mathbf{a}_2$, so that $[\mathbf{a}_1, \mathbf{a}_2]$ is Lagrange-reduced
1 if $\|\mathbf{a}_1\| < \|\mathbf{a}_2\|$ then
2     Swap $\mathbf{a}_1$ and $\mathbf{a}_2$ ;
3 repeat
4     $q = \lfloor \mathbf{a}_1^T\mathbf{a}_2/\|\mathbf{a}_2\|^2 \rceil$ ;
5     Set $\mathbf{Z} = \mathbf{I}_2$, except $z_{21} = -q$ ;
6     $[\mathbf{a}_1\ \mathbf{a}_2] \leftarrow [\mathbf{a}_1\ \mathbf{a}_2]\mathbf{Z}$ ;
7     Swap $\mathbf{a}_1$ and $\mathbf{a}_2$;
8 until $\|\mathbf{a}_1\| \leq \|\mathbf{a}_2\|$;
```

2.3 Generic Jacobi Method

A Jacobi method has a two dimensional workhorse. In the Jacobi method for the symmetric eigenvalue problem proposed by Jacobi in 1846, the workhorse is two dimensional eigenvalue decomposition [5]. In 2012, using the two dimensional Lagrange reduction, Qiao presented a generic Jacobi method for lattice basis reduction [20]. It repeatedly applies the Lagrange algorithm to every pair of the basis vectors for an n dimensional lattice. The generic Jacobi method produces a reduced basis defined as follows.

Definition 2.1 (Reduced)**.** A basis matrix $\mathbf{A} = [\mathbf{a}_1,\ \mathbf{a}_2, \ldots,\ \mathbf{a}_n]$ is *reduced*, if :

$$|\mathbf{a}_i^T \mathbf{a}_j| \leq \frac{1}{2}\|\mathbf{a}_i\|_2^2 \quad \text{(for all } 1 \leq i < j \leq n\text{);} \tag{2a}$$

$$\|\mathbf{a}_i\|_2 \leq \|\mathbf{a}_j\|_2 \quad \text{(for all } 1 \leq i < j \leq n\text{).} \tag{2b}$$

We can see that in a reduced basis as defined by Definition 2.1, every pair of basis vectors is Lagrange-reduced. Algorithm 2 shows the row-cyclic version of the generic Jacobi method.

Algorithm 2: Generic Jacobi Method

```
Input  : A basis matrix $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n]$
Output: The basis matrix $\mathbf{A}$ is overwritten by a reduced basis matrix defined by Definition 2.1
1 while not all pairs $(\mathbf{a}_i, \mathbf{a}_j)$ satisfy (2a) and (2b) do
2     for $i = 1$ to $n-1$ do
3         for $j = i+1$ to $n$ do
4             $[\mathbf{a}_i, \mathbf{a}_j] \leftarrow$ Lagrange($\mathbf{a}_i, \mathbf{a}_j$) ;
```

In [20], the gram matrix $\mathbf{G} = \mathbf{A}^T\mathbf{A}$ is used for efficiency. Note that, if $\mathbf{R}$ is the upper triangular matrix in the QR decomposition [5] of $\mathbf{A}$, then $\mathbf{G} = \mathbf{R}^T\mathbf{R}$. Also, $g_{ij} = \mathbf{a}_i^T\mathbf{a}_j = \mathbf{r}_i^T\mathbf{r}_j$ and $g_{jj} = \|\mathbf{a}_j\|^2 = \|\mathbf{r}_j\|^2$. Procedure `LagrangeIT`($\mathbf{G}, \mathbf{Z}, \mathbf{R}, i, j$) performs one iteration of the Lagrange reduction on the ith and jth basis vectors using $\mathbf{G}$ and the optional $\mathbf{R}$. Note that, different from the Lagrange reduction algorithm, in Procedure LagrangeIT the jth basis vector is always used to reduced the ith vector, where $j > i$. From the procedure, we can see that the output g_{jj} equals the input g_{jj} and the output g_{ii} is less than or equal to the input g_{jj}.

Procedure LagrangeIT($\mathbf{G}, \mathbf{Z}, \mathbf{R}, i, j$)

```
Input  : The matrices $\mathbf{G}, \mathbf{Z}$, optional $\mathbf{R}$ and a pair of indices $(i, j)$, $i < j$
Output: The input matrices are updated so that one Lagrange iteration is performed on the $i$th and $j$th
        $(i < j)$ basis vectors
1 $q = \lfloor g_{ij}/g_{jj} \rceil$ ;
2 Set $\mathbf{Z}_{ij} = I_n$ except $z_{ji} = -q$ ;
3 $\mathbf{G} \leftarrow \mathbf{Z}_{ij}^T \mathbf{G} \mathbf{Z}_{ij}$ ;
4 $\mathbf{Z} \leftarrow \mathbf{Z}\mathbf{Z}_{ij}$ ;
5 if $\mathbf{R}$ is present then
6     $\mathbf{R} \leftarrow \mathbf{R}\mathbf{Z}_{ij}$ ;
7     Restore the upper triangular structure of $\mathbf{R}$ ;
```

If $\mathbf{R}$ is not present, the procedure `LagrangeIT`($\mathbf{G}, \mathbf{Z}, i, j$) costs $O(n)$ operations (additions or multiplications), since it operates on two rows and two columns of $\mathbf{G}$ and two columns of $\mathbf{Z}$. If $\mathbf{R}$ is present, after the application of $\mathbf{Z}_{ij}$ to $\mathbf{R}$, the entries $r_{k,i}$, $k = i+1, ..., j$, become nonzero. The upper triangular structure of $\mathbf{R}$ can be restored by first eliminating $z_{j,i}, ..., z_{i+1,i}$, followed by eliminating $z_{i+2,i+1}, ..., z_{j,j-1}$ using the Givens rotations, as shown in Procedure `RestoreR`($\mathbf{R}, i, j$). Thus, restoring the upper triangular structure of $\mathbf{R}$ costs $O(n^2)$ operations. Thus, when the gram matrix $\mathbf{G}$ is used in the generic Jacobi method, line 4 in Algorithm 2 is replaced by the above Procedure `LagrangeIT` ($\mathbf{G}, \mathbf{Z}, i, j$), followed by swapping the ith and jth basis vectors.

Although experiments have shown that Algorithm 2 terminates within 10 sweeps for basis matrices of dimension less than 300, its complexity remains unknown.

Procedure RestoreR($\mathbf{R}, i, j$)

```
Input  : R and indices $i, j$ $(i < j)$
Output: Triangularized R
1 for $k = j$ downto $i + 1$ do
2     Find a Givens plane rotation $\mathbf{G}$ to eliminate $r_{k,i}$ using $r_{k-1,i}$ ;
3     $\mathbf{R} \leftarrow \mathbf{GR}$ ;
4 for $k = i + 1$ to $j - 1$ do
5     Find a Givens plane rotation $\mathbf{G}$ to eliminate $r_{k+1,k}$ using $r_{k,k}$ ;
6     $\mathbf{R} \leftarrow \mathbf{GR}$ ;
```

3 A Modified Jacobi Method

To ensure the termination of the generic Jacobi method, in this section, we modify the condition for the Lagrange reduction described in the previous section. Then, we present a conditional Jacobi method using Procedure LagrangeIT. Also, to improve the condition number of the reduced basis matrix produced by the algorithm, we integrate the size reduction into our conditional Jacobi method. We will show that the time complexity of our modified Jacobi method is $O(n^5 \log B)$, where n is the dimension of the input basis, and B is the maximum length of the input basis vectors.

3.1 Conditional Lagrange Reduction

Let $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2]$ be a basis matrix for a two-dimensional lattice $L(\mathbf{A})$, and $\mathbf{A}' = [\mathbf{a}'_1, \mathbf{a}'_2]$ be a reduced basis matrix for the same lattice produced by a lattice reduction algorithm. We then call the ratio $\|\mathbf{a}_1\|_2 \, \|\mathbf{a}_2\|_2 / (\|\mathbf{a}'_1\|_2 \, \|\mathbf{a}'_2\|_2)$ the *reduction factor* of the algorithm. Consider the Lagrange algorithm. It has been proven that the reduction factor of one Lagrange iteration is at least $\sqrt{3}$, except that in the first or the last iteration the reduction factor can be arbitrarily close to 1 [19, 23]. To ensure that the reduction factor of every Lagrange iteration is strictly larger than 1, that is, the basis vector length is strictly reduced, we modify the Lagrange-reduced definition (1) by introducing a condition with a parameter ω. We say that a two dimensional lattice basis matrix $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2]$ is *ω-Lagrange-reduced*, if

$$|\mathbf{a}_1^T \mathbf{a}_2| \le \frac{1}{2} \|\mathbf{a}_2\|_2^2, \tag{3a}$$

or

$$|\mathbf{a}_1^T \mathbf{a}_2| > \frac{1}{2} \|\mathbf{a}_2\|_2^2 \quad \text{and} \quad \|\mathbf{a}_1\|_2^2 < \omega^2 \|\mathbf{a}_2\|_2^2 \tag{3b}$$

where $1 < \omega \le \sqrt{3}$.

Since the reduction factors of the Lagrange iterations other than the first and the last are at least $\sqrt{3}$, to ensure the strict length reduction, we only need to inspect the first and the last iterations. For these two iterations, the only possible values of the integral scalar q on line 4 of Algorithm 1 are ± 1. Substituting $q = \pm 1$ into line 5, we can see from line 6 that the length of the newly computed vector $\mathbf{a}_1 - q\mathbf{a}_2$ is not longer than $\|\mathbf{a}_2\|_2$. Then, condition (3b) guarantees that the reduction factors of the first and last iterations are at least ω.

In general, we say that an n-dimensional basis matrix $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n]$ is ω-*reduced*, if

$$|\mathbf{a}_i^T \mathbf{a}_j| \leq \frac{1}{2} \|\mathbf{a}_j\|_2^2, \tag{4a}$$

or

$$|\mathbf{a}_i^T \mathbf{a}_j| > \frac{1}{2} \|\mathbf{a}_j\|_2^2 \quad \text{and} \quad \|\mathbf{a}_i\|_2^2 < \omega^2 \|\mathbf{a}_j\|_2^2, \tag{4b}$$

for all $1 \leq i < j \leq n$ and $1 < \omega \leq \sqrt{3}$. In terms of the gram matrix $\mathbf{G}$, the above inequalities are equivalent to

$$|g_{ij}| \leq \frac{1}{2} g_{jj}, \tag{5a}$$

or

$$|g_{ij}| > \frac{1}{2} g_{jj} \quad \text{and} \quad g_{ii} < \omega^2 g_{jj}, \tag{5b}$$

for all $1 \leq i < j \leq n$ and $1 < \omega \leq \sqrt{3}$. Algorithm 3 depicts the conditional Jacobi method. The condition on line 5 ensures that the reduction factor of each Lagrange iteration is at least ω. Since $1 < \omega \leq \sqrt{3}$ and the reduction factor of any iteration other than the first and last is at least $\sqrt{3}$, the condition on line 5 affects only the first and the last iterations. The generic Jacobi method Algorithm 2 is a special case of Algorithm 3 when $\omega = 1$.

Algorithm 3: Conditional Jacobi method

```
Input  : A basis matrix $\mathbf{A}$ and $\omega$ $(1 < \omega \leq \sqrt{3})$
Output: An $\omega$-reduced basis matrix $\mathbf{A}$
1 $\mathbf{G} = \mathbf{A}^T\mathbf{A}$, $\mathbf{Z} = \mathbf{I}_n$ ;
2 while not all elements $g_{ij}$ satisfy (5a) and (5b) do
3     for $i = 1$ to $n-1$ do
4         for $j = i+1$ to $n$ do
5             if $g_{ij}$ doesn't satisfy (5a) and (5b) then
6                 $[\mathbf{G}, \mathbf{Z}] \leftarrow$ LagrangeIT($\mathbf{G}, \mathbf{Z}, i, j$) ;
7                 Swap the $i$th and $j$th basis vectors ;
8 $\mathbf{A} = \mathbf{A}\mathbf{Z}$ ;
```

Denote $D = \prod_{i=1}^n g_{ii} = \prod_{i=1}^n \|\mathbf{a}_i\|^2$. Then, D has a lower bound $\prod_{i=1}^n (\lambda_1^2)$, where λ_1 is the first Minkowski minima of the lattice $L(\mathbf{A})$ [8, 18], which is the length of a shortest nonzero lattice vector. Suppose that B is the maximum length of the input basis vectors, then B^{2n} is an upper bound for D. Each iteration on line 6 of Algorithm 3 reduces g_{ii} for some i by a factor of at least $\omega^2 > 1$, while keeping g_{kk}, $k \neq i$, unchanged. Thus, each iteration reduces D by a factor of at least $\omega^2 > 1$. Thus the algorithm terminates after performing the Lagrange iteration at most $O(n \log B)$ times. Since the complexity of `LagrangeIT`($\mathbf{G}, \mathbf{Z}, i, j$) when $\mathbf{R}$ is not present is $O(n)$, the complexity of Algorithm 3 is $O(n^4 \log B)$, the same as the LLL algorithm. When the algorithm terminates, the computed basis is ω-reduced defined by (4a) and (4b).

Procedure PSizeReduce($\mathbf{G}, \mathbf{Z}, \mathbf{R}, i, j$)

```
Input  : $\mathbf{G}, \mathbf{Z}, \mathbf{R}$ and indices $i, j$, $i < j$
Output: Updated $\mathbf{G}, \mathbf{Z}, \mathbf{R}$, s. t. $\mathbf{AZ}$ is partially size reduced w.r.t. $(i, j)$
1 $\mathbf{Z}_{ij} = \mathbf{I}_n$ ;
2 for $k \leftarrow i$ downto 1 do
3     if $|r_{kj}| > |r_{kk}|/2$ then
4         $q = \lfloor r_{kj}/r_{kk} \rceil$ ;
5         Set $\mathbf{Z}_s = I_n$ except $z_{kj} = -q$ ;
6         $\mathbf{R} \leftarrow \mathbf{RZ}_s$, $\mathbf{Z}_{ij} \leftarrow \mathbf{Z}_{ij}\mathbf{Z}_s$ ;
7 $\mathbf{Z} \leftarrow \mathbf{ZZ}_{ij}$ ;
8 $\mathbf{G} \leftarrow \mathbf{Z}_{ij}^T \mathbf{G} \mathbf{Z}_{ij}$ ;
```

3.2 Size Reduction

The generic Jacobi method, which is based on the Lagrange algorithm, brings basis vectors closer to being orthogonal to each other. However, it is ineffective in reducing the lengths of the basis vectors, especially the condition number of the basis matrix. The size reduction [28], which is required by most lattice reductions, effectively shortens the basis vectors and improves the condition number of a basis matrix.

Let $\mathbf{A}$ be an n dimensional basis matrix and $\mathbf{A} = \mathbf{QR}$ be its QR decomposition [5], then $\mathbf{A}$ is called *size-reduced*, if the upper triangular matrix $\mathbf{R}$ satisfies

$$|r_{i,j}| \le \frac{1}{2}|r_{i,i}| \quad \text{(for all } 1 \le i < j \le n\text{)}. \tag{6}$$

For example, consider the basis matrix [21],

$$\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3] = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ 0 & 0 & \frac{1}{2} \end{bmatrix}.$$

It can be verified that $\mathbf{A}$ satisfies Definition 2.1 and its condition number is approximately 4.7387. After the size reduction, $\mathbf{a}_3$ is shortened to $[0\ 0\ 1/2]^T$ and the condition number of the basis matrix is improved to approximately 2.4495.

To improve the condition number of the basis matrix produced by the conditional Jacobi method, we integrate the size reduction into Algorithm 3. Since each Lagrange iteration modifies only one column of the basis matrix, we introduce a notion of partial size reduction, which is a generalization of the size reduction defined in (6). A basis matrix $\mathbf{A}$ is said to be *partially size reduced* with respect to an index pair (i, j), $i < j$, if

$$|r_{k,j}| \le \frac{1}{2}|r_{k,k}| \quad \text{(for } 1 \le k \le i\text{)}. \tag{7}$$

Thus, if $\mathbf{A}$ is partially size reduced with respect to $(i, i+1)$ for all i, $1 \le i < n$, then $\mathbf{A}$ is size reduced. The following procedure `PSizeReduce(`$\mathbf{G}, \mathbf{Z}, \mathbf{R}, i, j$`)` applies the partial size reduction with respect to (i, j) to the basis matrix associated with $\mathbf{R}$.

Algorithm 4: Modified Jacobi method

```
Input   : A basis matrix $\mathbf{A}$ and $\omega$ $(1 < \omega \le \sqrt{3})$
Output: A updated reduced basis matrix $\mathbf{A}$
1  $\mathbf{G} = \mathbf{A}^T\mathbf{A}$,  $\mathbf{Z} = \mathbf{I}_n$,  get $\mathbf{R}$ from the QR decomposition of $\mathbf{A}$ ;
2  while not all elements $g_{ij}$ satisfy (5a) and (5b) do
3      for $i = 1$ to $n-1$ do
4          for $j = i+1$ to $n$ do
5              if $g_{ij}$ doesn't satisfy (5a) and (5b) then
6                  $[\mathbf{G}, \mathbf{Z}, \mathbf{R}] \leftarrow$ LagrangeIT($\mathbf{G}, \mathbf{Z}, \mathbf{R}, i, j$) ;
7                  RestoreR($\mathbf{R}, i, j$) ;
8              Apply PSizeReduce($\mathbf{G}, \mathbf{Z}, \mathbf{R}, i, j$) only if $g_{jj}$ is reduced after the appplication ;
9          Find an index $k$ $(i \le k \le n)$, s.t. $g_{kk} = \min_{l=i}^{n} g_{ll}$ ;
10         if $k \ne i$ then
11             Swap the $i$th and $k$th basis vectors ;
12             RestoreR($\mathbf{R}, i, k$) ;
13 $\mathbf{A} = \mathbf{A}\mathbf{Z}$ ;
```

The procedure PSizeReduce($\mathbf{G}, \mathbf{Z}, \mathbf{R}, i, j$) costs $O(n^2)$ operations.

3.3 Modified Jacobi Method

The size reduction, however, does not always reduce basis vector lengths. Thus, in Algorithm 4, which is an integration of Algorithm 3 and the partial size reduction, we introduce an condition for the partial size reduction to prevent the basis vector lengths from increasing. Specifically, after the application of the procedure PSizeReduce($\mathbf{G}, \mathbf{Z}, \mathbf{R}, i, j$), if g_{jj} is not reduced, we roll back the procedure. Moreover, to make the length reduction of basis vectors more effective, at the end of each inner j-loop, we push the shorter basis vector to the front.

Since the cost of PSizeReduce($\mathbf{G}, \mathbf{Z}, \mathbf{R}, i, j$) or restoring the structure of $\mathbf{R}$ is $O(n^2)$ (the same as that of LagrangeIT($\mathbf{G}, \mathbf{Z}, \mathbf{R}, i, j$) when $\mathbf{R}$ is present), the cost of the modified Jacobi method is $O(n^5 \log B)$.

4 A Hybrid Method

To further improve Algorithm 4, we propose an enhancement for the algorithm by including a post-processing without compromising its complexity.

As shown in Algorithm 5, there are two parts in postprocessing, lines 2 to 11, and lines 12 to 15. Unlike the main `while` loop in Algorithm 4, the partial size reduction on line 8 in the first part of postprocessing is unconditional. In the second part of postprocessing, both Lagrange iteration and the partial size reduction are unconditional.

For the time complexity of Algorithm 5, we give the following fact:

Algorithm 5: Hybrid Method

```
Input  : A basis matrix $\mathbf{A}$ and $\omega$ $(1 < \omega \le \sqrt{3})$
Output: An updated reduced basis matrix $\mathbf{A}$
Run the modified Jacobi method and keep the matrices $\mathbf{G}$, $\mathbf{Z}$ and $\mathbf{R}$ ;
for $i = 1$ to $n-1$ do
    for $j = i+1$ to $n$ do
        if (5a) and (5b) are not satisfied then
            $[\mathbf{G},\mathbf{Z},\mathbf{R}] \leftarrow$ LagrangeIT($\mathbf{G},\mathbf{Z},\mathbf{R},i,j$) ;
            $\mathbf{R} =$ RestoreR($\mathbf{R},i,k$) ;
        $[\mathbf{G},\mathbf{Z},\mathbf{R}] \leftarrow$PSizeReduce($\mathbf{G},\mathbf{Z},\mathbf{R},i,j$) ;
    Find an index $k$ $(i \le k \le n)$, s.t. $g_{kk} = \min_{l=i}^{n} g_{ll}$ ;
    if $k \ne i$ then
        Swap the $i$th and $k$th basis vectors ;
        $\mathbf{R} =$ RestoreR($\mathbf{R},i,k$) ;
for $i = 1$ to $n-1$ do
    for $j = i+1$ to $n$ do
        $[\mathbf{G},\mathbf{Z},\mathbf{R}] \leftarrow$LagrangeIT($\mathbf{G},\mathbf{Z},\mathbf{R},i,j$) ;
        $[\mathbf{G},\mathbf{Z},\mathbf{R}] \leftarrow$PSizeReduce($\mathbf{G},\mathbf{Z},\mathbf{R},i,j$) ;
$\mathbf{A} = \mathbf{AZ}$ ;
```

Fact 4.1. *Let* $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n]$ *be a basis matrix of an n dimensional lattice* $L(\mathbf{A})$*, and let* $B = \max_{1\le i\le n} \|\mathbf{a}_i\|_2$. *Then, the number of arithmetic operations needed by Algorithm 5, the hybrid method for lattice basis reduction, is* $O(n^5 \log B)$.

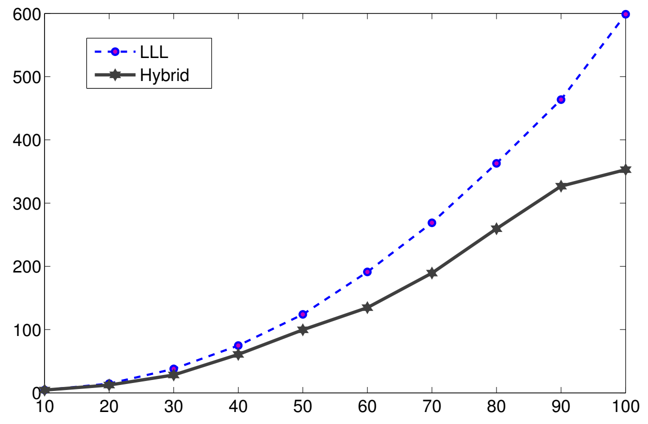
Since the costs of `PSizeReduce`$(\mathbf{G},\mathbf{Z},\mathbf{R},i,j)$, restoring the structure of $\mathbf{R}$, and `LagrangeIT`$(\mathbf{G},\mathbf{Z},\mathbf{R},i,j)$ are both $O(n^2)$, the cost of the postprocessing is $O(n^4)$. Hence, adding a postprocessing does not compromise the complexity of our algorithm. The time complexity of the hybrid method is $O(n^5 \log B)$.

5 Experimental Results

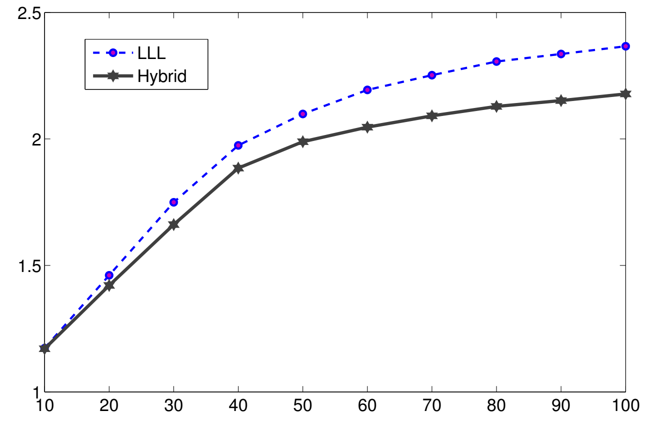
In this section, we present our experimental results on comparing the efficiency and output quality of the hybrid method with the LLL algorithm, a well-known and widely used polynomial time lattice reduction algorithm.

Our hybrid method and the LLL algorithm were implemented in `MATLAB`. We adopted the vector-operated version [15] of the LLL algorithm to achieve high efficiency, and set the parameter ω to 0.99 to get high quality outputs from the LLL algorithm. The reduction factor ω $(1 < \omega \le \sqrt{3})$ of the hybrid method was set to $\sqrt{2}$. We compared the basis matrices of dimensions up to 100, starting from dimension 10 with interval 10.

To get average results for each chosen dimension, we generated 1000 matrices with uniformly distributed random entries. The results shown in Figure 1 and Figure 2 are the measurements for the chosen dimen-



(a) Condition numbers



(b) Orthogonality defects

Figure 1: Comparison of quality measured by condition numbers and orthogonality defects

sions.

The quality of the computed basis matrices is measured by two measurements: the *condition number* and the *orthogonality defect* $\delta(\mathbf{A})$, defined by

$$\delta^n(\mathbf{A}) = \frac{\Pi_j ||\mathbf{a}_j||_2}{\sqrt{\det(\mathbf{A}^T\mathbf{A})}},$$

which is also called *Hadamard Ratio* [9]. From the *Hadamard's Inequality* [16], $\delta(\mathbf{A}) \geq 1$, where the equality holds if and only if the columns $\mathbf{a}_j$ are orthogonal to each other. The closer $\delta(\mathbf{A})$ is to 1, the shorter the geometric mean of the lengths of the columns is and the more orthogonal the columns in $\mathbf{A}$ are; hence the better the basis matrix $\mathbf{A}$ is.

The efficiency of the algorithms is measured by cpu time.

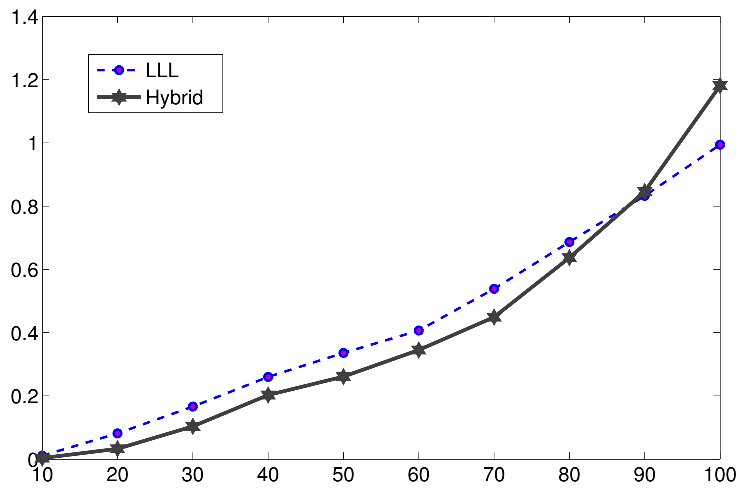


Figure 2: Comparison of efficiency (in seconds)

Figure 1 (a) and (b) illustrate the qualities of the computed bases by the two algorithms, shown by the two measurements: the condition number and the orthogonality defect. Both measurements are the smaller, the better. The figures show that the basis matrices computed by our hybrid method have better quality than those computed by the LLL algorithm, The difference becomes more significant as the dimension increases.

Figure 2 shows the cpu time (in seconds) of the two algorithms. Our hybrid method is faster than the LLL algorithm for the dimension under 90.

6 Conclusions

In this paper, we present a polynomial time hybrid lattice reduction algorithm. Firstly, by modifying the condition for the Lagrange reduction, a conditional Jacobi method is proposed. We show that its complexity is $O(n^4 \log B)$, where n is the dimension of the lattice and B is the maximum length of the input basis vectors. This complexity is the same as the well-known LLL algorithm. Then, we propose a modified Jacobi method by integrating the size reduction into the Lagrange reduction and show its complexity of $O(n^5 \log B)$. Finally, we present a hybrid method and show its complexity of $O(n^5 \log B)$.

Our experiments, comparing our hybrid method with the LLL algorithm, show that our algorithm consistently produces results with both better orthogonality and better condition number than the LLL algorithm. The differences between them grows as the problem size increases. Although the complexity of our hybrid method is higher than that of the LLL algorithm, our algorithm runs faster than the LLL algorithm for problems of sizes under 90. Jacobi method is inherently parallel, our future work includes parallel implementations of our algorithm and improvements on its complexity.

References

[1] Dan Boneh. Twenty years of attacks on the RSA cryptosystem. *Notices of the AMS*, 46:203–213, 1999.

[2] H. Cohen. *A Course in Computational Algebraic Number Theory*. Graduate Texts in Mathematics. Springer, 1993.

[3] Aharonov Dorit and Regev Oded. Lattice problems in NP and co-NP. *J. ACM*, 52:749–765, September 2005.

[4] Steven D. Galbraith. *Mathematics of public key cryptography*. Cambridge University Press, 2012.

[5] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 3rd edition, 1996.

[6] Guillaume Hanrot and Damien Stehlé. Improved analysis of Kannan's shortest lattice vector algorithm. In *Proceedings of the 27th annual international cryptology conference on Advances in cryptology*, CRYPTO'07, pages 170–186, Berlin, Heidelberg, 2007. Springer-Verlag.

[7] Babak Hassibi and Haris Vikalo. On the sphere-decoding algorithm I. expected complexity. *IEEE Trans. Sig. Proc*, pages 2806–2818, 2005.

[8] Bettina Helfrich. Algorithms to construct Minkowski reduced and Hermite reduced lattice bases. *Theoretical Computer Science*, 41:125 – 139, 1985.

[9] J. Hoffstein, J.C. Pipher, and J.H. Silverman. *An introduction to mathematical cryptography*. Undergraduate texts in mathematics. Springer, 2008.

[10] Ravi Kannan. Improved algorithms for integer programming and related lattice problems. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, STOC '83, pages 193–206, New York, NY, USA, 1983. ACM.

[11] Ravi Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12:415–440, August 1987.

[12] A. Korkine and G. Zolotareff. Sur les formes quadratiques. *Mathematische Annalen*, 6:366–389, 1873.

[13] A.K. Lenstra, Lenstra, and Lászlo Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261:515–534, 1982.

[14] H.W. Lenstra. *Integer programming with a fixed number of variables*. Report. Department of Mathematics. University of Amsterdam. Department, Univ., 1981.

[15] Franklin T. Luk and Daniel M. Tracy. An improved LLL algorithm. *Linear Algebra and its Applications*, 428(2-3):441 – 452, 2008.

[16] V. Mazya and T.O. Shaposhnikova. *Jacques Hadamard: A Universal Mathematician*. History of mathematics. American Mathematical Society, 1999.

[17] Daniele Micciancio and Shafi Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, Massachusetts, March 2002.

[18] H. Minkowski. Discontinuity region for arithmetical equivalence. *J. reine Angew*, (129):220–274, 1905.

[19] Phong Q. Nguyen and Damien Stehlé. Low-dimensional lattice basis reduction revisited. *ACM Trans. Algorithms*, 5(4):46:1–46:48, November 2009.

[20] Sanzheng Qiao. A Jacobi method for lattice basis reduction. In *Proceedings of 2012 Spring World Congress on Engineering and Technology (SCET2012)*, Vol.2. IEEE, pages 649–652, Xi'an China, May 2012.

[21] Igor A. Semaev. A 3-dimensional lattice reduction algorithm. In *CaLC*, pages 181–193, 2001.

[22] M. Taherzadeh, A. Mobasher, and A.K. Khandani. LLL reduction achieves the receive diversity in MIMO decoding. *Information Theory, IEEE Transactions on*, 53(12):4801–4805, Dec. 2007.

[23] Zhaofei Tian and Sanzheng Qiao. A complexity analysis of a Jacobi method for lattice basis reduction. In *Proceedings of the Fifth International C* Conference on Computer Science and Software Engineering*, C3S2E '12, pages 53–60, New York, NY, USA, 2012. ACM.

[24] Zhaofei Tian and Sanzheng Qiao. An enhanced Jacobi method for lattice-reduction-aided MIMO detection. In *Proceedings of IEEE China Summit and International Conference on Signal and Information Processing*, pages 39–43, Beijing, China, June 8-10 2013. IEEE.

[25] Brigitte Vallée and Antonio Vera. Lattice reduction in two dimensions: analyses under realistic probabilistic models. In *the Proceedings of AofA'07, Discrete Mathematics and Theoretical Computer Science*, 2007.

[26] Brigitte Vallée and Antonio Vera. Probabilistic analyses of lattice reduction algorithms. In *The LLL Algorithm*, Information Security and Cryptography, pages 71–143. Springer Berlin Heidelberg, 2010.

[27] D. Wubben, R. Bohnke, V. Kuhn, and K.-D. Kammeyer. Near-maximum-likelihood detection of MIMO systems using MMSE-based lattice reduction. In *Communications, 2004 IEEE International Conference on*, volume 2, pages 798 – 802 Vol.2, June 2004.

[28] D. Wübben, D. Seethaler, J. Jalden, and G. Matz. Lattice reduction: A survey with applications in wireless communications. *IEEE Signal Processing Magazine*, 28(3):70–91, May 2011.

[29] P. Xu, C. Shi, and J. Liu. Integer estimation methods for GPS ambiguity resolution: an applications oriented review and improvement. *Survey Review*, 44:59–71, Jan. 2012.

[30] Wen Zhang, Sanzheng Qiao, and Yimin Wei. HKZ and Minkowski reduction algorithms for Lattice-Reduction-Aided MIMO detection. *IEEE Transactions on Signal Processing*, 60(11):5963–5976, 2012.