Conditioning Properties of the LLL Algorithm

Franklin T. Luk[a] and Sanzheng Qiao[b]

[a]Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong
[b]Dept. of Computing and Software, McMaster Univ., Hamilton, Ontario L8S 4L7, Canada

ABSTRACT

Although the LLL algorithm[1] was originally developed for lattice basis reduction, the method can also be used[2] to reduce the condition number of a matrix. In this paper, we propose a pivoted LLL algorithm that further improves the conditioning. Our experimental results demonstrate that this pivoting scheme works well in practice.

Keywords: LLL algorithm, matrix conditioning, pivoting, unimodular matrix, lattice, reduced basis, Gauss transformation.

1. INTRODUCTION

The LLL algorithm, named after Lenstra, Lenstra, and Lovász[1], is a method useful for reducing a lattice basis. Recently, Luk and Tracy[2] propose a linear algebraic interpretation: Given a real nonsingular matrix A, the LLL algorithm computes a QRZ decomposition:

$$A = QRZ^{-1},$$

where the matrix Q is orthogonal, the matrix R is triangular and reduced (in the sense that its columns form a reduced basis as defined in Section 2), and the matrix Z is integer and unimodular. Since the publication of the LLL algorithm in 1982, many new applications (e.g., cryptography[3] and wireless communications[4,5]) have been found. The two related objectives of this paper are to show that the LLL algorithm can be used to reduce the condition number of a matrix and that a new pivoting scheme can further improve the conditioning.

The paper is organized as follows. In Section 2, we present the QRZ decomposition by Luk and Tracy[2] and illustrate it with a 2-by-2 matrix. In Section 3, we give a geometrical interpretation of the LLL algorithm and show how the algorithm reduces a lattice basis and improves the condition of a matrix. To further reduce the condition number, we present a pivoting scheme in Section 4 and prove that this scheme can further decrease the condition number, at least in the case of the 2-by-2 matrix. Empirical results in Section 5 demonstrate that our pivoting scheme can improve conditioning in general.

2. LLL ALGORITHM

Without loss of generality, we start with a nonsingular upper triangular matrix R, for otherwise a general matrix of full column rank can be reduced to this desired form by the QR decomposition[6]. Let us describe the matrix decomposition interpretation of the LLL algorithm as proposed by Luk and Tracy[2]; the algorithm computes the QRZ decomposition:

$$R = Q\tilde{R}Z^{-1},$$

where Q is orthogonal, $\tilde{R}$ is upper triangular and reduced, and Z is integer unimodular as defined below[7].

Definition 2.1 (Unimodular). *A nonsingular integer matrix M is said to be unimodular if* $\det(M) = \pm 1$.

A consequence of this definition is that a nonsingular integer matrix M is unimodular if and only if M^{-1} is an integer matrix. We adopt the definition of a reduced triangular matrix from Luk and Tracy[2]:

Send correspondence to S. Qiao: qiao@mcmaster.ca

Definition 2.2 (Reduced basis). *An upper triangular matrix R is reduced if*

$$|r_{i,i}| \geq 2|r_{i,j}|, \quad \textit{for all} \quad 1 \leq i < j \leq n, \tag{1}$$

and

$$r_{i,i}^2 + r_{i-1,i}^2 \geq \omega\, r_{i-1,i-1}^2, \quad \textit{for all} \quad 2 \leq i \leq n. \tag{2}$$

where $0.25 < \omega < 1$.

The algorithm comprises the following two building blocks.

Procedure 1 ($\texttt{Decrease}(i,j)$). *Given R and Z, calculate $\gamma = \lceil r_{i,j}/r_{i,i} \rfloor$, form $Z_{ij} = I_n - \gamma \mathbf{e}_i \mathbf{e}_j^{\mathrm{T}}$, where $\mathbf{e}_i$ is the ith unit vector, and apply Z_{ij} to both R and Z:*

$$R \leftarrow R Z_{ij} \quad \textit{and} \quad Z \leftarrow Z Z_{ij}.$$

Thus, if $|r_{i,i}| < 2|r_{i,j}|$ in the old R, then in the updated R, we have $|r_{i,i}| \geq 2|r_{i,j}|$ satisfying the condition (1).

Procedure 2 ($\texttt{SwapRestore}(i)$). *Given R, Z, and Q, compute the plane reflection*

$$G = \begin{bmatrix} c & s \\ s & -c \end{bmatrix}$$

such that

$$G \begin{bmatrix} r_{i-1,i-1} & r_{i-1,i} \\ 0 & r_{i,i} \end{bmatrix} P = \begin{bmatrix} \hat{r}_{i-1,i-1} & \hat{r}_{i-1,i} \\ 0 & \hat{r}_{i,i} \end{bmatrix}, \quad \textit{where} \quad P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$

and apply $Q_i = \mathrm{diag}([I_{i-2}\ J_i\ I_{n-i}])$ and permutation $\Pi_i = \mathrm{diag}([I_{i-2}\ P\ I_{n-i}])$ to R, Z, and Q:

$$R \leftarrow Q_i R \Pi_i, \quad Z \leftarrow Z \Pi_i, \quad Q \leftarrow Q Q_i.$$

The above procedure swaps columns $i-1$ and i of R and restores its upper triangular structure. If in the old R we have $r_{i,i}^2 + r_{i-1,i}^2 < \omega\, r_{i-1,i-1}^2$, where $|r_{i-1,i}| \leq |r_{i-1,i-1}|/2$, then in the updated R, we have $r_{i,i}^2 + r_{i-1,i}^2 \geq \omega\, r_{i-1,i-1}^2$. What follows is the improved LLL algorithm.[2]

Algorithm 1 (Improved LLL Algorithm). *Given an upper triangular matrix $R = [r_{i,j}]$ of order n and a parameter ω, $0.25 < \omega < 1$, this algorithm computes an orthogonal matrix Q and an integer unimodular matrix Z and overwrites R, so that the new upper triangular matrix R equals $Q^{\mathrm{T}} R Z$ and its columns form a reduced basis as defined in Definition 2.2.*

set $Z \leftarrow I$ and $Q \leftarrow I$;
$k \leftarrow 2$;
W1 while $k \leq n$
I1 if $|r_{k-1,k}/r_{k-1,k-1}| > 1/2$
I1.1 $\texttt{Decrease}(k-1,k)$;
endif
I2 if $r_{k,k}^2 + r_{k-1,k}^2 < \omega\, r_{k-1,k-1}^2$;
I2.1 $\texttt{SwapRestore}(k)$;
I2.2 $k \leftarrow \max(k-1, 2)$;
else
F1 for $i = k-2$ downto 1
I3 if $|r_{i,k}/r_{i,i}| > 1/2$
I3.1 $\texttt{Decrease}(i,k)$;
endif
endfor
I2.3 $k \leftarrow k+1$;
endif
endwhile

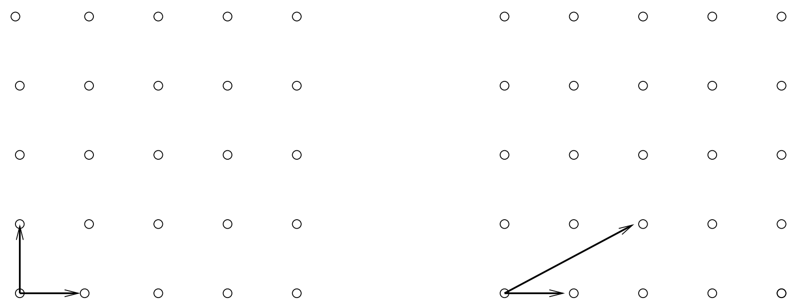

Figure 1. The identity matrix and the matrix (5) generate the same lattice points. The grid on the left shows the basis formed by the columns of the identity matrix, while the grid on the right shows the basis formed by the columns of the matrix (5).

Let us examine Algorithm 1. If $|r_{i,i}| < 2|r_{i,k}|$ on line I1 or I3, we call `Decrease`(i, k) on line I3.1 to ensure that the new $r_{i,i}$ and $r_{i,k}$ satisfy Condition (1). If $|r_{k-1,k-1}| \geq 2|r_{k-1,k}|$ satisfying (1) but $r_{k,k}^2 + r_{k-1,k}^2 < \omega\, r_{k-1,k-1}^2$ on line I2, we call `SwapRestore`(k) on line I2.1 so that the new $r_{k-1,k-1}$, $r_{k-1,k}$, and $r_{k,k}$ will satisfy the second condition (2).

In the original LLL algorithm, R is given in the form $R = DU$, where D is the diagonal part of R, and so U has a unit diagonal. The matrices D^2 and U are obtained by applying the modified Gram-Schmidt orthogonalization method to a general matrix. The original LLL algorithm is square-root-free by working on D^2 and U. See LLL[1] for details and Luk and Tracy[2] for the relation between the original LLL algorithm and the improved version.

EXAMPLE 1. *Let $\omega = 0.75$ and*

$$R_0 = \begin{bmatrix} 9/2 & 5/3 \\ 0 & \sqrt{2}/3 \end{bmatrix}. \tag{3}$$

It can be verified that R_0 satisfies the condition (1) but not the condition (2). That is, R_0 is not reduced. Applying `SwapRestore` and then `Decrease`, we get the QRZ decomposition:

$$R_0 = Q_1 R_1 Z_1^{-1} = \begin{bmatrix} 5\sqrt{3}/9 & \sqrt{6}/9 \\ \sqrt{6}/9 & -5\sqrt{3}/9 \end{bmatrix} \begin{bmatrix} \sqrt{3} & \sqrt{3}/2 \\ 0 & \sqrt{6}/2 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & -2 \end{bmatrix}^{-1}, \tag{4}$$

where the new matrix R_1 is reduced.

3. MATRIX CONDITIONING

Given a real matrix R of full column rank, the set of all points $\mathbf{x} = R\mathbf{u}$, where $\mathbf{u}$ is an integer vector, is called the lattice with basis formed by the columns of R. For example, the identity matrix I generates a rectangular lattice as shown in Figure 1. The basis is not uniquely determined by the lattice. For example, the upper triangular matrix

$$\begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix} \tag{5}$$

generates the same lattice points as in Figure 1. In general, two matrices A and B generate the same lattice points if $A = BM$, where M is an integer unimodular matrix.

The LLL algorithm reduces a lattice basis. When it is applied to the matrix in (5), the matrix is reduced to the identity. In Example 1, the LLL algorithm reduces R_0 to R_1. Both R_0 and R_1 generate the same lattice up to the orthogonal transformation Q_1. Figure 2 shows that the grid generated by R_0 is more skewed than the one generated by R_1. In terms of the matrix condition number, $\text{cond}(R_0) = 10.87$ and $\text{cond}(R_1) = 1.97$. Thus, the LLL algorithm can vastly improve the conditioning of a matrix.

How does the LLL algorithm reduce the condition number of a matrix? Condition (1) requires that the off-diagonal entries of a reduced R are small relative to the diagonal ones. In the extreme case when all the

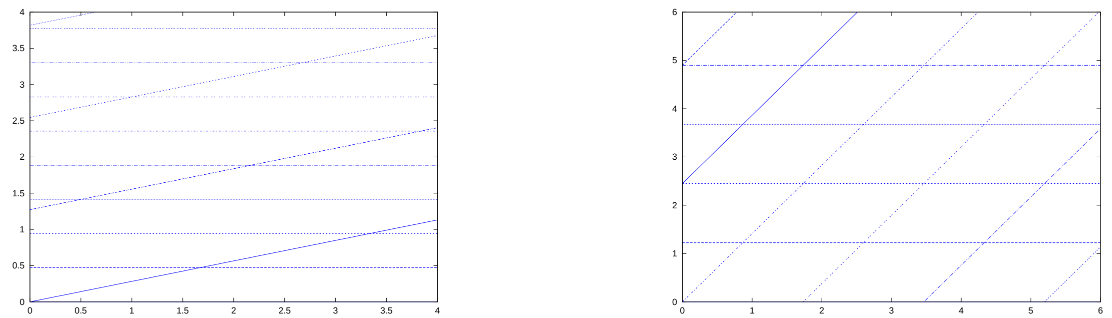

Figure 2. The grid on the left is generated by the matrix R_0 in (3) and the one on the right by the matrix R_1 in (4). The intersections of the two families of parallel lines in each grid form the lattice points.

off-diagonal entries equal zero, R is diagonal, meaning that the columns of R are orthogonal to each other; in other words, $\mathrm{cond}_2(R) = 1$. In general, one possible source of ill conditioning of a triangular matrix is from rows with off-diagonal entries which are large relative to the diagonal ones[8]. Thus one possible way of improving conditioning is to decrease off-diagonal elements. Therefore, the procedure `Decrease` in the LLL algorithm can improve the conditioning of a triangular matrix by enforcing Condition (1). Condition (2) requires that the diagonal elements be in loosely increasing order from top to bottom. The smaller the ω is, the more loosely the diagonal is ordered. The procedure `SwapRestore` itself does not change the condition, however, by pushing up small diagonal elements, the LLL algorithm can further improve the conditioning when `Decrease` is called, as shown in the above example.

Can the condition of R_1 in Example 1 be further improved by modifying the QRZ decomposition (4)? In the next section, we present a pivoting scheme that gives an affirmative answer to the question.

4. FURTHER CONDITIONING

It was pointed out in Section 3 that the LLL algorithm can improve conditioning by transforming a given matrix into a reduced one via imposing conditions (1) and (2). In this section, we propose a pivoting strategy into the LLL algorithm that may further reduce a reduced matrix.

Consider the 2×2 matrix:

$$R = \begin{bmatrix} 1 & a \\ 0 & b \end{bmatrix}, \quad b \neq 0. \tag{6}$$

Suppose that the matrix is in the reduced form, that is,

$$|a| \leq \frac{1}{2} \quad \text{and} \quad a^2 + b^2 \geq \omega. \tag{7}$$

Permuting its columns, we get

$$\begin{bmatrix} a & 1 \\ b & 0 \end{bmatrix}.$$

To restore the triangular structure, we apply the reflection:

$$\begin{bmatrix} \frac{a}{\sqrt{a^2+b^2}} & \frac{b}{\sqrt{a^2+b^2}} \\ \frac{b}{\sqrt{a^2+b^2}} & \frac{-a}{\sqrt{a^2+b^2}} \end{bmatrix}$$

and then get

$$\tilde{R} = \sqrt{a^2+b^2} \begin{bmatrix} 1 & \frac{a}{a^2+b^2} \\ 0 & \frac{b}{a^2+b^2} \end{bmatrix}. \tag{8}$$

The procedure `Decrease` will decrease the size of its (1,2)-entry when

$$\frac{2\,|a|}{a^2+b^2} > 1,$$

that is, $a^2 - 2\,|a| + b^2 < 0$, which implies that $1-\sqrt{1-b^2} < |a| \le 1/2$, provided that $b^2 < 3/4$. The inequality $1-\sqrt{1-b^2} < |a|$ is equivalent to $b^2 < |a|(2-|a|)$, which implies $b^2 < 3/4$ since $|a| \le 1/2$. In summary, assuming the matrix R in (6) is in the reduced form, that is, a and b satisfy the conditions (7), if

$$0 < b^2 < |a|(2-|a|), \tag{9}$$

then after calling the procedure `SwapRestore`, the resultant matrix $\tilde{R}$ in (8) becomes un-reduced. The application of the procedure `Decrease` to $\tilde{R}$ will decrease the size of its $(1,2)$-entry. Thus, in addition to the conditions (1) and (2), we introduce a third condition:

$$r_{i,i}^2 \ge |r_{i-1,i}|(2\,|r_{i-1,i-1}| - |\hat{r}_{i-1,i}|). \tag{10}$$

What is the effect of decreasing the size of $(1,2)$-entry? We will show that the 2-norm condition number of an upper triangular matrix of order two is improved. Without loss of generality, consider the upper triangular matrix

$$\begin{bmatrix} 1 & x \\ 0 & y \end{bmatrix}.$$

Its singular values are the square roots of

$$\frac{(x^2+y^2+1) \pm \sqrt{(x^2+y^2+1)^2 - 4y^2}}{2}.$$

It follows that its 2-norm condition number is

$$\frac{(x^2+y^2+1) + \sqrt{(x^2+y^2+1)^2 - 4y^2}}{2|y|},$$

which shows that the condition number will improve when $|x|$ is decreased.

Lemma 4.1. *Given a 2×2 upper triangular matrix R, if the condition (1) is not satisfied, then the procedure `Decrease` reduces the condition number of R.*

Since `SwapRestore` applies orthogonal transformations, we have the following lemma.

Lemma 4.2. *Given a 2×2 upper triangular matrix R, the procedure `SwapRestore` does not reduce the condition number of R.*

Example 2. *The reduced matrix R_1 in (4) in Example 1 does not satisfy the condition (10). Applying the procedure `SwapRestore` followed by `Decrease`, we get the QRZ decomposition*

$$R_1 = Q_2 R_2 Z_2^{-1} = \begin{bmatrix} \sqrt{3}/3 & \sqrt{6}/3 \\ \sqrt{6}/3 & -\sqrt{3}/3 \end{bmatrix} \begin{bmatrix} 3/2 & -1/2 \\ 0 & \sqrt{2} \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix}^{-1}. \tag{11}$$

Comparing with $\mathrm{cond}(R_1) = 1.97$, *the condition number is further reduced to* $\mathrm{cond}(R_2) = 1.41$. *Putting the decompositions (4) and (11) together, we have*

$$R = Q R_2 Z^{-1} = \begin{bmatrix} 7/9 & 4\sqrt{2}/9 \\ -4\sqrt{2}/9 & 7/9 \end{bmatrix} \begin{bmatrix} 3/2 & -1/2 \\ 0 & \sqrt{2} \end{bmatrix} \begin{bmatrix} 1 & -1 \\ -2 & 3 \end{bmatrix}^{-1}.$$

Recall that $\mathrm{cond}(R) = 10.87$ *and* $\mathrm{cond}(R_2) = 1.41$. *The final upper triangular R_2 satisfies all three conditions (1), (2), and (10). We may say that R_2 is strongly reduced.*

The grids generated by R_1 and R_2 in Figure 3 show that pivoting can further reduce a reduced lattice basis.

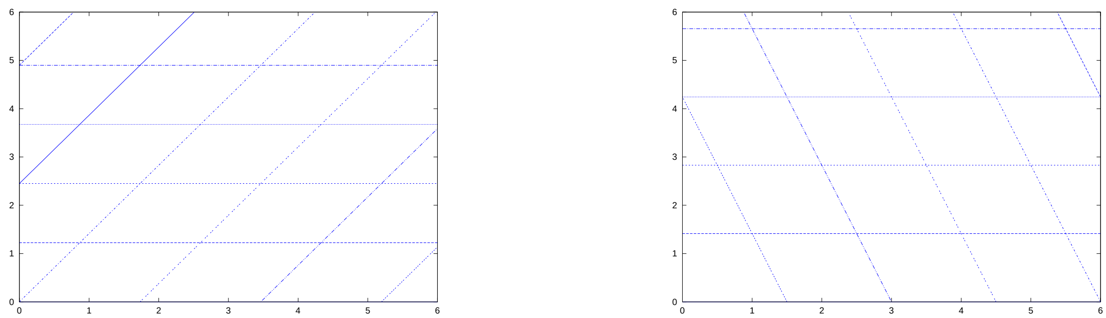

Figure 3. The grid on the left is generated by the matrix R_1 and the one on the right is generated by the matrix R_2 in Example 2, showing that a reduced basis can be further reduced by pivoting.

Algorithm 2 (LLL Algorithm with Pivoting). *Algorithm 1 with pivoting.*

set $Z \leftarrow I$ and $Q \leftarrow I$;
$k \leftarrow 2$;
W1 while $k \leq n$
I1   if $|r_{k-1,k}/r_{k-1,k-1}| > 1/2$
I1.1   `Decrease`$(k-1,k)$;
    endif
I2   if $r_{k,k}^2 + r_{k-1,k}^2 < \omega\, r_{k-1,k-1}^2$;
I2.1   `SwapRestore`(k);
I2.2 $k \leftarrow \max(k-1,2)$;
    else
        if $r_{k,k}^2 < |r_{k-1,k}|(2|r_{k-1,k-1}| - |r_{k-1,k}|)$ % pivoting
            `SwapRestore`(k);
            $k \leftarrow \max(k-1,2)$;
        else
F1   for $i = k-2$ downto 1
I3   if $|r_{i,k}/r_{i,i}| > 1/2$
I3.1   `Decrease`(i,k);
                endif
            endfor
I2.3 $k \leftarrow k+1$;
        endif
    endif
endwhile

5. NUMERICAL EXPERIMENTS

In this section, we present our experimental results to show that the LLL algorithm can dramatically improve the conditioning of an upper triangular matrix and the pivoting can further improve the conditioning.

We programmed our algorithms in Octave and ran our experiments on an Intel Core 2 Duo with 4 GB memory in Mac OS X v10.5.6. For each case, we generated random upper triangular matrices with a predetermined condition number, each of which was generated as follows. Given a condition number κ, a vector of singular values linearly spaced between one and $1/\kappa$ was generated. Two random orthogonal matrices U and V were obtained from the QR decompositions of two random matrices with entries uniformly distributed in $[-1,1]$. Then a random matrix A with condition number κ was formed by multiplying U, the diagonal singular value matrix, and V^T. The upper triangular matrix was obtained by the QR decomposition of A.

		LLL		Pivoted LLL		
ω	$\kappa(R_0)$	$\kappa(R_1)$	#calls Decrease	$\kappa(R_2)$	#pivotings	#calls Decrease
0.75	10^4	17.8	435.8	17.3	13.7	485.5
0.75	10^6	17.2	719.5	15.7	23.7	783.8
0.30	10^4	61.1	162.1	31.1	23.4	277.0
0.30	10^6	111.2	267.9	51.3	38.8	433.1

Table 1. Comparison of condition numbers of R_0, R_1, and R_2 with two different values of ω. The table also gives the numbers of calls to Procedure Decrease and the numbers of pivotings.

In our experiments, each case consisted of ten random upper triangular matrices of order 20. For each case, the averages of the condition numbers after the LLL algorithm, without and with pivoting, were calculated. Also, the average number of pivotings and the average number of calls to `Decrease` were recorded. Table 1 lists our experimental results with different condition numbers κ and different values of the iteration parameter ω. Our results show that the LLL algorithm can dramatically improve the conditioning and the pivoting can further improve the conditioning. For smaller ω, the improvement from pivoting is more noticeable.

As shown in Section 4, the procedure `Decrease` improves condition. For the same sets of matrices, we counted the average number of calls to `Decrease` in the LLL algorithms with and without pivoting. Table 1 lists the results, which shows empirically that pivoting can further improve the conditioning.

It should be pointed out that the above results are from experimenting with random matrices. The LLL algorithm does not always improve the conditioning of an upper triangular matrix. An example is given in Luk and Tracy[2]:

$$\begin{bmatrix} 1 & -0.5 & -0.5 & \cdots & \cdots & -0.5 \\ & 1 & -0.5 & \cdots & \cdots & -0.5 \\ & & 1 & \ddots & \cdots & -0.5 \\ & & & \ddots & \ddots & \vdots \\ & & & & 1 & -0.5 \\ & & & & & 1 \end{bmatrix}.$$

The $n \times n$ matrix is reduced and becomes very ill-conditioned when $n \gg 2$; indeed, its inverse is given by

$$\begin{bmatrix} 1 & 1.5^0/2 & 1.5^1/2 & \cdots & \cdots & 1.5^{n-2}/2 \\ & 1 & 1.5^0/2 & \cdots & \cdots & 1.5^{n-3}/2 \\ & & 1 & \ddots & \cdots & 1.5^{n-4}/2 \\ & & & \ddots & \ddots & \vdots \\ & & & & 1 & 1.5^0/2 \\ & & & & & 1 \end{bmatrix}.$$

REFERENCES

1. A. Lenstra, H. Lenstra, and L. Lovasz, "Factoring polynomials with rational coefficients," *Mathematicsche Annalen* **261**, pp. 515–534, 1982.
2. F. T. Luk and D. M. Tracy, "An improved LLL algorithm," *Linear Algebra and Its Applications* **428**(2-3), pp. 441–452, 2008.
3. O. Goldreich, S. Goldwasser, and S. Halevi, "Public-key cryptosystems from lattice reduction problems," in *Advances in Cryptology - CRYPTO97, 17th Ann. Int. Crypto. Conf.*, pp. 112–131, 1997.
4. B. Hassibi and H. Vikalo, "On the sphere-decoding algorithm i: Expected complexity," *IEEE Transactions on Signal Processing* **53**, pp. 2806–2818, 2005.

5. S. Qiao, "Integer least squares: Sphere decoding and the LLL algorithm," in *Proceedings of C3S2E-08*, B. C. Desai, ed., *ACM International Conference Proceedings Series*, pp. 69–80, Concordia University, (Montreal, QC), May 2008.
6. G. Golub and C. V. Loan, *Matrix Computations, 3rd Ed.*, The Johns Hopkins University Press, Baltimore, MD, 1996.
7. J. Cassels, *An Introduction to the Geometry of Numbers*, Springer-Verlag, Berlin, 1997.
8. N. J. Higham, *Accuracy and Stability of Numerical Algorithms, 2nd Ed.*, SIAM, Philadelphia, PA, 2002.