Radius Selection Algorithms for Sphere Decoding [*]

Fei Zhao
Department of Computing & Software,
McMaster University,
1280 Main St. West
Hamilton, Ontario, L8S 4L7, Canada.
zhaof3@mcmaster.ca

Sanzheng Qiao
Department of Computing & Software,
McMaster University,
1280 Main St. West
Hamilton, Ontario, L8S 4L7, Canada.
qiao@mcmaster.ca

ABSTRACT

The integer least squares problem arises from many applications such as communications, cryptography, and GPS. In this paper, we consider the sphere decoding method in communication applications. One of key issues in sphere decoding is the selection of an initial radius of the search hypersphere. We first present a deterministic radius selection algorithm using the Babai estimate. However, due to the rounding errors in floating-point computation, this method may produce a too small radius and cause sphere decoding to fail to find a solution. In this paper, we perform an error analysis and propose a modified radius selection algorithm by taking computational error into account. Our numerical experiments show that this modified method achieves high success rate without compromising performance.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Integer Least Squares—*error analysis and correction*; D.2.8 [**Software Engineering**]: Metrics—*performance measures*

General Terms

Improvement

Keywords

Integer least squares, sphere decoding, closest lattice point, Babai estimate radius, numerical error.

1. INTRODUCTION

Sphere decoding is widely used in communication applications [1]. It is a method for solving the integer least squares problem:

$$\min_{s \in \mathbb{Z}^m} \|Hs - y\|_2^2, \tag{1}$$

where $y \in \mathbb{R}^n$ and $H \in \mathbb{R}^{n \times m}$. That is to find an integer vector s minimizing $\|Hs - y\|_2^2$. Note that while s is an integer vector, both the matrix H and vector y are real. The general problem of integer least squares has many applications, such as, in addition to communications, cryptography [2] and GPS [3], among others.

The integer least squares problem (1) can be interpreted by a geometric lattice. All integer vectors s form a rectangular m-dimensional lattice. Figure 1 shows a rectangular 2-dimensional lattice.

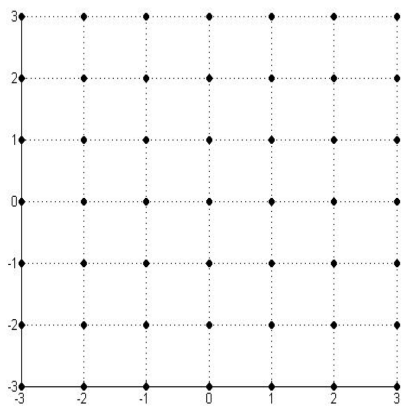

Figure 1: A 2-dimensional rectangular lattice.

The matrix H in (1), called lattice generating matrix, transforms m-dimensional integer vectors s into n-dimensional real vectors Hs forming a skewed lattice. Figure 2 depicts a skewed lattice, represented by solid dots, transformed by a 3-by-2 lattice generating matrix. The vector y in (1) is an n-dimensional real vector. In Figure 2, for example, the hollow point represents a 3-dimensional real vector. Thus, in that figure, the integer least squares problem is to find a solid point that is closest, in the Euclidean distance, to the hollow point.

It is shown in [4] that the complexity of solving a general integer least squares problem is NP-hard. A proof can be found in [5].

Sphere decoding is a particular method for solving the integer least squares problem in communication applications. As the standard way of solving least squares problems [6], assuming the matrix H in (1) is of full column rank, H is first reduced into an upper triangular matrix using orthogonal transformations, such as the Householder transformation, to obtain the QR decomposition:

$$H = Q \begin{bmatrix} R \\ 0 \end{bmatrix},$$

where $Q \in \mathbb{R}^{n \times n}$ is orthogonal and $R \in \mathbb{R}^{m \times m}$ is upper trian-

[*]This work is partially supported by NSERC of Canada.

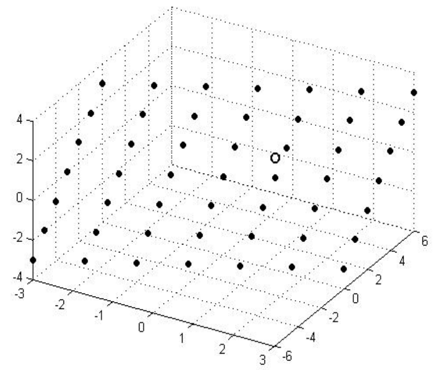

Figure 2: Geometric interpretation of the integer least squares problem

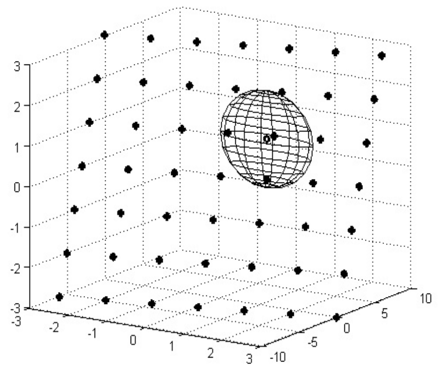

Figure 3: Geometric interpretation of a hypersphere in a lattice space.

gular. Partitioning $Q = [Q_1 \;\; Q_2]$, where Q_1 is $n \times m$ and Q_2 is $n \times (n-m)$, we get

$$\begin{aligned}
& \|Hs - y\|_2^2 \\
= & \left\| Q \begin{bmatrix} R \\ 0 \end{bmatrix} s - y \right\|_2^2 \\
= & \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} s - Q^T y \right\|_2^2 \\
= & \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} s - \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix} y \right\|_2^2 \\
= & \|Rs - Q_1^T y\|_2^2 + \|Q_2^T y\|_2^2. \qquad (2)
\end{aligned}$$

Note that the second term is independent of s. Denoting $\hat{y}$ as $Q_1^T y$, the integer least squares problem (1) is then reduced to the following triangular integer least squares problem:

$$\min_{s \in \mathbb{Z}^m} \|Rs - \hat{y}\|_2^2. \qquad (3)$$

Sphere decoding solves the triangular integer least squares problem (3) arising from communication applications. It searches a solution in a predetermined hypersphere centered at $\hat{y}$. We describe the sphere decoding algorithm in Section 2. One of key issues in sphere decoding is the selection of a search radius. In Section 3, we present a radius selection algorithm. In theory, this algorithm ensures success, however, in practice due to rounding errors, this algorithm sometimes fails. We perform an error analysis in Section 4 and propose a modified radius selection algorithm in Section 5. Our numerical experiments demonstrated in Section 6 show that our modified radius selection algorithm has perfect success rate without degrading performance. Finally, Section 7 concludes our paper.

2. SPHERE DECODING

Sphere decoding, introduced originally by Finke and Pohst in [7] in 1985, enumerates all lattice points in a hypersphere centered at a given vector. It searches a lattice point in a hypersphere of radius d and centered at $\hat{y}$ in (3) that is closest, in Euclidean distance, to the center. Therefore, by restricting the search area, it can reduce the computational complexity of solving (3). Figure 3 illustrates a hypersphere centered at a vector represented by the hollow point.

The complexity of sphere decoding depends on the following two issues:

- The radius d of the hypersphere.
  If d is too large, the hypersphere contains too many lattice points, then the search complexity may be exponential to d; if d is too small, on the other hand, the hypersphere may contain no lattice points. There are no general guidelines for selecting an appropriate d. It is application dependent.
- Search for all the lattice points inside the hypersphere.
  This requires the test of the Euclidean distance between each lattice point and the given central point to determine whether it is smaller than the radius d of the hypersphere.

2.1 Tree representation of sphere decoding

The process of sphere decoding can be informally explained as follows. It works in an order from the mth dimension down to the first dimension and in entrywise for lattice points. We know that an m-dimensional lattice point is determined by an m-dimensional integer vector. Suppose that we have determined the entries, from the mth down to the kth, of all lattice points in the hypersphere. Then, for each kth entry of a lattice point, we find all the possible $(k-1)$th entries so that the new lattice points lie in the hypersphere. Figure 4 shows the process of sphere decoding represented by a tree of depth $m+1$, where $m = 3$. Starting at the root, we first find the mth entries of all possible lattice points in the sphere, using the radius d. In Figure 4, there are three possible integer values of the mth entry. Then for each possible value of the mth entry, we find all possible values of the $(m-1)$th entry. Note that some of the nodes at level $k = 2$ may not lead to any possible nodes at the lower level. Thus, each node in the tree is assigned an integer and each path from the root to a leaf node at the lowest level gives a lattice point, an m-dimensional integer vector, in the sphere. In the figure, there are eleven lattice points in the sphere. From the viewpoint of this tree representation, the complexity of sphere decoding depends on the $size(density)$ of the tree, that is, the number of nodes of the tree visited by sphere decoding. The details of the sphere decoding algorithm is given in the next subsection.

2.2 Sphere decoding algorithm

The sphere of radius d and centered at y in (1) can be defined as

$$S = \{s \quad | \quad \|Hs - y\| \le d\},$$

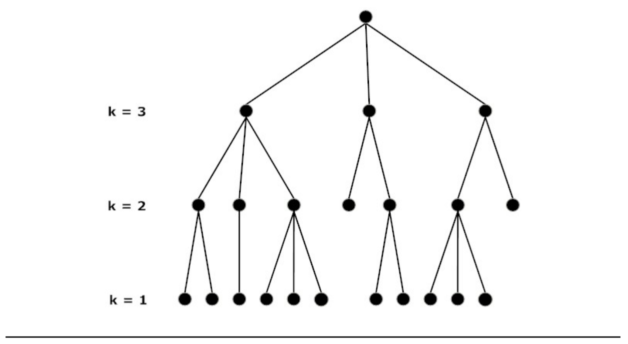

Figure 4: Tree representation of sphere decoding, where $m = 3$.

whose condition, from (2), is equivalent to

$$\|Rs - \hat{y}\|_2^2 \le \hat{d}^2, \tag{4}$$

where

$$\hat{d}^2 \triangleq d^2 - \|Q_2^T y\|_2^2.$$

Since R is upper triangular, we can rewrite the condition (4) in entrywise as

$$\hat{d}^2 \ge \sum_{i=1}^{m} \left(\sum_{j=i}^{m} r_{i,j} s_j - \hat{y}_i \right)^2 ,$$

where $r_{i,j}$, $j \ge i$, denotes the (i,j)th entry of R. The above inequality can then be expanded to

$$\begin{aligned} \hat{d}^2 \quad \ge \quad & (\hat{y}_m - r_{m,m} s_m)^2 \\ & + (\hat{y}_{m-1} - r_{m-1,m} s_m - r_{m-1,m-1} s_{m-1})^2 \\ & + \cdots . \end{aligned} \tag{5}$$

The first term in the right side of (5) depends only on the mth entry s_m of lattice point s, the second term depends on the entries s_m and s_{m-1}, and so on.

We can see that a necessary condition for Rs lying in the hypersphere of radius $\hat{d}$ is $\hat{d}^2 \ge (\hat{y}_m - r_{m,m} s_m)^2$, which is equivalent to the following condition for entry s_m:

$$\left\lceil \frac{-\hat{d} + \hat{y}_m}{r_{m,m}} \right\rceil \le s_m \le \left\lfloor \frac{\hat{d} + \hat{y}_m}{r_{m,m}} \right\rfloor . \tag{6}$$

Furthermore, for each integer s_m satisfying (6), define

$$\hat{d}_{m-1}^2 \triangleq \hat{d}^2 - (\hat{y}_m - r_{m,m} s_m)^2$$

and

$$\hat{y'}_{m-1} \triangleq \hat{y}_{m-1} - r_{m-1,m} s_m ,$$

then, if $\hat{d}_{m-1}^2 \ge (\hat{y'}_{m-1} - r_{m-1,m-1} s_{m-1})^2$, the condition for s_{m-1} is

$$\left\lceil \frac{-\hat{d}_{m-1} + \hat{y'}_{m-1}}{r_{m-1,m-1}} \right\rceil \le s_{m-1} \le \left\lfloor \frac{\hat{d}_{m-1} + \hat{y'}_{m-1}}{r_{m-1,m-1}} \right\rfloor .$$

Following the above procedure, we can obtain the intervals for s_{m-2}, s_{m-3}, and so on until we reach s_1. Then we are able to determine all the lattice points in the hypersphere of radius $\hat{d}$.

The sphere decoding algorithm is presented in Table 1.

Remarks. In the implementation of the sphere decoding algorithm, there are two issues need to be addressed:

Table 1: Algorithm 1 - Sphere Decoding Algorithm

```
In:  $R$, upper triangular matrix
     $\hat{y}$, center of the sphere
     $\hat{d}$, radius of the sphere
Out: $s$ or $null$

1.  $d_n \leftarrow \hat{d}$;
2.  $UB_n \leftarrow \lfloor (d_n + \hat{y}_n)/r_{n,n} \rfloor$, $LB_n \leftarrow \lceil (-d_n + \hat{y}_n)/r_{n,n} \rceil$;
3.  $minR \leftarrow d_n$;
4.  $interSum_n \leftarrow \hat{y}_n$;
5.  if $LB_n > UB_n$
6.    initial radius too small, return $null$;
7.  else $s_n \leftarrow LB_n$; end
8.  $k \leftarrow n$;
9.  while $s_n \le UB_n$ {top level not exhausted}
10.   if $s_k > UB_k$ {this level exhausted already}
11.     $k \leftarrow k + 1$; $s_k \leftarrow s_k + 1$; {go back to upper level}
12.   else {$s_k \le UB_k$ }
13.     if $k > 1$
14.       $k \leftarrow k - 1$;
15.       $d_k^2 \leftarrow d_{k+1}^2 - (interSum_{k+1} - r_{k+1,k+1} * s_{k+1})^2$;
16.       $interSum_k = \hat{y}_k - r_{k,k+1:n} * s_{k+1:n}$;
17.       $UB_k \leftarrow \lfloor (d_k + interSum_k)/r_{k,k} \rfloor$;
18.       $LB_k \leftarrow \lceil (-d_k + interSum_k)/r_{k,k} \rceil$;
19.       if $LB_k > UB_k$ {empty interval}
20.         $k \leftarrow k + 1$; $s_k \leftarrow s_k + 1$; {go back to upper level}
21.       else $s_k \leftarrow LB_k$; end
22.     else {$k = 1$}
23.       while $s_k \le UB_k$ {go through all $s_k$ at $k = 1$}
24.         if $minR > \|Rs - \hat{y}\|_2$
25.           found $s$;
26.           $minR \leftarrow \|Rs - \hat{y}\|_2$;
27.         end
28.         $s_k \leftarrow s_k + 1$;
29.       end {inner while}
30.       $k \leftarrow k + 1$; $s_k \leftarrow s_k + 1$; {go back to $k = 2$ level}
31.     end {if $k > 1$ else}
32.   end {if $s_k > UB_k$ else}
33. end {outer while}
34. return $s$ or return $null$;
```

- We start with $k = m$, decrease k until $k = 1$. It is possible that some intervals determined by inequalities like (6) contain no integers. In that case, we cannot go down further. Instead, we have to go back to the upper level and pick another integer in the upper level's interval, then compute its lower level's interval, and so on. This is why the generated tree structure of sphere decoding is not a balanced tree, that is, not all the branches have the same length. Some branches may be shorter than the depth of the generated tree. It is also possible that all the branches' lengths of the generated tree structure are less than $m + 1$. That means that the hypersphere contains no lattice point. In this case, it is necessary to increase the initial radius $\hat{d}$ of the searching hypersphere.

- In the hypersphere centered at $\hat{y}$, there may be more than one lattice point. Therefore, we have to search all the lattice points in the hypersphere and compare their Euclidean distances to $\hat{y}$ and find the one closest to $\hat{y}$.

From the above remarks, we can see that the decoding process sometimes increases the value of k and sometimes decreases the value of k. In other words, it sometimes goes up a level and sometimes goes down a level, but in different branches each time. It goes through the tree, except the root node, like a preorder traversal and performs depth-first searching.

Table 2: Algorithm 2 - Radius selection algorithm using the Babai estimate

Inputs:	R, where R is upper triangular.
	$\hat{y}$, where $\hat{y}$ is the y reduced by the QR decomposition.
Output:	the radius $\hat{d}$ of search sphere
1.	solve $s \in \mathbb{R}^m$ for $Rs = \hat{y}$;
2.	round: $\hat{s} = \lceil s \rfloor \in \mathbb{Z}^m$;
3.	set $\hat{d} = \|R\hat{s} - \hat{y}\|_2$;

3. RADIUS SELECTION USING BABAI ESTIMATE

In [9], Qiao proposed a deterministic method for selecting an initial hypersphere radius. This method is designed for communication applications. Since H represents a channel matrix for a communication channel, it can be assumed that the norm of Hs is not too large because of the power constraint of the channel. This means that when the channel matrix H is applied to the source signal vector s, it does not significantly magnify the length of the source signal vector. In other words, $\|Hs\|_2$ and $\|s\|_2$ are of the same magnitude order. Based on this assumption, an initial radius can be determined by the following deterministic method.

Suppose that the QR decomposition is already applied to (1) and it is reduced to (3). First, we find the real solution for the triangular system $Rs = \hat{y}$, which is the real least squares solution for the problem $\min \|Hs - y\|_2^2$. Then we round the entries of s to their nearest integers to obtain the lattice point:

$$\hat{s} = \lceil s \rfloor \in \mathbb{Z}^m.$$

This $\hat{s}$ is known as the Babai estimate [8]. A radius $\hat{d}$ is then set to the distance, in the Euclidean sense, between $R\hat{s}$ and $\hat{y}$:

$$\hat{d} = \|R\hat{s} - \hat{y}\|_2. \qquad (7)$$

In communication applications, this procedure is often referred to as Zero-Forcing (ZF) equalization.

It is clear that the hypersphere of radius $\hat{d}$ and centered at $\hat{y}$ contains at least one lattice point, namely $\hat{s}$. Thus the sphere decoding using this radius will find the integer least squares solution in this sphere, possibly on its surface.

If the real least squares solution s happens to be an integer vector, that is, $\hat{s} = s$, then the radius $\hat{d}$ is zero, meaning that s is the integer least squares solution. In communication application, this situation means that both channel and signal are perfect, no channel distortion on the transmitted signal and no additive noise to the transmitted signal, which is only possible in theory.

The algorithm for selecting a search radius $\hat{d}$ using the Babai estimate is presented in Table 2.

The size of $\hat{d}$ is also examined in [9]. Let $f = \hat{s} - s$, then from (7), we have

$$\begin{aligned} \hat{d} &= \|R\hat{s} - \hat{y}\|_2 \\ &= \|R(s+f) - \hat{y}\|_2 \\ &= \|Rf\|_2. \end{aligned}$$

Since $f = \hat{s} - s = \lceil s \rfloor - s$, $s \in \mathbb{R}^m$, we get $\|f\|_2 \le \sqrt{m}/2$. Thus

$$\begin{aligned} \hat{d} &= \|Rf\|_2 \\ &\le \frac{\sqrt{m}}{2}\|R\|_2 \\ &= \frac{\sqrt{m}}{2}\|H\|_2. \end{aligned}$$

The radius computed by Algorithm 2 ensures that the search sphere contains at least one lattice point, namely $\hat{s}$, therefore the integer least squares solution lies in the search sphere if the radius is computed exactly. In practice, however, in the presence of inexact arithmetic due to rounding errors introduced by floating-point computation, the computed radius contains error. If the computed radius is smaller than the exact radius and $\hat{s}$ happens to be the integer least squares solution, the computed search sphere will contain no lattice point and sphere decoding fails. Consider the following example.

Example 1:

In (1), let

$$H = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 0 & -2 \\ -1 & -1 & -1 \end{bmatrix} \quad \text{and} \quad y = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}.$$

In MATLAB (double precision), after the QR decomposition,

$$R \approx \begin{bmatrix} -2.4495 & 0.4082 & -0.4082 \\ 0 & 1.3540 & 1.6002 \\ 0 & 0 & 1.8091 \end{bmatrix}$$

and

$$\hat{y} \approx \begin{bmatrix} 1.2247 \\ 0.3693 \\ -0.6030 \end{bmatrix}.$$

The computed real solution for the triangular system $Rs = \hat{y}$ was

$$s \approx \begin{bmatrix} -0.3333 \\ 0.6667 \\ -0.3333 \end{bmatrix}.$$

Rounding the entries of s to their nearest integers, we got

$$\hat{s} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix},$$

which happened to be an integer least squares solution. The exact search radius should be $\sqrt{2}$, however, the radius computed by $\tilde{d} = \|R\hat{s} - \hat{y}\|_2$ (where $\tilde{d}$ denotes the computed radius) using double precision in MATLAB was slightly smaller than $\sqrt{2}$ due to rounding errors. Specifically, $\tilde{d} - \sqrt{2} \approx -2.22 \times 10^{-16}$. Using this computed search radius, the sphere decoding failed to find a lattice point in the computed sphere.

The above example shows that a practical radius selection algorithm must take computational error into account.

4. AN ERROR ANALYSIS

In this section, we perform an error analysis on our deterministic radius selection algorithm and derive an error bound for the computed radius.

Recall that we round the real least squares solution s to the integer vector $\hat{s}$ and then compute the radius $\hat{d} = \|R\hat{s} - \hat{y}\|_2$. So, it remains to find the computational error in computing $\|R\hat{s} - \hat{y}\|_2$. We denote the computed radius as $\tilde{d}$ and find $|\hat{d} - \tilde{d}|$. Let $u = R\hat{s}$

and $\tilde{u}$ be the computed $R\hat{s}$, which is a matrix-vector multiplication. A forward error analysis of matrix multiplication [10] shows that

$$\|u - \tilde{u}\|_2 \leq \gamma_m \sqrt{m}\, \|R\|_2 \|\hat{s}\|_2, \tag{8}$$

where

$$\gamma_m = \frac{m\mu}{1 - m\mu}$$

and μ is the unit of roundoff. Thus, using (8), the error in the computed radius we are looking for is

$$\begin{aligned} |\hat{d} - \tilde{d}| &= |\, \|R\hat{s} - \hat{y}\|_2 - \tilde{d}| \\ &= |\, \|u - \tilde{u} + \tilde{u} - \hat{y}\|_2 - \tilde{d}| \\ &\leq \|u - \tilde{u}\|_2 + |\, \|\tilde{u} - \hat{y}\|_2 - \tilde{d}| \\ &\leq \gamma_m \sqrt{m}\, \|R\|_2 \|\hat{s}\|_2 + |\, \|\tilde{u} - \hat{y}\|_2 - \tilde{d}|. \end{aligned} \tag{9}$$

The term $|\, \|\tilde{u} - \hat{y}\|_2 - \tilde{d}|$ is the computational error in computing the vector difference $\tilde{u} - \hat{y}$ and then the 2-norm of the computed $\tilde{u} - \hat{y}$. A standard forward error analysis of vector addition and inner product [10] gives

$$|\, \|\tilde{u} - \hat{y}\|_2 - \tilde{d}| \leq \gamma_m \tilde{d}.$$

Thus, from (9), we have

$$|\hat{d} - \tilde{d}| \leq \gamma_m \sqrt{m}\, \|R\|_2 \|\hat{s}\|_2 + \gamma_m \tilde{d}. \tag{10}$$

The above inequality gives an upper bound for the computational error in the computed radius $\tilde{d}$, which can be used to get an upper bound for the exact radius $\hat{d}$.

5. MODIFIED RADIUS SELECTION ALGORITHM

In this section, applying the error bound (10) for the computed radius derived in the previous section, we present a modified radius selection algorithm that takes the computational error into account.

The μ in γ_m is the unit of roundoff. For example, in single precision $\mu \approx 10^{-7}$ and in double precision $\mu \approx 10^{-16}$. We assume that

$$m\mu < 1/2,$$

since the typical values of m are much smaller than half of the machine precision. We then have $\gamma_m < 2m\mu$ since $1 - m\mu > 1/2$. Also note that $\|R\|_2 = \|H\|_2$. It then follows from (10) that

$$\hat{d} \leq \tilde{d} + 2m(\sqrt{m}\, \|H\|_2 \|\hat{s}\|_2 + \tilde{d})\mu. \tag{11}$$

The above inequality gives an upper bound for the exact radius $\hat{d}$ in term of the computed radius $\tilde{d}$, the norm of the channel matrix H, and the norm of the integer vector $\hat{s}$. Using this bound (11), we present a modified radius selection algorithm listed in Table 3. This algorithm incorporates the computational error into the computation of the search radius.

Example 2:

Following the Example 1,

$$\|H\|_2 \approx 2.676 \quad \text{and} \quad \|\hat{s}\|_2 = 1.$$

The modified radius selection algorithm computed, in double precision ($\mu \approx 10^{-16}$), a radius $\tilde{d}$ slightly larger than the exact radius $\hat{d} = \sqrt{2}$. Specifically, $\tilde{d} - \sqrt{2} \approx 3.33 \times 10^{-15}$. Consequently, sphere decoding succeeded in finding the solution $[0\ 1\ 0]^T$.

Table 3: Algorithm 3 - Modified radius selection algorithm

Inputs:	R, where R is upper triangular. $\|H\|_2$, the 2-norm of the channel matrix H. $\hat{y}$, where $\hat{y}$ is the y reduced by the QR decomposition.
Output:	the radius $\hat{d}$ of search sphere
1.	solve $s \in \mathbb{R}^m$ for $Rs = \hat{y}$;
2.	round: $\hat{s} = \lceil s \rfloor \in \mathbb{Z}^m$;
3.	compute $\tilde{d} = \|R\hat{s} - \hat{y}\|_2$;
4.	set $\hat{d} = \tilde{d} + 2m(\sqrt{m}\, \|H\|_2 \|\hat{s}\|_2 + \tilde{d})\mu$;

6. NUMERICAL EXPERIMENTS

In this section, we simulate a communication channel and target our experiments on this channel simulation. In communication applications, the channel matrix $H \in \mathbb{R}^{n \times m}$ usually has the following Toeplitz form:

$$H = \begin{bmatrix} h_1 & & & \\ h_2 & h_1 & & \\ \vdots & \ddots & \ddots & \\ h_l & \ddots & \ddots & h_1 \\ & h_l & \ddots & h_2 \\ & & \ddots & \vdots \\ & & & h_l \end{bmatrix}$$

where $h_i, i = 1, \cdots, l$ are the parameters of the channel and l is the order of the channel, thus, $n = m + l - 1$. The transmitted source signal is an integer vector. The noise vector v is additive white Gaussian $\mathcal{N}(0, \sigma^2)$ with mean zero and variance σ^2, and the noise vector is scaled based on the given signal-to-noise ratio (SNR). Therefore, the actual received signal vector y given to sphere decoding is computed by $y = Hs + v$.

We programed our algorithms in MATLAB. The entries of the channel matrix H were random numbers uniformly distributed over the interval $[0, 1]$. The entries of the source signal vector s were randomly chosen from the set $\{\pm 1, \pm 3\}$. The entries of the additive white noise v were random values drawn from a normal distribution with mean zero and a deviation computed from the SNR. For each randomly generated channel matirx H, one thousand random source signal vectors s and one thousand random white noise vectors v with $SNR = 20dB$ were generated, then one thousand received signal vectors y were computed. The order of channel was set to 3 or 5 ($l = 3$ or $l = 5$), and the length of source signal was set to 4 or 8 ($m = 4$ or $m = 8$).

Table 4 lists the average radius computed by the Algorithm 2 for each communication channel and the number of failures out of the one thousand cases. Table 5 shows the average radius computed by the Algorithm 3. From the two tables, we can see that

- Using the modified radius selection algorithm, sphere decoding achieved perfect success rate in our experiments.
- The radii computed by both radius selection algorithms were almost the same. Thus we expected negligible performance difference between the two radius selection algorithms, which was confirmed by our running time measurements.

7. CONCLUSION

Table 4: Results from Algorithm 2

Size of signal (m), Order of channel (l)	Average radius ($\hat{d}$)	Failure rates (%)
$m = 4, l = 3$	0.744781	66.7
$m = 4, l = 5$	0.861159	65.2
$m = 8, l = 3$	1.44361	20.5
$m = 8, l = 5$	1.49963	32.6

Table 5: Results from Algorithm 3

Size of signal (m), Order of channel (l)	Average radius ($\hat{d}$)	Failure rates (%)
$m = 4, l = 3$	0.752058	0
$m = 4, l = 5$	0.820661	0
$m = 8, l = 3$	1.45038	0
$m = 8, l = 5$	1.54609	0

This paper discusses the issue of selecting a search radius in sphere decoding in communication applications, more generally in solving integer least squares problems. We first present a radius selection algorithm using the Babai estimate and show that sphere decoding may fail due to rounding errors. Then we perform an error analysis of the algorithm and propose a modified radius selection algorithm by taking computational error into account. Our experiments show that the modified radius selection algorithm achieved perfect success rate without noticeable performance degradation.

8. REFERENCES

[1] B. Hassibi and H. Vikalo, "On the Sphere Decoding Algorithm I. Expected Complexity", *IEEE Transactions on Signal Processing*, vol. 53, no. 8, pp. 2806-2818, Aug. 2005.

[2] O. Goldreich, S. Goldwasser and S. Halevi, "Public-Key Cryptosystems from Lattice Reduction Problems", *Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*, pp. 112-131, Aug. 1997.

[3] A. Hassibi and S. Boyd, "Integer parameter estimation in linear models with applications to GPS", *IEEE Transactions on Signal Processing*, vol. 46, no. 11, pp. 2938-2952, Nov. 1998

[4] P. van Emde Boas, "Another NP-complete partition problem and the complexity of computing short vectors in lattices", *Tech. Rept. 81-04, Department of Mathematics, University of Amsterdam*, 1981.

[5] D. Micciancio, "The hardness of the closest vector problem with preprocessing", *IEEE Transactions on Information Theory*, vol. 47, no. 3, pp. 1212-1215, Mar. 2001.

[6] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd Edition, Johns Hopkins University Press, Baltimore, Maryland, 1996.

[7] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis", *Mathematics of Computation*, vol. 44, no. 170, pp. 463-471, Apr. 1985.

[8] M. Grotschel, L. Lovasz and A. Schriver, *Geometric Algorithms and Combinatorial Optimization*, 2nd Edition, Springer-Verlag, New York, 1993.

[9] Sanzheng Qiao, "Integer least squares: Sphere decoding and the LLL algorithm", *Proceedings of C3S2E-08, ACM International Conference Proceedings Series*, pp. 23-28, May 2008.

[10] Nicholas J. Higham, *Accuracy and Stability of Numerical Algorithms*, Second Edition, Society for Industrial and Applied Mathematics, 2002.