Invited paper: An efficient H^∞ estimation approach to speed up the sphere decoder

Mihailo Stojnic, Haris Vikalo and Babak Hassibi

Department of Electrical Engineering
California Institute of Technology
Pasadena, California, USA
E-mail:{mihailo, hvikalo, hassibi}@systems.caltech.edu

***Abstract*—Maximum-likelihood (ML) decoding often reduces to solving an integer least-squares problem, which is NP hard in the worst-case. On the other hand, it has recently been shown that, over a wide range of dimensions and signal-to-noise ratios (SNR), the sphere decoding algorithm finds the exact solution with an expected complexity that is roughly cubic in the dimension of the problem. However, the computational complexity of sphere decoding becomes prohibitive if the SNR is too low and/or if the dimension of the problem is too large. In this paper, we target these two regimes and attempt to find faster algorithms by pruning the search tree beyond what is done in the standard sphere decoder. The search tree is pruned by computing lower bounds on the possible optimal solution as we proceed to go down the tree. Using ideas from H^∞ estimation theory, we have developed a general framework to compute the lower bound on the integer least-squares. Several special cases of lower bounds were derived from this general framework. Clearly, the tighter the lower bound, the more branches can be eliminated from the tree. However, finding a tight lower bound requires significant computational effort that might diminish the savings obtained by additional pruning. In this paper, we propose the use of a lower bound which can be computed with only linear complexity. Its use for tree pruning results in significantly speeding up the basic sphere decoding algorithm.**

I. INTRODUCTION

In this paper we are interested in solving *exactly* the following problem

$$\min_{\mathbf{s}\in\mathcal{D}\subset\mathcal{Z}^m} \|\mathbf{x}-H\mathbf{s}\|_2, \tag{1}$$

where $\mathbf{x}\in\mathcal{R}^m$, $H\in\mathcal{R}^{m \quad m}$ and $\mathcal{D}$ refers to some subset of the integer lattice $\mathcal{Z}^m$. The main idea of the sphere decoder algorithm [1] for solving the previous problem is based on finding all points $\mathbf{s}$ such that $\|\mathbf{x}-H\mathbf{s}\|_2$ lies within some adequately chosen radius d, i.e., on finding all $\mathbf{s}$ such that

$$d^2 \geq \|\mathbf{x}-H\mathbf{s}\|_2^2, \tag{2}$$

and then choosing the one that minimizes the objective function. Using the QR-decomposition $H=QR$, with Q unitary and R upper triangular, we can reformulate (2) as

$$d^2 \geq \|\mathbf{y}-R\mathbf{s}\|_2^2, \tag{3}$$

where we have defined $\mathbf{y}=Q^*\mathbf{x}$.

This work was supported in part by the National Science Foundation under grant no. CCR-0133818, by a fellowship from the David and Lucille Packard Foundation, and by Caltech's Lee Center for Advanced Networking

Now using the upper-triangular property of R, (3) can be further rewritten as

$$d^2 \geq \|\mathbf{y}_{k:m}-R_{k:m,k:m}\mathbf{s}_{k:m}\|^2 + \|\mathbf{y}_{1:k-1}-R_{1:k-1,1:k-1}\mathbf{s}_{1:k-1}-R_{1:k-1,k:m}\mathbf{s}_{k:m}\|^2, \tag{4}$$

for any $2\leq k\leq m$, where the subscripts determine the entries the various vectors and matrices run over. A necessary condition for (3) can therefore be obtained by omitting the second term on the RHS of the above expression to yield

$$d^2 \geq \|\mathbf{y}_{k:m}-R_{k:m,k:m}\mathbf{s}_{k:m}\|^2. \tag{5}$$

The sphere decoder finds all points $\mathbf{s}$ in (2) by proceeding inductively on (5), starting from $k=m$ and proceeding to $k=1$. In other words for $k=m$, it determines all one-dimensional lattice points $\mathbf{s}_m$ such that

$$d^2 \geq (\mathbf{y}_m - R_{m,m}\mathbf{s}_m)^2,$$

and then for each such one-dimensional lattice point $\mathbf{s}_m$ determines all possible values for $\mathbf{s}_{m-1}$ such that

$$\begin{aligned} d^2 &\geq \|\mathbf{y}_{m-1:m}-R_{m-1:m,m-1:m}\mathbf{s}_{m-1:m}\|^2 \\ &= (\mathbf{y}_m - R_{m,m}\mathbf{s}_m)^2 \\ &+ (\mathbf{y}_{m-1}-R_{m-1,m-1}\mathbf{s}_{m-1}-R_{m-1,m}\mathbf{s}_m)^2. \end{aligned}$$

This gives all possible two-dimensional lattice points and one then proceeds in a similar fashion until $k=1$. The sphere decoder thus generates a tree, where the branches at the $m-k+1$th level of the tree correspond to all $m-k+1$-dimensional lattice points satisfying (5). In this manner at the bottom of the tree (the m-th level) all points satisfying (2) are found. (For more details on the sphere decoder and for an explicit description of the algorithm the reader may refer to [1], [8], [3].)

The computational complexity of the sphere decoder depends on how d is chosen. In communications we usually can assume

$$\mathbf{x}=H\mathbf{s}+\mathbf{w}, \tag{6}$$

where the entries of $\mathbf{w}$ are independent $\mathcal{N}(0,\sigma^2)$ random variables. In [3] it is shown that, if the radius is chosen appropriately based on the statistical characteristics of the noise $\mathbf{w}$, then over a wide range of SNRs and problem dimensions the expected complexity of the sphere decoder is roughly cubic.

The above assertion unfortunately fails and the computational complexity becomes increasingly prohibitive if the SNR is too low and/or if the dimension of the problem is too large. Increasing the dimension of the problem clearly is useful. Moreover, the use of the sphere decoder in low SNR situations is also important when one is interested in obtaining soft information to pass onto an iterative decoder (see, e.g., [6], [5]). To reduce the computational complexity one approach is to resort to suboptimal methods based either on heuristics (see, e.g., [4]) or some form of statistical pruning (see [7]).

In this paper, we attempt to reduce the computational complexity of the sphere decoder while still finding the *exact* solution. Let us surmise on how this may be done. As mentioned above, the sphere decoder generates a tree whose number of branches at each level corresponds to the number of lattice points satisfying (5). Clearly, the complexity of the algorithm depends on the size of this tree since each branch in the tree is visited and appropriate computations are then performed. Thus, one approach would be to reduce the size of the tree beyond that which is suggested by (5). To do so, suppose that we had some way of computing a lower bound on the optimal value of the second term of the RHS of (4):

$$LB = LB(\mathbf{y}_{1:k-1}, R_{1:k-1,1:m}, \mathbf{s}_{k:m}) \leq$$
$$\min_{\mathbf{s}_{1:k-1}\in\mathcal{D}\subset\mathcal{Z}^{k-1}} \|\mathbf{y}_{1:k-1} - R_{1:k-1,1:k-1}\mathbf{s}_{1:k-1} - R_{1:k-1,k:m}\mathbf{s}_{k:m}\|^2,$$

where we have emphasized the fact that the lower bound is a function of $\mathbf{y}_{1:k-1}$, $R_{1:k-1,1:m}$, and $\mathbf{s}_{k:m}$. Provided our lower bound is nontrivial, i.e., $LB > 0$, then we can replace (5) by [1]

$$d^2 - LB \geq \|\mathbf{y}_{k:m} - R_{k:m,k:m}\mathbf{s}_{k:m}\|^2. \qquad (7)$$

This is certainly a more restricted condition than (5) and so will lead to the elimination of more points from the tree. Note that (7) will not result in missing any lattice points from (2) since we have used a lower bound for the remainder of the cost in (4). Assuming that we have some way of computing a lower bound, we state the modification of the standard sphere decoder algorithm based on the use of (7) with $LB > 0$. The algorithm uses S-E strategy with the radius update.

Input: Q, R, x, $y = Q^*x$, $d = \hat{d}$, $ll_{1:m} = 0_{1\ m}$.

1. Set $k = m$, $d_m^2 = d^2$, $y_{m|m+1} = y_m$
2. (Bounds for s_k) Set $ub(s_k) = \left\lfloor \frac{\sqrt{d_k^2-(d^2-\hat{d}^2)}+y_{k|k+1}}{r_{k,k}} \right\rfloor$, $lb(s_k) = \left\lceil \frac{-\sqrt{d_k^2-(d^2-\hat{d}^2)}+y_{k|k+1}}{r_{k,k}} \right\rceil$, $l_k = \left\lfloor \frac{lb(s_k)+ub(s_k)+1}{2} \right\rfloor$, $u_k = l_k + 1$
3. (Zig-zag through s_k) if $ll_k = 0$, $s_k = l_k$, $l_k = l_k - 1$, $ll_k = 1$, otherwise $s_k = u_k$, $u_k = u_k + 1$, $ll_k = 0$.
   If $lb(s_k) \leq s_k \leq ub(s_k)$, go to 4, else go to 5.
4. if $LB + (y_{k|k+1} - r_{k,k}s_k)^2 - d_k^2 + (d^2 - \hat{d}^2) > 0$, go to 3, else go to 6.
5. (Increase k) $k = k + 1$; if $k = m + 1$ terminate algorithm, else go to 3.
6. (Decrease k) If $k = 1$ go to 7. Else $k = k - 1$, $y_{k|k+1} = y_k - \sum_{j=k+1}^{m} r_{k,j}s_j$, $d_k^2 = d_{k+1}^2 - (y_{k+1|k+2} - r_{k+1,k+1}s_{k+1})^2$, and go to 2.
7. Solution found. Save s and its distance from x, $\hat{d} = d_m^2 - d_1^2 + (y_1 - r_{1,1}s_1)^2$, and go to 3.

[1] $LB = 0$, of course, simply corresponds to the standard sphere decoder.

Now clearly, the tighter the lower bound LB, the more points that will be pruned from the tree. Of course, we cannot hope to find the optimal lower bound since this requires solving an integer least-squares problem (which was our original problem to begin with). Therefore in what follows we shall consider obtaining lower bounds on the integer least-squares problem

$$\min_{\mathbf{s}_{1:k-1}\in\mathcal{D}\subset\mathcal{Z}^{k-1}} \|\mathbf{z}_{1:k-1} - R_{1:k-1,1:k-1}\mathbf{s}_{1:k-1}\|^2, \qquad (8)$$

where we have defined $\mathbf{z}_{1:k-1} = \mathbf{y}_{1:k-1} - R_{1:k-1,k:m}\mathbf{s}_{k:m}$.

Before proceeding any further, however, we note that finding a lower bound on (8) requires *some* computational effort. Therefore, it is a natural question to ask whether the benefits of additional pruning outweigh the additional complexity incurred by computing a lower bound. An even more basic question perhaps, is what are the potential pruning capabilities of the lower bounding technique which we use to modify the sphere decoding algorithm. To illustrate this consider the lower bound on (8) that has been proposed in [9], which is based on duality and may be stated as

$$\begin{aligned} LB_{SDP} = \max \quad & \mathrm{Tr}(\Lambda) \\ \text{subject to} \quad & Q \succeq \Lambda, \quad \Lambda \ \text{ is diagonal}, \end{aligned} \qquad (9)$$

where

$$Q = \begin{bmatrix} \frac{1}{4}R_{1:k-1,1:k-1}^T R_{1:k-1,1:k-1} & -\frac{1}{2}R_{1:k-1,1:k-1}^T \mathbf{z}_{1:k-1} \\ -\frac{1}{2}\mathbf{z}_{1:k-1}^T R_{1:k-1,1:k-1} & \mathbf{z}_{1:k-1}^T\mathbf{z}_{1:k-1} \end{bmatrix}.$$

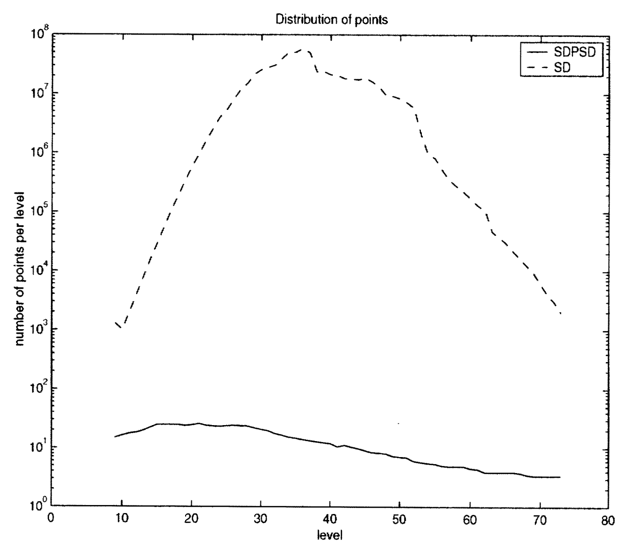

Fig. 1. Comparison of number of points per level in search tree, $m = 100$, snr=10dB,$\mathcal{D} = \{-\frac{1}{2}, \frac{1}{2}\}^{k-1}$

Figure 1 compares the number of points on each level of the search tree visited by the basic sphere decoding algorithm with the corresponding number of points visited by the sphere decoding algorithm which employs a lower bound LB_{SDP} of (9). We refer to the former as the SD-algorithm and to the latter as the SDSDP-algorithm. As evident from Figure 1, the number of points in the search tree visited by the SDSDP-algorithm is several orders of magnitude smaller than that visited by the SD-algorithm. [The additional pruning of the search tree nodes varies across the tree levels. The total number of the points visited by the SDSDP-algorithm is roughly 10^5 times smaller than that visited by the SD-algorithm.] Therefore, a good lower bound can help prune the tree much more efficiently than the standard sphere decoding alone. However, computing LB_{SDP} requires solving an SDP per each point in the search tree. Since this requires computational effort roughly cubic in k, the total flop count savings are not as significant as the savings in the number of examined tree points shown in Figure 1. Therefore, there is merit in searching for lower bounds that may not be as tight as (9), but which require significantly lower computational effort.

In this paper, we deduce such a lower bound as a special case of a general family of bounds on integer least-squares problems that was developed in [10] using ideas from H^∞ estimation theory. We show that the effort required to compute the bound is linear in k. We demonstrate that, when employed for additional pruning in sphere decoding, the newly proposed bound provides significant flop count savings.

II. H^∞ BASED LOWER BOUND

In this section, we derive the lower bound LB in (7) based on H^∞ theory. To simplify notation, we rewrite (8) as

$$\min_{\mathbf{a}\in\mathcal{D}\subset\mathcal{Z}^{k-1}} \|\mathbf{b}-L\mathbf{a}\|^2, \tag{10}$$

where $\mathbf{a}=\mathbf{s}_{1:k-1}$, $\mathbf{b}=\mathbf{z}_{1:k-1}$, and $L=R_{1:k-1,1:k-1}$.

Consider an estimation problem where $\mathbf{a}$ and $\mathbf{b}-L\mathbf{a}$ are unknown vectors, $\mathbf{b}$ is the observation, and the quantities we want to estimate are $\mathbf{a}$ and $\mathbf{b}$. In the H^∞ framework, the goal is to construct estimators $\hat{\mathbf{a}}=f_1(\mathbf{b})$ and $\hat{\mathbf{b}}=f_2(\mathbf{b})$, such that for some given γ and diagonal matrix $D>0$, we have

$$\frac{\|\mathbf{a}-\hat{\mathbf{a}}\|^2+\|\mathbf{b}-\hat{\mathbf{b}}\|^2}{\mathbf{a}^*D\mathbf{a}+\|\mathbf{b}-L\mathbf{a}\|^2}\le\gamma^2 \tag{11}$$

for *all* $\mathbf{a}$ and $\mathbf{b}$ (see, e.g., [11]).

Obtaining the desired lower bound from (11) is now straightforward. Note that for all $\mathbf{a}$ and $\mathbf{b}$ we can now write

$$\|\mathbf{b}-L\mathbf{a}\|^2\ge\gamma^{-2}\left(\|\mathbf{a}-\hat{\mathbf{a}}\|^2+\|\mathbf{b}-\hat{\mathbf{b}}\|^2\right)-\mathbf{a}^*D\mathbf{a}, \tag{12}$$

and, in particular,

$$\min_{\mathbf{a}\in\mathcal{D}}\|\mathbf{b}-L\mathbf{a}\|^2\ge\min_{\mathbf{a}\in\mathcal{D}}\left(\gamma^{-2}\|\mathbf{a}-\hat{\mathbf{a}}\|^2-\mathbf{a}^*D\mathbf{a}\right)+\gamma^{-2}\|\mathbf{b}-\hat{\mathbf{b}}\|^2. \tag{13}$$

Note that the minimization on the RHS of (13) is straightforward since it can be done componentwise. Thus, for any H^∞ estimators, $\hat{\mathbf{a}}=f_1(\mathbf{b})$ and $\hat{\mathbf{b}}=f_2(\mathbf{b})$, (13) provides a readily computable lower bound. The issue, of course, is to obtain the best $\hat{\mathbf{a}}$ and $\hat{\mathbf{b}}$ (and D and γ). To this end, let us assume that the estimators are linear, i.e., $\hat{\mathbf{a}}=K_1\mathbf{b}$ and $\hat{\mathbf{b}}=K_2\mathbf{b}$ for matrices K_1 and K_2 of the appropriate size. Introducing $\mathbf{c}=\begin{bmatrix}D^{1/2}\mathbf{a}\\ \mathbf{b}-L\mathbf{a}\end{bmatrix}$ and $T=\begin{bmatrix}(I-K_1L)D^{-1/2} & -K_1\\ (I-K_2)LD^{-1/2} & I-K_2\end{bmatrix}$ we have from (12) that for all $\mathbf{c}$ it must hold that

$$\mathbf{c}^*T^*T\mathbf{c}\le\gamma^2\mathbf{c}^*I\mathbf{c},$$

(see [11]). Since T is square, this implies either of the equivalent inequalities

$$TT^*\le\gamma^2I \quad\text{or}\quad T^*T\le\gamma^2I. \tag{14}$$

The tighter the bound in (14), the tighter will be the bound in (13). Hence we will attempt to choose K_1 and K_2 to make $\gamma^{-2}TT^*$ as close to identity as possible.

To this end post multiply

$$T=\begin{bmatrix}D^{-1/2} & 0\\ LD^{-1/2} & I\end{bmatrix}-\begin{bmatrix}K_1\\ K_2\end{bmatrix}\begin{bmatrix}LD^{-1/2} & I\end{bmatrix},$$

with the unitary matrix

$$=\begin{bmatrix}\nabla^{-1} & D^{-1/2}L^*\Delta^{-*}\\ -LD^{-1/2}\nabla^{-1} & \Delta^{-*}\end{bmatrix},$$

where $D^{-1/2}L^*LD^{-1/2}+I=\nabla^*\nabla$ and $LD^{-1}L^*+I=\Delta\Delta^*$, to obtain

$$T=\begin{bmatrix}A & B\\ 0 & C\end{bmatrix} \tag{15}$$

where $A=D^{-1/2}\nabla^{-1}$, $B=D^{-1}L^*\Delta^{-*}-K_1\Delta$, and $C=(I-K_2)\Delta$. Thus $TT^*\le\gamma^2I$ leads to

$$\begin{bmatrix}AA^*+BB^* & BC^*\\ CB^* & CC^*\end{bmatrix}\le\gamma^2I. \tag{16}$$

Note that we have many degrees of freedom to choose from. The point is to make judicious choices. To simplify things, let us choose K_2 such that $CC^*=\gamma_1^2I$ for some $0\le\gamma_1\le\gamma$. To make half the eigenvalues of $\gamma^{-2}TT^*$ unity, we can now set the Schur complement of the $(2,2)$ entry to zero, i.e.,

$$AA^*+BB^*-\gamma^2I-BC^*(CC^*-\gamma^2I)^{-1}CB^*=0. \tag{17}$$

Using $CC^*=C^*C=\gamma_1^2I$, it easily follows that

$$BB^*=(1-\frac{\gamma_1^2}{\gamma^2})(\gamma^2I-AA^*).$$

This implies that we should choose K_1 according to

$$K_1=D^{-1}L^*(LD^{-1}L^*+I)^{-1}-B\Delta^{-1}. \tag{18}$$

From the (1,1) entry of (16) we must have

$$\gamma^2I-(AA^*+BB^*)\ge0,$$

or in other words

$$\gamma^2\ge\frac{1}{\lambda_{min}(D+L^*L)}.$$

We can put everything together in the following theorem:

Theorem 1: Consider the integer least-squares problem (10). Then for any $\gamma^2\ge\frac{1}{\lambda_{min}(D+L^*L)}$, $0\le\gamma_1\le\gamma$, and any matrices D, B, and Δ satisfying $\Delta\Delta^*=I+LD^{-1}L^*$ and $BB^*=(1-\frac{\gamma_1^2}{\gamma^2})(\gamma^2I-(D+L^*L)^{-1})$

$$\min_{\mathbf{a}\in\mathcal{D}}\|\mathbf{b}-L\mathbf{a}\|^2\ge\min_{\mathbf{a}\in\mathcal{D}}\gamma^{-2}\|\mathbf{a}-D^{-1}L^*(LD^{-1}L^*+I)^{-1}\mathbf{b}+B\Delta^{-1}\mathbf{b}\|^2-\mathbf{a}^*D\mathbf{a}+\frac{\gamma_1^2}{\gamma^2}\|\Delta^{-1}\mathbf{b}\|^2.$$

Proof: Follows from the previous discussion, noting that

$$||\mathbf{b}-\hat{\mathbf{b}}||^2 = ||(I-K_2)\mathbf{b}||^2 = ||C\Delta^{-1}\mathbf{b}||^2 = \gamma_1^2||\Delta^{-1}\mathbf{b}||^2$$

and

$$AA^* = (D+L^*L)^{-1}.$$

■

The above Theorem can be written as follows.

Corollary 1: Consider the setting of the Theorem 1. Then

$$\min_{\mathbf{a}\in\mathcal{D}} ||\mathbf{b}-L\mathbf{a}||^2 \geq \min_{\mathbf{a}\in\mathcal{D}} \gamma^{-2}||\mathbf{a}-D^{-1}L^*(LD^{-1}L^*+I)^{-1}\mathbf{b} + B\phi||^2 - \mathbf{a}^*D\mathbf{a} + \frac{\gamma_1^2}{\gamma^2}||\phi||^2, \quad (19)$$

where B is the unique symmetric square root of $(1-\frac{\gamma_1^2}{\gamma^2})(\gamma^2 I - (D+L^*L)^{-1})$ and ϕ is *any* vector of the squared length $\mathbf{b}^*(I+LD^{-1}L^*)^{-1}\mathbf{b}$.

It should be noted that we have several degrees of freedom in choosing the parameters $(\gamma_1, \gamma, D, \phi)$ to tighten the bound in (19) as much as possible. Optimizing simultaneously over all these parameters appears to be rather difficult. However, we can simplify the problem and let $\gamma_1 \to \gamma$. This has two benefits: it maximizes the second term in (19) and it sets $B=0$ so that we need not worry about the vector ϕ. Finally, to maximize the first term, we need to take γ as its smallest value, i.e., we set

$$\gamma^2 = \frac{1}{\lambda_{min}(D+L^*L)}.$$

This leads to the following result.

Corollary 2: Consider the setting of the Theorem 1. Then

$$\min_{\mathbf{a}\in\mathcal{D}} ||\mathbf{b}-L\mathbf{a}||^2 \geq \min_{\mathbf{a}\in\mathcal{D}} (\lambda_{min}(D+L^*L)||\mathbf{a}- D^{-1}L^*(LD^{-1}L^*+I)^{-1}\mathbf{b}||^2 - \mathbf{a}^*D\mathbf{a}) + \mathbf{b}^*(I+LD^{-1}L^*)^{-1}\mathbf{b}. \quad (20)$$

In [10], we use (20) to deduce several special cases of lower bounds. In particular, appropriate choices of the parameters in (20) lead to bounds which correspond to spherical and polytope relaxations of the search space. As one might expect, the tighter the bound, the more branches are eliminated from the search tree; however, the greater the computational effort required to find it. In fact, finding bounds that correspond to spherical and polytope relaxations of the search space require computational effort that is at least quadratic in k.

III. EIGEN-BOUND

In general, the lower bound in (20) still requires an optimization over the diagonal matrix $D \geq 0$. A particular choice that may be useful from a computational point of view is $D = \alpha I$, for some α. However in this section we shall take the most simple choice of $D=0$. Although this may appear to yield too loose a lower bound, it turns out that it still yields an algorithm with improved flop count compared to the standard sphere decoder. The key observation is that, with $D=0$, it is possible to perform all the computations required at any point in the tree in time that is linear in the dimension. (The standard sphere decoder also requires a linear number of operations per point.) For reasons to be made in a moment we will refer to this bound as the *eigen-bound*.

When $D=0$, (20) reduces to

$$LB_{eigb} = \lambda_{min}(L^*L) \min_{\mathbf{a}\in\mathcal{D}} ||\mathbf{a}-L^{-1}\mathbf{b}||^2. \quad (21)$$

Clearly, since (21) is a special case of (20), it is a valid lower bound on the integer least-squares problem (8). Note that it appears as if (21) may not be a good bound since $\lambda_{min}(L^*L)$ may become very small. However, since the minimization $\min_{\mathbf{a}\in\mathcal{D}} ||\mathbf{a}-L^{-1}\mathbf{b}||^2$ in (21) is performed over integers, the resulting bound turns out to be still sufficiently large to serve our purposes (i.e., tree pruning in sphere decoding), especially in the case of higher symbol constellations. Furthermore, we will show that the additional computation that we need to perform to find LB_{eigb} in a node at a level k in the search tree is linear in k.

The key observation which enables efficient computation of the lower bound LB_{eigb} in (21) is that the value of the vector $L^{-1}\mathbf{b}$ can be propagated as the search progresses down in the tree. Before proceeding any further, we will simplify the notation. First, we recall that

$$\begin{aligned} L &= R_{1:k-1,1:k-1} \quad \text{and} \\ \mathbf{b} &= \mathbf{z}_{1:k-1} = \mathbf{y}_{1:k-1} - R_{1:k-1,k:m}\mathbf{s}_{k:m}. \end{aligned}$$

Let us denote $F_{1:k-1,1:k-1} = R^{-1}_{1:k-1,1:k-1}$, and introduce

$$\begin{aligned} \mathbf{f}^{(k-1)} &= L^{-1}\mathbf{b} = F_{1:k-1,1:k-1}\mathbf{z}_{1:k-1} = \\ &= F_{1:k-1,1:k-1}(\mathbf{y}_{1:k-1} - R_{1:k-1,k:m}\mathbf{s}_{k:m}). \end{aligned}$$

We wish to show that computation of the vector $\mathbf{f}^{(k-2)}$, using the already calculated vector $\mathbf{f}^{(k-1)}$, requires number of operations that is linear in k. Now, introducing $X_1 = F_{1:k-2,1:k-2}, X_2 = F_{1:k-2,1:k-1}, X_3 = F_{k-1,1:k-1}, X_4 = F_{1:k-2,k-1}, X_5 = F_{k-1,k-1}$, we can write

$$\begin{aligned} \mathbf{f}^{(k-1)} &= F_{1:k-1,1:k-1}(\mathbf{y}_{1:k-1} - R_{1:k-1,k:m}\mathbf{s}_{k:m}) \\ &= \begin{bmatrix} X_2 \\ X_3 \end{bmatrix} \mathbf{y}_{1:k-1} - \begin{bmatrix} X_1 & X_4 \\ 0 & X_5 \end{bmatrix} \begin{bmatrix} R_{1:k-2,k:m} \\ R_{k-1,k:m} \end{bmatrix} \mathbf{s}_{k:m} \\ &= \begin{bmatrix} X_2\mathbf{y}_{1:k-1} \\ X_3\mathbf{y}_{1:k-1} \end{bmatrix} - \begin{bmatrix} X_1R_{1:k-2,k:m} + X_4R_{k-1,k:m} \\ X_5R_{k-1,k:m} \end{bmatrix} \mathbf{s}_{k:m} = \\ &\begin{bmatrix} X_1(\mathbf{y}_{1:k-2} - R_{1:k-2,k:m}\mathbf{s}_{k:m}) + X_4(y_{k-1} - R_{k-1,k:m}\mathbf{s}_{k:m}) \\ X_3\mathbf{y}_{1:k-1} - X_5R_{k-1,k:m}\mathbf{s}_{k:m} \end{bmatrix} \end{aligned} \quad (22)$$

From (22), we see that

$$\mathbf{f}^{(k-1)}_{1:k-2} = X_1\mathbf{y}_{1:k-2} - X_1R_{1:k-2,k:m}\mathbf{s}_{k:m} + X_4y_{k-1} - X_4R_{k-1,k:m}\mathbf{s}_{k:m} \quad (23)$$

Similar to the way we expressed $\mathbf{f}^{(k-1)}$ in (22), we can write $\mathbf{f}^{(k-2)}$ for a given k,

$$\begin{aligned} \mathbf{f}^{(k-2)} &= X_1(\mathbf{y}_{1:k-2} - R_{1:k-2,k-1:m}\mathbf{s}_{k-1:m}) \\ &= X_1\mathbf{y}_{1:k-2} - X_1\begin{bmatrix} R_{1:k-2,k-1} & R_{1:k-2,k:m} \end{bmatrix} \begin{bmatrix} s_{k-1} \\ \mathbf{s}_{k:m} \end{bmatrix} \\ &= X_1(\mathbf{y}_{1:k-2} - R_{1:k-2,k:m}\mathbf{s}_{k:m} - R_{1:k-2,k-1}s_{k-1}) \end{aligned} \quad (24)$$

Using expressions (23) and (24), we can relate $\mathbf{f}^{(k-1)}$ and $\mathbf{f}^{(k-2)}$ in the following manner,

$$\mathbf{f}^{(k-2)} = \mathbf{f}^{(k-1)}_{1:k-2} + F_{1:k-2,k-1}R_{k-1,k:m}\mathbf{s}_{k:m} - F_{1:k-2,k-1}y_{k-1} - F_{1:k-2,1:k-2}R_{1:k-2,k-1}s_{k-1}. \quad (25)$$

All operations in the recursion (25) are linear, except for the matrix-vector multiplication $F_{1:k-2,1:k-2}R_{1:k-2,k-1}$ which is quadratic. However, this multiplication needs to be computed only once for each level of the tree and the resulting term is used for computing (25) in all points at a level which are visited by the algorithm. Therefore, this multiplication may be treated as a part of pre-processing, i.e., we compute it for all k before actually running the sphere decoding algorithm. Hence, updating vector $L^{-1}\mathbf{b}$ in the lower bound (21) requires a computational effort that is linear in k. Furthermore, since it is done component-wise, the minimization in (21) also has complexity that is linear in k. Hence we conclude that the complexity of computing the eigen-bound is linear in k.

Now we state the complete sphere decoding algorithm with modification based on the use of the previously derived eigen-bound.

Input: Q, R, x, $y = Q^*x$, $d = \hat{d}$, $F = R^{-1}$, $\lambda_k = \min \text{eig}(F^*_{1:k-1,1:k-1}F_{1:k-1,1:k-1}), 2 \leq k \leq m$, $FR_{1:k-1,k} = F_{1:k-1,1:k-1}R_{1:k-1,k}, 2 \leq k \leq m$, $ll_{1:m} = 0_{1\ m}$.

1. Set $k = m$, $d_m^2 = d^2$, $y_{m|m+1} = y_m$
2. (Bounds for s_k) Set $ub(s_k) = \lfloor \frac{\sqrt{d_k^2-(d^2-\hat{d}^2)}+y_{k|k+1}}{r_{k,k}} \rfloor$, $lb(s_k) = \lceil \frac{-\sqrt{d_k^2-(d^2-\hat{d}^2)}+y_{k|k+1}}{r_{k,k}} \rceil$, $l_k = \lfloor \frac{lb(s_k)+ub(s_k)+1}{2} \rfloor$, $u_k = l_k + 1$
3. (Zig-zag through s_k)
   if $ll_k = 0$
   $s_k = l_k$, $l_k = l_k - 1$, $ll_k = 1$
   else
   $s_k = u_k$, $u_k = u_k + 1$, $ll_k = 0$
   end if

   If $lb(s_k) \leq s_k \leq ub(s_k)$, go to 4, else go to 5.
4. a) (Compute LB)
   if $k = m$
   $\mathbf{f}^{k-1} = F_{1:k-1,1:k-1}(y_{1:k-1} - R_{1:k-1,k:m}\mathbf{s}_{k:m})$
   else if $1 < k < m$
   $\mathbf{f}^{k-1} = \mathbf{f}^{k}_{1:k-1} + F_{1:k-1,k}R_{k,k+1:m}\mathbf{s}_{k+1:m} - F_{1:k-1,k}y_k - FR_{1:k-1,k}s_k$
   end if
   if $k > 1$ $LB = \lambda_k \min_{\mathbf{a}\in\mathcal{D}} ||\mathbf{a}-\mathbf{f}^{k-1}||^2$ else $LB = 0$
   b) if $LB + (y_{k|k+1} - r_{k,k}s_k)^2 - d_k^2 + (d^2 - \hat{d}^2) > 0$, go to 3, else go to 6.
5. (Increase k) $k = k + 1$; if $k = m + 1$ terminate algorithm, else go to 3.
6. (Decrease k) If $k = 1$ go to 7. Else $k = k - 1$, $y_{k|k+1} = y_k - \sum_{j=k+1}^{m} r_{k,j}s_j$, $d_k^2 = d_{k+1}^2 - (y_{k+1|k+2} - r_{k+1,k+1}s_{k+1})^2$, and go to 2.
7. Solution found. Save s and its distance from x, $\hat{d} = d_m^2 - d_1^2 + (y_1 - r_{1,1}s_1)^2$, and go to 3.

A. Simulation results

We refer to the modification of the sphere decoding algorithm which makes use of the lower bound LB_{eigb} in (21) as EIGSD-algorithm and study its expected computational complexity in Figure 2. In particular, Figure 2 compares the expected complexity of the EIGSD-algorithm to the expected complexity of the standard sphere decoder algorithm (SD-algorithm). We employ the algorithms for detection in the multi-antenna communication system with 6 antennas, where the components of the transmitted symbol vectors are points in a 256-QAM constellation. Note that the signal-to-noise ratio in Figure 2 is defined as $\text{snr} = 10\log_{10}\frac{255m}{12\sigma^2}$, where σ^2 is the variance of each component of the noise vector $\mathbf{w}$. Both algorithms choose the initial search radius statistically as in [3], and perform the Schnor-Euchner search strategy updating the radius every time the bottom of the tree is reached. As the simulation results in Figure 2 indicate, the EIGSD-algorithm runs more than 4.5 times faster than the SD-algorithm. [Needless to say, the bit error-rate performance of both algorithms coincide with the maximum-likelihood.]

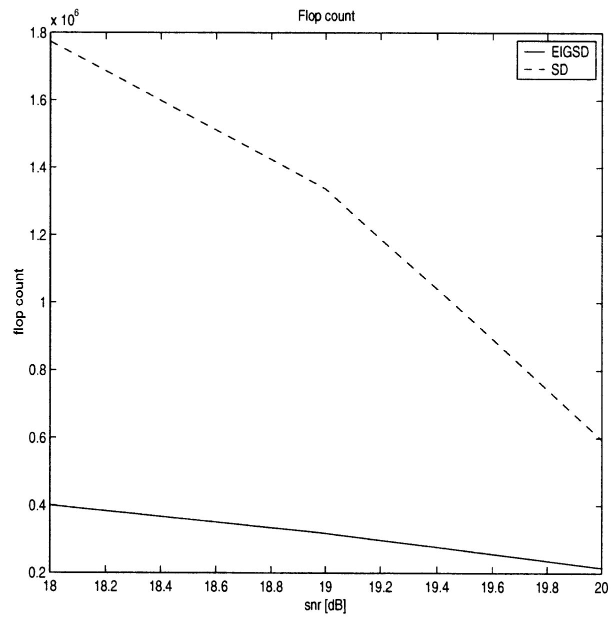

Fig. 2. Eigen-bound, $m = 12$, $\mathcal{D} = \{-\frac{15}{2}, -\frac{13}{2}, \ldots, \frac{13}{2}, \frac{15}{2}\}^{k-1}$

IV. Summary and Discussion

In this paper, we attempted to improve the computational complexity of sphere decoding in the regimes of low SNR and/or high dimensions, by further pruning points from the search tree. The main idea is based on computing a lower bound on the remainder of the cost function as we descend down the search tree (the standard sphere decoder simply uses a lower bound of zero). If the sum of the current cost at a given node and the lower bound on the remaining cost from that node exceeds the cost of an already found solution, then that node (and all its descendants) are pruned from the search tree. In this sense, we are essentially using a "branch and bound" technique.

Adding a lower bound on the remainder of the cost function has the potential to prune the search tree significantly more than the standard sphere decoder algorithm prunes. However, more significant pruning of the search tree does not, in general, guarantee that the modified algorithm will perform faster than the standard sphere decoder algorithm. This is due to the additional computations required by the modified algorithm to find a lower bound in each node of the search tree. Hence a natural conclusion of our work; the lower bound on one hand has to be as tight as possible in order to prune the search tree as much as possible, and on the other hand it should be efficiently computable. Led by these two main characteristics, in this paper we first discuss a general framework for computing the desired lower bounds. The framework is based on a connection, that we established, between the H^∞ estimation theory and the problem of bounding the integer least-squares problem. Several special cases of the lower bounds on the integer least-squares problem could be deduced from this framework. Two of such bounds, which correspond to relaxation of the search space to a sphere or a polytope, were considered elsewhere. In this paper, another special case which corresponds to a bound via smallest eigenvalue, is proposed. The computation of the latter bound requires smaller effort than any of the other, earlier considered, bounds.

Simulation results show that the modified sphere decoding algorithm, incorporating the lower bound based on the smallest eigenvalue, outperforms the basic sphere decoder algorithm. This is not always the case with the aforementioned alternative bounds, and is due to the efficient implementation which is only linear in the dimension of the problem.

We should also point out that although we derive it in order to improve the speed of the sphere decoder algorithm, the lower bound on the integer least-squares problem is an interesting result in itself. In particular, the proposed H^∞-estimation approach provides a general framework for the efficient computation of lower bounds on the difficult integer least-squares problem and may find applications beyond the scope of the current paper.

The results we present indicate potentially significant improvements in the speed of the sphere decoder algorithm. However, we should note that the proposed H^∞-estimation based framework for bounding integer least-squares problem is only partially utilized. In particular, there are several degrees of freedom in the general H^∞-based bound that are not exploited. Instead, a simple and not as tight as possible but computationally efficient special case is used. It is certainly of interest to attempt to exploit the previously mentioned degrees of freedom and tighten the lower bound. If, in addition, this can be done efficiently, it might even further improve the speed of the modified sphere decoding algorithm. We suspect that the choice of $D = \alpha I$, for an appropriately chosen $\alpha \geq 0$ should lead to better results.

REFERENCES

[1] 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, pp. 463-471, April 1985.

[2] G. Golub and C. Van Loan, "*Matrix Computations*," John Hopkins University Press, 3rd edition, 1996.

[3] B. Hassibi, and H. Vikalo, "On the sphere decoding algorithm: Part I, The expected complexity," to appear in *IEEE Trans. on Signal Processing*.

[4] H. Artes, D. Seethaler and F. Hlawatsch, "Effi cient detection algorithms for MIMO channels: A geometrical approach to approximate ML detection," *IEEE Trans. on Signal Processing*, November, 2003.

[5] H. Vikalo, B. Hassibi and T. Kailath, "Iterative decoding for MIMO channels via modifi ed sphere decoder", *IEEE Trans. on Wireless Communications*, November, 2004.

[6] B. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel", *IEEE Trans. on Communications*, March, 2003.

[7] R. Gowaikar and B. Hassibi, "Effi cient maximum-likelihood decoding via statistical pruning", submitted to *IEEE Trans. on Information Theory*.

[8] M. O. Damen, A. Chkeif and J.-C. Belfore, "Lattice code decoder for space-time codes"' *IEEE Comm. Let.* pp. 161-163, May 2000.

[9] M. Stojnic, H. Vikalo and B. Hassibi, "A branch and bound approach to speed up the sphere decoder ", *ICASSP, International conference on Acoustics, Signal and Speech Processing*, Philadelphia, March, 2005.

[10] M. Stojnic, H. Vikalo and B. Hassibi, "A combination of branch and bound and H^∞ approaches to speed up the sphere decoder ", to be submitted to *IEEE Trans. on Signal Processing*.

[11] B. Hassibi, A. H. Sayed, and T. Kailath, *Indefinite-Quadratic Estimation and Control*, SIAM, 1999.