# Speeding up the Sphere Decoder With $H^\infty$ and SDP Inspired Lower Bounds

Mihailo Stojnic, Haris Vikalo, and Babak Hassibi

***Abstract*—It is well known that maximum-likelihood (ML) decoding in many digital communication schemes 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 $N$ and signal-to-noise ratios (SNRs), the sphere decoding algorithm can be used to find the exact ML solution with an expected complexity that is often less than $N^3$. 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 decoding algorithm. The search tree is pruned by computing lower bounds on the optimal value of the objective function as the algorithm proceeds to descend down the search tree. We observe a tradeoff between the computational complexity required to compute a lower bound and the size of the pruned tree: the more effort we spend in computing a tight lower bound, the more branches that can be eliminated in the tree. Using ideas from semidefinite program (SDP)-duality theory and $H^\infty$ estimation theory, we propose general frameworks for computing lower bounds on integer least-squares problems. We propose two families of algorithms, one that is appropriate for large problem dimensions and binary modulation, and the other that is appropriate for moderate-size dimensions yet high-order constellations. We then show how in each case these bounds can be efficiently incorporated in the sphere decoding algorithm, often resulting in significant improvement of the expected complexity of solving the ML decoding problem, while maintaining the exact ML-performance.**

***Index Terms*—Branch-and-bound algorithm, $H^\infty$ estimation, convex optimization, expected complexity, integer least-squares, maximum-likelihood (ML) detection, sphere decoding.**

## 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 $\mathbf{x}\in\mathcal{R}^n$, $H\in\mathcal{R}^{n\times m}$ 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 $H\mathbf{s}$ lies within a sphere of some adequately chosen radius $d_s$ centered at $\mathbf{x}$, i.e., on finding all $\mathbf{s}$ such that

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

and then choosing the one that minimizes the objective function. Using the $QR$-decomposition $H=[\,Q_1 \quad Q_2\,]\begin{bmatrix} R \\ 0_{n-m\times m}\end{bmatrix}$, where $R$ is $m\times m$ upper triangular, and $Q_1\in R^{n\times m}$ and $Q_2\in R^{n\times(n-m)}$ are such that $Q=[\,Q_1 \quad Q_2\,]$ is unitary, we can reformulate (2) as

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

where we have defined $d^2=d_s^2-\|Q_2^*\mathbf{y}\|^2$ and $\mathbf{y}=Q_1^*\mathbf{x}$.

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

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

for any $2\leq k\leq m$, where the subscripts determine the entries the various vectors and matrices run over (e.g., $\mathbf{y}_{1:k-1}$ is a column vector whose components are $\mathbf{y}_1,\mathbf{y}_2,\ldots,\mathbf{y}_{k-1}$, and similarly $R_{1:k-1,k:m}$ is a $(k-1)\times(m-k+1)$ matrix and $R_{i,k},R_{i,k+1},\ldots,R_{i,m}$ are the components of its $i$th row). A necessary condition for (3) can therefore be obtained by omitting the second term on the right-hand side (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

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

Manuscript received September 3, 2006; revised May 11, 2007. This work was supported in part by the National Science Foundation under grant CCR-0133818, by the David and Lucille Packard Foundation, and by Caltech's Lee Center for Advanced Networking.

The authors are with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125 USA (e-mail: mihailo@systems.caltech.edu; hvikalo@systems.caltech.edu; hassibi@systems.caltech.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2007.906697

This gives all two-dimensional lattice points that satisfy (3); we proceed in a similar fashion until $k = 1$. The sphere decoding algorithm 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). Therefore, 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],[4], and [10].)

The computational complexity of the sphere decoding algorithm depends on how $d$ is chosen. In a digital communication context, $\mathbf{x}$ is the received signal, i.e., a noisy version of the symbol vector $\mathbf{s}$ transmitted across the channel $H$

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

where the entries of the additive noise vector $\mathbf{w}$ are independent, identically distributed (i.i.d.) $\mathcal{N}(0, \sigma^2)$ random variables. In [4], it is shown that, if elements of $H$ are i.i.d. Gaussian with zero mean and unit variance and if the radius is chosen appropriately based on the statistical characteristics of the noise $\mathbf{w}$, then over a wide range of signal-to-noise ratios (SNRs) and problem dimensions, the expected complexity of the sphere decoding algorithm is low and comparable to the complexity of the best known heuristics, which are 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 (in fact as shown in [19] the expected computational complexity of the sphere decoder is always exponential). Increasing the dimension of the problem clearly is useful.[1] Moreover, the use of the sphere decoder in low-SNR situations is also important, especially when one is interested in obtaining soft information to pass onto an iterative decoder (see, e.g., [7] and [8]). One way to reduce the computational complexity is to resort to suboptimal methods based either on heuristics (see, e.g., [6]) or some form of statistical pruning (see [9]). Also, the interested reader may find more about recent improvements and alternative techniques in [20]–[24].

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 decoding algorithm 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 to decrease the complexity is to reduce the size of the tree beyond that which is suggested by (5). To this end, consider a lower bound (LB) on the optimal value of the second term on the RHS of (4)

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

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., $\mathrm{LB} > 0$, we can replace (5) by[2]

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

This is certainly a more restrictive condition than (5), and so will lead to eliminating 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) (for more on branch and bounding ideas, the interested reader may refer to, e.g., [25] and the references therein). Assuming that we have some way of computing a lower bound $\mathrm{LB} > 0$ as suggested above, we state the modification of the standard sphere decoding algorithm based on (7). The algorithm uses the Schnorr–Euchner (S-E) strategy with radius update [11]. [Note that in this paper, we consider several modifications of the sphere decoding algorithm, and all are implemented using Algorithm 1 below. The difference between the various modifications is how the value of LB in step 4 of the algorithm is computed. Also, note that in the Algorithm 1 given below, $\mathcal{D}$ is the full integer lattice while later in the paper in different modifications of the original algorithm it will be restricted to its subsets.]

*Algorithm 1:*

*Input:* $Q$, $R$, $\mathbf{x}$, $\mathbf{y} = Q_1^*\mathbf{x}$, $d = \hat{d}$, $\mathbf{ll}_{1:m} = \mathbf{0}_{1\times m}$.

1. Set $k = m$, $d_m^2 = d^2$, $\mathbf{y}_{m|m+1} = \mathbf{y}_m$.

2. (Bounds for $\mathbf{s}_k$) Set $ub(\mathbf{s}_k) = \lfloor\sqrt{d_k^2 - (d^2 - \hat{d}^2) + \mathbf{y}_{k|k+1}}/r_{k,k}\rfloor$,
$\mathrm{lb}(\mathbf{s}_k) = \lceil -\sqrt{d_k^2 - (d^2 - \hat{d}^2)} + \mathbf{y}_{k|k+1}/r_{k,k}\rceil$,
$l_k = \lfloor \mathrm{lb}(\mathbf{s}_k) + ub(\mathbf{s}_k) + 1/2\rfloor$, $u_k = l_k + 1$.

3. (Zig-zag through $\mathbf{s}_k$)

If $\mathbf{ll}_k = 0$, $\mathbf{s}_k = l_k$, $l_k = l_k - 1$, $\mathbf{ll}_k = 1$, otherwise $\mathbf{s}_k = u_k$, $u_k = u_k + 1$, $\mathbf{ll}_k = 0$.

If $\mathrm{lb}(\mathbf{s}_k) \leq \mathbf{s}_k \leq ub(\mathbf{s}_k)$, go to 4, else go to 5.

4. If

$$\begin{aligned}&\mathrm{LB}(\mathbf{y}_{1:k-1}, R_{1:k-1,1:m}, \mathbf{s}_{k:m})\\ &\qquad + (\mathbf{y}_{k|k+1} - r_{k,k}\mathbf{s}_k)^2 - d_k^2 + (d^2 - \hat{d}^2) > 0\end{aligned}$$

go to step 3, else go to step 6.

5. (Increase $k$) $k = k + 1$; if $k = m + 1$ terminate algorithm, else go to step 3.

6. (Decrease $k$) If $k = 1$ go to step 7. Else
$k = k - 1$, $\mathbf{y}_{k|k+1} = \mathbf{y}_k - \sum_{j=k+1}^{m} r_{k,j}\mathbf{s}_j$,
$d_k^2 = d_{k+1}^2 - (\mathbf{y}_{k+1|k+2} - r_{k+1,k+1}\mathbf{s}_{k+1})^2$, and go to step 2.

7. Solution found. Save $\mathbf{s}$ and its distance from $\mathbf{x}$,
$\hat{d} = d_m^2 - d_1^2 + (\mathbf{y}_1 - r_{1,1}\mathbf{s}_1)^2$, and go to step 3.

[1]Various space–time codes result in integer least-squares problems where the problem dimension is much larger than the number of transmit antennas. Also, in distributed space–time codes for wireless relay networks, the problem dimension is equal to the number of relay nodes which can be quite large [36], [37].

[2]$\mathrm{LB} = 0$, of course, simply corresponds to the standard sphere decoder.

Clearly, the tighter the lower bound LB, the more points 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 focus on obtaining computationally feasible 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, for simplicity, we introduced $\mathbf{z}^{(k-1)} = \mathbf{z}_{1:k-1} = \mathbf{y}_{1:k-1} - R_{1:k-1,k:m}\mathbf{s}_{k:m}$. Also, in the rest of the paper, we will assume

$$\mathcal{D} = \left\{-\frac{M-1}{2}, -\frac{M-3}{2}, \ldots, \frac{M-3}{2}, \frac{M-1}{2}\right\}^m$$

the case which is of interest in communications applications.

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 that we use to modify the sphere decoding algorithm. To illustrate this, consider a simple lower bound (which is only valid in the binary case, i.e., if $\mathcal{D} = \{-(1/2), (1/2)\}^m$) on (8), used earlier in [12] and further considered in Section II, which is based on duality and may be computed by solving the following semidefinite program (SDP):

$$\max_{\Lambda} \ \mathrm{Tr}(\Lambda) \ \text{ subject to } \ Q \succeq \Lambda, \ \ \Lambda \ \text{ is diagonal} \qquad (9)$$

where

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

We mention that bounds of this type are very well known in the literature on semidefinite programming relaxations. More on them and their history can be found in [27]. Here, we would like to only briefly mention the reason for their popularity. Although it is difficult to prove how tight these bounds are, it turns out that in practice they perform very well. On the top of that, the optimization problem given above is *convex* (the objective function is convex and the region of optimization is convex as well), which means that these bounds can be computed very efficiently using a host of numerical methods [5]. Even more surprising, it can be proved that they can be computed in polynomial time.

Although these bounds have been known for a very long time, they attracted enormous interest in the algorithms and optimization areas after the work of [26]. Quite remarkably, in [26] the authors were able to give a hard bound on the performance of the previously mentioned SDP-relaxation bound for a specific case of the matrix $Q$. Since then the SDP-relaxation techniques have become a standard tool in solving complicated combinatorial optimization problems. Naturally, many of these techniques have also been applied in detection problems (see, e.g., [28]–[32] and [35]). More specifically, in [30] and [31], the authors considered applications of SDP relaxation to problems in multiuser detection in CDMA systems. In [28], [29], [32], and [35], the authors applied the SDP relaxation to the problem of maximum-likelihood (ML) detection in multiple-input multiple-output (MIMO) systems (the same one that we consider in this paper). In [35], the authors generalized the applications of SDP algorithm from binary to larger constellations, and in [29] the authors proved that under certain conditions related to the dimension of the problem in high SNR regime the SDP relaxation is tight.

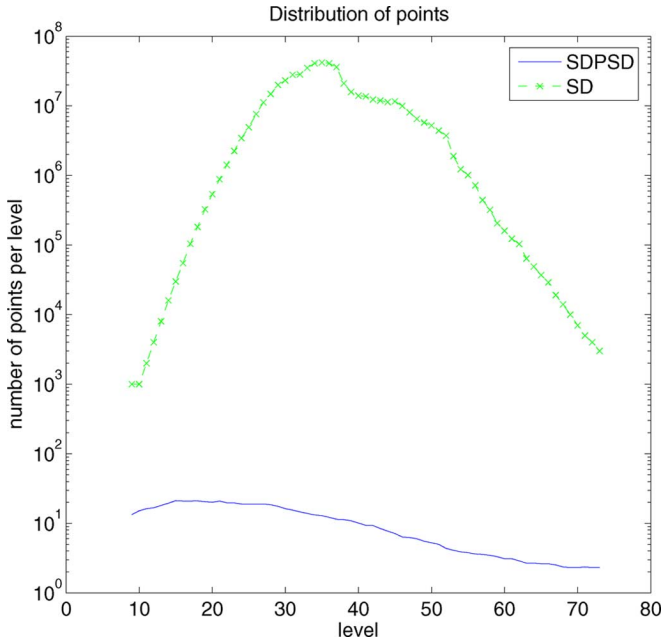



Fig. 1. Comparison of the number of points per level in the search tree visited by the SD and the SDSDP algorithm, $m = 100$, SNR $= 10$ dB, $\mathcal{D} = \{-(1/2), (1/2)\}^m$.

As demonstrated in these references, the SDP technique can be very powerful in producing a very good approximate solution of the original integer least-squares problem. However, in this work, we focus on solving the integer least-squares problem *exactly*, and therefore we will only use its lower-bounding feature. It should also be noted that although in general suboptimal, the SDP technique can sometimes produce the *exact* solution to the original problem (for more on when this happens, see, e.g., [32]).

After a brief chronology on the SDP relaxations, we now turn our attention again to (9). We denote the optimal value of the objective function in (9) by $\mathrm{LB}_{\mathrm{SDP}}$. Fig. 1 compares the number of points (for more on why the number of points may be important, interested readers can refer to [40] and [41]) on each level of the search tree visited by the basic sphere decoding algorithm with the corresponding number of points visited by the modified sphere decoding algorithm which employs $\mathrm{LB}_{\mathrm{SDP}}$ for additional, lower bound based, pruning. We refer to the former as the SD algorithm and to the latter as the SDSDP algorithm. As evident from Fig. 1, for a problem of dimension $m = 100$, SNR $= 10$ dB, and $\mathcal{D} = \{-(1/2), (1/2)\}^m$ (i.e., BPSK modulation), 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 varies across the tree levels. The total number of the points visited by the SDSDP algorithm is roughly $10^6$ 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 $\mathrm{LB}_{\mathrm{SDP}}$ requires solving an SDP per each point in the search tree. The computational effort of solving an SDP is $O(k^{3.5})$, which is significantly greater than the linear complexity of the operations performed by the standard sphere decoding algorithm at every visited node. Furthermore, although the complexity scaling behavior of solving an SDP is provably $O(N^{3.5})$, even for moderately large $N$ ($30 < N < 70$), the real complexity of solving the SDP given in (9) is $\gg N^{3.5}$. On the other hand, the standard sphere decoder performs per each node a number of operations that is $\approx N$. Therefore, there is merit in searching not only for tight lower bounds such as the one in (9), but also for those that may not be as tight but which require significantly smaller computational effort.

In this paper, we therefore introduce a lower bound $\mathrm{LB}_{\mathrm{sdp}}$ on the quantity $\mathrm{LB}_{\mathrm{SDP}}$ which can be computed with complexity linear in $k$. The idea is based on efficient propagation of $\mathrm{LB}_{\mathrm{sdp}}$ through the search tree. We will show that the lower bound $\mathrm{LB}_{\mathrm{sdp}}$ significantly improves the expected complexity of the standard sphere decoder. However, $\mathrm{LB}_{\mathrm{SDP}}$ defined in (9) (and hence $\mathrm{LB}_{\mathrm{sdp}}$) is a valid lower bound only when $\mathcal{D} = \{-(1/2), (1/2)\}^m$. To address the case of general $\mathcal{D}$, we derive another family of lower bounds on integer least-squares problems using ideas from $H^\infty$ estimation theory. We show that several lower bounds that may otherwise be obtained by relaxing the optimization constraints, are in fact special cases of our general $H^\infty$ based lower bound. When employing the above lower bounds to modify sphere decoding, we observe a tradeoff between the computational complexity required to compute a lower bound and the size of the pruned tree: the more effort we spend in computing a tight lower bound, the more branches can be eliminated from the tree. We show that the most computationally efficient among the special cases, the so-called eigenbound, provides a significant improvement in the expected complexity over the sphere decoding algorithm.

The paper is organized as follows. In Section II we derive a computationally efficient lower bound $\mathrm{LB}_{\mathrm{sdp}}$ on $\mathrm{LB}_{\mathrm{SDP}}$. In Section III, we derive the general $H^\infty$ estimation-based lower bound on the integer least-squares problem. In Sections IV, V, and VII, special cases of this general bound are considered. In particular, the so-called spherical relaxation bound is derived in Section IV, the polytope relaxation bound is considered in Section V, and the eigenbound is studied in Section VII. The effects of the aforementioned bounds on the number of search tree points and/or the total expected complexity of the modified sphere decoding algorithm are studied throughout. Some simulations are presented in Section VI, and finally, Section VIII contains conclusions and a discussion of potential extensions of the current work.

## II. SDP-Based Lower Bound

Let $\mathrm{LB}_{\mathrm{SDP}}^{(k-1)}, 1 \le k \le m$ denote the optimal value of the following optimization problem:

$$\mathrm{LB}_{\mathrm{SDP}}^{(k-1)} = \max\ \mathrm{Tr}(\Lambda)\ \text{ subject to } Q_{k-1} \succeq \Lambda,\ \Lambda \text{ is diagonal} \tag{10}$$

where

$$Q_{k-1} = \begin{bmatrix} \frac{1}{4} R^T_{1:k-1,1:k-1} R_{1:k-1,1:k-1} & -\frac{1}{2} R^T_{1:k-1,1:k-1} \mathbf{z}^{(k-1)} \\ -\frac{1}{2} (\mathbf{z}^{(k-1)})^T R_{1:k-1,1:k-1} & (\mathbf{z}^{(k-1)})^T \mathbf{z}^{(k-1)} \end{bmatrix}.$$

In this section, we derive $\mathrm{LB}_{\mathrm{sdp}}^{(k-1)}$, a lower bound on $\mathrm{LB}_{\mathrm{SDP}}^{(k-1)}$. To this end, let $\hat{\Lambda}$ denote the optimal solution of

$$\max\ \mathrm{Tr}(\Lambda)\ \text{ subject to } Q \succeq \Lambda,\ \Lambda \text{ is diagonal} \tag{11}$$

where

$$Q = \begin{bmatrix} \frac{1}{4} R^T R & -\frac{1}{2} R^T \mathbf{y} \\ -\frac{1}{2} \mathbf{y}^T R & \mathbf{y}^T \mathbf{y} \end{bmatrix}.$$

Let $GG^T = (1/4) R^T R - \hat{\Lambda}_{1:m,1:m}$, where $G$ is a lower triangular matrix. Also, let $M = G^{-1} R^T$. Using the fact that the matrices $G$ and $R^T$ are lower triangular, we obtain

$$\begin{aligned} &(G_{1:k-1,1:k-1})^{-1} \\ &= (G^{-1})_{1:k-1,1:k-1}, \\ &\quad G_{1:k-1,1:k-1} (G_{1:k-1,1:k-1})^T \\ &= \frac{1}{4} R^T_{1:k-1,1:k-1} R_{1:k-1,1:k-1} - \hat{\Lambda}_{1:k-1,1:k-1} \end{aligned}$$

and $M_{1:k-1,1:k-1} = (G_{1:k-1,1:k-1})^{-1} R^T_{1:k-1,1:k-1}$. Furthermore, let

$$\begin{aligned} \lambda_k = &(\mathbf{z}^{(k-1)})^T \mathbf{z}^{(k-1)} \\ &- \frac{1}{4} (\mathbf{z}^{(k-1)})^T M^T_{1:k-1,1:k-1} M_{1:k-1,1:k-1} \mathbf{z}^{(k-1)} \end{aligned} \tag{12}$$

and let

$$\mathrm{LB}_{\mathrm{sdp}}^{(k-1)} = \begin{cases} \sum_{i=1}^{k-1} \hat{\Lambda}_{i,i} + \lambda_k & \text{if } \sum_{i=1}^{k-1} \hat{\Lambda}_{i,i} + \lambda_k \ge 0 \\ 0 & \text{if } \sum_{i=1}^{k-1} \hat{\Lambda}_{i,i} + \lambda_k < 0. \end{cases} \tag{13}$$

Now, it is clear that $\mathrm{LB}_{\mathrm{sdp}}^{(k-1)} \le \mathrm{LB}_{\mathrm{SDP}}^{(k-1)}$ since

$$\mathrm{LB}_{\mathrm{sdp}}^{(k-1)} = \mathrm{Tr}(\mathrm{diag}(\hat{\Lambda}_{1,1}, \hat{\Lambda}_{2,2}, \ldots, \hat{\Lambda}_{k-1,k-1}, \lambda_k))$$

and $\mathrm{diag}(\hat{\Lambda}_{1,1}, \hat{\Lambda}_{2,2}, \ldots, \hat{\Lambda}_{k-1,k-1}, \lambda_k)$ is an admissible matrix in (10). On the other hand, if $\sum_{i=1}^{k-1} \hat{\Lambda}_{i,i} + \lambda_k < 0$, $\mathrm{LB}_{\mathrm{sdp}}^{(k-1)} = 0$, and clearly $\mathrm{LB}_{\mathrm{sdp}}^{(k-1)} \le \mathrm{LB}_{\mathrm{SDP}}^{(k-1)}$.

We refer to Algorithm 1 which, in step 4, makes use of $\mathrm{LB}_{\mathrm{sdp}}^{(k-1)}$ as the SDsdp algorithm. The subroutine for computing $\mathrm{LB}_{\mathrm{sdp}}^{(k-1)}$ is given below. Clearly, using $\mathrm{LB}_{\mathrm{sdp}}^{(k-1)}$ instead of $\mathrm{LB}_{\mathrm{SDP}}^{(k-1)}$ results in pruning fewer points in the search tree. However, the computation of $\mathrm{LB}_{\mathrm{sdp}}^{(k-1)}$ is quite more efficient

than the cubic computation of $\mathrm{LB}_{\mathrm{SDP}}^{(k-1)}$. In particular, unlike in the SDSDP-algorithm, we need to solve only one SDP—the one given by (11). Then, we may compute $\mathrm{LB}_{\mathrm{sdp}}^{(k-2)}$ recursively from $\mathrm{LB}_{\mathrm{sdp}}^{(k-1)}$, which requires complexity linear in $k$ [15]. This is shown next.

Recall that $\mathbf{z}^{(k-1)} = \mathbf{y}_{1:k-1} - R_{1:k-1,k:m}\mathbf{s}_{k:m}$. It is easy to see that we can compute $\mathbf{z}^{(k-2)}$ from $\mathbf{z}^{(k-1)}$ as

$$\mathbf{z}^{(k-2)} = \mathbf{z}_{1:k-2}^{(k-1)} - R_{1:k-2,k-1}s_{k-1}. \tag{14}$$

Furthermore, note that $\mathbf{p}^{(k-1)} = M_{1:k-1,1:k-1}\mathbf{z}^{(k-1)}$ can be computed recursively as

$$\begin{aligned}\mathbf{p}^{(k-2)} &= M_{1:k-2,1:k-2}\mathbf{z}^{(k-2)}\\ &= M_{1:k-2,1:k-2}(\mathbf{z}_{1:k-2}^{(k-1)} - R_{1:k-2,k-1}s_{k-1})\\ &= M_{1:k-2,1:k-2}(\mathbf{z}^{(k-1)})_{1:k-2}\\ &\quad - M_{1:k-2,1:k-2}R_{1:k-2,k-1}s_{k-1}\\ &= \mathbf{p}_{1:k-2}^{(k-1)} - (MR)_{1:k-2,k-1}s_{k-1}.\end{aligned} \tag{15}$$

Using $\mathbf{p}^{(k-2)}$ and $\mathbf{z}^{(k-2)}$, we compute $\lambda_{k-1}$ from (12), and $\mathrm{LB}_{\mathrm{sdp}}^{(k-2)}$ from (13). The computation of $\mathrm{LB}_{\mathrm{sdp}}^{(k-2)}$ in each node at the $(m-(k-2))$th level of the search tree requires $4(k-2)$ additions and $2(k-2)$ multiplications. For the basic sphere decoder, the number of operations per each node at the $(m-k+1)$th level is $(2k+17)$. This essentially means that the SDsdp algorithm performs about four times more operations per each node of the tree than the standard sphere decoder algorithm. In other words, if the SDsdp algorithm prunes at least four times more points than the basic sphere decoder, the new algorithm is faster in terms of the flop count.

---

*Subroutine for computing* $\mathrm{LB}_{\mathrm{sdp}}$*:*

*Input:* $R$, $\mathbf{y}$, $\mathbf{s}$, $M = G^{-1}R^T$, $MR$, $\mathbf{p}^{(k-1)}$, $\mathbf{z}^{(k-1)}$.

1. If $k = m$, solve (11) and set $\hat{\Lambda}$ to be the optimal solution of (11);

$$\lambda_k = (\mathbf{z}^{(k-1)})^T\mathbf{z}^{(k-1)} - \frac{1}{4}(\mathbf{z}^{(k-1)})^T M_{1:k-1,1:k-1}^T M_{1:k-1,1:k-1}\mathbf{z}^{(k-1)}.$$

2. If $1 < k < m$,

   2.1 $\mathbf{z}^{(k-1)} = \mathbf{z}_{1:k-1}^{(k)} - R_{1:k-1,k}s_k$, $\mathbf{p}^{(k-1)} = \mathbf{p}_{1:k-1}^{(k)} - (MR)_{1:k-1,k}s_k$.

   2.2 $\lambda_k = (\mathbf{z}^{(k-1)})^T\mathbf{z}^{(k-1)} - (1/4)(\mathbf{p}^{(k-1)})^T\mathbf{p}^{(k-1)}$.

3. If $\lambda_k \geq 0$, $\mathrm{LB}_{\mathrm{sdp}}^{(k-1)} = \sum_{i=1}^{k-1}\hat{\Lambda}_{i,i} + \lambda_k$, otherwise $\mathrm{LB}_{\mathrm{sdp}} = 0$.

---

Fig. 2 compares the expected complexity of the SDsdp algorithm to the expected complexity of the standard sphere decoder algorithm (SD algorithm). The two algorithms are employed for solving a high dimensional binary integer least-squares problem. The SNR in Fig. 2 is defined as $\mathrm{SNR} = 10\log_{10}(m/4\sigma^2)$, where $\sigma^2$ is the variance of each component of the noise vector $\mathbf{w}$. Both algorithms choose an initial search radius statistically as in [4] (the sequence of $\epsilon$'s, $\epsilon = 0.9$, $\epsilon = 0.99$, $\epsilon = 0.999$, etc.), and update the radius every time the bottom of the tree is reached.

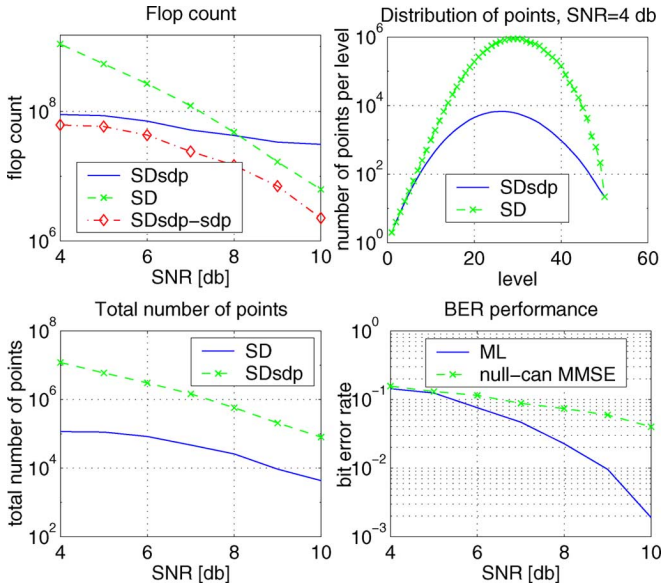



Fig. 2. Computational complexity of the SD and SDsdp algorithms, $m = 50$, $\mathcal{D} = \{-(1/2), (1/2)\}^{50}$.

As can be seen from Fig. 2, the SDsdp algorithm can run up to ten times faster than the SD algorithm at SNR 4–5 dB. At higher SNR, the speed-up decreases, and at SNR 8 dB, the SD algorithm is faster. We can attribute this to the complexity of performing the original SDP (11). In fact, Fig. 2, subplot 1, shows the flop count of the SDsdp, when the computations of the SDP (11) are removed (denoted there as SDsdp-sdp), which can be seen to be uniformly faster than the SD. Thus, the main bottleneck is solving (11), and any computational improvement there can have a significant impact on our algorithm. In our numerical experiments, we solved (11) exactly, i.e., with very high numerical precision, which requires a significant computational effort. This is, of course, not necessary. In fact, how precisely (11) needs to be solved is a very interesting question. For this reason, we emphasize again that constructing faster SDP algorithms would significantly speed up the SDsdp algorithm.

In subplot 2 of Fig. 2, the distribution of points per level in the search tree is shown for both SD and SDsdp algorithms. As stated in [40] and [41], in some practical realizations, the size of the tree may be as important as the overall number of multiplication and addition operations. In subplot 3 of Fig. 2, the comparison of the total number of points kept in the tree by SD and SDsdp algorithms is shown. As expected, the SDsdp algorithm keeps significantly less points in the tree than SD algorithm.

Finally, in subplot 4 of Fig. 2, the comparison of the bit error rate (BER) performance of the exact ML detector (SDsdp algorithm) and the approximate minimum mean-square

error (MMSE) nulling and canceling with optimal ordering heuristic is shown. Over the range of SNRs considered here, the ML detector outperforms the MMSE detector significantly, thereby justifying our efforts in constructing more efficient ML algorithms.

*Remark:* Recall that the lower bound introduced in this section is valid only if the original problem is binary, i.e., $\mathcal{D} = \{-(1/2),(1/2)\}^{k-1}$. A generalization to case $\mathcal{D} = \{-(3/2),-(1/2),(1/2),(3/2)\}^{k-1}$ can be found in [42]. It is not difficult to generalize it to any

$$\mathcal{D} = \left\{-\frac{L-1}{2}, -\frac{L-2}{2}, \ldots, \frac{L-3}{2}, \frac{L-1}{2}\right\}^{k-1}$$

by noting that any $k-1$-dimensional vector whose elements are numbers from $\{-L+1, -L+2, \ldots, L-2, L-1\}$ can be represented as a linear transformation of a $(k-1)(L-1)$-dimensional vector from $\mathcal{D} = \{-(1/2),(1/2)\}^{(k-1)(L-1)}$ (interested readers can find more on this in [35]). However, this significantly increases the dimension of the SDP problem in (11), which may cause our algorithm to be inefficient. Motivated by this, in the following section, we consider a different framework, based on $H^\infty$ estimation theory, which will (as we will see in Section VII) produce as a special case a general lower bound applicable for any $\mathcal{D}$.

## III. $H^\infty$-Based Lower Bound

In this section, we derive the lower bound LB in (7) based on $H^\infty$ estimation theory [13]. In estimation theory, $H^\infty$ is a concept where the goal is to minimize the worst-case energy gain from the disturbances to the estimation errors. In what follows, we will try to exploit mathematical similarity between the problem at hand and the $H^\infty$ concept.

To simplify the notation, we rewrite (8) as

$$\min_{\mathbf{a}\in\mathcal{D}\subset\mathcal{Z}^{k-1}} \|\mathbf{b} - L\mathbf{a}\|^2 \tag{16}$$

where we introduced $\mathbf{a} = \mathbf{s}_{1:k-1}$, $\mathbf{b} = \mathbf{z}_{1:k-1}$, 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 $H^\infty$ 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 $\gamma$, some $\beta \geq 0$, and some diagonal matrix $D > 0$, we have

$$\frac{\beta\|\mathbf{a}-\hat{\mathbf{a}}\|^2 + \|\mathbf{b}-\hat{\mathbf{b}}\|^2}{\mathbf{a}^* D\mathbf{a} + \|\mathbf{b}-L\mathbf{a}\|^2} \leq \gamma^2 \tag{17}$$

for *all* $\mathbf{a}$ and $\mathbf{b}$ (see, e.g., [16]).

Obtaining a desired lower bound from (17) is now straightforward. Note that for all $\mathbf{a}$ and $\mathbf{b}$, we can write

$$\|\mathbf{b}-L\mathbf{a}\|^2 \geq \gamma^{-2}\left(\beta\|\mathbf{a}-\hat{\mathbf{a}}\|^2 + \|\mathbf{b}-\hat{\mathbf{b}}\|^2\right) - \mathbf{a}^* D\mathbf{a} \tag{18}$$

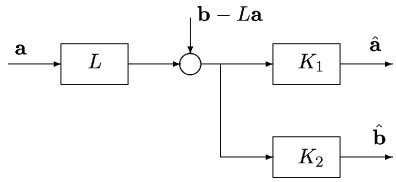



Fig. 3. $H^\infty$ estimation analogy used in deriving a lower bound on integer least-squares problem.

and, in particular

$$\min_{\mathbf{a}\in\mathcal{D}} \|\mathbf{b}-L\mathbf{a}\|^2 \geq \min_{\mathbf{a}\in\mathcal{D}} \left(\gamma^{-2}\beta\|\mathbf{a}-\hat{\mathbf{a}}\|^2 - \mathbf{a}^* D\mathbf{a}\right) + \gamma^{-2}\|\mathbf{b}-\hat{\mathbf{b}}\|^2. \tag{19}$$

Note that the minimization on the RHS of (19) is straightforward since it can be done componentwise (which is why we chose $D > 0$ diagonal). Thus, for any $H^\infty$ estimators $\hat{\mathbf{a}} = f_1(\mathbf{b})$ and $\hat{\mathbf{b}} = f_2(\mathbf{b})$, (19) provides a readily computable lower bound. The issue, of course, is how to obtain the best $\hat{\mathbf{a}}$ and $\hat{\mathbf{b}}$ (and $D$ and $\gamma$). 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 some matrices $K_1$ and $K_2$ (see Fig. 3).

Introducing $\mathbf{c} = \begin{bmatrix} D^{1/2}\mathbf{a} \\ \mathbf{b}-L\mathbf{a} \end{bmatrix}$ and noting that

$$T = \begin{bmatrix} D^{-1/2} & 0 \\ LD^{-1/2} & I \end{bmatrix} - \begin{bmatrix} K_1 \\ K_2 \end{bmatrix} [\,LD^{-1/2} \quad I\,] = \begin{bmatrix} \sqrt{\beta}(I-K_1L)D^{-1/2} & -\sqrt{\beta}K_1 \\ (I-K_2)LD^{-1/2} & I-K_2 \end{bmatrix}$$

maps $\mathbf{c}$ to $\begin{bmatrix} \sqrt{\beta}(\mathbf{a}-\hat{\mathbf{a}}) \\ \mathbf{b}-\hat{\mathbf{b}} \end{bmatrix}$, from (18) we see that for all $\mathbf{c}$ it must hold that

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

(see [16]). Since $T$ is square, this implies either of the equivalent inequalities

$$TT^* \leq \gamma^2 I \quad \text{or} \quad T^*T \leq \gamma^2 I. \tag{20}$$

The tighter the bound in (20), the tighter the bound in (19). In other words, the closer $\gamma^{-1}T$ is to a unitary matrix, the tighter (19) becomes. Hence, we attempt to choose $K_1$ and $K_2$ to make $\gamma^{-2}TT^*$ as close to identity as possible.

To this end, postmultiply $T$ with the unitary matrix

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

$\nabla$ and $\Delta$ are found via the factorizations

$$D^{-1/2}L^*LD^{-1/2} + I = \nabla^*\nabla \quad \text{and} \quad LD^{-1}L^* + I = \Delta\Delta^* \tag{21}$$

to obtain

$$T\Phi = \begin{bmatrix} A & B \\ 0 & C \end{bmatrix} \tag{22}$$

where

$$A = \sqrt{\beta} D^{-1/2} \nabla^{-1},$$
$$B = \sqrt{\beta}(D^{-1} L^* \Delta^{-*} - K_1 \Delta), \text{ and } C = (I - K_2)\Delta. \tag{23}$$

Thus, $TT^* \leq \gamma^2 I$ implies

$$\begin{bmatrix} AA^* + BB^* & BC^* \\ CB^* & CC^* \end{bmatrix} \leq \gamma^2 I. \tag{24}$$

Note that we have many degrees of freedom when choosing $K_1$ and $K_2$ and wish to make judicious choices. So, to simplify things, let us choose $K_2$ such that $CC^* = \gamma_1^2 I$ for some $0 \leq \gamma_1 \leq \gamma$. (Clearly, this can always be done, since from (21) we have that $\Delta$ is invertible, and the simple choice $K_2 = I - \gamma_1 \Delta^{-1}$ will do the job.) To make half the eigenvalues of $\gamma^{-2} TT^*$ unity, we set the Schur complement of the (2,2) entry of (24) to zero, i.e.,

$$AA^* + BB^* - \gamma^2 I - BC^*(CC^* - \gamma^2 I)^{-1} CB^* = 0. \tag{25}$$

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

$$BB^* = \left(1 - \frac{\gamma_1^2}{\gamma^2}\right)(\gamma^2 I - AA^*). \tag{26}$$

Using the definitions of $A$ and $B$ from (23), we obtain

$$\sqrt{\beta} K_1 = \sqrt{\beta} D^{-1} L^* (LD^{-1} L^* + I)^{-1} - B\Delta^{-1}. \tag{27}$$

From the (1,1) entry of (24), it follows that

$$\gamma^2 I - (AA^* + BB^*) \geq 0$$

which is the only constraint on $\gamma$. Combining this constraint with the definition of $A$ from (23), the definition of $\nabla$ from (21), and the expression for $BB^*$ from (26), we obtain that

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

We summarize the results of this section in the following theorem:

*Theorem 1: Consider the integer least-squares problem (16). Then, for any $\gamma^2 \geq \beta / \lambda_{\min}(D + L^* L)$, $0 \leq \gamma_1 \leq \gamma$, and any matrices $D \geq 0$, $B$, and $\Delta$ satisfying $\Delta\Delta^* = I + LD^{-1} L^*$ and $BB^* = (1 - \gamma_1^2/\gamma^2)(\gamma^2 I - \beta(D + L^* L)^{-1})$*

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

*Proof:* It 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 $AA^* = \beta(D + L^* L)^{-1}$.

The next corollary directly follows from Theorem 1.

*Corollary 1: Consider the setting of the Theorem 1 and let $\beta = 1$. Then*

$$\min_{\mathbf{a} \in \mathcal{D}} \|\mathbf{b} - L\mathbf{a}\|^2$$
$$\geq \min_{\mathbf{a} \in \mathcal{D}} \gamma^{-2} \|\mathbf{a} - D^{-1} L^* (LD^{-1} L^* + I)^{-1} \mathbf{b} + B\phi\|^2$$
$$- \mathbf{a}^* D\mathbf{a} + \frac{\gamma_1^2}{\gamma^2} \|\phi\|^2 \tag{28}$$

*where $B$ is the unique symmetric square root of $(1 - \gamma_1^2/\gamma^2)(\gamma^2 I - (D + L^* L)^{-1})$, and $\phi$ 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 $(\gamma_1, \gamma, D, \phi)$, and we can exploit that to tighten the bound in (28) 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 third term in (28), and it sets $B = 0$ so that we need not worry about the vector $\phi$. Finally, to maximize the first term, we need to take $\gamma$ as its smallest possible 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 and let $\beta = 1$. Then*

$$\min_{\mathbf{a} \in \mathcal{D}} \|\mathbf{b} - L\mathbf{a}\|^2$$
$$\geq \lambda_{\min}(L^* L + D) \|\mathbf{a} - (L^* L + D)^{-1} L^* \mathbf{b}\|^2$$
$$- \mathbf{a}^* D\mathbf{a} + \mathbf{b}^*(I - L((L^* L + D)^{-1}) L^*)\mathbf{b}. \tag{29}$$

*Remark:* We would like to note that the bound given in the previous Corollary could have been also obtained in a faster way. Below we show a possible derivation that an anonymous reviewer has provided to us.

Let $D$ be a diagonal matrix such that $D \geq 0$. Then, we have

$$\|\mathbf{b} - L\mathbf{a}\|^2$$
$$= \mathbf{a}^* L^* L\mathbf{a} - 2\mathbf{b}^* L\mathbf{a} + \mathbf{b}^*\mathbf{b}$$
$$= \mathbf{a}^*(L^* L + D)\mathbf{a} - 2\mathbf{b}^* L\mathbf{a} + \mathbf{b}^*\mathbf{b} - \mathbf{a}^* D\mathbf{a}$$
$$= (\mathbf{a} - (L^* L + D)^{-1} L^* \mathbf{b})^* (L^* L + D)(\mathbf{a} - (L^* L + D)^{-1} L^* \mathbf{b})$$
$$- \mathbf{b}^* L((L^* L + D)^{-1}) L^* \mathbf{b} + \mathbf{b}^*\mathbf{b} - \mathbf{a}^* D\mathbf{a}$$
$$\geq \lambda_{\min}(L^* L + D) \|\mathbf{a} - (L^* L + D)^{-1} L^* \mathbf{b}\|^2$$
$$- \mathbf{a}^* D\mathbf{a} + \mathbf{b}^*(I - L((L^* L + D)^{-1}) L^*)\mathbf{b}.$$

It is not difficult to see that this is precisely the same bound as the bound given in the Corollary 2. The interested reader can find more on this type of bounds in, e.g., [33] and [34].

In the following sections, we show how various choices of the free parameters in the general lower bound from Theorem 1

yield several interesting special cases of lower bounds. In particular, in Section IV, we show that the lower bound obtained by solving a related convex optimization problem, where the search space is relaxed from integers to a sphere, can be deduced as a special case of the lower bound from Theorem 1. Then, in Section V, we show that the lower bound obtained by solving another convex optimization problem, where the search space is now relaxed from integers to a polytope, can also be deduced as a special case of the lower bound from Theorem 1. Finally, in Section VII, we use (29) to deduce the lower bound based on the minimum eigenvalue of $L^*L$.

## IV. Spherical Relaxation

Assume the setting of the Theorem 1. Let $\gamma_1 \rightarrow \gamma$, $\beta \rightarrow 0$, $D = (1/\alpha)I$, and $\Delta_{\rm sph}\Delta_{\rm sph}^* = \alpha LL^* + I$. Then

$$\mathrm{LB}_{\rm sph}^{(1)} = \|\Delta_{\rm sph}^{-1}\mathbf{b}\|^2 - \frac{k-1}{4\alpha} \tag{30}$$

is a special case of the general bound given in Theorem 1 and, therefore, a lower bound on the integer least-squares problem (8). In addition, since being a special case, it is less tight than the general bound given in Theorem 1. Clearly, to make $\mathrm{LB}_{\rm sph}^{(1)}$ as tight as possible, we should maximize (30) over $\alpha$.

Consider the singular value decomposition (SVD) of $L$, $L = U\Sigma V^T$, where $U$ and $V$ are unitary matrices, and where $\Sigma$ is diagonal matrix. Let $\sigma_i$ be the $i$th component on the main diagonal of $\Sigma$, and let $r$ be the rank of $L$. Also, let $\mathbf{q}_{1:k-1} = U^T\mathbf{z}_{1:k-1}$. Then we can write

$$\mathrm{LB}_{\rm sph}^{(1)} = \sum_{i=1}^{r}\frac{\alpha^{-1}\mathbf{q}_i^2}{\sigma_i^2+\alpha^{-1}} - \alpha^{-1}\frac{k-1}{4}. \tag{31}$$

To maximize over $\alpha$, we differentiate to obtain

$$\frac{d\mathrm{LB}_{\rm sph}^{(1)}}{d(\alpha^{-1})} = \sum_{i=1}^{r}\left(\frac{\sigma_i\mathbf{q}_i}{\sigma_i^2+\alpha^{-1}}\right)^2 - \frac{k-1}{4}. \tag{32}$$

Let $\hat{\alpha}$ denote the value of $\alpha$ which maximizes $\mathrm{LB}_{\rm sph}^{(1)}$. Then, it easily follows that

$$\sum_{i=1}^{r}\left(\frac{\sigma_i\mathbf{q}_i}{\sigma_i^2+\hat{\alpha}^{-1}}\right)^2 = \frac{k-1}{4}. \tag{33}$$

Note that if $\sum_{i=1}^{k-1}(\mathbf{q}_i/\sigma_i)^2 - k - 1/4 \leq 0$, we set $\hat{\alpha}^{-1} = 0$. Hence, we can state a lower bound on (8) as

$$\hat{\mathrm{LB}}_{\rm sph}^{(1)} = \|\hat{\Delta}_{\rm sph}^{-1}\mathbf{b}\|^2 - \frac{k-1}{4\hat{\alpha}} \tag{34}$$

where $\hat{\Delta}_{\rm sph}$ is any matrix such that $\hat{\Delta}_{\rm sph}\hat{\Delta}_{\rm sph}^* = \hat{\alpha}LL^* + I = \hat{\alpha}R_{1:k-1,1:k-1}R_{1:k-1,1:k-1}{}^* + I$, $\mathbf{b} = \mathbf{z}_{1:k-1}$, and $\hat{\alpha}^{-1}$ is the unique solution of (33) if $\sum_{i=1}^{k-1}(\mathbf{q}_i/\sigma_i)^2 - k - 1/4 > 0$, and zero otherwise.

To obtain an interpretation of the bound we have derived, let us consider a bound obtained by a simple spherical relaxation. To this end, let us denote $\hat{\mathrm{LB}}_{\rm sph}^{(2)} = \|\mathbf{z}_{1:k-1} - R_{1:k-1,1:k-1}\hat{\mathbf{s}}_{1:k-1}\|_2^2$, where $\hat{\mathbf{s}}_{1:k-1}$ is the solution of the following optimization problem:

$$\min_{\mathbf{s}_{1:k-1}} \|\mathbf{z}_{1:k-1} - R_{1:k-1,1:k-1}\mathbf{s}_{1:k-1}\|_2^2 \text{ subject to } \sum_{i=1}^{k-1}\mathbf{s}_i^2 \leq \frac{k-1}{4}. \tag{35}$$

This is a lower bound since the integer constraints have been relaxed to a spherical constraint that includes $\{-(1/2),(1/2)\}^k$. The solution of (35) can be found via Lagrange multipliers (see. e.g., [3]), and it turns out that the optimal value of its objective function coincides with (34). Therefore, we conclude that

$$\hat{\mathrm{LB}}_{\rm sph}^{(2)} = \hat{\mathrm{LB}}_{\rm sph}^{(1)}$$

and the lower bound obtained via spherical relaxation is indeed a special case of the general lower bound given in Theorem 1.

We would also like to note that we could get a tighter bound in (35) if we replace inequality with an equality. Although the resulting problem would be nonconvex, following the procedure from, e.g., [3] and [39], we would obtain a result similar to the one obtained in (34). The difference would be that now in (33), $\hat{\alpha}^{-1}$ would be allowed to take negative values as well. This would certainly give a bound which is tighter than $\hat{\mathrm{LB}}_{\rm sph}^{(1)}$. However, in general, we did not find that solving (33) for negative $\hat{\alpha}^{-1}$ would be more useful for our algorithms than solving it only for positive $\hat{\alpha}^{-1}$. In addition, we would like to emphasize that the bound given in (35) is valid for the binary case. It can, however, be used for $M > 2$ if the constraint in (35) is replaced by $\sum_{i=1}^{k-1}\mathbf{s}_i^2 \leq (M-1)^2(k-1/4)$. However, we believe that this type of bound is more useful in the binary case.

Now, let us unify the notation and write $\mathrm{LB}_{\rm sph} = \hat{\mathrm{LB}}_{\rm sph}^{(1)} = \hat{\mathrm{LB}}_{\rm sph}^{(2)}$. We employ $\mathrm{LB}_{\rm sph}$ to modify the sphere decoding algorithm by substituting it in place of the lower bound in step 4 of Algorithm 1. The subroutine for computing $\mathrm{LB}_{\rm sph}$ is given below.

---

*Subroutine for computing* $\mathrm{LB}_{\rm sph}$*:*

---

*Input:* $\mathbf{y}_{1:k-1}, R_{1:k-1,k:m}, \mathbf{s}_{k:m}, R_{1:k-1,1:k-1}$.

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

2. Compute the SVD of $R_{1:k-1,1:k-1}$, $R_{1:k-1,1:k-1} = U\Sigma V^T, V = [\mathbf{v}_1,\ldots,\mathbf{v}_{k-1}]$.

3. Set $\mathbf{q}_{1:k-1} = U^T\mathbf{z}_{1:k-1}$ and $r = \mathrm{rank}(R_{1:k-1,1:k-1})$.

4. If $\sum_{i=1}^{r}(\mathbf{q}_i/\sigma_i)^2 > k - 1/4$, find $\lambda^*$ such that $\sum_{i=1}^{r}(\sigma_i\mathbf{q}_i/\sigma_i^2+\lambda^*)^2 = k - 1/4$, and compute $\hat{\mathbf{s}}_{1:k-1} = \sum_{i=1}^{r}(\sigma_i\mathbf{q}_i/\sigma_i^2+\lambda^*)\mathbf{v}_i$ and $\mathrm{LB}_{\rm sph} = \sum_{i=1}^{r}(\lambda^*\mathbf{q}_i/\sigma_i^2+\lambda^*)\mathbf{v}_i$.

5. If $\sum_{i=1}^{r}(\mathbf{q}_i/\sigma_i)^2 \leq k - 1/4$, set $\hat{\mathbf{s}}_{1:k-1} = \sum_{i=1}^{r}(\mathbf{q}_i/\sigma_i)\mathbf{v}_i$ and $\mathrm{LB}_{\rm sph} = 0$.

---

The computational complexity of finding the spherical lower bound by the above subroutine is quadratic in $k$, and the bound needs to be computed at each point visited by Algorithm 1. That the complexity is only quadratic may not immediately seem obvious since we do need to compute the SVD of the matrix $R_{1:k-1,1:k-1}$. Fortunately, however, this operation has to be performed only once for each level of the search tree, and hence can be done in advance (i.e., before Algorithm 1 even starts). Computing the SVD of matrices $R_{1:k-1,1:k-1}, 2 \leq k \leq m$ would require performing factorizations that are cubic in $k$ for any $2 \leq k \leq m$. However, using the results from [17] and [18], it can be shown that all $m$ SVDs can, in fact, be computed with complexity that is cubic in $m$.

The computational effort required for finding $\mathrm{LB}_{\mathrm{sph}}$ beyond performing the SVD is clearly quadratic in $k$ at the $(m-k+1)$th level of the search tree. Note that unlike the SVD, these remaining operations do need to be performed per *each point* visited by the algorithm. In particular, computing the vector $\mathbf{q}$ requires finding $U^T\mathbf{z}_{1:k-1}$, which is quadratic in $k$. Now, the matrix $U^T$ is constant at each level of the search tree, but the vector $\mathbf{z}_{1:k-1}$ differs from node to node. Clearly, this is the most significant part of the cost, and the computational complexity of finding $\mathrm{LB}_{\mathrm{sph}}$ is indeed quadratic.

### A. Generalized Spherical Relaxation

In this subsection, we propose a generalization of the spherical lower bound. This generalization is given by

$$\mathrm{LB}_{\mathrm{gsph}} = \begin{cases} \mathrm{LB}_{\mathrm{sph}} + D\mathrm{LB}, & \text{if } \|\hat{\Delta}_{\mathrm{sph}}^{-1}\mathbf{b}\|^2 - \frac{k-1}{4\hat{\alpha}} > 0, \\ 0, & \text{otherwise} \end{cases} \tag{36}$$

where, as in (34),

$$\mathrm{LB}_{\mathrm{sph}} = \|\hat{\Delta}_{\mathrm{sph}}^{-1}\mathbf{b}\|^2 - \frac{k-1}{4\hat{\alpha}}$$

$\hat{\Delta}_{\mathrm{sph}}\hat{\Delta}_{\mathrm{sph}}^{*} = \hat{\alpha}LL^* + I$, $L = R_{1:k-1,1:k-1}$, $\mathbf{b} = \mathbf{z}_{1:k-1}$, $\mathbf{q} = U^T\mathbf{b}$, $L = U\Sigma V^T$, $\hat{\alpha}^{-1}$ is the unique solution of (33) if $\sum_{i=1}^{k-1}(\mathbf{q}_i/\sigma_i)^2 - k - 1/4 > 0$, and 0 otherwise, and where

$$D\mathrm{LB} = \min_{\mathbf{a}\in\mathcal{D}} \left(\frac{1}{\hat{\alpha}} + \lambda_{\min}(L^*L)\right) \|\mathbf{a} - \hat{\alpha}L^*(\hat{\alpha}LL^* + I)^{-1}\mathbf{b}\|^2. \tag{37}$$

Clearly, (36) is obtained from (29) by setting $D = (1/\hat{\alpha})I$ and is, therefore, a lower bound on the integer least-squares problem (8). Also, since $\mathrm{LB}_{\mathrm{gsph}} \geq \mathrm{LB}_{\mathrm{sph}}$, the generalized spherical bound is tighter than the spherical bound. It is interesting to mention that $\mathrm{LB}_{\mathrm{gsph}}$ was also obtained in [33] based on a different approach.

We refer to Algorithm 1 with $\mathrm{LB} = \mathrm{LB}_{\mathrm{gsph}}$ as the GSPHSD algorithm. Since the generalized spherical bound is at least as tight as the spherical bound, we expect that the GSPSD algorithm prunes more points from the search tree than the SPHSD algorithm.

We give the subroutine for computing $\mathrm{LB}_{\mathrm{gsph}}$ below.

*Subroutine for computing* $\mathrm{LB}_{\mathrm{gsph}}$*:*

*Input:* $\mathbf{y}_{1:k-1}, R_{1:k-1,k:m}, \mathbf{s}_{k:m}, R_{1:k-1,1:k-1}$

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

2. Compute the SVD of $R_{1:k-1,1:k-1}$, $R_{1:k-1,1:k-1} = U\Sigma V^T, V = [\mathbf{v}_1, \ldots, \mathbf{v}_{k-1}]$.

3. Set $\mathbf{q}_{1:k-1} = U^T\mathbf{z}_{1:k-1}$ and $r = \mathrm{rank}(R_{1:k-1,1:k-1})$.

4. If $\sum_{i=1}^{r}(\mathbf{q}_i/\sigma_i)^2 > k - 1/4$, find $\lambda^*$ such that $\sum_{i=1}^{r}(\sigma_i\mathbf{q}_i/\sigma_i^2 + \lambda^*)^2 = k - 1/4$, and compute $\hat{\mathbf{s}}_{1:k-1} = \sum_{i=1}^{r}(\sigma_i\mathbf{q}_i/\sigma_i^2 + \lambda^*)\mathbf{v}_i$ and

$$\mathrm{LB}_{\mathrm{gsph}} = \min_{\mathbf{a}\in\mathcal{D}}(\lambda^* + \lambda_{\min}(R_{1:k-1,1:k-1}^{*}R_{1:k-1,1:k-1}))\|\mathbf{a} - \sum_{1}^{r}\frac{\sigma_i\mathbf{q}_i}{\sigma_i^2+\lambda^*}\mathbf{v}_i\|^2 + \sum_{i=1}^{r}\left(\frac{\lambda^*\mathbf{q}_i}{\sigma_i^2+\lambda^*}\right)\mathbf{v}_i.$$

5. If $\sum_{i=1}^{r}(\mathbf{q}_i/\sigma_i)^2 \leq k - 1/4$, set $\hat{\mathbf{s}}_{1:k-1} = \sum_{i=1}^{r}(\mathbf{q}_i/\sigma_i)\mathbf{v}_i$ and $\mathrm{LB}_{\mathrm{gsph}} = 0$.

At first, the complexity of computing $D\mathrm{LB}$ in (37) may seem cubic in $k$; however, it can actually be reduced to quadratic. Clearly, finding the inverse of $\hat{\alpha}LL^* + I$ is of cubic complexity and required in each node of the search tree ($L$ is constant per level but $\hat{\alpha}$ differs from node to node). However, instead of inverting the matrix $\hat{\alpha}LL^* + I$ directly, we can do it in several steps. In particular, using the SVD $L = U\Sigma V^T$, we can write

$$\hat{\alpha}L^*(\hat{\alpha}LL^* + I)^{-1}\mathbf{b} = \hat{\alpha}V\Sigma(\hat{\alpha}\Sigma^2 + I)^{-1}U\mathbf{b}.$$

Since $\Sigma$ is a diagonal matrix, the inversion of $\hat{\alpha}\Sigma^2 + I$ is only linear in $k$. Therefore, the computationally dominant operation in finding $\hat{\alpha}V\Sigma(\hat{\alpha}\Sigma^2 + I)^{-1}U\mathbf{b}$ is multiplication of a matrix and a vector, which requires quadratic complexity. Recall what we argued earlier in this section: although the SVD decomposition of the matrix $L$ is of cubic complexity, it can be performed offline since $L$ is constant on each level in the search tree. Furthermore, instead of computing separately SVDs of all matrices $R_{1:k-1,1:k-1}, 2 \leq k \leq m$, we can employ efficient techniques from [17] and [18] to obtain all relevant matrices in these SVDs with complexity cubic in $m$. Therefore, computing $D\mathrm{LB}$ is essentially quadratic in $k$. Since, computing $\mathrm{LB}_{\mathrm{sph}}$ is also quadratic, the computational effort required for finding $\mathrm{LB}_{\mathrm{gsph}}$ in (36) is quadratic as well.

## V. Polytope Relaxation

In this subsection, we show that the lower bound on the integer least-squares problem (8) obtained by solving the related convex optimization where the search space is relaxed from integers to a polytope is yet another special case of the lower bound derived in Section III.

Assume the setting of the Theorem 1. Let $\gamma_1 \to \gamma$, $\beta \to 0$, and $\Delta_{\rm plt}\Delta_{\rm plt}^* = LD^{-1}L^* + I$. Then

$$\mathrm{LB}_{\rm plt}^{(1)} = \|\Delta_{\rm plt}^{-1}\mathbf{b}\|^2 - \frac{\mathrm{Tr}D}{4}. \tag{38}$$

is a special case of the general bound given in Theorem 1 and, therefore, a lower bound on the integer least-squares problem (8). Now, since the matrix $D$ is a free parameter, we can make the bound (38) tighter by optimizing over $D$. Hence, we can obtain a lower bound to the integer least-squares problem (8) as

$$\hat{\mathrm{LB}}_{\rm plt}^{(1)} = \max_{D\geq 0} \|\Delta_{\rm plt}^{-1}\mathbf{b}\|^2 - \frac{\mathrm{Tr}D}{4}. \tag{39}$$

Clearly, $\hat{\mathrm{LB}}_{\rm plt}^{(1)}$ is also a lower bound on the integer least-squares problem (8). Furthermore, since (39) allows for any positive-semidefinite diagonal matrix $D$, while (30) allows only for scaled version of identity, it is clear that the bound in (39) will be tighter than the one in (30). However, as we will see in the rest of this section, computing (39) is of greater complexity than computing (30).

Now, before further discussing and comparing the merits of the bounds defined in (30) and (39), we will show that the lower bound (39) is equivalent to the lower bound obtained by relaxing the search space in the integer least-squares problem (8) to a polytope and solving the resulting convex optimization problem. In particular, such a relaxation yields

$$\min \ \ \|\mathbf{b} - L\mathbf{d}\|^2 \ \text{ subject to } -\frac{1}{2} \leq d_i \leq \frac{1}{2}. \tag{40}$$

Let us denote $\hat{\mathrm{LB}}_{\rm plt}^{(2)} = \|\mathbf{b} - L\hat{\mathbf{d}}\|^2$, where $\hat{\mathbf{d}}$ is a solution of (40). We want to show that $\hat{\mathrm{LB}}_{\rm plt}^{(1)} = \hat{\mathrm{LB}}_{\rm plt}^{(2)}$. To this end, consider the Lagrange dual of the problem (40)

$$\begin{aligned}\mathcal{L}(\boldsymbol{\xi}) &= \|\mathbf{b} - L\mathbf{d}\|^2 + \sum_{i=1}^{k-1} \xi_i \left(d_i^2 - \frac{1}{4}\right) \\ &= \mathbf{d}^*(L^*L + \Xi)\mathbf{d} - 2\mathbf{b}^*L\mathbf{d} + \mathbf{b}^*\mathbf{b} - \frac{\mathrm{Tr}\Xi}{4} \\ &= \mathbf{d}^*\Omega\Omega^*\mathbf{d} - 2\mathbf{b}^*L\Omega^{-*}\Omega^*\mathbf{d} + \mathbf{b}^*\mathbf{b} - \frac{\mathrm{Tr}\Xi}{4} \\ &\quad + \mathbf{b}^*L\Omega^{-*}\Omega^{-1}L^*\mathbf{b} - \mathbf{b}^*L\Omega^{-*}\Omega^{-1}L^*\mathbf{b} \\ &= (\Omega^*\mathbf{d} - \Omega^{-1}L^*\mathbf{b})^2 + \mathbf{b}^*\mathbf{b} - \mathbf{b}^*L\Omega^{-*}\Omega^{-1}L^*\mathbf{b} - \frac{\mathrm{Tr}\Xi}{4}\end{aligned}$$

where $\Omega$ is any matrix such that $\Omega\Omega^* = L^*L + \Xi$. Using $\mathcal{L}(\boldsymbol{\xi})$, we can pose a dual problem to the primal in (40) as

$$\max_{\Xi} \min_{\mathbf{d}} (\Omega^*\mathbf{d} - \Omega^{-1}L^*\mathbf{b})^2 + \mathbf{b}^*\mathbf{b} - \mathbf{b}^*L\Omega^{-*}\Omega^{-1}L^*\mathbf{b} - \frac{\mathrm{Tr}\Xi}{4}$$
$$\text{subject to } \Xi \geq 0, \quad \Xi \text{ is diagonal.}$$

Clearly, the previous problem is equivalent to

$$\max_{\Xi} \ \mathbf{b}^*\mathbf{b} - \mathbf{b}^*L\Omega^{-*}\Omega^{-1}L^*\mathbf{b} - \frac{\mathrm{Tr}\Xi}{4} \ \text{ subject to } \ \Xi \geq 0, \ \Xi \text{ is diagonal}$$

which, after straightforward algebraic transformations involving the matrix inversion lemma, can be written as

$$\max_{\Xi} \quad \mathbf{b}^*(I + L\Xi^{-1}L^*)^{-1}\mathbf{b} - \frac{\mathrm{Tr}\Xi}{4}$$
$$\text{subject to} \quad \Xi \geq 0, \quad \Xi \text{is diagonal.} \tag{41}$$

Since the primal problem is strictly feasible, the duality gap between the problems in (40) and (41) is zero. Therefore, if we denote the optimal solution of (41) by $\hat{\Xi}$

$$\hat{\mathrm{LB}}_{\rm plt}^{(2)} = \mathbf{b}^*(I + L\hat{\Xi}^{-1}L^*)^{-1}\mathbf{b} - \frac{\mathrm{Tr}\hat{\Xi}}{4}. \tag{42}$$

Comparing (39) and (42), we conclude that

$$\hat{\mathrm{LB}}_{\rm plt}^{(1)} = \hat{\mathrm{LB}}_{\rm plt}^{(2)}$$

which implies that the bound on the integer least-squares problem (8) is indeed a special case of the general bound we derived in Section II. To unify the notation, we write $\mathrm{LB}_{\rm plt} = \hat{\mathrm{LB}}_{\rm plt}^{(1)} = \hat{\mathrm{LB}}_{\rm plt}^{(2)}$. We refer to Algorithm 1 which, in step 4, makes use of $\mathrm{LB}_{\rm plt}$ as the PLTSD algorithm. The subroutine for computing $\mathrm{LB}_{\rm plt}$ is given below.

---

*Subroutine for computing* $\mathrm{LB}_{\rm plt}$*:*

---

*Input:* $\mathbf{y}_{1:k-1}, R_{1:k-1,k:m}, \mathbf{s}_{k:m}, R_{1:k-1,1:k-1}$

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

2.

$$\mathrm{LB}_{\rm plt} = quadprog\Big(R^*_{1:k-1,1:k-1}R_{1:k-1,1:k-1}, -2R^*_{1:k-1,1:k-1}\mathbf{z}_{1:k-1}, [], [], [], [], -\frac{1}{2}, \frac{1}{2}\Big);$$

(*quadprog* is MATLAB function for solving quadratic optimization problems).

---

The lower bound studied in this subsection is tighter than the spherical one considered earlier in the paper. It is clear that (39) results in a tighter lower bound than (30) since (39) includes maximization over all diagonal positive semi-definite matrices $D$, whereas (30) assumes only the special case $D = (1/\alpha)I$. The geometric interpretation implies the difference between (30) and (39) as well. In particular, in (30) the set of integers from the basic problem (8) is relaxed to a sphere, while in (39) the same set of integers is relaxed to a polytope, i.e., to a smaller set. However, although the lower bound based on the polytope relaxation is tighter than the one based on the spherical relaxation, the total computational effort is not necessarily improved. The reason is the additional computational effort required to calculate the $\mathrm{LB}_{\rm sph}$ per each node; these additional computations are of quadratic complexity, while the additional operations for calculating the $\mathrm{LB}_{\rm plt}$ are cubic (see, e.g., [2]).

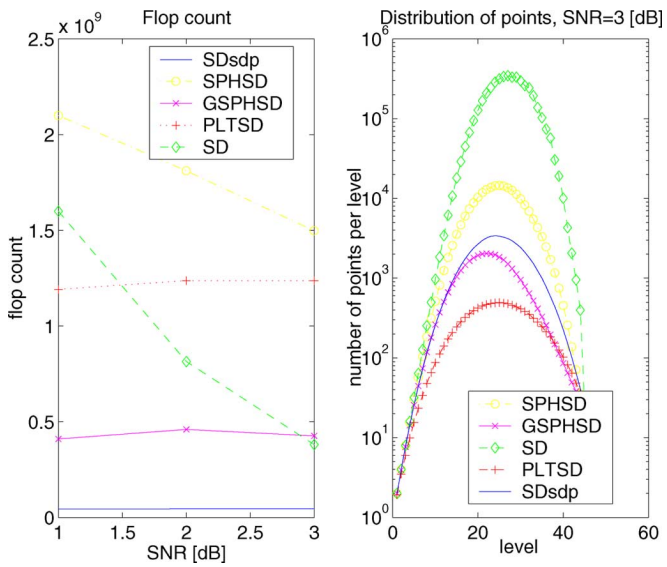



Fig. 4. Computational complexity and the distribution of the points in the search tree for SD, SPHSD, GSPHSD, PLTSD, and SDsdp algorithms, $m = 45$, $\mathcal{D} = \{-(1/2), (1/2)\}^{45}$.

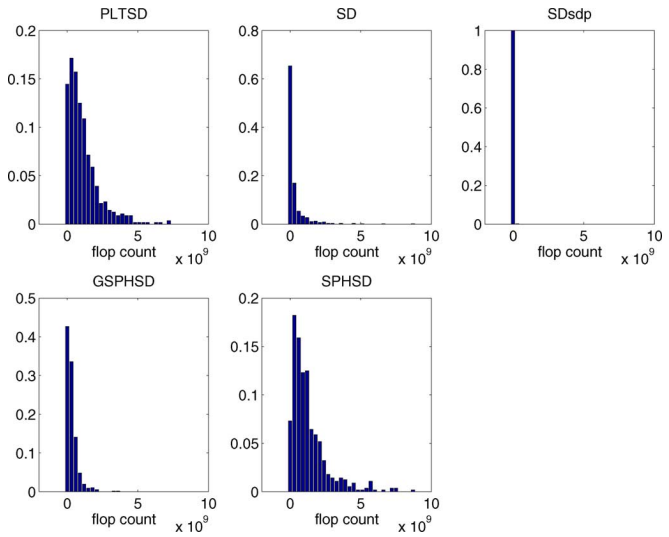



Fig. 5. Flop count histograms for SD, SPHSD, GSPHSD, PLT, and SDsdp algorithms, $m = 45$, SNR $= 3$ [dB], $\mathcal{D} = \{-(1/2), (1/2)\}^{45}$.

Therefore, there is no general answer to which bound is better for improving the standard sphere decoding algorithm.

## VI. Performance Comparison

In this section, we study and compare the performances of the SPHSD, GSPHSD, PLTSD, SDsdp, and SD algorithms.

### A. Flop Count

In Fig. 4 the average flop count and the distribution of the number of visited nodes per each level of the search tree are shown for each of the SPHSD, GSPHSD, PLTSD, SD, and SDsdp algorithms. The parameters of the system are $m = 45$, $\mathcal{D} = \{-(1/2), (1/2)\}^m$, and SNR $= 10\log_{10}(m/4\sigma^2)$. The initial search radius was chosen statistically as in [4] (the sequence of $\epsilon$'s, $\epsilon = 0.9$, $\epsilon = 0.99$, $\epsilon = 0.999$, etc.), and updated every time the bottom of the tree is reached. As can be seen, the SPHSD, GSPHSD, PLTSD, and SDsdp prune more points than the SD algorithm. Also, as it is expected, the PLTSD prunes more points than the SPHSD and GSPHSD since it uses a tighter lower bound. However, the large improvement in tree pruning does not always reflect in improving the overall flop count. The reason is, as we have already said, the additional amount of computation that has to be performed at each node of the search tree. For the system parameters simulated on Fig. 4, we have that the SDsdp algorithm has the best flop count, the GSPHSD still has better flop count than the SD algorithm, and the PLTSD and SPHSD have worse flop count than the SD algorithm.

### B. Flop Count Histogram

In Fig. 5, the flop count histograms of the SPHSD, GSPHSD, PLTSD, SD, and SDsdp algorithms are shown obtained from performing 560 numbers of independent runs of the algorithms. As before the parameters of the system are $m = 45$, $\mathcal{D} = \{-(1/2), (1/2)\}^m$, and SNR $= 3$ [dB]. It can be seen that the GSPHSD and SDsdp have significantly better shaped (shorter tail) histograms than the SD algorithm. This implies that the probability of encountering large flop counts is significantly less. It should be noted that SPHSD and PLTSD have longer tail than the SD.

## VII. Eigenbound

In principle, the lower bound (29) still requires an optimization over the diagonal matrix $D \geq 0$. A particular choice that may be computationally feasible is $D = \alpha I$, for some $\alpha$. However, in this section, we focus on an even more simple choice for $D$. Namely, as noted in [14], letting $D \to 0$ in Corollary 2, we obtain

$$\mathrm{LB}_{\mathrm{eigb}} = \lambda_{\min}(L^*L) \min_{\mathbf{a}\in\mathcal{D}} \|\mathbf{a} - L^{-1}\mathbf{b}\|^2. \tag{43}$$

We also mention that this bound could have been obtained in an easier fashion as

$$\begin{aligned}\mathrm{LB}_{\mathrm{eigb}} &= \lambda_{\min}(L^*L) \min_{\mathbf{a}\in\mathcal{D}} \|\mathbf{a} - L^{-1}\mathbf{b}\|^2 \\ &\leq \min_{\mathbf{a}\in\mathcal{D}} (\mathbf{a} - L^{-1}\mathbf{b})^* \lambda_{\min}(L^*L) I (\mathbf{a} - L^{-1}\mathbf{b}) \\ &\leq \min_{\mathbf{a}\in\mathcal{D}} (\mathbf{a} - L^{-1}\mathbf{b})^* (L^*L) (\mathbf{a} - L^{-1}\mathbf{b}) \\ &\leq \min_{\mathbf{a}\in\mathcal{D}} \|L\mathbf{a} - \mathbf{b}\|^2. \end{aligned} \tag{44}$$

Although this may raise concern that the resulting bound will be too loose, it turns out that it yields an algorithm with smaller flop count than the standard sphere decoder. The key observation is that, with $D = 0$, all the computations required at any point in the tree are linear in the dimension. (The standard sphere decoder also requires a linear number of operations per point.) Since it is based on the minimum eigenvalue, we refer to this bound as the *eigenbound*.

Clearly, since (43) can be regarded as a special case of (29), it is a lower bound on the integer least-squares problem (8).

Note that it appears as if (43) may not be a good bound since $\lambda_{\min}(L^*L)$ could become very small. However, since the minimization in (43) is performed over integers, the resulting bound turns out to be 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 computation required to find $\mathrm{LB}_{\mathrm{eigb}}$ for a node at a level $k$ in the search tree is linear in $k$.

The key observation that enables efficient computation of $\mathrm{LB}_{\mathrm{eigb}}$ in (43) is that the vector $L^{-1}\mathbf{b}$ can be propagated as the search progresses down the tree. Before proceeding any further, we will simplify notation. First, recall that

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

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

$$\begin{aligned}\mathbf{f}^{(k-1)}&=L^{-1}\mathbf{b}=F_{1:k-1,1:k-1}\mathbf{z}_{1:k-1}\\&=F_{1:k-1,1:k-1}(\mathbf{y}_{1:k-1}-R_{1:k-1,k:m}\mathbf{s}_{k:m}).\end{aligned}\tag{45}$$

We wish to find a recursion that relates the vector $\mathbf{f}^{(k-2)}$ to the already calculated vector $\mathbf{f}^{(k-1)}$. See equation (46) at the bottom of the page. From (46), we see that

$$\begin{aligned}\mathbf{f}^{(k-1)}_{1:k-2}&=F_{1:k-2,1:k-2}\mathbf{y}_{1:k-2}+F_{1:k-2,k-1}y_{k-1}\\&\quad-F_{1:k-2,1:k-2}R_{1:k-2,k:m}\mathbf{s}_{k:m}-F_{1:k-2,k-1}R_{k-1,k:m}\mathbf{s}_{k:m}.\end{aligned}\tag{47}$$

Similarly

$$\begin{aligned}\mathbf{f}^{(k-2)}&=F_{1:k-2,1:k-2}\mathbf{y}_{1:k-2}-R_{1:k-2,k-1:m}\mathbf{s}_{k-1:m}\\&=F_{1:k-2,1:k-2}\mathbf{y}_{1:k-2}\\&\quad-F_{1:k-2,1:k-2}\begin{bmatrix}R_{1:k-2,k-1} & R_{1:k-2,k:m}\end{bmatrix}\begin{bmatrix}s_{k-1}\\ \mathbf{s}_{k:m}\end{bmatrix}\\&=F_{1:k-2,1:k-2}\mathbf{y}_{1:k-2}-F_{1:k-2,1:k-2}R_{1:k-2,k:m}\mathbf{s}_{k:m}\\&\quad-F_{1:k-2,1:k-2}R_{1:k-2,k-1}s_{k-1}.\end{aligned}\tag{48}$$

Using (47) and (48), we relate $\mathbf{f}^{(k-1)}$ and $\mathbf{f}^{(k-2)}$ as

$$\begin{aligned}\mathbf{f}^{(k-2)}&=\mathbf{f}^{(k-1)}_{1:k-2}+F_{1:k-2,k-1}R_{k-1,k:m}\mathbf{s}_{k:m}\\&\quad-F_{1:k-2,k-1}y_{k-1}-F_{1:k-2,1:k-2}R_{1:k-2,k-1}s_{k-1}.\end{aligned}\tag{49}$$

All operations in the recursion (49) 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 (49) for all points visited by the algorithm at a level. Therefore, this multiplication may be treated as a part of preprocessing, i.e., we compute it for all $k$ before actually running Algorithm 1. Hence, updating the vector $L^{-1}\mathbf{b}$ in the (43) requires a computational effort that is linear in $k$. Furthermore, since it is done componentwise, the minimization in (43) also has complexity that is linear in $k$. Hence, we conclude that the complexity of computing the eigenbound is linear in $k$. Also, it should be noted that in addition to standard sphere decoder, we have to compute $\lambda_k=\min \mathrm{eig}(F^*_{1:k-1,1:k-1}F_{1:k-1,1:k-1}), 1\le k\le m$. However, computing these $\lambda_k$'s requires an effort that is negligible to the overall flop count for the model parameters that we will consider.

We state the subroutine for computing $\mathrm{LB}_{\mathrm{eig}}$ below.

*Subroutine for computing* $\mathrm{LB}_{\mathrm{eig}}$*:*

*Input:* $R$, $\mathbf{y}_{1:k-1}, \mathbf{s}_{k:m}$, $F=R^{-1}$, $\lambda_k=\min \mathrm{eig}(F^*_{1:k-1,1:k-1}F_{1:k-1,1:k-1}), 1\le k\le m$, $FR_{1:k-2,k-1}=F_{1:k-2,1:k-2}R_{1:k-2,k-1}, 1\le k\le m$, $\mathbf{f}^k_{1:k-1}$.

1. If $k=m$, $\mathbf{f}^{k-1}=F_{1:k-1,1:k-1}(y_{1:k-1}-R_{1:k-1,k:m}\mathbf{s}_{k:m})$; otherwise, $\mathbf{f}^{k-1}=\mathbf{f}^k_{1:k-1}+F_{1:k-1,k}R_{k,k+1:m}\mathbf{s}_{k+1:m}-F_{1:k-1,k}y_k-FR_{1:k-1,k}\mathbf{s}_k$.

2. if $k>1$, $\mathrm{LB}_{\mathrm{eigb}}=\lambda_k\min_{\mathbf{a}\in\mathcal{D}}\|\mathbf{a}-\mathbf{f}^{k-1}\|^2$, otherwise, $\mathrm{LB}_{\mathrm{eigb}}=0$.

We refer to the modification of the sphere decoding algorithm which makes use of the lower bound $\mathrm{LB}_{\mathrm{eigb}}$ as EIGSD algorithm and study its expected computational complexity in the following subsection.

### A. Eigenbound-Performance Comparison

In this subsection we study the performance of EIGSD-algorithm.

In particular, Fig. 6 compares the expected complexity and total number of points in the tree of the EIGSD algorithm to the expected complexity and total number of points of the standard sphere decoder algorithm. We employ both algorithms for

$$\begin{aligned}\mathbf{f}^{(k-1)}&=F_{1:k-1,1:k-1}(\mathbf{y}_{1:k-1}-R_{1:k-1,k:m}\mathbf{s}_{k:m})\\&=\begin{bmatrix}F_{1:k-2,1:k-1}\\ F_{k-1,1:k-1}\end{bmatrix}\mathbf{y}_{1:k-1}-\begin{bmatrix}F_{1:k-2,1:k-2} & F_{1:k-2,k-1}\\ 0 & F_{k-1,k-1}\end{bmatrix}\begin{bmatrix}R_{1:k-2,k:m}\\ R_{k-1,k:m}\end{bmatrix}\mathbf{s}_{k:m}\\&=\begin{bmatrix}F_{1:k-2,1:k-1}\mathbf{y}_{1:k-1}\\ F_{k-1,1:k-1}\mathbf{y}_{1:k-1}\end{bmatrix}-\begin{bmatrix}F_{1:k-2,1:k-2}R_{1:k-2,k:m}+F_{1:k-2,k-1}R_{k-1,k:m}\\ F_{k-1,k-1}R_{k-1,k:m}\end{bmatrix}\mathbf{s}_{k:m}\\&=\begin{bmatrix}F_{1:k-2,1:k-2}\mathbf{y}_{1:k-2}+F_{1:k-2,k-1}y_{k-1}\\ F_{k-1,1:k-1}\mathbf{y}_{1:k-1}\end{bmatrix}-\begin{bmatrix}F_{1:k-2,1:k-2}R_{1:k-2,k:m}\mathbf{s}_{k:m}+F_{1:k-2,k-1}R_{k-1,k:m}\mathbf{s}_{k:m}\\ F_{k-1,k-1}R_{k-1,k:m}\mathbf{s}_{k:m}\end{bmatrix}.\end{aligned}\tag{46}$$

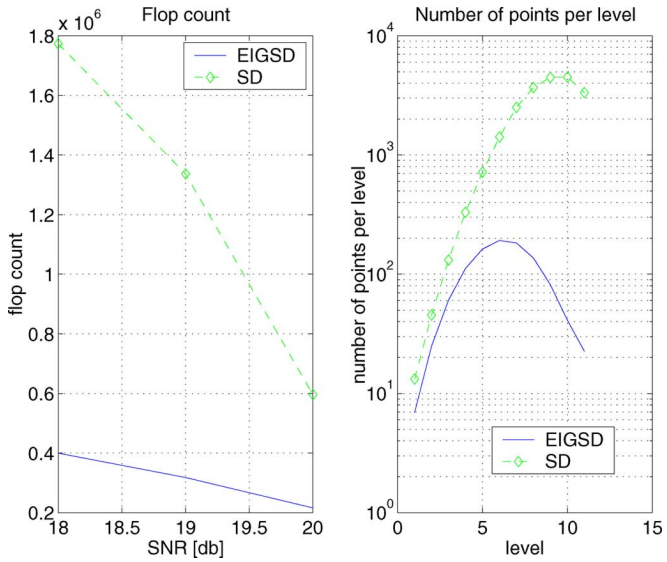



Fig. 6. Computational complexity of the SD and EIGSD algorithms, $m = 12$, $\mathcal{D} = \{-(15/2), -(13/2), \ldots, (13/2), (15/2)\}^{12}$.

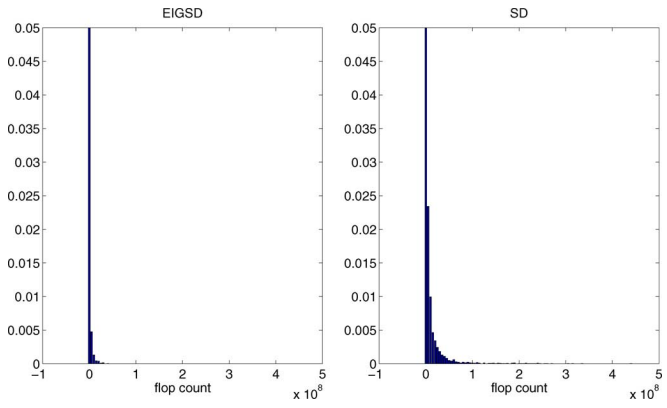



Fig. 7. Flop count histograms for SD and EIGSD algorithms, $m = 12$, SNR $=$ 18 [dB], $\mathcal{D} = \{-(15/2), -(13/2), \ldots, (13/2), (15/2)\}^{12}$.

detection in a multiantenna communication system with 6 antennas, where the components of the transmitted symbol vectors are points in a 256-QAM constellation. Note that the SNR in Fig. 6 is defined as SNR $= 10 \log_{10}(255m/12\sigma^2)$, where $\sigma^2$ is the variance of each component of the noise vector $\mathbf{w}$. Both algorithms choose the initial search radius statistically as in [4] (the sequence of $\epsilon$'s, $\epsilon = 0.9$, $\epsilon = 0.99$, $\epsilon = 0.999$, etc.), and employ the Schnor–Euchner search strategy updating the radius every time the bottom of the tree is reached. As the simulation results in Fig. 6 indicate, the EIGSD algorithm runs more than 4.5 times faster than the SD algorithm.

In Fig. 7, the flop count histograms of SD and EIGSD algorithms are shown. As can be seen, the EIGSD algorithm has significantly better shaped (shorter tail) distribution of the flop count than the SD algorithm.

We would also like to point out that the EIGSD algorithm is not restricted to applications in communication systems. In Fig. 2, we show what its potential can be if applied to a random integer least squares problem. In the problem simulated in Fig. 8, $H$ was generated as an $m \times m$ matrix with i.i.d. Gaussian entries and entries of $\mathbf{y}$ were generated uniformly

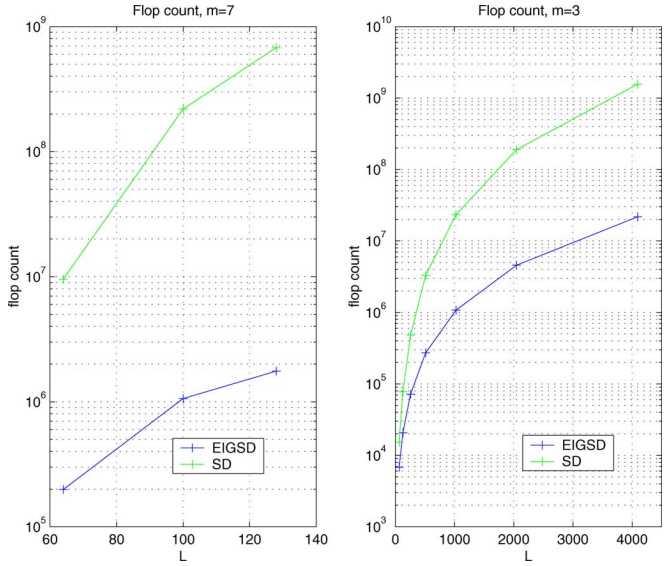



Fig. 8. Computational complexity of the SD and EIGSD algorithms,

$$\mathcal{D} = \left\{-\frac{M-1}{2}, -\frac{M-3}{2}, \ldots, \frac{M-3}{2}, \frac{M-1}{2}\right\}^m.$$

from the interval $[-(M-10/2), (M-10/2)]$. The problem that we were solving was again

$$\min_{\mathbf{s}\in\mathcal{D}^m} ||\mathbf{x} - H\mathbf{s}||_2 \tag{50}$$

where

$$\mathcal{D} = \left\{-\frac{M-1}{2}, -\frac{M-3}{2}, \ldots, \frac{M-3}{2}, \frac{M-1}{2}\right\}.$$

The initial radius $d_g$ was chosen as

$$d_g = ||\mathbf{x} - H\hat{\mathbf{s}}|| \tag{51}$$

where $\hat{s}$ is obtain by rounding the components of $H^{-1}\mathbf{x}$ to the closest element in $\mathcal{D}$. $\hat{s}$ generated in this way is sometimes called the Babai estimate [38]. Fig. 8 compares the expected flop count of the EIGSD and SD algorithms for different large values of $M$. As it can be seen, the larger the set of allowed integers, the better the EIGSD performance.

## VIII. 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 decoding algorithm. However, more significant pruning of the search tree does not, in general, guarantee that the modified algorithm will perform faster than the standard sphere decoding 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: a 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 requirements, in this paper, we introduced a general framework, based on the $H^\infty$ estimation theory, for computing the desired lower bounds. Several special cases of lower bounds were deduced from this framework. We explicitly studied four such lower bounds, and employed them for sphere decoding. The first two correspond to relaxation of the search space to either a sphere or a polytope, while the third one is a slight generalization of the spherical lower bound. The last special case corresponds to bounding the integer least-squares problem with the smallest eigenvalue and requires smaller computational effort than any of the previously mentioned bounds. In addition to $H^\infty$ framework for computing lower bound on the integer least-squares problem, we introduced an SDP-based framework for computing desired lower bound relevant in cases when the original problem is binary.

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

Effectively all algorithms developed in this paper can be divided in two groups depending on the type of the problem that they were designed for. The first group (which includes SDsdp, GSPHSD, SPHSD, and PLTSD) is specifically designed for binary problems, while the second group (EIGSD) is specifically designed for higher order constellation problems. From the results that we presented, the SDsdp, GSPHSD, and EIGSD algorithms seem to outperform the standard SD in the simulated regimes in terms of flop count. Furthermore, the distributions of their flop counts have significantly shorter tail than the distribution of the SD. However, SPHSD and PLTSD do not perform as well as the standard SD in terms of the flop count and flop count histogram. These results suggest that using a lower-bounding technique is useful, but only if the lower bound can be computed in a fast manner.

We should also point out that although we derived it in order to improve the speed of the sphere decoding algorithm, the general lower bound on integer least-squares problems is an interesting result in itself. In fact, the proposed $H^\infty$ estimation framework for the efficient computation of lower bounds on the difficult integer least-squares problems may find applications beyond the scope of the current paper.

The results we present indicate potentially significant improvements in the speed of the sphere decoding algorithm. However, we should note that the proposed $H^\infty$ estimation based framework for bounding integer least-squares problem is only partially utilized. In fact, there are several degrees of freedom in the general $H^\infty$ based bound that are not fully exploited. It is certainly of interest to extend the current work and use the previously mentioned degrees of freedom to further 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.

## REFERENCES


[1] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," *Math. Comput.*, vol. 44, pp. 463–471, Apr. 1985.

[2] T. Coleman and Y. Li, "A reflective Newton method for minimizing a quadratic function subject to bounds on some of the variables," *SIAM J. Optim.*, vol. 6, no. 4, pp. 1040–1058, 1996.

[3] G. Golub and C. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD: The Johns Hopkins Univ. Press, 1996.

[4] B. Hassibi and H. Vikalo, "On the sphere decoding algorithm. Part I: The expected complexity," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pt. 1, pp. 2806–2818, Aug. 2005.

[5] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[6] H. Artes, D. Seethaler, and F. Hlawatsch, "Efficient detection algorithms for MIMO channels: A geometrical approach to approximate ML detection," *IEEE Trans. Signal Process.*, vol. 51, no. 11, pp. 2808–2820, Nov. 2003.

[7] H. Vikalo, B. Hassibi, and T. Kailath, "Iterative decoding for MIMO channels via modified sphere decoder," *IEEE Trans. Wireless Commun.*, vol. 3, no. 6, pp. 2299–2311, Nov. 2004.

[8] B. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel," *IEEE Trans. Commun.*, vol. 51, no. 3, pp. 389–399, Mar. 2003.

[9] R. Gowaikar and B. Hassibi, "Statistical pruning for near-Maximum likelihood decoding," *IEEE Trans. Signal Process.*, vol. 55, no. 6, pp. 2661–2675, Jun. 2007.

[10] M. O. Damen, A. Chkeif, and J.-C. Belfore, "Lattice code decoder for space–time codes," *IEEE Commun. Lett.*, vol. 4, no. 5, pp. 161–163, May 2000.

[11] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Trans. Inf. Theory*, vol. 48, no. 8, pp. 2001–2214, Aug. 2002.

[12] M. Stojnic, H. Vikalo, and B. Hassibi, "A branch and bound approach to speed up the sphere decoder," in *Proc. Int. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, Mar. 18–23, 2005, vol. 3, pp. 429–432.

[13] M. Stojnic, H. Vikalo, and B. Hassibi, "An $H^\infty$-based lower bound to speed up the sphere decoder," presented at the Signal Processing Its Applications in Wireless Communications (SPAWC), New York, Jun. 5–8, 2005.

[14] M. Stojnic, H. Vikalo, and B. Hassibi, "An efficient $H^\infty$ estimation approach to speed up the sphere decoder," presented at the World's Premier Int. Conf. Wireless Networks, Communications, Mobile Computing (Wirelesscomm), Maui, HI, Jun. 13–16, 2005.

[15] M. Stojnic, H. Vikalo, and B. Hassibi, "Further results on speeding up the sphere decoder," in *Int. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, May 15-19, 2006, vol. 3.

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

[17] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear Estimation*. Englewood Cliffs, NJ: Prentice-Hall, 2000.

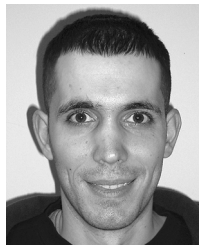
[18] B. Hassibi, "A efficient square-root algorithm for BLAST," *IEEE Trans. Signal Process.*, submitted for publication.

[19] J. Jalden and B. Ottersten, "On the complexity of the sphere decoding in digital communications," *IEEE Trans. Signal Process.*, vol. 53, pp. 1474–1484, 2005.

[20] T. Cui, C. Tellambura, and W. Chen, "Reduced complexity sphere decoding using forcing rules," in *Proc. 38th Asilomar Conf. Signals, Systems, Computers*, Nov. 7–10, 2004, vol. 1, pp. 1218–1221.

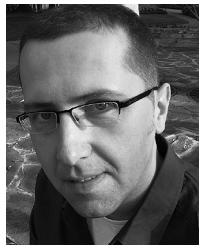
[21] K.-K. Wong and A. Paulraj, "On the decoding order of MIMO maximum-likelihood sphere decoder: Linear and non-linear receivers," in *Proc. IEEE Vehicular Technology Conf.*, May 17–19, 2004, vol. 2, pp. 698–702.

[22] Z. Guo and P. Nilsson, "Algorithm and implementation of the K-best sphere decoding for MIMO detection," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 3, pp. 491–503, Mar. 2006.
[23] G. Latsoudas and D. Sidiropoulus, "A hybrid probabilistic data association-sphere decoding detector for multiple-input-multiple-output systems," *IEEE Signal Process. Lett.*, vol. 12, pp. 309–312, Apr. 2005.
[24] Z. Walnun and G. Giannakis, "Reduced complexity closest point decoding algorithms for random lattices," *IEEE Trans. Wireless Commun.*, vol. 5, no. 1, pp. 101–111, Jan. 2006.
[25] A. D. Murugan, H. E. Gamal, M. O. Damen, and G. Caire, "A unified framework for tree search decoding: Rediscovering the sequential decoder," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 933–953, Mar. 2006.
[26] M. Goemans and Williamnson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *J. ACM*, vol. 42, no. 6, pp. 1115–1145, 1995.
[27] H. Wolkowicz, R. Saigal, and L. Vandenberghe, *Handbook of Semidefinite Programming: Theory, Algorithms, and Applications*. Boston, MA: Kluwer Academic, 2000.
[28] W. K. Ma, T. N. Davidson, K. M. Wong, Z.-Q. Luo, and P. C. Ching, "Quasi-maximum-likelihood multiuser detection using semi-definite relaxation," *IEEE Trans. Signal Process.*, vol. 50, no. 4, pp. 912–922, Apr. 2002.
[29] M. Kisialiou and Z.-Q. Luo, "Performance analysis of quasi-maximum-likelihood detector based on semi-definite programming," in *Proc. Int. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, Philadelphia, PA, Mar. 2005, pp. 433–436.
[30] P. H. Tan and L. K. Rasmussen, "The application of semidefinite programming for detection in CDMA," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 8, pp. 1442–1449, Aug. 2001.
[31] M. Abdi, H. E. Nahas, A. Jard, and E. Moulines, "Semidefinite positive relaxation of the maximum-likelihood criterion applied to multiuser detection in a CDMA context," *IEEE Signal Process. Lett.*, vol. 9, no. 6, pp. 165–167, Jun. 2002.
[32] J. Jalden, C. Martin, and B. Ottersten, "Semidefinite programming for detection in linear systems—Optimality conditions and space–time decoding," in *Proc. Int. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, Apr. 2003.
[33] H. van Maaren and J. P. Warners, "Bound and fast approximation algorithms for binary quadratic optimization problems with application on MAX 2SAT," *Discrete Appl. Math.*, vol. 107, pp. 225–239, 2000.
[34] S. Poljak, F. Rendl, and H. Wolkowicz, "A recipe for semidefinite relaxation for (0,1)-quadratic programming," *J. Global Optim.*, vol. 7, no. 1, pp. 51–73, 1995.
[35] A. Mobasher, A. Taherzadeh, R. Sotirov, and A. Khandani, "A near maximum likelihood decoding algorithm for MIMO systems based on semi-definite programming," in *Proc. IEEE Int. Symp. Information Theory (ISIT'05)*, Adelaide, Australia, Sep. 4–9, 2005, pp. 1686–1690.
[36] J. N. Laneman and G. Wornell, "Distributed space–time protocols for exploiting cooperative diversity in wireless networks," *IEEE Trans. Inf. Theory*, vol. 49, no. 10, Oct. 2003.
[37] Y. Jing and B. Hassibi, "Distributed space–time coding in wireless relay networks," *IEEE Trans. Wireless Commun.*, vol. 5, pp. 3524–3536, Dec. 2006.
[38] M. Grotschel, L. Lovasz, and A. Schriver, *Geometric Algorithms and Combinatorial Optimization*, 2nd ed. New York: Springer-Verlag, 1993.
[39] Y. Eldar and A. Beck, "Hidden convexity based near maximum-likelihood CDMA detection," in *Proc. 39th Annu. Conf. Information Sciences Systems (CISS 2005)*.
[40] A. Burg, M. Borgmann, M. Wenk, Zellweger, W. Fichtner, and H. Bolcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm," *IEEE J. Solid-State Circuits*, vol. 40, no. 7, pp. 1566–1577, Jul. 2005.
[41] J. Jalden and B. Ottersten, "Parallel implementation of a soft output sphere decoder," in *Proc. 39th Asilomar Conf. Signals, Systems, Computers*, 2005.
[42] A. Wiesel, Y. Eldar, and S. Shamai, "Semidefinite relaxation for detection of 16-QAM signalling in MIMO channels," *IEEE Signal Process. Lett.*, vol. 12, no. 9, pp. 653–656, Sep. 2005.



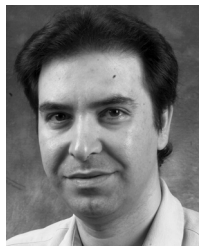
**Mihailo Stojnic** received the M.S. degree in electrical engineering from the California Institute of Technology, Pasadena, in 2003, where he is currently working towards the Ph.D. degree.

His research interests are in mathematics applied in electrical engineering problems.



**Haris Vikalo** was born in Tuzla, Bosnia and Herzegovina. He received the B.S. degree from University of Zagreb, Croatia, in 1995, the M.S. degree from Lehigh University, Bethlehem, PA, in 1997, and the Ph.D. degree from Stanford University, Stanford, CA, in 2003, all in electrical engineering.

He held a short-term appointment at Bell Laboratories, Murray Hill, NJ, in summer 1999. From January 2003 to July 2003, he was a Postdoctoral Researcher, and from July 2003 to August 2007, he was an Associate Scientist at the California Institute of Technology, Pasadena. Since September 2007, he has been with the Department of Electrical and Computer Engineering, The University of Texas at Austin, where he is currently an Assistant Professor. His research interests include genomic signal and information processing, wireless communications, and statistical signal processing.



**Babak Hassibi** was born in Tehran, Iran, in 1967. He received the B.S. degree from the University of Tehran, Iran, in 1989, and the M.S. and Ph.D. degrees from Stanford University, Stanford, CA, in 1993 and 1996, respectively, all in electrical engineering.

From October 1996 to October 1998, he was a Research Associate at the Information Systems Laboratory, Stanford University, and from November 1998 to December 2000, he was a Member of the Technical Staff in the Mathematical Sciences Research Center at Bell Laboratories, Murray Hill, NJ. Since January 2001, he has been with the Department of Electrical Engineering at the California Institute of Technology, Pasadena, where he is currently an Associate Professor. He has also held short-term appointments at Ricoh California Research Center, the Indian Institute of Science, and Linkoping University, Sweden. His research interests include wireless communications, robust estimation and control, adaptive signal processing, and linear algebra. He is the coauthor of the books *Indefinite Quadratic Estimation and Control: A Unified Approach to $H^2$ and $H^\infty$ Theories* (New York: SIAM, 1999) and *Linear Estimation* (Englewood Cliffs, NJ: Prentice-Hall, 2000).

Dr. Hassibi is a recipient of an Alborz Foundation Fellowship, the 1999 O. Hugo Schuck Best Paper Award of the American Automatic Control Council, the 2002 National Science Foundation Career Award, the 2002 Okawa Foundation Research Grant for Information and Telecommunications, the 2003 David and Lucille Packard Fellowship for Science and Engineering, and the 2003 Presidential Early Career Award for Scientists and Engineers (PECASE). He has been a Guest Editor for the IEEE TRANSACTIONS ON INFORMATION THEORY Special Issue on Space-Time Transmission, Reception, Coding and Signal Processing and was Associate Editor for Communications of the IEEE TRANSACTIONS ON INFORMATION THEORY during 2003–2006.