EFFICIENT MAXIMUM LIKELIHOOD DETECTION FOR COMMUNICATION OVER MULTIPLE INPUT MULTIPLE OUTPUT CHANNELS

BY

Karen Su
(Trinity Hall)

Laboratory for Communication Engineering
Department of Engineering
University of Cambridge

Efficient Maximum Likelihood detection for communication over Multiple Input Multiple Output channels

Laboratory for Communication Engineering
Cambridge University Engineering Department
University of Cambridge

by Karen Su
February 2005

Abstract

The Maximum Likelihood (ML) detection of signals transmitted over Multiple Input Multiple Output (MIMO) channels is an important problem in modern communications that is well-known to be NP-complete. However, recent advances in signal processing techniques have led to the development of the *Sphere Decoder (SD)*, which offers ML detection for MIMO channels at an average case polynomial time complexity. Even so, existing SDs are not without their weaknesses: For most current proposals, the decoder performance is highly sensitive to the value chosen for the search radius parameter. The complexity coefficient can also become very large when the Signal-to-Noise Ratio (SNR) is low or when the problem dimension is high, e.g., at high spectral efficiencies.

We report here on two novel contributions designed to address these key weaknesses exhibited by existing schemes. First we present a new decoder, dubbed the *Automatic Sphere Decoder (ASD)* because of its ability to perform sphere decoding without a search radius, thereby eliminating any sensitivity to this parameter. We also define the notion of sphere decoder *efficiency* and prove that the ASD achieves the optimal efficiency over the set of known SDs. Secondly, we propose a pre-processing stage that dramatically improves the efficiency of sphere decoding. The pre-processing is itself computationally efficient and is shown to be effective when applied before existing SDs. The advantage offered is particularly great at low SNRs and at high spectral efficiencies, two important operating regions where current SD algorithms are prohibitively complex.

Combined, the ASD and the proposed pre-processing stage make ML detection of signals transmitted over MIMO channels feasible in practice, even at low SNRs and at high spectral efficiencies. Therefore we believe that these contributions will play an important role in the next generation of wireless communication systems.

Preface

The work detailed in this report was undertaken as a component of my approved course of research for the degree of PhD at the University of Cambridge. My dissertation is tentatively entitled "On Detection and Coding for Communication over Linear MIMO Channels." This report is concerned with the detection aspects of the greater investigation. Previous work by other researchers is appropriately cited in the text; all major results and theorems presented are original contributions. Parts of this work have been accepted for presentation at the *IEEE International Conference on Communications* in May 2005 [15].

I would like to thank my PhD supervisor Dr Ian Wassell for his support of my research at the Laboratory for Communication Engineering. My thanks also to Dr Miguel Rodrigues, for the many insightful discussions that have so greatly enriched my overall research experience, and to Mr Colin Jones, for yet more discussions that, among other things, led to the formulation of the core ideas behind the Automatic Sphere Decoder. Finally, I most gratefully acknowledge the generous assistance of Universities UK, the Cambridge Commonwealth Trust, the Natural Sciences and Engineering Research Council of Canada, and Trinity Hall in providing financial support for my studies.

Karen Su
Cambridge, 2005.

Contents

Chapter 1

Introduction

Driven by the demand for increasingly sophisticated connectivity anytime, anywhere, wireless communications has emerged as one of the largest and most rapidly growing sectors of the global telecommunications industry. One of the most significant technological developments of the last decade, which promises to play a key role in fuelling this tremendous growth, is communication using Multiple Input Multiple Output (MIMO) antenna architectures [11].

In a MIMO system, multiple element antenna arrays are deployed at both the transmitter and the receiver. The communications challenge lies in designing the sets of signals simultaneously sent by the transmit antennas and the algorithms for processing those observed by the receive antennas, so that the quality of the transmission (i.e., bit error probability) and/or its data rate are superior to those supported by traditional single antenna systems. These gains can then provide increased reliability, reduced power requirements and higher composite data rates. What is especially exciting about the benefits offered by MIMO technology is that they can be attained *without the need for additional spectral resources*, which are not only expensive but also extremely scarce.

Over the last ten years, the greatly enhanced performance that is possible over realistic wireless MIMO channels has been both shown theoretically and demonstrated in experimental laboratory settings [2, 3, 8, 16, 17]. Hence the recent explosion of interest from both academic and industrial researchers in the area of signal processing techniques for MIMO systems. Particularly, at the receiver end of the MIMO channel, signal detection has been the subject of intensive study. One of the most important and industrially relevant algorithms to emerge from these efforts is the Sphere Decoder (SD) [7, 9, 18].

The Maximum Likelihood (ML) or optimal detection of signals transmitted over MIMO channels is well-known to be an NP-complete problem. However, the SD has been shown to offer ML detection at a computational complexity that is polynomial in the average case [9].

So bright are its prospects for application in future communication systems that hardware implementations have already been reported in the literature (e.g., [5]).

Even so, existing SDs exhibit two major weaknesses: First, the performance of most current proposals is highly sensitive to the value chosen for the search radius parameter. The successful termination of the algorithm, i.e., returning an optimal solution, as well as its time complexity, are heavily dependent on the search radius [18]. Secondly, although its time complexity is polynomial in the average case, the complexity coefficient can become very large when the SNR is low, or when the problem dimension is high, e.g., at the high spectral efficiencies required to support higher communication rates.

We have developed two novel contributions that successfully tackle these critical challenges. Our presentation begins in Chapter 2 with a formal definition of the ML detection problem. Algebraic tools that are instrumental in the study of sphere decoding are then overviewed; geometric tools are included for the interested reader in Appendix A. We also introduce two generic SDs, based respectively on the Fincke-Pohst and Schnorr-Euchner enumerations. These schemes can be considered as representative of the majority of existing decoders, which have been built around either of these two enumeration strategies, and are therefore used as our benchmarks.

The first of our contributions addresses the sensitivity of the sphere decoder's performance to its radius parameter. Chapter 3 details a new decoding algorithm, dubbed the Automatic Sphere Decoder (ASD) because of its ability to find an ML solution without invoking any notion of search radius, thereby eliminating any sensitivity to this parameter. The concept of sphere decoder *computational efficiency* is also introduced and we prove theoretically that the ASD achieves the optimal efficiency over the set of known SDs.

The SD detects multiple transmitted symbols given multiple observations in a sequential manner. In Chapter 4 we show that the order of symbol detection has a significant effect both on the average computational efficiency of a SD, as well as on the variance of its computation time. Our analysis, distinguished from other works by its geometric approach, reveals a previously unreported property that we find to be an indicator of sphere decoding efficiency. Based on this study, we propose a heuristic ordering rule designed to maximize the computational efficiency of the ASD and demonstrate via simulation that a dramatic reduction in computation time is achievable. The improvement is particularly great at low SNRs and at high spectral efficiencies, two important operating regions where SD algorithms have traditionally been considered to be prohibitively complex.

Chapter 5 concludes this report on our work, which puts forward effective solutions addressing the key issues facing state-of-the-art sphere decoding proposals, under certain conditions at a reduced time complexity. Thus making the sphere decoder an even stronger candidate MIMO detector that is viable for use over a larger range of channel conditions.

Chapter 2

Maximum Likelihood detection

Consider the linear MIMO system diagram shown in Fig. 2.1.[1] To communicate over this channel, we are faced with the task of detecting a set of M transmitted symbols from a set of N observed signals. Our observations are corrupted by the non-ideal communication channel, typically modelled as a linear system followed by an additive noise vector.

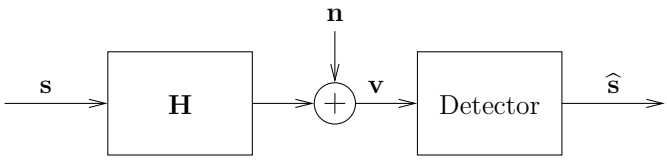


Figure 2.1: A simplified linear MIMO communication system diagram showing the following discrete time signals: transmitted symbol vector $\mathbf{s} \in \mathcal{X}^M$, channel matrix $\mathbf{H} \in \mathbb{R}^{N\times M}$, additive noise vector $\mathbf{n} \in \mathbb{R}^N$, received vector $\mathbf{v} \in \mathbb{R}^N$, and detected symbol vector $\widehat{\mathbf{s}} \in \mathbb{R}^M$.

To assist us in achieving our goal, we draw the transmitted symbols from a known finite alphabet $\mathcal{X} = \{x_1, \ldots, x_B\}$ of size B. The detector's role is then to choose one of the B^M possible transmitted symbol vectors based on the available data. Our intuition correctly suggests that an optimal detector should return $\widehat{\mathbf{s}} = \mathbf{s}_*$, the symbol vector whose (posterior) probability of having been sent, given the observed signal vector $\mathbf{v}$, is the largest:

$$\mathbf{s}_* \triangleq \underset{\mathbf{s}\in\mathcal{X}^M}{\operatorname{argmax}}\, P(\mathbf{s} \text{ was sent} \,|\, \mathbf{v} \text{ is observed}) \tag{2.1}$$

$$= \underset{\mathbf{s}\in\mathcal{X}^M}{\operatorname{argmax}}\, \frac{P(\mathbf{v} \text{ is observed} \,|\, \mathbf{s} \text{ was sent})P(\mathbf{s} \text{ was sent})}{P(\mathbf{v} \text{ is observed})}. \tag{2.2}$$

Equation (2.1) is known as the *Maximum A posteriori Probability (MAP)* detection rule. Making the standard assumption that the symbol vectors $\mathbf{s} \in \mathcal{X}^M$ are equiprobable, i.e.,

[1] Note that all of the signals and coefficients used in our theoretical derivations are represented as real numbers. This mathematical convenience does not limit our results since the complex case where $\mathbf{s} \in (\mathcal{X}^2)^M$ is a vector of M QAM modulated signals, $\mathbf{v} \in \mathbb{C}^N$ and $\mathbf{H} \in \mathbb{C}^{N\times M}$ can be written as an equivalent problem in twice the number of real dimensions, i.e., with $\mathbf{v} \in \mathbb{R}^{2N}$ and $\mathbf{H} \in \mathbb{R}^{2N\times 2M}$, as shown in Appendix C.

that $P(\mathbf{s}$ was sent) is constant, the optimal MAP detection rule can be written as:

$$\mathbf{s}_* = \underset{\mathbf{s}\in\mathcal{X}^M}{\operatorname{argmax}} P(\mathbf{v} \text{ is observed} \,|\, \mathbf{s} \text{ was sent}). \tag{2.3}$$

A detector that always returns an optimal solution satisfying (2.3) is called a *Maximum Likelihood (ML) detector*. If we further assume that the additive noise $\mathbf{n}$ is white and Gaussian, then we can express the ML detection problem of Fig. 2.1 as the minimization of the squared Euclidean distance metric to a target vector $\mathbf{v}$ over an M-dimensional finite discrete search set:

$$\mathbf{s}_* = \underset{\mathbf{s}\in\mathcal{X}^M}{\operatorname{argmin}} |\mathbf{v} - \mathbf{H}\mathbf{s}|^2, \tag{2.4}$$

where borrowing terminology from the optimization literature we call the elements of $\mathbf{s}$ *optimization variables* and $|\mathbf{v} - \mathbf{H}\mathbf{s}|^2$ the *objective function*.

Examples of wireless communications problems that can be modelled in this way, by appropriately defining the entries of channel matrix $\mathbf{H}$, include ML detection of lattice coded signals and QAM-modulated signals transmitted over MIMO flat fading channels and frequency selective fading channels, as well as multi-user channels.

2.1 Mathematical preliminaries

In our derivations, we assume an overdetermined problem, i.e., that $M \leq N$, and that $\mathbf{H}$ is of full rank M. For communication over MIMO flat fading channels, this assumption means that there are at least as many receive antennas (N) as transmit antennas (M).

We make use of the following notational conveniences: Given a square $M \times M$ matrix $\mathbf{A}$, let $\mathbf{a}_i$ denote the i^{th} column vector, a_{ii} the element in the i^{th} row and column position, $\mathbf{A}_{\setminus i}$ the tall submatrix comprised of all columns but the i^{th}, and $\mathbf{A}_{\setminus ii}$ the square submatrix formed by removing the i^{th} row and column. Given a vector $\mathbf{z}$, let z_i denote the i^{th} element and $\mathbf{z}_{\setminus i}$ the vector comprised of all elements but the i^{th}.

We also denote by $\mathbf{0}$ an appropriately sized vector or matrix of zeros, by $\mathbf{I}_M$ the square $M \times M$ identity matrix, by $\mathbf{e}_i$ the i^{th} elementary vector of appropriate length, by $\mathcal{I}$ the index set $\{1, \ldots, M\}$, and by the superscript ^{T} the vector or matrix transpose operation. Unless otherwise indicated, transposition takes precedence over column selection, i.e., given an $M \times M$ matrix $\mathbf{A}$, $\mathbf{A}_i^T$ denotes the i^{th} row of $\mathbf{A}$ written as a column vector and $(\mathbf{a}_i)^T$ the i^{th} column of $\mathbf{A}$ written as a row vector. When it is necessary to distinguish between the optimization variables themselves and the values that they may take, we use the underline notation $\underline{s}_i$ or $\underline{\mathbf{s}}$ to refer to the actual variables, and $s_i \in \mathcal{X}$ or $\mathbf{s} \in \mathcal{X}^M$ to indicate particular values taken.

2.2 Sphere decoding fundamentals

Sphere decoding is based on the enumeration of points in the search set that are located within a sphere of some radius centered at a *target*, e.g., the received signal point. The Fincke-Pohst (F-P) and Schnorr-Euchner (S-E) techniques are two computationally efficient means of realizing this enumeration [7], and so they have come to form the foundation of most existing sphere decoders [6, 9].

Underlying both the F-P and S-E enumerations, and in fact all known SDs, is the QR factorization of the channel matrix: Every N by $M \leq N$ matrix $\mathbf{H}$ with linearly independent columns can be factored into

$$\mathbf{H} = \mathbf{Q} \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}, \tag{2.5}$$

where $\mathbf{Q}$ is $N \times N$ and orthogonal, $\mathbf{R}$ is $M \times M$, upper triangular and invertible, and $\mathbf{0}$ is an $(N - M) \times M$ matrix of zeros.

Since the objective function is invariant under orthogonal transformation, minimization problem (2.4) can be written as

$$\underset{\mathbf{s}\in\mathcal{X}^M}{\operatorname{argmin}} |\mathbf{v} - \mathbf{H}\mathbf{s}|^2 = \underset{\mathbf{s}\in\mathcal{X}^M}{\operatorname{argmin}} \left| \mathbf{Q}^T\mathbf{v} - \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix} \mathbf{s} \right|^2 \tag{2.6}$$

$$= \underset{\mathbf{s}\in\mathcal{X}^M}{\operatorname{argmin}} |\widetilde{\mathbf{v}} - \mathbf{R}\mathbf{s}|^2, \tag{2.7}$$

where $\widetilde{\mathbf{v}} = \left[\mathbf{Q}^T\mathbf{v}\right]_1^M$ extracts the first M elements of the orthogonally transformed target.

The upper triangular structure of the factored matrix then enables the decoder to decompose the equivalent objective function (2.7) recursively as follows:

$$|\widetilde{\mathbf{v}} - \mathbf{R}\mathbf{s}|^2 = d^2(\widetilde{v}_M, r_{MM}s_M) + \left|(\widetilde{\mathbf{v}} - \mathbf{r}_M s_M)_{\setminus M} - \mathbf{R}_{\setminus MM}\mathbf{s}_1^{M-1}\right|^2 \tag{2.8}$$

$$= d^2(\widetilde{v}_M, r_{MM}s_M) + \left|\widetilde{\mathbf{y}}(s_M) - \mathbf{R}_{\setminus MM}\mathbf{s}_1^{M-1}\right|^2 \tag{2.9}$$

$$= \sum_{D=M-1}^{0} d^2\left(\widetilde{y}\left(\mathbf{s}_{D+1}^M\right)_D, r_{DD}s_D\right), \tag{2.10}$$

where $d^2(\cdot)$ is the squared Euclidean distance metric and we call

$$\widetilde{\mathbf{y}}\left(\mathbf{s}_{D+1}^M\right) \triangleq \begin{cases} \widetilde{\mathbf{y}}(\emptyset) = \widetilde{\mathbf{v}}, & D = M \\ \left(\widetilde{\mathbf{y}}\left(\mathbf{s}_{D+2}^M\right) - \mathbf{r}_{D+1}s_{D+1}\right)_{\setminus D+1}, & D = M-1, \ldots, 0 \end{cases} \tag{2.11}$$

a *residual target*.[2] Residual targets are parameterized by a set of $L = M - D$ constraint values applied to optimization variables $\underline{s}_{D+1}, \ldots, \underline{s}_M$. We overmark $\widetilde{\mathbf{y}}$ with a tilde to indicate that it resides in the same orthogonally transformed space as $\widetilde{\mathbf{v}}$. As shown in Appendix A, this space is D-dimensional.

Thus the QR factorization provides a means of evaluating the objective function more efficiently, through decomposition into the summations of (2.10), which contain many shared terms. For instance, the first term of (2.9) is involved in computing the values of the objective function for all B^{M-1} points in the search set satisfying $\underline{s}_M = s_M$. We can therefore associate the constraint $\underline{s}_M = s_M$ with this term.

The summation in (2.10) lends itself naturally to a weighted B-ary tree representation, as shown in Fig. 2.2 for the case where $M = B = 2$ and $\mathcal{X} = \{-1, 1\}$. In this diagram, each of the terms in the summations of (2.10) is associated with a constraint as well as with a branch. Each node then encapsulates a set of constraints $\underline{\mathbf{s}}_{D+1}^{M} = \mathbf{s}_{D+1}^{M}$ that have been applied, as specified by the branches traversed along its path from the root node. By computing (2.11), a residual target can also be associated with each node. We observe that because of the underlying QR factorization, the variables must be constrained in order from $\underline{s}_M$ to $\underline{s}_1$.

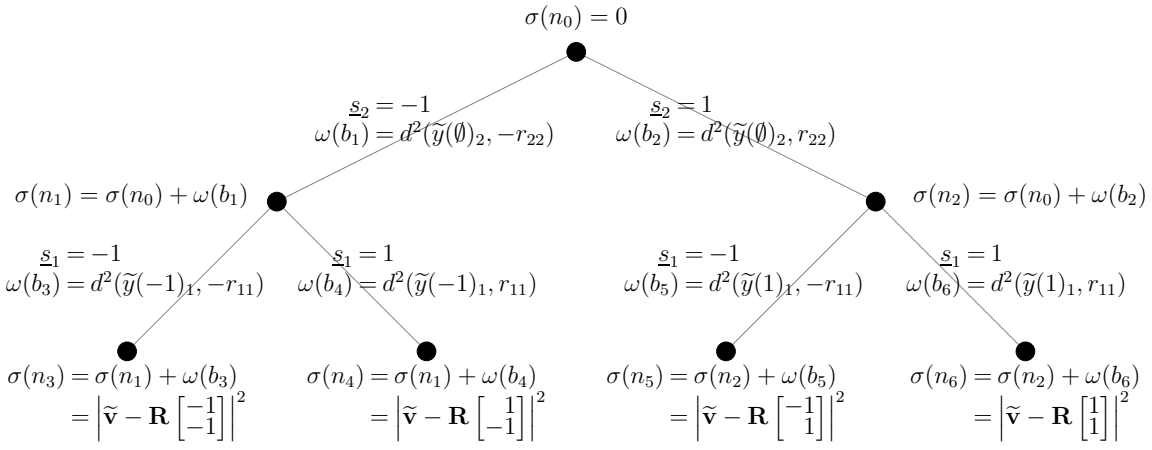


Figure 2.2: A weighted B-ary tree for computing (2.10) with $M = 2$ and $\mathcal{X} = \{-1, 1\}$.

The tree shown in Fig. 2.2 is uniquely determined by problem parameters $\mathbf{v}$, $\mathbf{H}$, and $\mathcal{X}$. We highlight here a few properties of the tree that are important in the study of sphere decoding algorithms. The interested reader is referred to Appendix B for a more formal graph theoretic treatment.

1. The nodes are distributed over $M + 1$ *levels*, numbered from the *root* node n_0 at level 0 to the *leaf* nodes at level M. *Non-leaf* nodes are those at levels 0 through $M - 1$.

[2]Making a slight abuse of notation, we take only the non-zero elements of $\mathbf{r}_i$ in (2.11), thus making it an appropriately-sized vector of length K. We also make the logical interpretation $\mathbf{s}_{M+1}^{M} = \emptyset$.

2. All branches between nodes at levels L and $L+1$ ($L = 0, \ldots, M-1$) are associated with variable $\underline{s}_D$, where we let $D = M - L$ in the statements to follow.

3. The tree is B-ary, i.e., each non-leaf node is the parent of exactly B child nodes. Each child branch corresponds to one of the B values in $\mathcal{X}$ that the associated variable can take. Therefore there are B^L nodes at level L, and each is associated with a set of L constraints $\underline{\mathbf{s}}_{D+1}^{M} = \mathbf{s}_{D+1}^{M}$. In particular, each leaf node is associated with a full vector of constraints $\underline{\mathbf{s}} = \mathbf{s}$, where $\mathbf{s} \in \mathcal{X}^M$ specifies a point in the search set.

4. The tree is a *weighted* tree; non-negative weights $\omega(b_j)$ and $\sigma(n_k)$ are associated with the branches and nodes, respectively. We assign to the root node n_0 the weight 0.

5. Branches from nodes at level L to $L+1$ are assigned weights $d^2\left(\widetilde{y}\left(\mathbf{s}_{D+1}^{M}\right)_D, r_{DD}s_D\right)$, where the constraints $\underline{\mathbf{s}}_{D+1}^{M} = \mathbf{s}_{D+1}^{M}$ are associated with the parent node at level L (Property 3), and $\underline{s}_D = s_D$ is the constraint correponding to the branch itself.

6. Each node weight is the sum of the branch weights along its path from the root, or equivalently the sum of the weights of its parent node and the connecting branch.

7. Along any path from the root to a leaf node, the node weights are non-decreasing.

8. The leaf node weights are precisely equal to the values of summation (2.10), i.e., the values of the objective function evaluated at each of the points in the search set.

Properties 3 and 8 imply that the ML solution is specified by the point in the search set associated with the smallest weight leaf node in the tree of Fig. 2.2. However, there remains an exponential number of leaf nodes to consider, and a comparable number of non-leaf nodes whose weights must all be computed in order to determine those of the leaf nodes. In the next section we discuss how existing sphere decoders are able to reduce the number of computations from an exponential to an average case polynomial number.

2.3 The Fincke-Pohst and Schnorr-Euchner enumerations

A Sphere Decoder (SD) searches for the smallest weight leaf node starting from the root. Because of the recursive definition of the node weights, it must begin at the root and can only expand its knowledge by computing the weights of connected branches and nodes. Through clever pruning of the tree based on intermediate node weights, it is able to declare an ML solution after computing only a polynomial number of weights, in the average case [9]. This pruning is made possible by Property 7.

Recall that geometrically, the weight of each leaf node corresponds to the squared Euclidean distance from a point in the search set to the target. Then given $C \in \mathbb{R}$, we can

enumerate the points located within a sphere of radius C centered at the target by exploring from the root along all branches until a node n_k is encountered such that $\sigma(n_k) > C^2$. Because of Property 7, all descendants of node n_k have weights at least as great as $\sigma(n_k)$. Therefore the points associated with leaf nodes that are descendants of node n_k must lie outside of the search sphere and are not of interest.

The computation time of the tree-based search can then be reduced by pruning at node n_k, i.e., the weights of branches and nodes that are descendants of node n_k need not be computed. The search continues, traversing the tree depth-first [13, Ch. 29], from left to right, until all nodes having weights not greater than C^2 are discovered. It then returns a list of leaf nodes that correspond to points located within the search sphere. This description summarizes the behaviour of the Fincke-Pohst (F-P) enumeration with respect to the tree in Fig. 2.2. More details on its implementation can be found in works such as [1, 9].

An important characteristic of the F-P strategy is that a search radius must be specified. However, if C is too large, many node weights will have to be computed and a large number of leaf nodes may also be returned. If it is too small, no leaf nodes will be found and the decoder must then be restarted with a larger search radius. Both of these factors negatively impact the overall computation time, and thus it is well-known that one of the main weaknesses of the F-P decoder is the sensitivity of its performance to the choice of C. A typical choice is the distance to the *Babai point* [1], which is a point in the search set and therefore we can be assured that at least one leaf node will be found. Our first benchmark decoder, referred to as the *FPB*, uses this value of C and the F-P enumeration.

The Schnorr-Euchner (S-E) enumeration adds a small but significant refinement to the F-P approach. In the F-P strategy, the tree is traversed depth-first and from left to right, i.e., the children of a node are considered in order of increasing $s_i \in \mathcal{X}$, where i is the level of the parent node and we recall that each of its children is associated with applying the additional constraint $\underline{s}_i = s_i$. The S-E strategy also advocates traversing the tree depth-first, but instead of considering child nodes from left to right, it computes their connecting branch weights and then explores them in increasing order of these weights.

As a result, it has been shown that the S-E enumeration discovers eligible leaf nodes more quickly than the F-P enumeration [1]. If this were the only refinement, the S-E enumeration would still have to compute the same number of branch and node weights as the F-P strategy. However, observe that once a leaf node n_l is discovered, the search radius can be adaptively reduced to $C = \sqrt{\sigma(n_l)}$. In other words, after having discovered a point in the search set, we become interested only in locating those points that are even closer to the target than that point. By adaptively adjusting the search radius, this decoder based on the S-E enumeration has become the current state-of-the art [5–7]. Thus it is our second benchmark decoder, referred to as the *SEA*.

Chapter 3

Automatic sphere decoding

In Section 2.2 we saw that sphere decoding amounts to searching for the smallest weight leaf node in a B-ary tree with $M+1$ levels. We have also seen that one of the main weaknesses of current proposals is the sensitivity of their computation time to the value chosen for the search radius parameter C. To tackle this problem, algorithms have been proposed that adaptively adjust the search radius during execution of the decoder [1, 7], which are shown to be less sensitive to its value [6], and where preprocessing is applied to find an optimal search radius before starting the decoding stage [10].

In our work, we take a fresh approach to the problem. Recall that a Sphere Decoder (SD) starts from the root node and expands its knowledge of other nodes in the tree by computing the weights of connected branches and nodes. However, its purpose is not to completely explore the tree. Instead it seeks to explore just enough of the tree in order to find a point in the search set that specifies an ML solution.

To develop this idea more formally, let a *node* n be a 4-tuple $(\sigma, L, \widetilde{\mathbf{y}}, \mathbf{s}_{D+1}^{M})$, where the weight σ is in $\mathbb{R}_{\geq 0}$, the level L is in $\{0, \ldots, M\}$, the associated residual target $\widetilde{\mathbf{y}}(\mathbf{s}_{D+1}^{M})$ is in $\mathbb{R}^{D}$, the associated vector of applied constraint values $\mathbf{s}_{D+1}^{M}$ is in $\mathcal{X}^{L}$, and we recall that $D = M - L$ as before. We can then define the action of *expanding* a node:

Definition 3.1 *Given upper triangular matrix* $\mathbf{R}$*, alphabet* $\mathcal{X} = \{x_1, \ldots, x_B\}$ *of size* B *and non-leaf node* $n = (\sigma, L, \widetilde{\mathbf{y}}, \mathbf{s}_{D+1}^{M})$*, let* expanding n *be defined as processing* σ*,* L*,* $\widetilde{\mathbf{y}}$*, and* $\mathbf{s}_{D+1}^{M}$ *to generate its* B *children* $\{n_{c_1}, \ldots, n_{c_B}\}$ *as follows:*

$$n_{c_j} = \left(\sigma + d^2(\widetilde{y}_D, r_{DD} x_j),\ L+1,\ (\widetilde{\mathbf{y}} - \mathbf{r}_D x_j)_{\setminus D},\ \begin{bmatrix} x_j \\ \mathbf{s}_{D+1}^{M} \end{bmatrix} \right). \tag{3.2}$$

We begin by asking the following question: Assuming a tree-based search is applied to the problem, what is the smallest number of nodes in Fig. 2.2 that must be expanded in order to obtain an ML solution? Given the weight of the smallest weight leaf node, denoted σ_*, it is clear that no more than all nodes having weights less than σ_* must be expanded.

Expanding any non-leaf node whose weight is greater than or equal to σ_* can only lead to the discovery of leaf nodes whose weights are also greater than or equal to σ_*.

Therefore, both the FPB and the SEA decoders may expand more nodes than necessary because they rely on a search radius to dictate whether or not a node should be expanded during the enumeration. Even though it may be adaptively reduced, if the squared search radius is ever larger than σ_*, then it is possible for nodes having weights greater than or equal to σ_* to be expanded. In contrast, the Automatic Sphere Decoder (ASD) is designed to expand precisely the number of nodes that is needed to establish an ML solution. It is also able to do so without prior knowledge of the optimal leaf node weight σ_*.

3.1 A novel approach

The ASD makes use of the tree structure in Fig. 2.2 to eliminate the need for a radius parameter. It efficiently searches for the smallest weight leaf node by maintaining a list of nodes $\mathcal{N}_b$ that define the border between the explored and unexplored parts of the tree.

Initially this list contains only the root node, to which is assigned the 4-tuple $(0, 0, \widetilde{\mathbf{v}}, \emptyset)$. In each iteration, the ASD selects and expands the border node with the smallest weight.[1] The expanded node is then deleted from $\mathcal{N}_b$, since it is no longer on the border, and replaced by its B children. Note that although the decoder knows about the border nodes, they are considered to be unexplored. The first time a leaf node n_{l_0} is selected for expansion, the decoder returns the associated vector of applied constraint values $\mathbf{s}(n_{l_0})$, along with its weight $\sigma(n_{l_0})$, and terminates. Pseudocode is given in Algorithm 1:

Algorithm 1 Automatic Sphere Decoder $ASD(\mathbf{v}, \mathbf{H}, \mathcal{X})$

Initialize $\mathcal{N}_b$ as $\{n_0\}$	Initialize border nodelist
$n \longleftarrow$ Get&DeleteMin($\mathcal{N}_b$)	Select root node
while n is not a leaf node **do**	Until leaf node selected
$\{n_{c_1}, \ldots, n_{c_B}\} \longleftarrow$ Expand(n)	Expand selected node (3.2)
Insert n_{c_j} into $\mathcal{N}_b$, $j = 1, \ldots, B$	Insert children into nodelist
$n \longleftarrow$ Get&DeleteMin($\mathcal{N}_b$)	Select smallest weight node
end while	
Return $\mathbf{s}_* = \mathbf{s}$, $C_* = \sqrt{\sigma}$	Report optimal solution and search radius

Intuitively, the strategy employed by the ASD ensures that whenever a node is selected for expansion, *all nodes in the tree having weights less than that node will already have been explored*. Consequently, all unexplored nodes must have weights that are at least as great as that of the selected node. Thus, when the first leaf node n_{l_0} is selected for expansion, it must be the case that the weights of the other leaf nodes, none of which have yet been explored (although they may also be on the border), are greater than or equal to $\sigma(n_{l_0})$. Since these

[1] In the event of a tie, the border node with the smallest weight and the lowest level is selected.

weights correspond to values of the objective function evaluated for a given constraint value vector, $\mathbf{s}(n_{l_0})$ can be declared an ML solution.

The reader is referred to Appendix B for proofs that the constraint value vector $\mathbf{s}$ returned by Algorithm 1 is an ML solution to minimization problem (2.4), and that the square root of the returned weight $\sqrt{\sigma}$ is the optimal search radius, denoted C_*.

3.2 The computational efficiency of sphere decoding

One of the main difficulties encountered when directly comparing the computation times of different sphere decoders, e.g., in terms of floating-point operations, is the implementation-dependent nature of such a comparison. In our work, we break down the operations performed by SDs into three categories: expanding nodes, determining the next node to expand, and maintaining a nodelist, if necessary. We claim that all computations involved in sphere decoding can be grouped into one of these categories. Further, we argue that in order to provide a fair comparison, the computation time required for a single node expansion must be fixed, i.e., equally optimized for all of the decoders being compared.

Therefore, we propose to evaluate and compare the computational performance of different SDs in a theoretical framework, by considering the number of nodes expanded during their execution, denoted ν. We believe that ν is an important characteristic for distinguishing between different decoding algorithms, and begin our study of this quantity by providing a lower bound on its value:

Proposition 3.3 *The number of nodes expanded by a sphere decoder satisfies* $\nu \geq M$.

Proof We recall that the tree structure shown in Fig. 2.2 underlies the operation of sphere decoders. Because all of the leaf nodes are at level M, when exploring from the root, the smallest number of expansions required to reach the first leaf node is M. ∎

The lower bound of M on the number of nodes expanded by a sphere decoder suggests the following definition of efficiency:

Definition 3.4 *Given target* $\mathbf{v}$*, channel matrix* $\mathbf{H}$*, finite alphabet* $\mathcal{X}$*, and search radius* C*, let the number of nodes expanded by sphere decoder* $SD(\mathbf{v}, \mathbf{H}, \mathcal{X}, C)$ *be denoted* $\nu_{SD}(\mathbf{v}, \mathbf{H}, \mathcal{X}, C)$*. The* computational efficiency *of algorithm* $SD(\mathbf{v}, \mathbf{H}, \mathcal{X}, C)$ *is then defined as*

$$\eta_{SD}(\mathbf{v}, \mathbf{H}, \mathcal{X}, C) \triangleq \frac{M}{\nu_{SD}(\mathbf{v}, \mathbf{H}, \mathcal{X}, C)}. \tag{3.5}$$

Definition 3.4 ensures that $0 < \eta_{SD} \leq 1$ and that $\eta_{SD} = 1 \longleftrightarrow \nu_{SD} = \nu_{min}$. Thus providing an intuitively satisfying description of the efficiency of a sphere decoding algorithm. Armed with this definition, we can then present the following result:

Theorem 3.6 *Given target* $\mathbf{v}$*, channel matrix* $\mathbf{H}$*, and finite alphabet* $\mathcal{X}$*, the computational efficiency of Algorithm 1 satisfies*

$$\eta_{ASD}(\mathbf{v}, \mathbf{H}, \mathcal{X}) \geq \eta_{SD}(\mathbf{v}, \mathbf{H}, \mathcal{X}, C), \tag{3.7}$$

for all other optimal sphere decoding algorithms $SD(\mathbf{v}, \mathbf{H}, \mathcal{X}, C)$*.*

Proof Corollary B.9 asserts that all nodes expanded by Algorithm 1 satisfy $\sigma(n) \leq C_*^2$. Since it is the optimal search radius, $C_* \leq C$ for the radius C of any sphere decoder that terminates successfully. Therefore n is also expanded by these decoders and $\nu_{ASD}(\mathbf{v}, \mathbf{H}, \mathcal{X}) \leq \nu_{SD}(\mathbf{v}, \mathbf{H}, \mathcal{X}, C)$ for any other optimal sphere decoding algorithm. The result follows from the inverse relationship between ν and η given in (3.5). ■

Thm. 3.6 demonstrates that there is no known SD that expands fewer nodes than the ASD during the decoding process. Thus the ASD achieves the optimal computational efficiency. However, note that this result does not necessarily imply that the overall computation time of the ASD is lower than that of all known decoders, since we have not taken the node selection and nodelist maintenance operations into consideration.

3.3 Complexity analysis and results

Two key advantages of the automatic approach are that the search radius parameter is no longer needed and the number of nodes expanded by the proposed decoder is always less than or equal to that expanded by existing sphere decoders. Since the time complexity of a sphere decoding algorithm is dominated by the product of the number of nodes expanded ν and the per node processing time τ, both of these factors serve to improve the time complexity of the automatic decoder with respect to that of existing sphere decoders.

In Fig. 3.1 we compare the average number of nodes expanded by the ASD to that expanded by the FPB and SEA benchmark decoders over a wide range of SNRs. The simulations were conducted over a (complex) MIMO Rayleigh flat fading channel with Additive White Gaussian Noise (AWGN), four transmit and four receive antennas. The decoders were tested using the QPSK, 16-QAM and 64-QAM symbol alphabets, which correspond to spectral efficiencies of 8, 16 and 24 bits/s/Hz, respectively.[2] The experimental setup is described in more detail in Appendix C.

Fig. 3.1 shows that the SEA and ASD decoders significantly outperform the FPB with respect to the average number of nodes expanded during the decoding stage. As predicted by Thm. 3.6, $\nu_{ASD} \leq \nu_{SEA}$ regardless of the SNR. For all of the decoders, there is a lower

[2]Results for the FPB decoder are only shown for the QPSK case, as even its average performance was found to be degrade very quickly with increasing search space dimension.

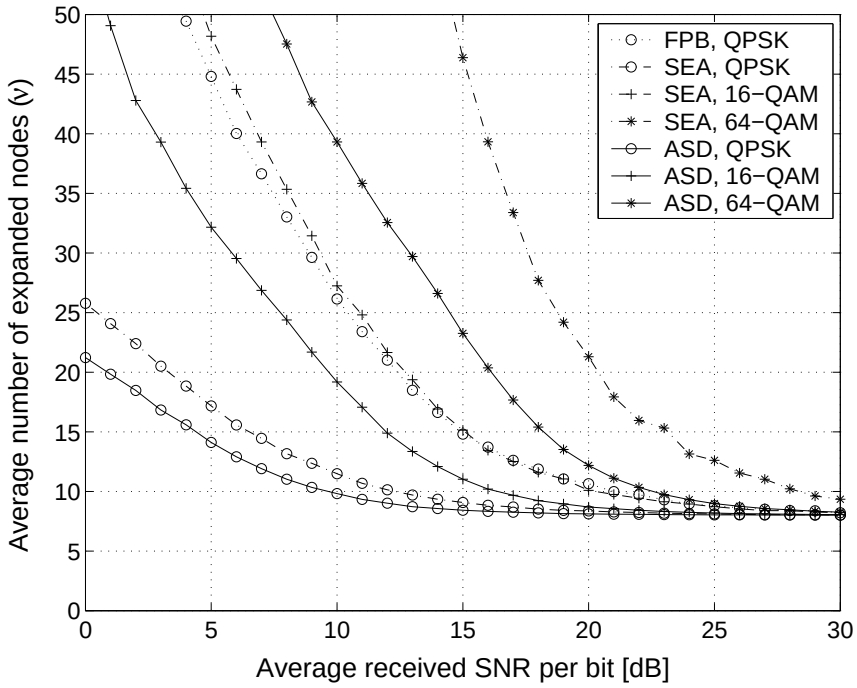


Figure 3.1: Average number of expanded nodes (ν) vs. average received SNR per bit for the FPB, SEA and proposed automatic sphere decoders over a 4:4 MIMO flat fading channel using QPSK, 16-QAM and 64-QAM modulations.

bound of $\nu \geq 2M$, where we recall that M is the number of transmit antennas; the factor of two arises because there are two real dimensions per complex dimension.

In addition to ν, the overall computation time of a SD depends on the per node processing time τ. This parameter is difficult to quantify because of its strong implementation-dependence. In [9], it is argued that the FPB decoder performs a number of elementary operations per expansion that is linear in i, where $i = 1, \ldots, M$ is the level of the node being expanded. Therefore the per node processing time is $\mathcal{O}(M)$. This base set of operations required per expansion persists in the SEA and ASD algorithms. On top of these, the SEA incurs a constant complexity penalty due to the S-E enumeration, which imposes an expansion order on the nodes at each level. Therefore, τ_{SEA} is also linear in M.

The ASD decoder incurs a slightly greater penalty because it imposes an expansion order on all nodes across all levels. In our implementation of ASD, the nodelist $\mathcal{N}_b$ is stored as a *(binary) heap* [13, Ch. 11]. Using this efficient data structure, the Get&DeleteMin and Insert functions can each be completed in at most $\mathcal{O}(M \log_2 B)$ time. Thus the complexity penalty is linear, and like τ_{SEA}, τ_{ASD} is still linear in M, albeit with a larger coefficient. In practice we have found that the time complexity penalty is not significant, as long as appropriate data structures are used in the implementation.

Chapter 4

Ordering for computational efficiency

Even when using the more efficient S-E enumeration, it is known that the computational cost of sphere decoding is highly sensitive to the ordering of the columns of the channel matrix [7]. In this chapter, we investigate the impact of different orderings on the computational efficiency of SD algorithms. Unlike previous proposals, where ordering decisions are based only on the channel matrix [7], we demonstrate that the optimal ordering for sphere decoding depends on the channel matrix and also on the target. Drawing on our findings, we develop an efficient algorithm for computing an enhanced ordering and evaluate its effectiveness in reducing the computational cost incurred by various sphere decoders.

4.1 Case study: Orderings and sphere decoding

To study their effect on sphere decoders, we begin by defining an *ordering* Π as a permutation of the index set $\mathcal{I}$ and write its elements as the ordered set

$$\Pi = \{i_M, i_{M-1}, \ldots, i_1\}\,, \tag{4.1}$$

where the mapping from $\mathcal{I}$ to Π is a bijection. Π also implicitly defines an $M \times M$ permutation matrix $\mathbf{P}_\Pi \triangleq \left[\, e_{i_M}\ e_{i_{M-1}}\ \cdots\ e_{i_1} \,\right]$ that can be used to permute the columns of channel matrix $\mathbf{H}$ through right multiplication:

$$\mathbf{H}\mathbf{P}_\Pi = [\, \mathbf{h}_{i_M} \cdots \mathbf{h}_{i_1} \,]\,. \tag{4.2}$$

Recall from Section 2.2 that the underlying QR factorization forces sphere decoders to constrain the optimization variables in order from $\underline{s}_M$ to $\underline{s}_1$, i.e., in the reverse order of the columns of $\mathbf{H}$. By applying ordering Π to the channel matrix before computing its QR

factorization, we allow the variables to be constrained in a strategic rather than random order $\underline{s}_{i_1}, \ldots, \underline{s}_{i_M}$.

Since the ASD achieves the optimal sphere decoding efficiency over the set of known SDs (Thm. 3.6), we propose to use $\eta_{ASD}(\mathbf{v}, \mathbf{HP}_\Pi, \mathcal{X})$, or simply $\nu_{ASD}(\mathbf{v}, \mathbf{HP}_\Pi, \mathcal{X})$, to quantify the computational efficiency of ordering Π. Because it expands those nodes contained in the search sphere of optimal radius C_* centered at the target $\mathbf{v}$ (Corollary B.9), ν_{ASD} captures an essential property of sphere decoder geometry. Therefore, unlike other decoders, when it is invoked with different orderings applied to the channel matrix, any variations in ν_{ASD} can be attributed solely to changes in the decoder geometry induced by the ordered channel matrix. This approach enables us to decouple the ordering part of the problem from the sphere decoder itself.

Next, we consider how the ASD decodes a received signal transmitted using BPSK modulation ($\mathcal{X} = \{-1, 1\}$) over a sample 2×2 channel

$$\mathbf{H} = \begin{bmatrix} 1.13 & -5.65 \\ 6.78 & -2.20 \end{bmatrix}. \tag{4.3}$$

Fig. 4.1 shows a map of the optimal ordering regions of the columns of sample channel matrix $\mathbf{H}$ over the domain of possible targets $\mathbf{v}$, where we call ordering Π_* and permuted channel matrix $\mathbf{H}_* = \mathbf{HP}_{\Pi_*}$ *optimal* if

$$\nu_{ASD}(\mathbf{v}, \mathbf{H}_*, \mathcal{X}) \leq \nu_{ASD}(\mathbf{v}, \mathbf{HP}, \mathcal{X}) \tag{4.4}$$

for all $M!$ possible permutation matrices $\mathbf{P}$.

Given the two-column channel matrix in (4.3), there are two possible orderings of $\mathbf{H}$. On the diagram in Fig. 4.1, light shading indicates points $\mathbf{v}$ for which $\nu_{ASD}(\mathbf{v}, \mathbf{H}[\,\mathbf{e}_1\ \mathbf{e}_2\,], \mathcal{X}) = \nu_{ASD}(\mathbf{v}, \mathbf{H}[\,\mathbf{e}_2\ \mathbf{e}_1\,], \mathcal{X})$, i.e., where both orderings result in equivalent decoder efficiencies, medium shading shows where the decoder using ordering $\Pi = \{1, 2\}$ expands fewer nodes than the alternative, and dark shading, where ordering $\Pi = \{2, 1\}$ is favoured.

The optimal ordering map in Fig. 4.1 illustrates an important property of ordering for sphere decoding that has been identified through our work: *The optimal ordering for efficient sphere decoding depends not only on the channel matrix* $\mathbf{H}$*, but also on the target* $\mathbf{v}$. This characterization is different from that underlying previous proposals, e.g., applying the Vertical Bell Labs Layered Space-Time (V-BLAST) ordering borrowed from the spatial multiplexing literature [19] to sphere decoding [7], since only $\mathbf{H}$ is considered in such designs.

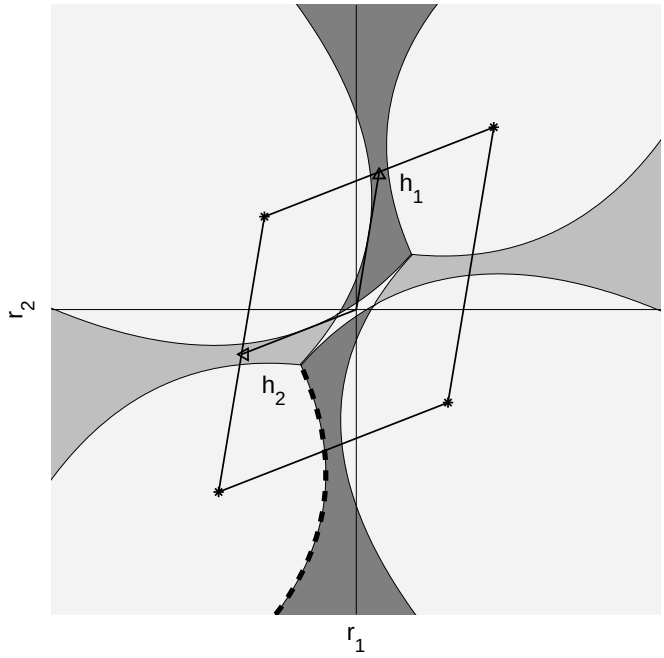


Figure 4.1: Optimal ordering of the columns of channel matrix $\mathbf{H}$ as a function of target $\mathbf{v}$: Light shading indicates that both orderings are equivalent, medium shading that $\mathbf{H}_* = \mathbf{H}[\,\mathbf{e}_1\ \mathbf{e}_2\,]$, and dark shading that $\mathbf{H}_* = \mathbf{H}[\,\mathbf{e}_2\ \mathbf{e}_1\,]$.

4.2 An enhanced ordering scheme

To develop an ordering that promotes sphere decoder efficiency, we consider the tree structure in Fig. 2.2. Recall that one optimization variable is associated with all of the branches originating from the nodes at each level. Each of the $M!$ possible orderings associates different variables with these sets of branches and therefore defines a distinct set of branch and node weights. Because the ASD expands those nodes having weights less than the squared optimal search radius C_*^2 (Corollary B.9), it is this difference in the node weights that so greatly affects the computational performance of SDs under different orderings.

Our strategy is to start at the root node and descend to a leaf, possibly not the one with the smallest weight, in M steps. The resulting path then implicitly defines an ordering. Initially, we are armed only with the structure of the tree; no optimization variables are associated with any of the branches. At each level, we choose the best variable to constrain given the available information. Once a variable $\underline{s}_{i'}$ is chosen, it is removed from contention in subsequent steps by applying one of the B constraints $\underline{s}_{i'} = x_j$ associated with the chosen variable, i.e., traversing one of the outgoing branches to a child node.

To choose the next variable to constrain, the following decision criterion is proposed:

$$i' = \underset{i \text{ remaining}}{\operatorname{argmax}}\ \omega\left(b(\underline{s}_i = x_{[2]})\right), \tag{4.5}$$

where we recall that ω is a branch weight. Its parameter $b\left(\underline{s}_i = x_{[2]}\right)$ is the *second branch associated with variable* $\underline{s}_i$*, taken in order of ascending weight*, i.e., $\left\{x_{[1]}, \ldots, x_{[B]}\right\}$ is a permutation of alphabet $\mathcal{X}$ such that $\omega\left(b\left(\underline{s}_i = x_{[1]}\right)\right) \leq \cdots \leq \omega\left(b\left(\underline{s}_i = x_{[B]}\right)\right)$.

The rationale behind decision criterion (4.5) is to select at each level the variable whose resulting branch weights may be expected to most encourage heavy pruning of the search tree. To achieve this goal, we recall that B adjacent branches originate from each non-leaf node. Observe that the branches with the smallest weights $b\left(\underline{s}_i = x_{[1]}\right)$ are by definition the most likely to be explored by the ASD, i.e., they are not likely to be pruned. Therefore our heuristic ordering rule is designed to choose the optimization variable such that the weight of the adjacent branch having the *second smallest* weight $\omega\left(b\left(\underline{s}_i = x_{[2]}\right)\right)$ is maximized. Thus making it more likely that all $B-1$ adjacent nodes are pruned by the ASD.

Having chosen variable i' as the next to constrain, we then traverse branch $b\left(\underline{s}_{i'} = x_{[1]}\right)$, i.e., the one that we also expect the ASD to explore. This process is repeated until all M selections are made. Pseudocode to compute proposed ordering $\widehat{\Pi}$ is given in Algorithm 2, where the function $\mathsf{Child}(n, b)$ returns the child of node n along branch b:

Algorithm 2 An Enhanced Ordering $\widehat{\Pi}(\mathbf{v}, \mathbf{H}, \mathcal{X})$

$\mathcal{I} \longleftarrow \{1, 2, \ldots, M\}$	Initialize index set
$n \longleftarrow n_0$	Select root node
for each level L from 1 to M **do**	
$\quad i_L \longleftarrow \text{argmax}_{i \in \mathcal{I}}\, \omega\left(b\left(\underline{s}_i = x_{[2]}\right)\right)$	Assign L^{th} variable to constrain
$\quad \mathcal{I} \longleftarrow \mathcal{I} \setminus i_L$	Remove i_L from index set
$\quad n \longleftarrow \mathsf{Child}\left(n, b\left(\underline{s}_{i_L} = x_{[1]}\right)\right)$	Traverse smallest weight branch to next node
end for	
Return $\widehat{\Pi} = \{i_M, \ldots, i_1\}$	

4.3 Performance evaluation

We present three evaluations of the performance of the enhanced ordering scheme. First we consider how closely the orderings of Algorithm 2 correspond to the optimal ones confirmed by simulation (Fig. 4.1). Fig. 4.2 depicts a map of the proposed ordering decisions $\widehat{\mathbf{H}} = \mathbf{HP}_{\widehat{\Pi}}$ over the domain of possible targets $\mathbf{v}$, given the same sample channel as before (4.3).

In this two-dimensional example, the match between the orderings computed by Algorithm 2 and the optimal orderings can easily be verified graphically. A similar correspondence is evidenced when using higher order modulations such as 16-QAM; graphical results for this case are shown in Fig. 4.3. We also provide a proof of the optimality of the proposed ordering for all channel matrices $\mathbf{H}$ and targets $\mathbf{v}$ when $M = B = 2$ in Appendix D.

One of the key hypotheses underlying our work is that an ordering effective in enhancing the performance of the ASD should also enhance that of other sphere decoding algorithms.

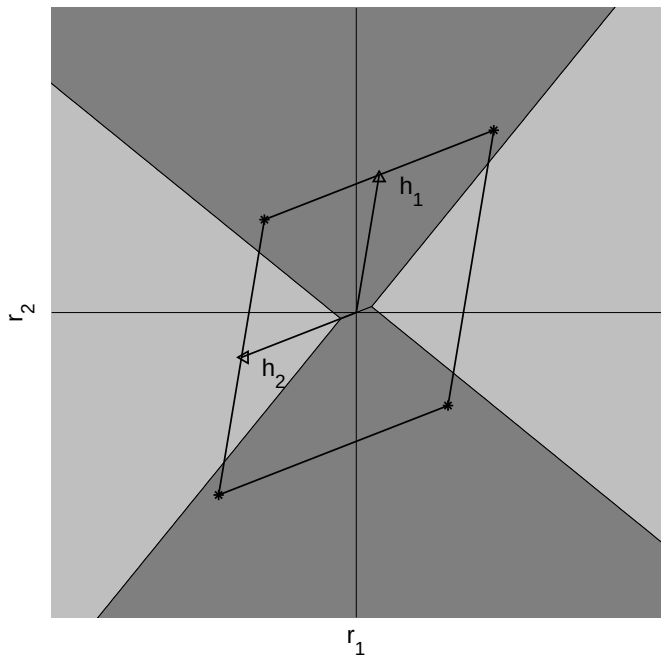
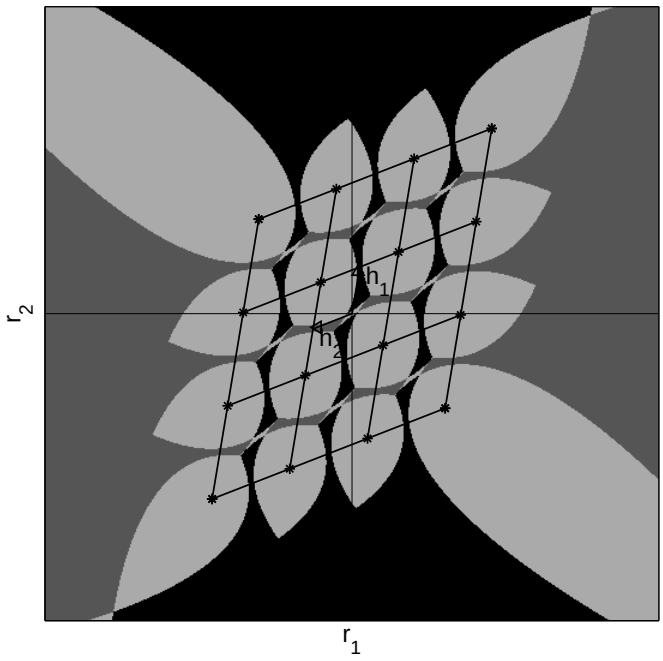


Figure 4.2: Proposed ordering of the columns of a sample two-dimensional (real) channel matrix $\mathbf{H}$ as a function of target $\mathbf{v}$: Medium shading indicates where $\widehat{\mathbf{H}} = \mathbf{H}[\,\mathbf{e}_1\ \mathbf{e}_2\,]$, and dark shading where $\widehat{\mathbf{H}} = \mathbf{H}[\,\mathbf{e}_2\ \mathbf{e}_1\,]$.

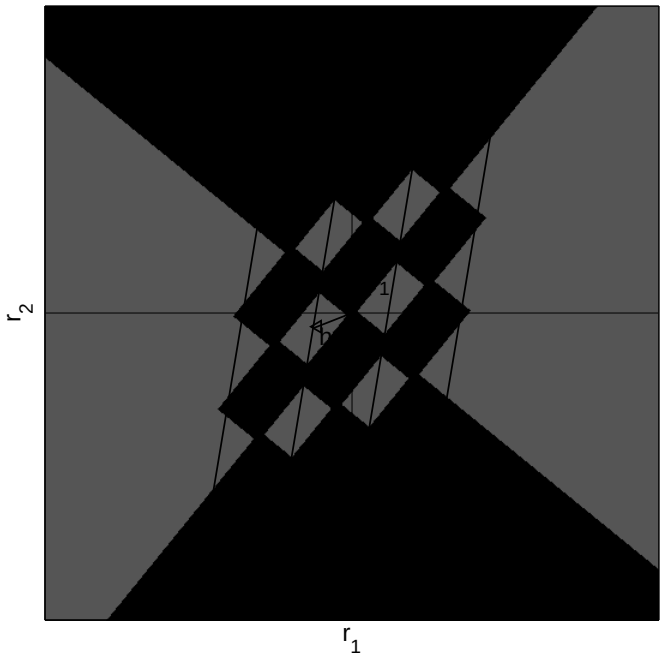
Thus another important evaluation of the performance of enhanced ordering scheme is given by how successful it is in reducing the number of nodes expanded by a standard sphere decoder. Fig. 4.4 shows the average number of nodes expanded by the state-of-the-art SEA benchmark decoder described in Section 2.3 as a function of the SNR. The performance curves obtained under three orderings are reported: Random ordering, that used in V-BLAST detection [19], and that proposed by Algorithm 2.

A vast improvement is realized by the new ordering scheme, especially at low SNRs, where the complexity of existing sphere decoders is not considered to be competitive. What is particularly noteworthy is that the improvement remains significant even for higher modulation orders. Although the enhanced ordering was derived using the ASD, simulation results demonstrate that the advantage is transferable to other sphere decoders as well.

Finally, we consider how the enhanced ordering fares over different realizations of the channel matrix $\mathbf{H}$: Is it the case that there are a few "easy-to-order" channels, or is it generally able to provide some benefit? Figs. 4.5(a)-4.5(c) present histograms showing estimates of the probability density functions $f_{\nu_{ASD}}(\nu_{SEA})$ of the number of nodes expanded by the SEA decoder when using random ordering, that used in V-BLAST detection, and the proposed approach. In Fig. 4.5(d), the density function $f_{\nu_{ASD}}(\nu_{ASD})$ obtained when combining the enhanced ordering with the ASD is also shown for comparison. 10,000 different channel realizations were generated for this experiment, which was conducted at an SNR



(a) Optimal ordering map.



(b) Proposed ordering map.

Figure 4.3: Optimal and proposed orderings of the columns of channel matrix $\mathbf{H}$ as a function of target $\mathbf{v}$: Light shading indicates that both orderings are equivalent, medium shading where $\mathbf{H}_*$ or $\widehat{\mathbf{H}} = \mathbf{H}[\,\mathbf{e}_1\ \mathbf{e}_2\,]$, and dark shading where $\mathbf{H}_*$ or $\widehat{\mathbf{H}} = \mathbf{H}[\,\mathbf{e}_2\ \mathbf{e}_1\,]$.

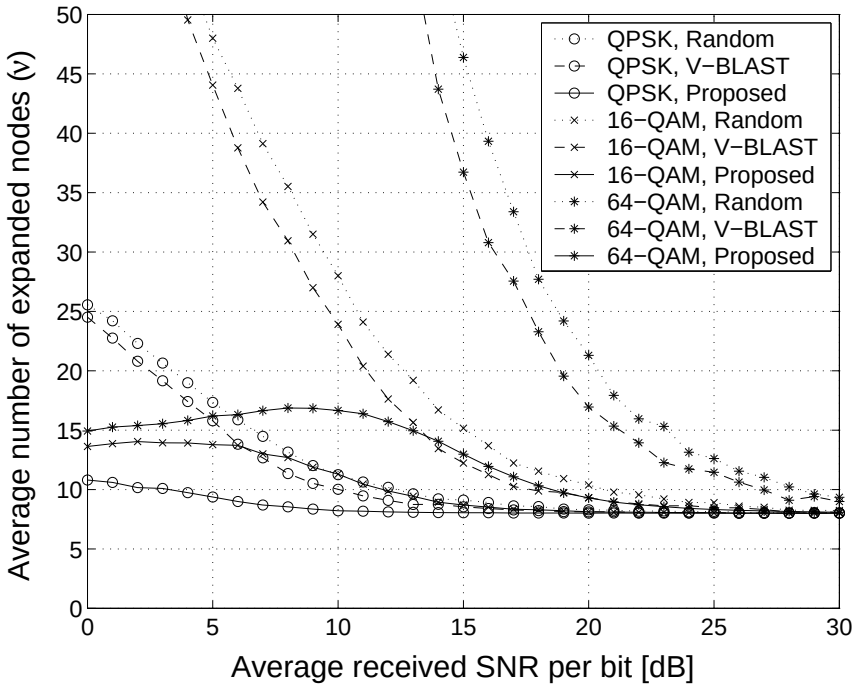
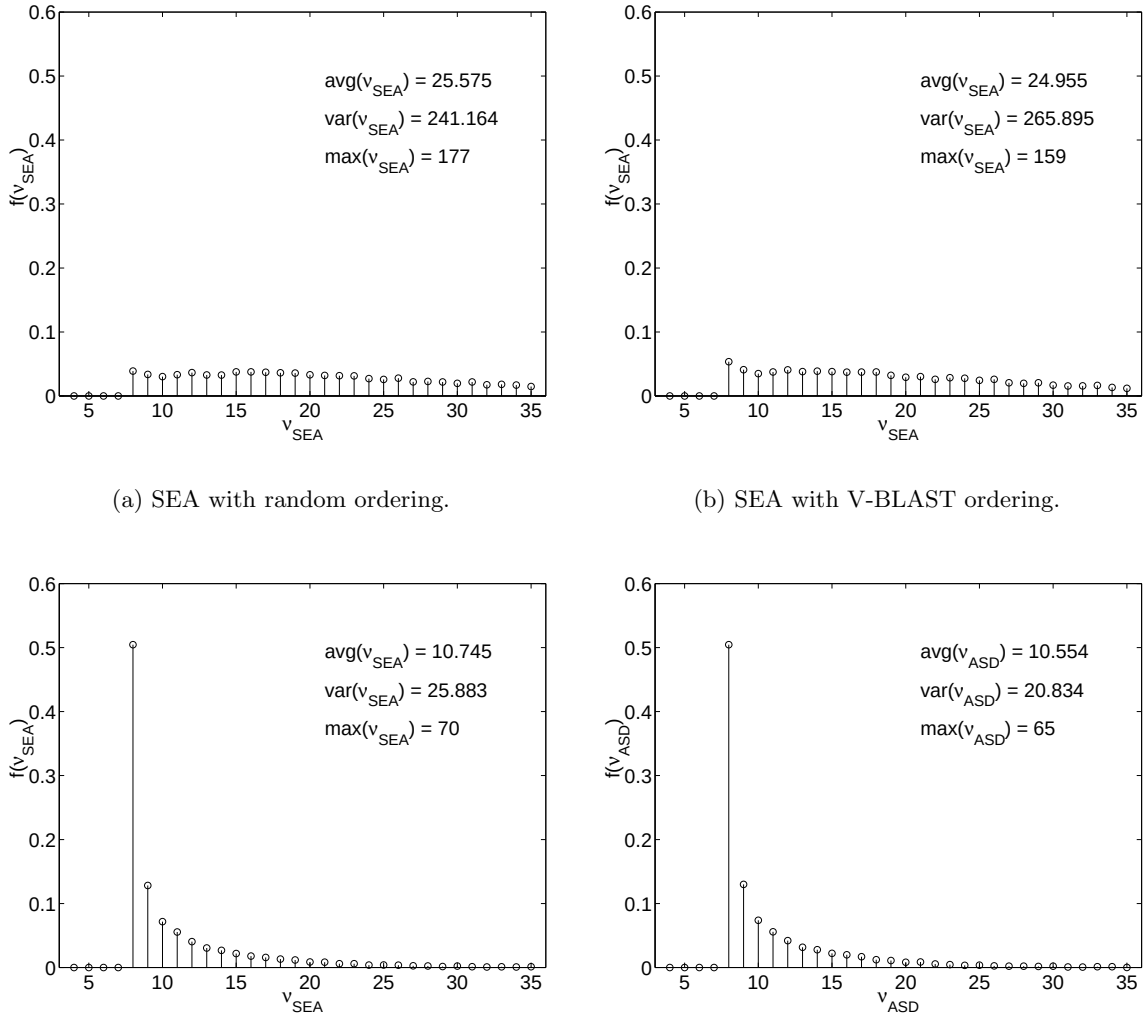


Figure 4.4: Average number of expanded nodes by the SEA decoder (ν_{SEA}) vs. average received SNR per bit under the random, V-BLAST and proposed orderings over a 4:4 MIMO flat fading channel using QPSK, 16-QAM and 64-QAM modulations.

of 0dB, i.e., under extremely poor channel conditions. We see that not only is the average number of expanded nodes dramatically reduced by the enhaned ordering, but the variance and maximum values of ν are likewise greatly reduced. These performance metrics are even further improved when the pre-processing stage is followed by the ASD.

Thus we have shown that the proposed ordering scheme is an extremely and consistently effective pre-processing stage that can be readily combined with existing sphere decoders. The time complexity of Algorithm 2 is $\mathcal{O}(M^3)$, roughly comparable to a matrix inversion and a few QR factorizations. In practice, we have observed immensely reduced decoding times at low SNRs using unoptimized implementations of the proposed ordering. As before, we see in Fig. 4.4 that as the SNR exceeds a certain modulation-dependent level, the number of nodes expanded by all sphere decoders, even under random ordering, converges to the lower bound of $2M$. Thus at such SNRs it may not be efficient to apply any pre-processing.



(a) SEA with random ordering.

(b) SEA with V-BLAST ordering.

(c) SEA with proposed ordering.

(d) ASD with proposed ordering.

Figure 4.5: Empirical estimates of the probability density functions $f_{\nu_{SEA}}(\nu_{SEA})$ and $f_{\nu_{ASD}}(\nu_{ASD})$ under the random, V-BLAST and proposed orderings over a 4:4 MIMO flat fading channel at an SNR of 0dB using QPSK modulation. The average value of ν, as well as its variance and maximum value over 10,000 channel realizations, are also indicated on each figure.

Chapter 5

Conclusions

In this report we discuss two contributions that we have put forward to the wireless communications research community on the subject of efficient Maximum Likelihood (ML) detection for communication over linear Multiple Input Multiple Output (MIMO) channels. Our solutions are both effective in addressing critical weaknesses of current proposals and timely, as the sphere decoder vies for a central role in current 4G standardization efforts.

The new Automatic Sphere Decoder (ASD) is capable of ML detection at a competitive complexity without the need for specifying a radius parameter. The removal of this parameter leads to reduced pre-processing times, since it is no longer necessary to compute an initial search radius before invoking the decoder. We also show that the philosophy behind the ASD means that it expands fewer nodes, i.e., has a better computational efficiency, than all known sphere decoders, over all SNRs and modulation orders.

Next, applying the theoretical framework developed in the design of the ASD, we show that the order of detection of the transmitted symbols can have an extremely significant impact on the computational efficiency of sphere decoding algorithms. Therefore, we propose a heuristic ordering rule designed to maximize sphere decoding efficiency. Simulations demonstrate that our enhanced ordering not only offers an immense reduction in the computational cost of sphere decoding, but it is also compatible with existing decoders.

Of particular note is its effectiveness in the low SNR regime, which has traditionally been a prohibitively expensive one for sphere decoders. Another strength of the new ordering is that it is of increasing influence at higher spectral efficiencies, thus enabling sphere decoders to achieve ML detection at a competitive computational cost over a wider range of operating parameters.

Because they facilitate communication over MIMO channels at low SNRs and at higher spectral efficiencies, offering great computational improvements over current proposals, we believe that these contributions and their underlying ideas will play a key role in the design of future wireless systems.

Bibliography

[1] Erik Agrell, Eriksson Thomas, Alexander Vardy, and Kenneth Zeger. Closest point search in lattices. *IEEE Transactions on Information Theory*, 48(8):2201–2214, August 2002.

[2] Siavash M. Alamouti. A simple transmit diversity technique for wireless communications. *IEEE Journal on Selected Areas in Communications*, 16(8):1451–1458, October 1998.

[3] Helmut Bölcskei, David Gesbert, and Arogyaswami J. Paulraj. On the capacity of OFDM-based spatial multiplexing systems. *IEEE Transactions on Communications*, 50(2):225–234, February 2002.

[4] Béla Bollobás. *Graph Theory*. Springer-Verlag, 1979.

[5] Andreas Burg, Moritz Borgmann, Claude Simon, Markus Wenk, Martin Zellweger, and Wolfgang Fichtner. Performance tradeoffs in the VLSI implementation of the sphere decoding algorithm. In *IEE 3G Mobile Communication Technologies Conference*, October 2004.

[6] Albert M. Chan and Inkyu Lee. A new reduced-complexity sphere decoder for multiple antenna systems. In *IEEE International Conference on Communications*, volume 1, pages 460–464, April 2002.

[7] Mohamed Oussama Damen, Hesham El Gamal, and Giuseppe Caire. On maximum-likelihood detection and the search for the closest lattice point. *IEEE Transactions on Information Theory*, 49(10):2389–2402, October 2003.

[8] Gerard J. Foschini. Layered space-time architecture for wireless communication in a fading environment when using multiple antennas. *Bell Labs Technical Journal*, 1(2):41–59, September 1996.

[9] Babak Hassibi and Haris Vikalo. On the sphere decoding algorithm: Part I, The expected complexity. *To appear in IEEE Transactions on Signal Processing*, 2004.

[10] Wai Ho Mow. Universal lattice decoding: Principle and recent advances. *Wireless Communications and Mobile Computing*, 3(5):553–569, August 2003.

[11] Theodore S. Rappaport, A. Annamalai, R. M. Buehrer, and William H. Tranter. Wireless communications: Past events and a future perspective. *IEEE Communications Magazine*, 40(5):148–161, May 2002.

[12] R. Tyrrell Rockafellar. *Convex analysis*. Princeton University Press, 1970.

[13] Robert Sedgewick. *Algorithms*. Addison-Wesley, 1988.

[14] Gilbert Strang. *Linear algebra and its applications*. Harcourt, 1988.

[15] Karen Su and Ian J. Wassell. An enhanced ordering for efficient sphere decoding. In *IEEE Int'l Conf. on Communications (to appear)*, May 2005.

[16] Vahid Tarokh, Nambi Seshadri, and A. Robert Calderbank. Space-time codes for high data rate wireless communication: Performance criterion and code construction. *IEEE Transactions on Information Theory*, 44(2):744–765, March 1998.

[17] I. Emre Telatar. Capacity of multi-antenna Gaussian channels. Technical report, October 1995.

[18] Emanuele Viterbo and Joseph Boutros. A universal lattice code decoder for fading channels. *IEEE Transactions on Information Theory*, 45(5):1639–1642, July 1999.

[19] Peter W. Wolniansky, Gerard J. Foschini, Glen D. Golden, and Reinaldo A. Valenzuela. V-BLAST: An architecture for realizing very high data rates over the rich-scattering wireless channel. In *International Symposium on Signals, Systems, and Electronics*, pages 295–300, September 1998.

Appendix A

The geometry of sphere decoding

Chapter 2 considers sphere decoding from an algebraic perspective. By applying the QR factorization and decomposing the objective function into the summations of (2.11), we arrive at the underlying tree structure depicted in Fig. 2.2. This appendix presents a geometric interpretation of sphere decoding, leading to an equivalent tree structure that likewise facilitates efficient decoding. First we review some important geometric notions.[1]

Definition A.1 *Given matrix* $\mathbf{H}$ *of full rank* M*, let the* linear subspace *spanned by the columns of* $\mathbf{H}$ *be defined as*

$$\mathcal{S}(\mathbf{H}) \triangleq \left\{ \mathbf{z} \,\middle|\, \mathbf{z} = \mathbf{H}\mathbf{a},\, \mathbf{a} \in \mathbb{R}^M \right\}. \tag{A.2}$$

The ML detection problem involves working within this space to find the point in the search set that is closest to a target $\mathbf{v}$.

Definition A.3 *Given matrix* $\mathbf{H}$ *of full rank* M *and alphabet* $\mathcal{X}$ *of size* B*, let the* finite lattice *of points in the search set be defined as*

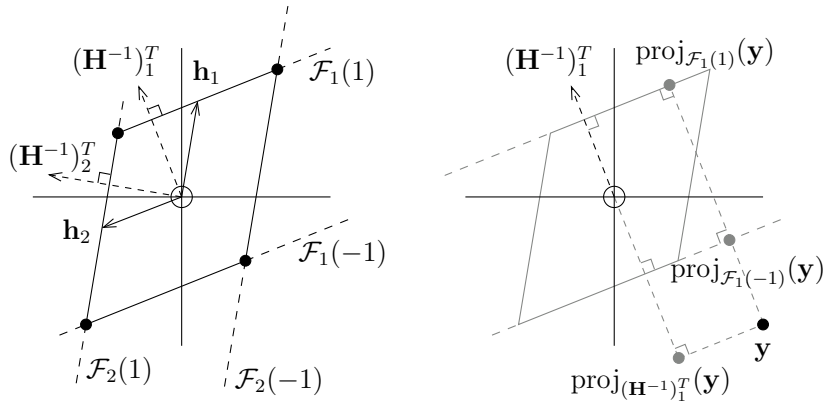
$$\mathcal{L}(\mathbf{H}) \triangleq \left\{ \mathbf{z} \,\middle|\, \mathbf{z} = \mathbf{H}\mathbf{s},\, \mathbf{s} \in \mathcal{X}^M \right\}. \tag{A.4}$$

There are B^M lattice points in $\mathcal{L}(\mathbf{H})$, as shown in Fig. A.1(a) for the case where $B = M = 2$. It can be partitioned in M ways into B sets of B^{M-1} lattice points, where each set of points is embedded in one of B parallel *affine sets* forming a collection. For instance, given $i \in \mathcal{I}$, the i^{th} collection $\{\mathcal{F}_i(s_i) \mid s_i \in \mathcal{X}\}$ contains B affine sets defined as

$$\mathcal{F}_i(s_i) \triangleq \left\{ \mathbf{z} \,\middle|\, \left\langle \mathbf{z} - \mathbf{h}_i s_i, (\mathbf{H}^{-1})_i^T \right\rangle = 0 \right\}, \tag{A.5}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product.

[1] For more details on linear algebraic structures and their geometry, please see [14].



(a) A finite lattice with $M = 2$ and $\mathcal{X} = \{-1, 1\}$.

(b) Projections of a point $\mathbf{y}$ onto the affine sets in collection $i = 1$.

Figure A.1: Some geometric entities useful in the analysis of sphere decoding.

We note that an affine set is generally defined as a set $\mathcal{F} = \mathcal{S} + \mathbf{a}$ for some linear subspace $\mathcal{S}$ and offset $\mathbf{a}$ [12, Sec. 1]. Thus an equivalent definition of the affine sets in colletion i is

$$\mathcal{F}_i(s_i) \triangleq \mathcal{S}(\mathbf{H}_{\setminus i}) + \mathbf{h}_i s_i. \tag{A.6}$$

Equation (A.5) defines the affine sets in terms of their shared *null space*, which is spanned by normal vector $(\mathbf{H}^{-1})_i^T$, whereas (A.6) is concerned instead with their *column space*, which is also shared and is spanned by the columns of $\mathbf{H}_{\setminus i}$. The *orthogonal projection* of a vector $\mathbf{y}$ onto affine set $\mathcal{F}_i(s_i)$ can then be defined as

$$\text{proj}_{\mathcal{F}_i(s_i)}(\mathbf{y}) \triangleq \mathbf{y} - \frac{\left\langle \mathbf{y} - \mathbf{h}_i s_i, (\mathbf{H}^{-1})_i^T \right\rangle}{\left|(\mathbf{H}^{-1})_i^T\right|^2} (\mathbf{H}^{-1})_i^T, \tag{A.7}$$

and the corresponding *orthogonal distance* as

$$d(\mathbf{y}, \mathcal{F}_i(s_i)) \triangleq \left| \mathbf{y} - \text{proj}_{\mathcal{F}_i(s_i)}(\mathbf{y}) \right|. \tag{A.8}$$

It should be clear that $\text{proj}_{\mathcal{F}_i(s_i)}(\mathbf{y})$ is the point in the affine set that is closest in Euclidean distance to $\mathbf{y}$, as depicted in Fig. A.1(b). Algebraically, $\mathcal{F}_i(s_i)$ contains the feasible set of lattice points satisfying constraint $\underline{s}_i = s_i$, which we define as

$$\mathcal{L}_i(s_i) \triangleq \mathcal{L}(\mathbf{H}_{\setminus i}) + \mathbf{h}_i s_i, \tag{A.9}$$

where we refer to $\mathbf{h}_i$ and s_i as the *offset vector* and *offset coefficient*, respectively, of $\mathcal{L}_i(s_i)$ and $\mathcal{F}_i(s_i)$. In particular, observe that $\mathcal{L}_i(s_i) - \mathbf{h}_i s_i$ is nothing more than a finite lattice having one less dimension than the original lattice $\mathcal{L}(\mathbf{H})$. The objective function $|\mathbf{v} - \mathbf{H}\mathbf{s}|^2$

can then be decomposed as follows:

$$|\mathbf{v}-\mathbf{H}\mathbf{s}|^2 = |\mathbf{v}-\mathbf{z}|^2,\ \mathbf{z}\in\mathcal{L}(\mathbf{H}) \tag{A.10}$$

$$= d^2\left(\mathbf{v},\mathcal{F}_{i_1}\left(s_{i_1}\right)\right) + \left|\mathrm{proj}_{\mathcal{F}_{i_1}(s_{i_1})}(\mathbf{v}) - \mathbf{z}'\right|^2,\ \mathbf{z}'\in\mathcal{L}_{i_1}\left(s_{i_1}\right). \tag{A.11}$$

Before completing this recursive decomposition, it will be helpful to observe some similarities and differences between the geometric step in (A.11) and the algebraic one in (2.8). There is clearly a strong structural resemblance between these two expressions. As before, the first term of (2.8) is associated with a constraint, $\underline{s}_{i_1} = s_{i_1}$, which is applied in the geometric framework by projecting the target $\mathbf{v}$ onto affine set $\mathcal{F}_{i_1}\left(s_{i_1}\right)$. It is involved in computing the values of the objective function for all B^{M-1} points where $\underline{s}_{i_1} = s_{i_1}$. We emphasize that these points are all contained in the affine set.

Because the geometric interpretation is not restricted by the QR factorization, we may choose any collection $i_1 \in \mathcal{I}$ onto which to project the target in the first step, i.e., we may choose to constrain any of the M optimization variables. In contrast, the QR-based algebraic decomposition (2.8) must extract the M^{th} variable s_M in the first step.

Geometrically, (A.11) expresses the objective function as the sum of the squared orthogonal distance to an affine set, i.e., from the target $\mathbf{v}$ to a projection $\mathrm{proj}_{\mathcal{F}_{i_1}(s_{i_1})}(\mathbf{v})$, and a squared distance from this projection to a point in the search set satisfying $\underline{s}_{i_1} = s_{i_1}$. Since all such points are in the same affine set as the projection, it should be clear that subsequent steps of the recursion take place within that affine set $\mathcal{F}_{i_1}\left(s_{i_1}\right)$.

To capture this notion in our recursive expressions, we extend the definitions of the affine sets of interest to cases where $L > 1$ constraints are satified:

$$\mathcal{F}_{i_L,\ldots,i_1}\left(\mathbf{s}_{i_L,\ldots,i_1}\right) \triangleq \mathcal{S}\left(\mathbf{H}_{\setminus i_1,\ldots,i_L}\right) + \mathbf{H}_{i_L,\ldots,i_1}\mathbf{s}_{i_L,\ldots,i_1}. \tag{A.12}$$

We can then complete the recursive decomposition as follows:

$$|\mathbf{v}-\mathbf{H}\mathbf{s}|^2 = d^2\left(\mathbf{v},\mathcal{F}_{i_1}\left(s_{i_1}\right)\right) + \left|\mathbf{w}_{i_1}\left(s_{i_1}\right) - \mathbf{z}'\right|^2,\ \mathbf{z}'\in\mathcal{L}_{i_1}\left(s_{i_1}\right) \tag{A.13}$$

$$= \sum_{L=1}^{M} d^2\left(\mathbf{w}_{i_{L-1},\ldots,i_1}\left(\mathbf{s}_{i_{L-1},\ldots,i_1}\right), \mathcal{F}_{i_L,\ldots,i_1}\left(s_{i_L,\ldots,i_1}\right)\right), \tag{A.14}$$

where, making the logical interpretations $\{i_0, i_1\}$, $\{\mathbf{s}_{i_0,i_1}\} = \emptyset$, the *affine residual targets* are defined as

$$\mathbf{w}_{i_L,\ldots,i_1}\left(\mathbf{s}_{i_L,\ldots,i_1}\right) \triangleq \begin{cases} \mathbf{w}_{\emptyset}(\emptyset) = \mathbf{v}, & L = 0 \\ \mathrm{proj}_{\mathcal{F}_{i_L,\ldots,i_1}(\mathbf{s}_{i_L,\ldots,i_1})}\left(\mathbf{w}_{i_{L-1},\ldots,i_1}\left(\mathbf{s}_{i_{L-1},\ldots,i_1}\right)\right), & L = 1,\ldots,M. \end{cases} \tag{A.15}$$

We can see immediately that the summations of (A.14) have exactly the same recursive structure as those of (2.10). Therefore this geometrically-oriented decomposition also enables us to perform sphere decoding by exploring nodes of a B-ary tree with $M+1$ levels, as depicted in Fig. 2.2 and studied in Section 2.2. In addition, L gives the level of the residuals defined according to (A.15).

As expected from their name, the affine residual targets $\mathbf{w}$ are closely related to the orthogonally transformed (linear) residuals $\widetilde{\mathbf{y}}$ previously defined in Section 2.2. Because they are projections onto affine sets as given by (A.12) it should be clear that each level L residual resides in a space of dimension $D = M - L$. In fact, if the variables are constrained in the same order as that enforced by the QR factorization, we can write the following explicit relationship between these entities via the untransformed linear residual targets $\mathbf{y}$:

Proposition A.16 *Given target* $\mathbf{v}$*, channel matrix* $\mathbf{H}$*, and alphabet* $\mathcal{X}$*, let* $D = M - L$*. Then the* linear residual targets *can be expressed as*

$$\mathbf{y}\left(\mathbf{s}_{D+1}^{M}\right) = \mathbf{w}_{D+1,\ldots,M}\left(\mathbf{s}_{D+1}^{M}\right) - \mathbf{H}_{D+1}^{M}\mathbf{s}_{D+1}^{M}, \text{ or} \tag{A.17}$$

$$\mathbf{Q}^{T}\mathbf{y}\left(\mathbf{s}_{D+1}^{M}\right) = \begin{bmatrix} \widetilde{\mathbf{y}}\left(\mathbf{s}_{D+1}^{M}\right) \\ \mathbf{0} \end{bmatrix}, \tag{A.18}$$

where $\mathbf{Q}$ *is obtained from the QR factorization of* $\mathbf{H}$ *and the orthogonally transformed residual targets are defined as previously in* (2.11)*.*

Proof In the trivial case where $L = 0$, $\{M+1, M\}$, $\{\mathbf{s}_{M+1}^{M}\} = \emptyset$, and $\mathbf{H}_{M+1}^{M}\mathbf{s}_{M+1}^{M} = \mathbf{0}$. Therefore $\mathbf{y}(\emptyset) = \mathbf{w}_{\emptyset}(\emptyset) - \mathbf{0} = \mathbf{v}$ and $\mathbf{Q}^{T}\mathbf{y}(\emptyset) = \mathbf{Q}^{T}\mathbf{v} = \widetilde{\mathbf{v}} = \widetilde{\mathbf{y}}(\emptyset)$.

In the more general case, we consider first (A.17): The affine residual is defined in (A.15) as a projection onto affine set $\mathcal{F}_{D,\ldots,M}\left(\mathbf{s}_{D}^{M}\right)$, which has underlying linear subspace $\mathcal{S}(\mathbf{H}_{1}^{D-1})$ and offset $\mathbf{H}_{D}^{M}\mathbf{s}_{D}^{M}$. Therefore, to compute the projection of $\mathbf{w}_{D+1,\ldots,M}\left(\mathbf{s}_{D+1}^{M}\right)$ onto this set, we first remove the offset and then subtract away any remaining components that lie in the null space of $\mathcal{S}(\mathbf{H}_{1}^{D-1})$. From the QR factorization, we have that the null space of $\mathcal{S}(\mathbf{H}_{1}^{D-1})$ is spanned by the columns of $\mathbf{Q}_{D}^{M}$ and so the orthogonal projection can be written as

$$\mathbf{w}_{D,\ldots,M}\left(\mathbf{s}_{D}^{M}\right) = \mathbf{w}_{D+1,\ldots,M}\left(\mathbf{s}_{D+1}^{M}\right) - \mathbf{Q}_{D}^{M}\left(\mathbf{Q}_{D}^{M}\right)^{T}\left(\mathbf{w}_{D+1,\ldots,M}\left(\mathbf{s}_{D+1}^{M}\right) - \mathbf{H}_{D}^{M}\mathbf{s}_{D}^{M}\right) \tag{A.19}$$

$$= \mathbf{w}_{D+1,\ldots,M}\left(\mathbf{s}_{D+1}^{M}\right) - \mathbf{Q}_{D}^{M}\left(\mathbf{Q}_{D}^{M}\right)^{T}\left(\mathbf{y}\left(\mathbf{s}_{D+1}^{M}\right) - \mathbf{h}_{D}s_{D}\right). \tag{A.20}$$

Then removing the offset and applying orthogonal transform $\mathbf{Q}^T$ to (A.19) and (A.20) we obtain the following:

$$\mathbf{Q}^T\mathbf{y}\left(\mathbf{s}_D^M\right) = \left[\mathbf{Q}^T - \mathbf{Q}^T\mathbf{Q}_D^M\left(\mathbf{Q}_D^M\right)^T\right]\left(\mathbf{w}_{D+1,\ldots,M}\left(\mathbf{s}_{D+1}^M\right) - \mathbf{H}_D^M\mathbf{s}_D^M\right) \tag{A.21}$$

$$= \begin{bmatrix}\left(\mathbf{Q}_1^{D-1}\right)^T \\ \mathbf{0}\end{bmatrix}\left(\mathbf{y}\left(\mathbf{s}_{D+1}^M\right) - \mathbf{h}_D s_D\right). \tag{A.22}$$

The proof then proceeds by induction on L: For the base case where $L = 0$, $D = M$ and (A.22) becomes

$$\mathbf{Q}^T\mathbf{y}(s_M) = \begin{bmatrix}\left(\mathbf{Q}_1^{M-1}\right)^T \\ \mathbf{0}\end{bmatrix}(\mathbf{v} - \mathbf{h}_M s_M) \tag{A.23}$$

$$= \begin{bmatrix}(\widetilde{\mathbf{v}} - \mathbf{r}_M s_M)_{\setminus M} \\ \mathbf{0}\end{bmatrix} \tag{A.24}$$

$$= \begin{bmatrix}\widetilde{\mathbf{y}}(s_M) \\ \mathbf{0}\end{bmatrix}. \tag{A.25}$$

Finally, making the assumptions that $\mathbf{y}\left(\mathbf{s}_{D+1}^M\right) = \mathbf{w}_{D+1,\ldots,M}\left(\mathbf{s}_{D+1}^M\right) - \mathbf{H}_{D+1}^M\mathbf{s}_{D+1}^M$ and that $\mathbf{Q}^T\mathbf{y}(\mathbf{s}_{D+1}^M) = \begin{bmatrix}\widetilde{\mathbf{y}}(\mathbf{s}_{D+1}^M) \\ \mathbf{0}\end{bmatrix}$, the inductive step can be shown in a similar manner:

$$\mathbf{Q}^T\mathbf{y}\left(\mathbf{s}_D^M\right) = \begin{bmatrix}\left(\mathbf{Q}_1^{D-1}\right)^T \\ \mathbf{0}\end{bmatrix}\left(\mathbf{y}\left(\mathbf{s}_{D+1}^M\right) - \mathbf{h}_D s_D\right) \tag{A.26}$$

$$= \begin{bmatrix}\left(\widetilde{\mathbf{y}}\left(\mathbf{s}_{D+1}^M\right) - \mathbf{r}_D s_D\right)_{\setminus D} \\ \mathbf{0}\end{bmatrix} \tag{A.27}$$

$$= \begin{bmatrix}\widetilde{\mathbf{y}}(\mathbf{s}_D^M) \\ \mathbf{0}\end{bmatrix}. \tag{A.28}$$

■

Proposition A.16 confirms that the orthogonally transformed linear residual targets (2.11) are related to the affine residuals (A.15) by the appropriate affine offset and the orthogonal matrix $\mathbf{Q}$ from the QR factorization. Therefore, if the optimization variables are constrained in the same order as that enforced by the QR factorization, the summations in (A.14) and (2.10) become identical.

Corollary A.29 *Given target* $\mathbf{v}$*, channel matrix* $\mathbf{H}$*, and alphabet* $\mathcal{X}$*, let* $D = M - L$*. Then the terms in summations* (2.10) *and* (A.14) *are identical, i.e.,*

$$d\left(\widetilde{y}\left(\mathbf{s}_{D+1}^M\right)_D, r_{DD}s_D\right) = d\left(\mathbf{w}_{D+1,\ldots,M}\left(\mathbf{s}_{D+1}^M\right), \mathcal{F}_{D,\ldots,M}\left(\mathbf{s}_D^M\right)\right) \tag{A.30}$$

Proof Combining (A.15) and (A.20) with Proposition A.16 gives

$$d\left(\mathbf{w}_{D+1,\ldots,M}\left(\mathbf{s}_{D+1}^{M}\right),\mathcal{F}_{D,\ldots,M}\left(\mathbf{s}_{D}^{M}\right)\right) = d\left(\mathbf{w}_{D+1,\ldots,M}\left(\mathbf{s}_{D+1}^{M}\right),\mathbf{w}_{D,\ldots,M}\left(\mathbf{s}_{D}^{M}\right)\right) \tag{A.31}$$

$$= d\left(\mathbf{Q}_{D}^{M}\left(\mathbf{Q}_{D}^{M}\right)^{T}\mathbf{y}\left(\mathbf{s}_{D+1}^{M}\right),\mathbf{Q}_{D}^{M}\left(\mathbf{Q}_{D}^{M}\right)^{T}\mathbf{h}_{D}s_{D}\right) \tag{A.32}$$

$$= d\left(\begin{bmatrix}\mathbf{0}\\ \left(\mathbf{Q}_{D}^{M}\right)^{T}\end{bmatrix}\mathbf{y}\left(\mathbf{s}_{D+1}^{M}\right),\begin{bmatrix}\mathbf{0}\\ \left(\mathbf{Q}_{D}^{M}\right)^{T}\end{bmatrix}\mathbf{h}_{D}s_{D}\right) \tag{A.33}$$

$$= d\left(\widetilde{\mathbf{y}}\left(\mathbf{s}_{D+1}^{M}\right)_{D},\left(\mathbf{r}_{D}s_{D}\right)_{D}\right). \tag{A.34}$$

■

Fig. A.2 provides a graphical illustration of the entities discussed in this Appendix. The left hand side considers the decomposition (2.9) based on the QR factorization of a channel matrix $\mathbf{H} = [\,\mathbf{h}_1\ \mathbf{h}_2\,]$ ($M = 2$). It also shows the transformed residual targets computed when performing the first step of the recursion given a target $\mathbf{v}$. The right hand side shows the relationship between these residuals and those computed in the geometric decomposition (A.11) with $i_1 = M$. This link enables us to study the problem from a geometric perspective, as in Chapter 4, while still performing computations using the efficient upper triangular structure of $\mathbf{R}$.

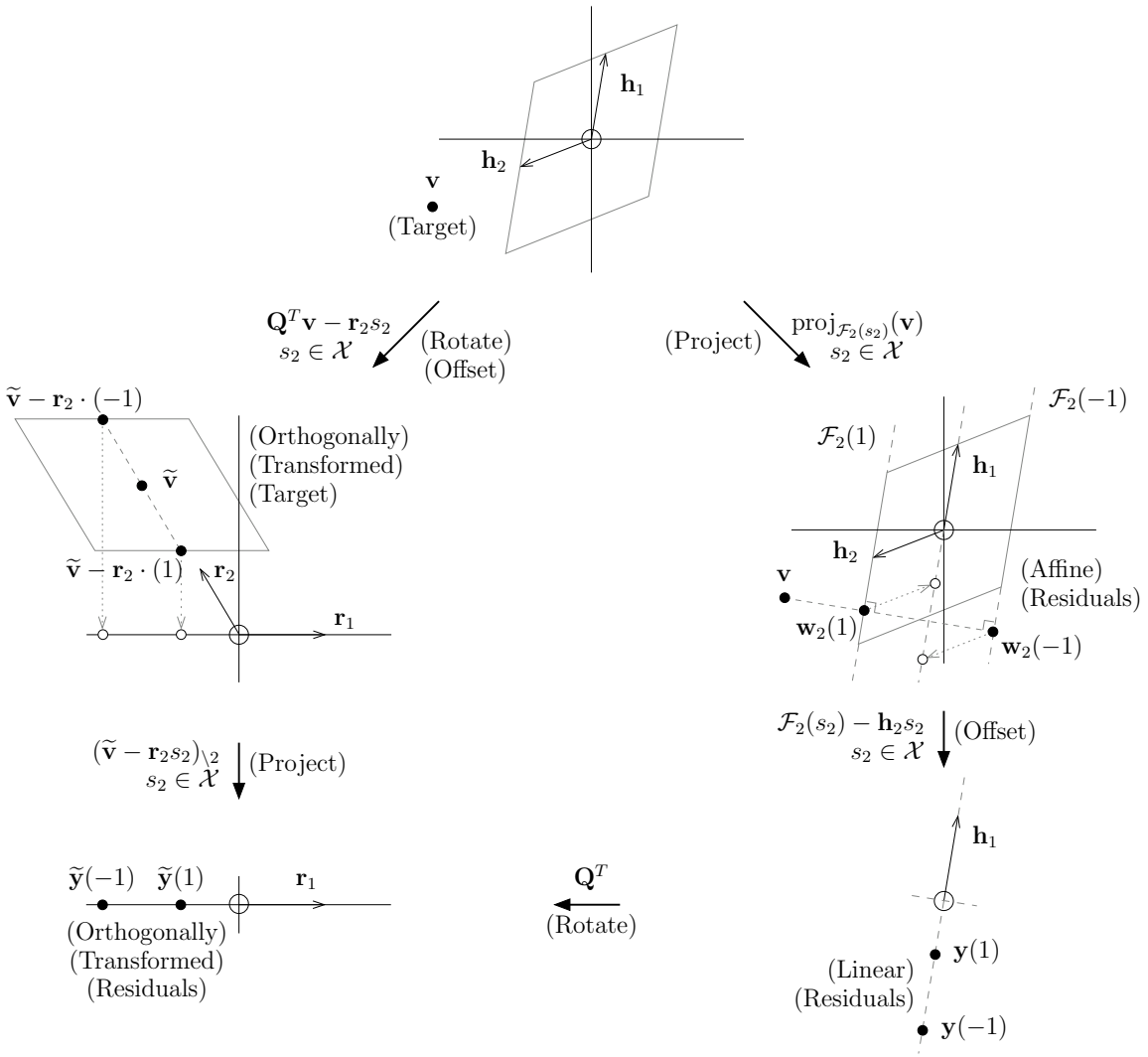


Figure A.2: An illustration of the relationship between some of the algebraic and geometric entities involved in sphere decoding for the case where $M = 2$ and $\mathcal{X} = \{-1, 1\}$. Observe that the problem dimension, i.e., the dimension of space in which the residual targets reside, is reduced by one at each step of the recursion.

Appendix B

Proof of the optimality of the automatic sphere decoder

The Automatic Sphere Decoder (ASD) makes use of the tree structure induced by the QR factorization to find an ML solution without the need to specify a radius parameter. A sample tree is shown in Fig. B.1 for the case where $M = B = 2$. The ASD efficiently explores the tree in search of the smallest weight leaf node. It starts from the root and expands the smallest number of nodes necessary to obtain a solution while guaranteeing its optimality. To prove that it returns an ML solution, we begin by recalling and formalizing some properties of this tree that were previously introduced in Section 2.2 and Chapter 3.

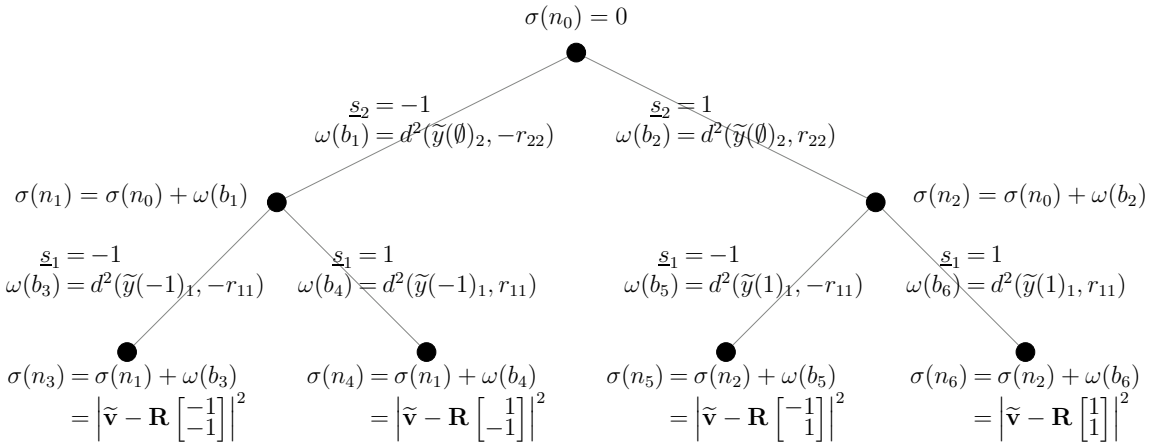


Figure B.1: A weighted B-ary tree for sphere decoding with $M = 2$ and $\mathcal{X} = \{-1, 1\}$.

Using standard graph theoretic notation, we define the *tree* $\mathcal{T}(\mathbf{v}, \mathbf{H}, \mathcal{X})$ as the ordered pair $(\mathcal{N}, \mathcal{B})$. Then given two distinct nodes n_i, $n_j \in \mathcal{N}$ there is a unique *path* from n_i to n_j, denoted $\mathcal{P}(n_i, n_j)$.[1] Observe that $\mathcal{P}(n_i, n_j)$ is nothing more than a connected subgraph of

[1]Recall that a path is a *walk* over distinct nodes, i.e., an alternating sequence of nodes and branches $n_i \ldots n_j$ in which no node appears more than once. See [4] for more details.

$\mathcal{T}$, which is in turn defined by the ordered pair $\left(\mathcal{N}_{\mathcal{P}(n_i,n_j)}, \mathcal{B}_{\mathcal{P}(n_i,n_j)}\right)$ comprising the sets of nodes and branches along the path. Next, we choose one of the nodes of $\mathcal{T}$ to be the *root* n_0, and define a *rooted path* as a path having n_0 as one of its endnodes. Therefore for all $n \in \mathcal{N} \setminus n_0$, there exists a unique rooted path $\mathcal{P}(n_0, n)$.

The nodes of tree $\mathcal{T}(\mathbf{v}, \mathbf{H}, \mathcal{X})$ are distributed over $M+1$ levels numbered from the *leaf* nodes at level M to the root at level 0. We associate optimization variable $\underline{s}_D$ with all branches bridging levels L and $L+1$. Since $\underline{s}_D$ takes values $s_D \in \mathcal{X}$, B branches originate at all *non-leaf* nodes, each associated with a constraint of the form $\underline{s}_D = s_D$, as shown in Fig. B.1. Based on (2.10), the *branch weight* of a branch b bridging levels L and $L+1$ is then computed by applying the associated constraint:

$$\omega(b) = d^2(\widetilde{y}\left(\mathbf{s}_{D+1}^M\right)_D, r_{DD}s_D). \tag{B.1}$$

Therefore, by definition branch weights are non-negative. Note that the weight of a branch bridging levels L and $L+1$ also depends on the set of constraints $\underline{\mathbf{s}}_{D+1}^M - \mathbf{s}_{D+1}^M$ associated with its parent node at level L.

From the recursive form of (2.10) and the branch weights (B.1), the *node weight* of node n is computed as

$$\sigma(n) = \sum_{b \in \mathcal{B}_{\mathcal{P}(n_0,n)}} \omega(b), \tag{B.2}$$

i.e., the sum of the weights of the branches in its rooted path. Similarly, the set of constraints associated with n is given by those associated with the branches in its rooted path.

The node level L corresponds to the number of constrained variables; the dimension of the residual target vector is then $D = M - L$. In particular, at a leaf node n_l, all M optimization variables are constrained, i.e., $\mathbf{s}(n_l) \in \mathcal{X}^M$, and $\sigma(n_l)$ represents one of the summations in in (2.10), i.e., the value of the objective function evaluated at $\mathbf{s}(n_l)$.

Although the structure of $\mathcal{T}$ is fixed, initially, only the properties of the root node are known. The proposed decoder maintains a list of nodes $\mathcal{N}_b$ defining the border between the explored and unexplored parts of the tree. At the start of decoding it contains only the root node n_0. In each iteration, it selects and expands the border node with the smallest weight. Recall that in the case of a tie, the border node having the smallest weight and the lowest level is selected for expansion. The expanded node is then deleted from $\mathcal{N}_b$, since it is no longer on the border, and replaced by its B children. When the smallest weight border node is a leaf, the decoder returns its associated constraint value vector $\mathbf{s}$, its weight σ, and then terminates. We now show that $\mathbf{s}$ is an ML solution to minimization problem (2.4).

In the derivations to follow, let $\mathcal{N}_e(k)$, $\mathcal{N}_u(k)$ and $\mathcal{N}_b(k)$ denote the lists of expanded, unexpanded and border nodes, respectively, at the beginning of iteration k of Algorithm 1. It should be clear from their descriptions that $\mathcal{N}_u(k) = \mathcal{N} \setminus \mathcal{N}_e(k)$ and $\mathcal{N}_e(k) \cap \mathcal{N}_b(k) = \emptyset$.

Lemma B.3 *Given $\mathcal{N}_e(k)$, $\mathcal{N}_b(k)$, and $n \in \mathcal{N}_u(k)$, there exists a node n' in rooted path $\mathcal{P}(n_0, n)$ such that $n' \in \mathcal{N}_b(k)$.*

Proof (By induction) When $k = 1$, $\mathcal{N}_e(1) = \emptyset$ and $\mathcal{N}_b(1) = \{n_0\}$. Since $n' = n_0 \in \mathcal{N}_{\mathcal{P}(n_0,n)}$ for all n, the condition is trivially met in the base case.

Next, suppose that for all $n \in \mathcal{N}_u(k)$ such a node $n' \in \mathcal{N}_b(k)$ exists at the beginning of iteration k. If n' is not expanded in iteration k, then $n' \in \mathcal{N}_b(k+1)$. Otherwise, if n' is expanded, then either $n \neq n'$, in which case one of the child nodes of n' is in path $\mathcal{P}(n_0, n)$, or $n = n'$. In the latter case $n' \notin \mathcal{N}_b(k+1)$ but also $n \notin \mathcal{N}_u(k+1)$. Therefore the condition still holds at the end of iteration k. Verification of the inductive step completes the proof. ∎

The purpose of Lemma B.3 is to show that $\mathcal{N}_b$ *encloses* $\mathcal{N}_e$, and so we can explore the larger set $\mathcal{N}_u$ by considering only nodes in the smaller border set $\mathcal{N}_b$.

Lemma B.4 *Given rooted path $\mathcal{P}(n_0, n)$, the weights of the nodes $n' \in \mathcal{N}_{\mathcal{P}(n_0,n)}$ satisfy $0 = \sigma(n_0) \leq \sigma(n') \leq \sigma(n)$.*

Proof The result follows from the non-negativity of branch weights (B.1) and the definition of node weights given by (B.2). ∎

Lemma B.4 establishes that node weights are monotonically non-decreasing along rooted paths, enabling us to show the following:

Lemma B.5 *Given $\mathcal{N}_e(k)$ and $\mathcal{N}_b(k)$, the next node selected by Algorithm 1, i.e., $n = \operatorname{argmin}_{n' \in \mathcal{N}_b(k)} \sigma(n')$, also satisfies $n = \operatorname{argmin}_{n' \in \mathcal{N}_u(k)} \sigma(n')$.*

Proof Let $n = \operatorname{argmin}_{n' \in \mathcal{N}_b(k)} \sigma(n')$ and suppose that there exists $n' \in \mathcal{N}_u(k) \setminus \mathcal{N}_b(k)$ such that $\sigma(n') < \sigma(n)$. Then by Lemmas B.3 and B.4, there also exists a node n'' such that $n'' \in \mathcal{N}_b(k)$, $\sigma(n'') \leq \sigma(n')$, and hence $\sigma(n'') < \min_{n' \in \mathcal{N}_b(k)} \sigma(n')$. This contradiction completes the proof. ∎

In other words, until its termination, Algorithm 1 expands the nodes of $\mathcal{N}$ in order of increasing weight. Next, we show the optimality of the solution returned by the automatic sphere decoder.

Theorem B.6 *Given target $\mathbf{v}$, QR factored channel matrix $\mathbf{H} = \mathbf{QR}$, and finite alphabet $\mathcal{X}$, Algorithm 1 returns an optimal solution $\mathbf{s}_* \in \mathcal{X}^M$ such that $|\widetilde{\mathbf{v}} - \mathbf{R}\mathbf{s}_*|^2 \leq |\widetilde{\mathbf{v}} - \mathbf{R}\mathbf{s}|^2$, and hence $|\mathbf{v} - \mathbf{H}\mathbf{s}_*|^2 \leq |\mathbf{v} - \mathbf{H}\mathbf{s}|^2$, $\forall\ \mathbf{s} \in \mathcal{X}^M$.*

Proof When Algorithm 1 terminates, $L = M$, $\mathbf{s} \in \mathcal{X}^M$, and $\sigma = |\widetilde{\mathbf{v}} - \mathbf{R}\mathbf{s}|^2$. Since no leaf nodes will have been expanded, $\mathcal{N}_l \subset \mathcal{N}_u$. Therefore by Lemma B.5, we have that $\sigma \leq \sigma(n)\ \forall\ n \in \mathcal{N}_l$. Recalling that the leaf node weights are equal to the values of the orthogonally transformed objective function evaluated at each in the search set, and that the objective function is invariant under such transformation, we obtain the desired result. ■

A few useful corollaries follow immediately:

Corollary B.7 *The weight σ returned by Algorithm 1 is the square of the* optimal search radius*, which is defined as*

$$C_* \triangleq \min_{\mathbf{s}\in\mathcal{X}^M} |\mathbf{v} - \mathbf{H}\mathbf{s}|\,. \tag{B.8}$$

Corollary B.9 *A node n is expanded by Algorithm 1 if and only if it satisfies $\sigma(n) \leq C_*^2$, or equivalently its associated affine residual target $\mathbf{w}_{D+1,\ldots,M}\left(\mathbf{s}_{D+1}^M(n)\right)$ is contained in the* (closed) optimal search sphere

$$\overline{\mathcal{S}}(\mathbf{v}, C_*) \triangleq \{\mathbf{z} : |\mathbf{z} - \mathbf{v}| \leq C_*\}\,. \tag{B.10}$$

Appendix C

Description of the experimental setup

In this appendix we describe the experimental setup used in our work. A block diagram is presented in Fig. C.1. It is parameterized by the size of the (real) transmission alphabet $B = |\mathcal{X}|$, the number of transmit antennas M, and the number of receive antennas N. Across the top of the figure are shown the four key quantities that are required to simulate each channel use: a sequence of data bits $\mathbf{b} \in \{0,1\}^{M \log_2 B^2}$, a power scaling factor $\alpha \in \mathbb{R}$, a complex channel matrix $\mathbf{H} \in \mathbb{C}^{N \times M}$ and a complex noise vector $\mathbf{n} \in \mathbb{C}^N$.

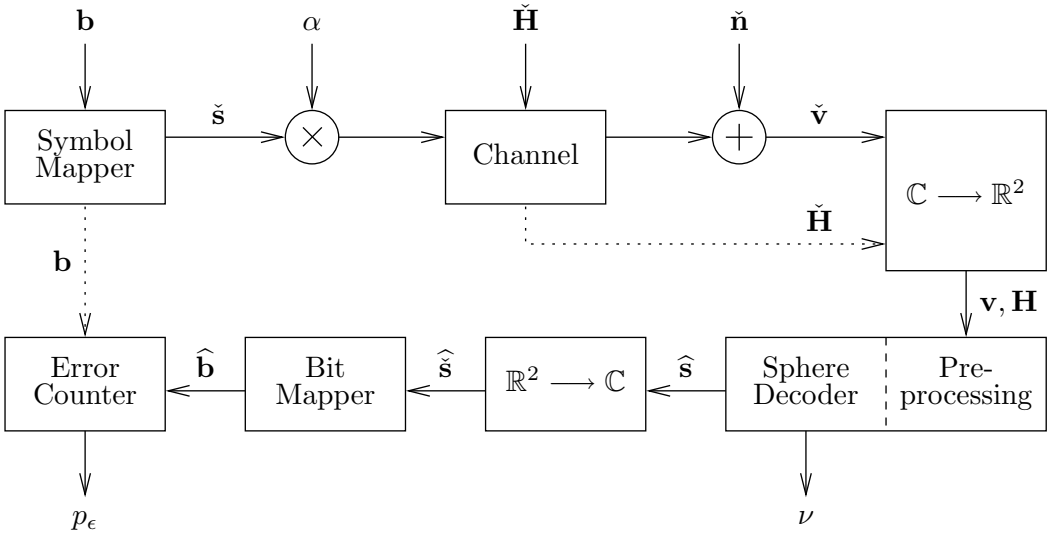
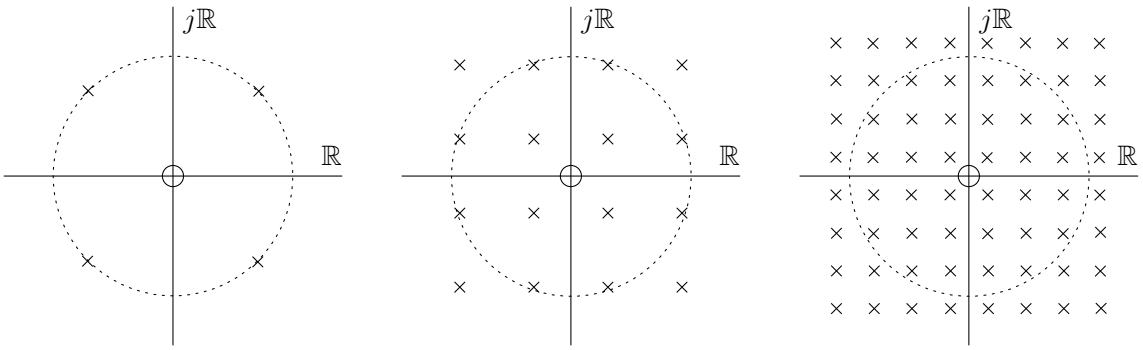


Figure C.1: Block diagram of experimental setup in complex baseband representation.

Each channel use begins by drawing $M \log_2 B^2$ bits from the uniform distribution. These are then taken $\log_2 B^2$ bits at a time and mapped to one of B^2 complex symbols $\check{\mathbf{s}} \in \{x_R + jx_I \,|\, x_R, x_I \in \mathcal{X}\}^M$ depending on the modulation scheme being simulated. Figs. C.2(a) to C.2(c) depict the symbol maps or constellations for QPSK ($B = 2$), 16-QAM ($B = 4$)

and 64-QAM ($B = 8$) modulation. The power of the constellations is normalized, with the dotted circle of unit radius in each of the figures below indicating its average power, i.e., before power scaling, $\mathsf{E}(|\check{s}_i|^2) = 1$ and $\mathsf{E}(|\check{\mathbf{s}}|^2) = M$. Therefore the symbol alphabet can be written as

$$\mathcal{X} = \sqrt{\frac{3}{2(B^2-1)}}\{-B+1, \ldots, -1, +1, \ldots, B-1\}. \tag{C.1}$$



(a) QPSK modulation. (b) 16-QAM modulation. (c) 64-QAM modulation.

Figure C.2: Symbol constellations for various complex modulation schemes of interest.

The purpose of the power scaling factor α is to give the desired Signal-to-Noise Ratio (SNR). The total energy transmitted per bit is

$$E_b = \frac{\mathsf{E}(|\alpha\check{\mathbf{s}}|^2)}{M \log_2 B^2} \tag{C.2}$$

$$= \frac{\alpha^2}{\log_2 B^2}. \tag{C.3}$$

Then to simulate a received SNR (in dB) of ρ, we set the scaling factor as follows:

$$\rho = 10 \log_{10} \frac{N E_b}{N_0} \tag{C.4}$$

$$E_b = N_0 \cdot 10^{\frac{\rho}{10}} \tag{C.5}$$

$$\alpha = \sqrt{\frac{N_0 \log_2 B^2 \cdot 10^{\frac{\rho}{10}}}{N}}, \tag{C.6}$$

where for numerical convenience, we may arbitrarily choose $N_0 = 1$. The factor of N arises because each receive antenna observes a transmitted energy per bit of E_b. Therefore the overall SNR seen across the receive array is given by (C.4).

The transmission channel is modelled as a complex Multiple Input Multiple Output (MIMO) linear system with complex Additive White Gaussian Noise (AWGN). As discussed in [7], a number of important channels can be simulated in this manner, including MIMO flat fading wireless channels, frequency selective fading channels, and multi-user channels. Of interest in our work is the multiple antenna Rayleigh spatially independent flat fading wireless channel, for which the coefficients of $\check{\mathbf{H}}$ are drawn independently from the circularly symmetric complex Gaussian distribution of zero mean and unit variance $\mathcal{CN}(0,1)$.

The $N \times M$ channel matrix represents the case where there are M transmit antennas and N receive antennas. In addition to the distortion caused by $\check{\mathbf{H}}$, additive noise is seen at the output of the channel, encapsulated by the complex random vector $\check{\mathbf{n}}$. The noise samples are independently drawn from the circularly symmetric complex Gaussian distribution $\mathcal{CN}(0, N_0)$, where recall that we assigned $N_0 = 1$ in computing the power scaling factor.

The vector of complex signals seen at the receiver is then given by

$$\check{\mathbf{v}} = \alpha\check{\mathbf{H}}\check{\mathbf{s}} + \check{\mathbf{n}}. \tag{C.7}$$

Observe from Fig. C.1 that we also make the common assumption of perfect channel state information at the receiver, i.e., that an uncorrupted copy of $\check{\mathbf{H}}$ is available at the detector.

Next, in order to perform (real) sphere decoding, we expand the complex signals into real ones as follows:

$$\mathbf{v} = \begin{bmatrix} \mathrm{Re}\{\check{\mathbf{v}}\} \\ \mathrm{Im}\{\check{\mathbf{v}}\} \end{bmatrix} \tag{C.8}$$

$$= \alpha \begin{bmatrix} \mathrm{Re}\{\check{\mathbf{H}}\} & -\mathrm{Im}\{\check{\mathbf{H}}\} \\ \mathrm{Im}\{\check{\mathbf{H}}\} & \mathrm{Re}\{\check{\mathbf{H}}\} \end{bmatrix} \begin{bmatrix} \mathrm{Re}\{\check{\mathbf{s}}\} \\ \mathrm{Im}\{\check{\mathbf{s}}\} \end{bmatrix} + \begin{bmatrix} \mathrm{Re}\{\check{\mathbf{n}}\} \\ \mathrm{Im}\{\check{\mathbf{n}}\} \end{bmatrix} \tag{C.9}$$

$$= \alpha\mathbf{H}\mathbf{s} + \mathbf{n}. \tag{C.10}$$

Received vector or target $\mathbf{v}$, channel matrix $\mathbf{H}$ and symbol alphabet $\mathcal{X}$ are then passed to the appropriate detector algorithms, depending on the test being conducted.

For the results presented in this report, we have been concerned primarily with averaging ν, the number of nodes expanded by a sphere decoder, over a large number of channel uses. These experiments were repeated at a wide range of SNRs, using the three symbol constellations shown in Fig. C.2, combined with a variety of sphere decoders and pre-processing schemes. Generally speaking, the number of channel uses must be large enough to ensure that the variance of the reported value is sufficiently low. Therefore we can feel confident that it provides a reliable representation of the average performance. To this end, thousands to hundreds of thousands of realizations were generated for each point on the final curves. Fig. C.1 also illustrates how the error rate performance p_ϵ can be assessed.

Appendix D

Proof of the optimality of the enhanced ordering when M=B=2

In this appendix we show the optimality of the proposed ordering with respect to the computational efficiency of a subsequent application of the Automatic Sphere Decoder (ASD) when $M = B = 2$. The direct proof is surprisingly straightforward and based on the simplified search tree depicted in Fig. D.1.

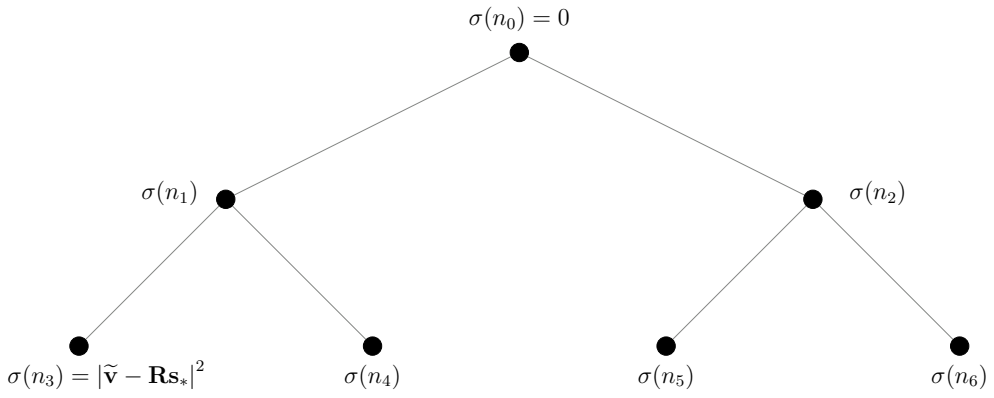


Figure D.1: A weighted binary tree for sphere decoding with $M = B = 2$, where the ML solution is associated with node n_3, and hence the optimal search radius is $C_* = \sqrt{\sigma(n_3)}$.

We begin by deriving an upper bound on the number of nodes expanded by a SD:

Proposition D.1 *Given target* $\mathbf{v}$*, channel matrix* $\mathbf{H}$*, and finite alphabet* $\mathcal{X}$ *of size* B*, the problem dimension is* $M = \operatorname{rank}(\mathbf{H})$*. Then the number of nodes expanded by a sphere decoder satisfies*

$$\nu \leq \sum_{i=0}^{M-1} B^i. \tag{D.2}$$

Proof Recalling the $(M+1)$-level B-ary tree structure that underlies the operation of sphere decoders, the maximum number of nodes expanded is clearly the number of non-leaf nodes, an expression for which is provided in (D.2). ■

Theorem D.3 *Given a matrix* $\mathbf{H} \in \mathbb{R}^{N\times 2}$ *and finite alphabet* $\mathcal{X}$ *of size 2, the ordering* Π_p *computed by Algorithm 2 is optimal in the sense of* (4.4) *for all targets* $\mathbf{v} \in \mathbb{R}^N$.

Proof Without loss of generality, we assume that the ML solution is associated with node n_3, and hence also that the optimal search radius is $C_* = \sqrt{\sigma(n_3)}$. Then it is clear that nodes n_0 and n_1 are expanded by all sphere decoders. Proposition D.1 states that the number of nodes expanded is at most 3. Therefore in order to evaluate the computational efficiency induced by an ordering, it remains to consider whether node n_2 is expanded.

Since $M = 2$, there are two possible orderings of the columns of $\mathbf{H}$. For both orderings, the structure of the tree in Fig. D.1 applies, and the weights of the root and leaf nodes are also the same by definition. Thus the only difference between the performance achieved under the two orderings arises from the weight of node n_2, specifically $\nu_{ASD} = 3$ if and only if $\sigma(n_2) < C_*^2$, otherwise $\nu_{ASD} = 2$.

Let the weight of node n_2 and the number of nodes expanded by the ASD under ordering $\{2,1\}$ be denoted as σ_1 and $\nu_{ASD,1}$, respectively, and those under ordering $\{1,2\}$ as σ_2 and $\nu_{ASD,2}$. Then based on decision criterion (4.5), Algorithm 2 proposes ordering $\{1,2\}$ if $\sigma_1 < \sigma_2$, and ordering $\{2,1\}$ if $\sigma_2 < \sigma_1$. If $\sigma_2 = \sigma_1$, it makes an arbitrary choice. All of the possible cases are enumerated in Table D.1:

σ_1 ? σ_2	? $< C_*$	? $\geq C_*$	$\nu_{ASD,1}$	$\nu_{ASD,2}$	Π_*	Π_p
$\sigma_1 < \sigma_2$		$\sigma_1, \sigma_2 \geq C_*$	2	2	either	$\{1,2\}$
	$\sigma_1 < C_*$	$\sigma_2 \geq C_*$	3	2	$\{1,2\}$	$\{1,2\}$
	$\sigma_1, \sigma_2 < C_*$		3	3	either	$\{1,2\}$
$\sigma_1 = \sigma_2$		$\sigma_1, \sigma_2 \geq C_*$	2	2	either	either
	$\sigma_1, \sigma_2 < C_*$		3	3	either	either
$\sigma_1 > \sigma_2$		$\sigma_2, \sigma_1 \geq C_*$	2	2	either	$\{2,1\}$
	$\sigma_2 < C_*$	$\sigma_1 \geq C_*$	2	3	$\{2,1\}$	$\{2,1\}$
	$\sigma_2, \sigma_1 < C_*$		3	3	either	$\{2,1\}$

Table D.1: Enumeration of possible relationships between σ_1, σ_2 and C_*, as well as the resulting optimal and proposed orderings (Algorithm 2).

It is clear that in all cases the proposed ordering agrees with the optimal choice. ■