Semidefinite Relaxation-Based Soft MIMO Demodulation via Efficient Dual Scaling

SEMIDEFINITE RELAXATION-BASED SOFT MIMO DEMODULATION VIA EFFICIENT DUAL SCALING

BY

MAHSA SALMANI, B.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

Master of Applied Science (2014) McMaster University
(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE:	Semidefinite Relaxation-Based Soft MIMO Demodulation via Efficient Dual Scaling
AUTHOR:	Mahsa Salmani B.Sc., (Electrical Engineering) University of Tehran, Tehran, Iran
SUPERVISOR:	Dr. T. N. Davidson
NUMBER OF PAGES:	xiii, 118

To my beloved husband:

Foad

Abstract

Soft multiple-input multiple-output (MIMO) demodulators are a core component of iterative receivers for MIMO communication systems that employ bit-interleaved coded modulation (BICM). The role of these demodulators is to extract a good approximation of the posterior likelihood of each bit transmitted at each channel use. The main challenge in designing a soft MIMO demodulator is to achieve the desired level of performance at a reasonable computational cost. This is important because in the case of a memoryless MIMO channel, the computational cost of the exact soft demodulator increases exponentially with the number of bits transmitted per channel use, and the cost grows faster in the case of the channels with memory.

Several approximate low-complexity soft demodulators for memoryless channels have been proposed in the literature. In this thesis, we develop a low-complexity soft MIMO demodulator that is based on semidefinite relaxation (SDR) and uses the max-log approximation to reduce the cost of the demodulation. In particular, we develop a customized dual-scaling algorithm to solve the semidefinite program that constitutes the core computational task of the SDR-based soft demodulator. The computational cost per iteration of the customized dual algorithm is about half that of the existing customized primal-dual algorithm, and this leads to a reduction in the overall computational cost. We apply the customized dual-scaling algorithm to two different list-based soft demodulators, the list-SDR and single-SDR demodulators, and compare the performance, computational cost, and EXIT chart characteristics of these demodulators with other existing methods. This comparison shows that the developed demodulator provides a desirable trade-off between performance and complexity.

Acknowledgements

First and foremost, I would like to express my deepest appreciation to my supervisor, Prof. Tim Davidson for his support, encouragement and invaluable comments. The complementation of this master thesis would have not been possible without his supervision and constant help.

The administrative team of the Electrical and Computer Engineering Department deserves my special thanks for their help in many matters.

I take this opportunity to express the profound gratitude from my deep heart to my beloved parents, Zahra and M. Hossein, for their endless love and support through my life.

Last but not least, I owe my deepest thanks to my loving husband, Foad, for his love, patience and encouragements. I appreciate all his help and support through my research. I dedicate this thesis to him with all my love.

Abbreviations

AWGN	Additive White Gaussian Noise
BCJR	Bahl-Cocke-Jelinek-Raviv
BER	Bit Error Rate
BICM	Bit Interleaved Coded Modulation
BPSK	Binary Phase Shift Keying
CDMA	Code Devision Multiple Access
FLOP	Floating Point Operation
IDD	Iterative Demodulation and Decoding
LISS	List Sequential
LLR	Log Likelihood Ratio
MAP	Maximum a Posteriori Probability
MIMO	Multiple Input Multiple Output
ML	Maximum Likelihood
MMSE	Minimum Mean Square Error
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Keying
RTS	Repeated Tree Searching

SD	Sphere decoding
SDP	Semidefinite Programming
SDR	Semidefinite Relaxation
SIC	Soft Interference Cancellation
SNR	Signal to Noise Ratio
STS	Single Tree Searching

Glossary of Notations

$\mathbf{b}$	Transmitted bit vector in one channel use
b_i	The i^{th} transmitted bit in a bit vector
$\mathbf{H}$	Channel matrix
$\tilde{\mathbf{H}}$	Real representation of the channel matrix
$\mathbf{s}$	Transmitted symbol vector for one channel use
$\mathbf{s}^t$	Transmitted codeword
$\tilde{\mathbf{s}}$	Real representation of the transmitted symbol vector
s_i	The i^{th} transmitted symbol in a symbol vector
$\mathbf{v}$	Additive noise vector over one channel use
$\tilde{\mathbf{v}}$	Real representation of the additive noise vector
$\mathbf{y}$	Received signal in one channel use
$\mathbf{y}^t$	Signal received while a codeword is transmitted over the channel
$\tilde{\mathbf{y}}$	Real representation of the received signal
$\boldsymbol{\lambda}^A$	The input soft information of the demodulator or decoder
$\boldsymbol{\lambda}^D$	The output soft information of the demodulator or decoder
$\boldsymbol{\lambda}^E$	The extrinsic soft information of the demodulator or decoder

Contents

List of Figures

Chapter 1

Introduction

The broadcast nature of wireless transmission, and the conventional methods for managing the resulting interference, mean that in the design of wireless communication systems it is important to ensure that the available spectrum is used in an efficient way. Systems with multiple antennas at the transmitter and multiple antennas at the receiver, known as multiple-input multiple-output (MIMO) systems, have the potential to provide increased efficiency over single antenna systems (e.g., Tse and Viswanath, 2005; Goldsmith, 2005). However, these systems are substantially more complex than single antenna systems. As a result, the design of practical MIMO systems involves trade-offs between reliability, data rate and computational cost that tend to be stronger than the corresponding trade-off in single antenna systems. The goal of this thesis is to contribute to the development of receivers that provide a compelling balance between computational cost and performance.

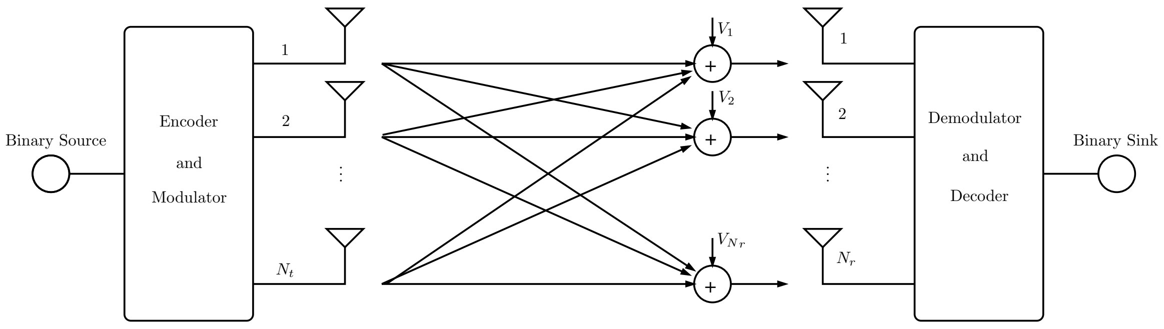

Figure 1.1: A standard MIMO system

1.1 Multiple-Input Multiple-Output Systems

An abstract model for a generic MIMO communication system is shown in Fig. 1.1. In this model, the encoder observes the binary message that is to be sent, and based on that message it synthesizes signals to be transmitted from each antenna. These signals propagate through the medium and are sensed by the antennas at the receiver. The decoder takes the measured signals and estimates the message that was encoded by the transmitter. This estimation can be based on information that the receiver has about the channel through which the signals propagate and, in certain cases, information about the probability with which each message will be sent.

To examine the potential benefits of a MIMO system, let us consider a simple scenario in which the channel from each transmitter antenna to each receiver antenna is linear, approximately flat in frequency across bandwidth of transmitted signal (i.e., the coherence bandwidth of the channel is significantly larger than the bandwidth of the signal), and approximately constant over the duration of communication (i.e., the coherence time of the channel is significantly larger than the duration of communication). In that setting, each receiver antenna observes a (typically different) linear combination of the signals that were sent from the transmitter antennas.

To build a model for that setting, we assume perfect carrier and symbol synchronization at the receiver, and in the k^{th} usage of the channel, we let $x_m[k]$ denote the baseband equivalent of the signal transmitted from transmitter antenna m and $y_n[k]$ denote the baseband equivalent of the signal observed by receiver antenna n. With this notation in place, we can write

$$y_n[k] = \sum_{m=1}^{N_t} h_{nm}[k]x_m[k] + v_n[k], \tag{1.1}$$

where $v_n[k]$ is the additive noise at the n^{th} receiver antenna, and $h_{nm}[k]$ is the baseband equivalent channel from transmitter antenna m to receiver antenna n at the k^{th} channel usage. Therefore, if we write all the signals that are received by the receiver antennas at the k^{th} channel use, we can construct a vector of received signal as $\mathbf{y}_k$ and rewrite (1.1) as

$$\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k, \tag{1.2}$$

where $\mathbf{H}_k$ is the channel matrix at the k^{th} channel use.

If the transmitted signals are correlated across the antennas, then the receiver can take advantage of the received different linear combination to improve the reliability of detection; for example, if a single signal was repeated from all transmitter antennas, that signal would see N_tN_r channel gains as it propagates to the receiver antennas. The receiver antenna can take advantage of this "diversity" to improve the reliability of detection.

An alternative scenario would be to send different signals from each transmitter antenna; a technique that is referred to as spatial multiplexing. Although spatial multiplexing can enable the successful transmission of higher data rates, the receiver has to deal with the self-interference of the signals that are transmitted simultaneously, and this dramatically increases the computational cost of the receiver.

Given the difference between the repetition approach and spatial multiplexing, it is interesting to ask whether there are any intermediate points that provide a trade-off. Indeed, there are. In fact, in channels that change slowly, there is a fundamental trade-off between diversity and spatial multiplexing (Zheng and Tse, 2003). Recent interpretations of where on that trade-off one should operate have suggested that it should be at the spatial multiplexing end (Lozano and Jindal, 2010), and hence, in this thesis we focus on spatial multiplexing. That said, many of the insights and algorithms that will be developed herein can be applied directly to systems that employ space-time block coding schemes from the broad class of linear dispersion codes (Hassibi and Hochwald, 2002).

Even though spatial multiplexing is quite simple to implement at the transmitter, the optimal receiver is complicated due to the interference between the symbols transmitted at a given channel use.. One approach to designing efficient spatial multiplexing schemes is to structure the abstract model in Fig. 1.1 by employing the principles of Bit-Interleaved Coded Modulation (BICM) with iterative soft demodulation and decoding (IDD) (e.g., i Fàbregas *et al.*, 2008). In that case, the encoder and decoder of Fig. 1.1 take the forms in Fig. 1.2.

In BICM-IDD, the information bits that constitute the message are first encoded by a binary encoder. Then the encoded bits are interleaved and modulated to symbols from a scalar constellation. In the case of spatial multiplexing, these scalar symbols are demultiplexed and transmitted from the different antennas. The interleaver has the effect of dispersing correlated encoded bits across the codeword so that the local

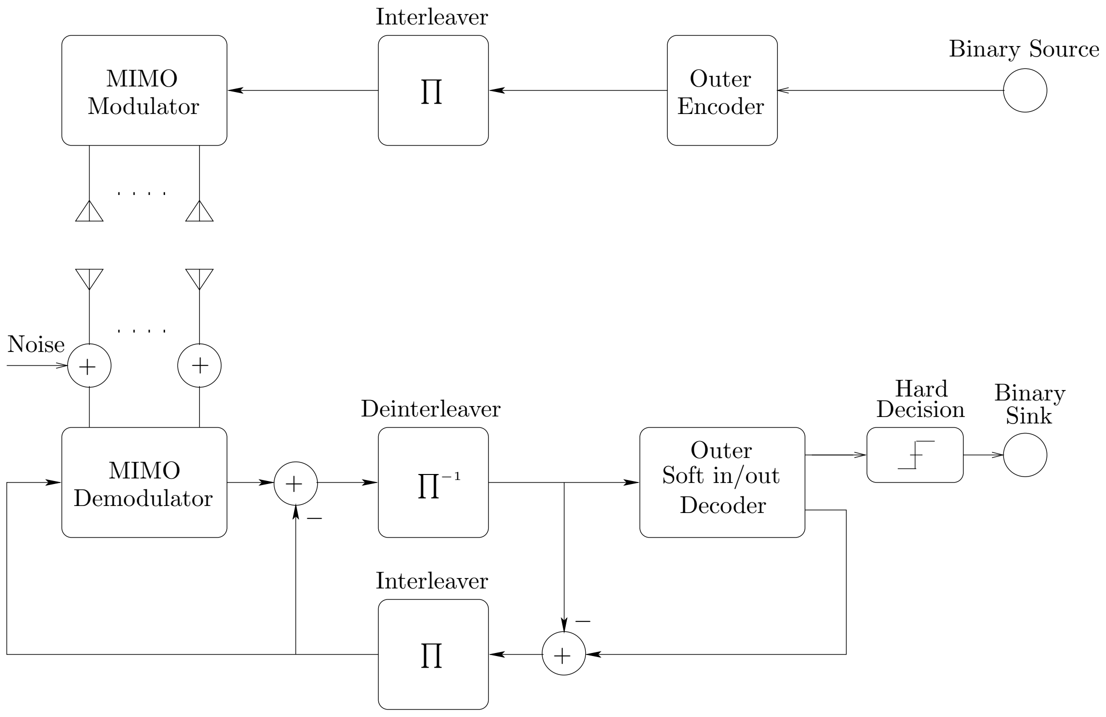

Figure 1.2: MIMO BICM-IDD transceiver

correlation is typically weak or non-existent. This serves two purposes. First, it improves the robustness of the system to bursts of demodulation errors, such as those that occur when the channel is in deep fading. Second, it enables some approximations to be made in the demodulator that significantly reduce its computational cost.

On the other side of the channel, the iterative demodulation and decoding (IDD) receiver is structured in a way that is matched to the transmitter. The MIMO demodulator works on a "per-channel-use" basis. The demodulator extracts an estimation of the posterior probability of each encoded bit that was transmitted in the current channel use. (That probability is often represented by the posterior log-likelihood ratio.) In the second and subsequent demodulation steps, the demodulator uses the estimation of *a priori* probability of these bits that were produced by the previous decoding operation. (In the first iteration the bits are presumed to be equally likely.) Once the demodulation step has been completed for each of the channel uses required to transmit the encoded bits that represent the message, the demodulator's estimate of the posterior probabilities are passed to a soft-input soft-output decoder that seeks to maximize the *a posteriori* probability of each bit in the codeword. The decoder's estimate of these *a posteriori* probabilities are then fed back to the demodulator for the next demodulation iteration. The feedback structure of the receiver in Fig. 1.2 has often been referred to as a "turbo" structure, although care needs to be taken to avoid the destabilizing effects of positive feedback, and hence the subtraction blocks in the figure are used.

One of the convenient features of BICM is that we can leverage a long history of development of binary coding schemes (Costello and Forney Jr, 2007), such as convolutional codes, turbo, and low density parity check codes (e.g. Lin and Costello, 2004; Richardson and Urbanke, 2008; Hanzo *et al.*, 2002), and the corresponding soft-input soft-output decoders such as Bahl-Cocke-Jelinek-Raviv (BCJR, Bahl *et al.*, 1974) or message passing, (e.g., Kschischang *et al.*, 2001).

The main focus of this thesis is the development of computationally-efficient high-performance soft demodulators for use in the IDD receiver. Given the iterative nature of the receiver, the demodulator must be computationally-efficient. However, it must also be quite accurate in order to reduce the number of iterations that are required, and to reduce convergence problems. Distinct from many of the existing approaches in the literature, which employ the principles of tree search or minimum mean square error (MMSE) estimation, the proposed demodulators will be based on an optimization procedure known as semidefinite relaxation (Luo *et al.*, 2010).

To provide further background for the proposed work, in the next chapter, we will describe in more detail the system model for soft MIMO demodulation, and the max-log approximation that is used in a variety of reduced-complexity soft modulators. We will also classify soft demodulators in two different categories. In Chapter 3, we will provide context for the proposed demodulators by describing some of the existing reduced-complexity soft demodulators in each of the two classes introduced in Chapter 2.

1.2 Contribution

In this thesis, we develop a low-complexity soft MIMO demodulator which is based on semidefinite relaxation. In this method, the optimal solution of the underlying detection problem is approximated by using semidefinite relaxation. The solution to the semidefinite program is used to guide a randomization procedure that is used to generate a list of candidate bit vectors, from which the soft output is extracted.

The main contribution in this thesis is to propose an algorithm that can be used to solve the semidefinite program (SDP) in a low-complexity way. Recently, a customized primal-dual interior point method was used to solve the SDP for the case of QPSK and 16-QAM (e.g., Nekuii, 2008; Nekuii and Davidson, 2009b; Nekuii *et al.*, 2011). In this thesis, we propose customized dual-scaling algorithm to solve the SDP for general rectangular QAM alphabets. The algorithm is based on the principles that underlie an existing generic algorithm (Benson *et al.*, 2000), but the proposed algorithm is tailored in such a way that we can take advantage of the sparse structure of the matrices.

We also exploit other features of the communication application and the result is an algorithm that, in our experiments, as about twice as fast as the existing customized primal-dual algorithm per iteration and the final cost is also cheaper.

1.3 Thesis Outline

In Chapter 2, we will describe the system model and formulate the problem of interest of this thesis. We will describe the structure of BICM in more detail and will review the "max-log" approximation of log-likelihood ratio, which is employed in many soft demodulators. We will also classify soft demodulators into the classes of "hard demodulation-based" and "list-based" demodulators.

In Chapter 3, we will review some of the existing approaches to soft MIMO demodulation. Firstly, we will consider some hard demodulation-based demodulators, such as the repeated tree search and single tree search methods. Subsequently, some list-based soft demodulators will be reviewed, including list sphere decoding and list sequential decoding (LISS). We will also quickly review some other list-based and fixed-complexity soft demodulators. Finally, methods that fall outside these classes, such as the minimum mean square error soft interference canceller (MMSE-SIC) will be described.

In Chapter 4, the basis of semidefinite relaxation will be explained. We will show how the problem of finding the maximum *a posteriori* solution can be approximated by a semidefinite program. Two classes of list-based demodulators that use SDR will be introduced in this chapter. These demodulators differ in the way that the list is generated.

In Chapter 5, we will develop a customized dual-scaling algorithm to solve the semidefinite program SDP that arises from the semidefinite relaxation of the demodulation problem. We will provide the structured matrix formulation that we use in order to solve the SDP using the dual-scaling algorithm. We will then introduce the dual potential function and the way it is used to solve the problem iteratively. Some customizations will also be stated as these help to further reduce the computational cost of the dual-scaling algorithm.

Finally in Chapter 6, the simulation results are shown. First, we compare the performance of different soft demodulators in terms of the bit error rate at different SNRs. In order to compare the complexity of these methods, we count the floating points operations (FLOPs) required by each demodulator and present the CDF of the FLOP counts. We also compare these demodulators in terms of their extrinsic information transfer chart (EXIT-chart; e.g., ten Brink, 2001). These charts help to provide some insight into the performance of demodulators and detectors in iterative demodulation and decoding receivers.

In Chapter 7, we will conclude the thesis and provide some suggestions for the directions of future work.

Chapter 2

Problem Statements and System Model

In the previous chapter, we described MIMO wireless systems in which there are several transmitter and receiver antennas. In the case of spatial multiplexing transmission, at each channel use different symbols are transmitted from each antenna. If we let $\mathbf{s}_n$ denote the vector of symbols transmitted at the n^{th} channel use, then for a linear narrow-band MIMO channel model, with additive noise at the receiver, the received signal at n^{th} channel use can be written as

$$\mathbf{y}_n = \mathbf{H}_n \mathbf{s}_n + \mathbf{v}_n, \tag{2.1}$$

where $\mathbf{v}_n$ is a vector of additive noise, which we will model as being zero-mean white additive circular complex Gaussian noise i.e. $\mathbb{E}\{\mathbf{v}_n \mathbf{v}_m\} = \sigma^2 \delta[n-m]\mathbf{I}$. As in the case for the scalar additive white Gaussian noise channel, most coding schemes for the model in (2.1) are based on multiple use of the channel. If we consider a set of

N channel uses, the total received signal can be written as

$$\begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_N \end{bmatrix} = \begin{bmatrix} \mathbf{H}_1 & 0 & \dots & 0 \\ 0 & \mathbf{H}_2 & \dots & 0 \\ \vdots & \ddots & \dots & 0 \\ 0 & 0 & \dots & \mathbf{H}_N \end{bmatrix} \begin{bmatrix} \mathbf{s}_1 \\ \mathbf{s}_2 \\ \vdots \\ \mathbf{s}_N \end{bmatrix} + \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_N \end{bmatrix}, \tag{2.2}$$

or, more compactly, as $\mathbf{y}^t = \mathbf{H}^t\mathbf{s}^t + \mathbf{v}^t$, where $\mathbf{s}^t$ will be referred to as the transmitted codeword.

Given knowledge of the set of codewords that the transmitter might send, which we denote by $\mathcal{S}$, the goal of a coherent receiver is to determine, based on the received signal $\mathbf{y}^t$ and knowledge of the channels $\{\mathbf{H}_i\}_{i=1}^N$, which codeword was transmitted. If all codewords are equally likely, the optimal decoder is the maximum likelihood decoder, that decoder solves the problem

$$\hat{\mathbf{s}}^t_{\mathrm{ML}} = \max_{\mathbf{s}^t \in \mathcal{S}} p(\mathbf{y}^t|\mathbf{s}^t), \tag{2.3}$$

where $p(\mathbf{y}^t|\mathbf{s}^t)$ is the probability of the received signal vector given that $\mathbf{s}^t$ was transmitted.

To assess the computational effort required to solve (2.3), consider a scenario in which we have N channel uses and an overall data rate of r bits per channel use. In this case, the size of the codebook is 2^{rN}. Since the codebook is a discrete set, in the worst case, finding $\hat{\mathbf{s}}^t_{\mathrm{ML}}$ will require 2^{rN} evaluations of $p(\mathbf{y}^t|\mathbf{s}^t)$. Hence, the computational cost grows exponentially with the number of transmitted bits. Effective coding for MIMO channels is based on developing a structured code that enables the optimal decoder to be approximated in a way that enables a good estimate of the optimal

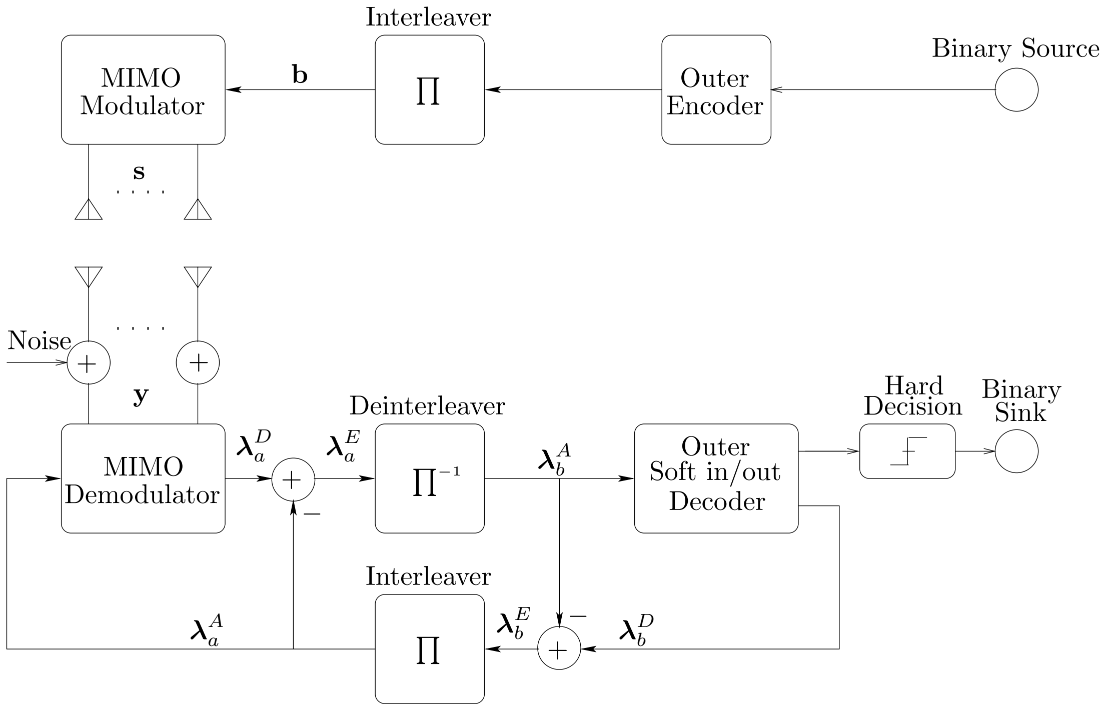
Interleaver
Binary Source
MIMO Modulator
b
Π
Outer Encoder
s
Noise
y
Deinterleaver
Hard Decision
Binary Sink
$\boldsymbol{\lambda}_a^D$
$\boldsymbol{\lambda}_a^E$
MIMO Demodulator
Π^{-1}
$\boldsymbol{\lambda}_b^A$
Outer Soft in/out Decoder
Interleaver
$\boldsymbol{\lambda}_a^A$
$\boldsymbol{\lambda}_b^E$
$\boldsymbol{\lambda}_b^D$

Figure 2.1: MIMO BICM-IDD transceiver

solution in (2.3) to be found with high probability and at low computational cost.

There are several transceiver structures that have been developed to obtain good performance at a reasonable computational cost. As stated in Chapter 1, one of the popular structures is Bit-Interleaved Coded Modulation (BICM) with iterative detection and decoding (IDD) which was illustrated in Fig. 1.2 and has been repeated in Fig. 2.1.

The receiver in this structure is an iterative demodulation and decoding receiver in which soft information regarding the transmitted bits is exchanged between the demodulator and the decoder. The demodulator works on a "per channel use" basis, whereas the decoder works on the whole codeword. In a given iteration, when it comes to the n^{th} channel use, the demodulator observes the signal received at that

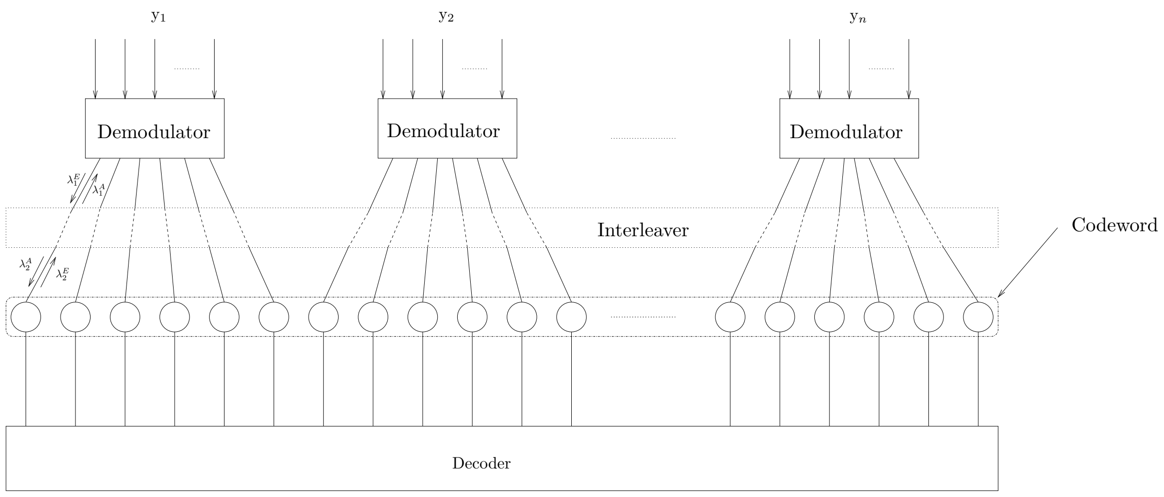

Figure 2.2: A graphical model of turbo scheme.

channel use, $\mathbf{y}_n$, and the estimate of the probability of each bit that indexes $\mathbf{s}_n$ that was obtained from the previous decoding operation. That estimate is the n^{th} block of $\boldsymbol{\lambda}_a^A$ in Fig. 2.1. The demodulator processes these inputs to refine its estimate of the *a posteriori* probabilities of the bits that index $\mathbf{s}_n$. Those estimates are stored in the n^{th} block of $\boldsymbol{\lambda}_a^D$. Once the demodulator has processed each channel use, the outer decoder receives the deinterleaved *a posteriori* likelihood estimates from the demodulator, $\boldsymbol{\lambda}_b^A$, and refines its estimate of the likelihood, producing $\boldsymbol{\lambda}_b^D$. That refined estimate is then sent to the demodulator, and the iteration is repeated.

This iterative structure can also be modeled using a graphical model, (e.g., Boutros and Caire, 2002; Hu and Duman, 2008), as is shown in Fig. 2.2. In this model the demodulator and the decoder are nodes in the graph and extracted soft information is the message passed between these two nodes.

BICM-IDD has proven to be a rather effective pragmatic coding scheme for scalar wireless communication systems (Caire *et al.*, 1998; i Fàbregas *et al.*, 2008). However, in the MIMO case, the demodulation step is still quite computationally expensive, as the computational cost of the optimal demodulator grows exponentially with the number of bits transmitted per channel use. Therefore, an important goal in the design of soft MIMO demodulators is to achieve acceptable performance (near optimal demodulator) in systems that operate at rates that are a significant fraction of the fundamental limit, and yet incur a reasonable computational cost. The focus of this thesis is to develop such a demodulator using a perspective on the problem that is substantially different from that of the common strategies.

To place that contribution in context, we will first describe in more detail the system that we will consider, and the soft demodulation process for that system. We will describe existing soft demodulators in Chapter 3 and the considered demodulators in Chapter 4 and Chapter 5.

2.1 System Model

In this thesis, we consider a narrow band MIMO system with N_t transmitter antennas and N_r receiver antennas. Encoded bits are assumed to be mapped to constellation symbols and the symbols are sent to the antennas according to the spatial multiplexing scheme. Hence at each channel use different symbols are transmitted from each antenna. If we let $\mathbf{s}_n$ denote the transmitted symbol vector in n^{th} channel use, then as in (2.1) the received signal can be written as

$$\mathbf{y}_n = \mathbf{H}_n \mathbf{s}_n + \mathbf{v}_n, \tag{2.4}$$

where $\mathbf{H}_n$ is the $N_r \times N_t$ matrix of channel gains which is assumed to be known at the receiver, and $\mathbf{v}_n$ is a vector of zero-mean additive white circular complex

Gaussian noise with covariance equal to $\sigma^2\mathbf{I}$. As $\mathbf{s}_n$ is a symbol vector of a constellation generated by the modulator, it is a mapping of bits to symbol vectors. If the scalar constellation is denoted by $\mathcal{C}$, then $\mathbf{s}_n \in \mathcal{C}^{N_t}$. To capture the ways in which the bits are mapped to the symbols, we will let $\mathbf{b}_n$ denote the block of the interleaved codeword that indexes $\mathbf{s}_n$, and we will write

$$\mathbf{s}_n = \mathcal{M}(\mathbf{b}_n), \tag{2.5}$$

where $\mathcal{M}(.)$ denotes the mapping scheme used in the modulator. In the schemes that we will consider, this mapping scheme operates on a symbol-by-symbol basis, which means that each symbol is the mapping of a different subblock of $\mathbf{b}_n$. However, for convenience we will use the vector notation in (2.5).

To simplify the notation used in the description of soft MIMO demodulation, we will leave the fact that the demodulator works on a per-channel-use basis implicit. As a result, we will drop the channel use index and write the system model in (2.4) as

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{v}, \tag{2.6}$$

where $\mathbf{s} = \mathcal{M}(\mathbf{b})$.

2.2 Soft MIMO Demodulation

In this section we outline the basic principles of soft MIMO demodulation. A review of specific soft MIMO demodulators that have been proposed in the literature will be provided in Chapter 3.

We will consider systems in which the scalar constellation $\mathcal{C}$ has 2^B points. Therefore, the total number of bits transmitted per channel use is BN_t. We will denote the vector of those bits as $\mathbf{b} = [b_1, \dots, b_{BN_t}]$. We will consider systems in which the output of the soft demodulator is in the form of the posterior log-likelihood ratio (LLR). For the i^{th} bit of $\mathbf{b}$ that ratio is

$$\lambda_i^D \triangleq \log \frac{p(b_i = +1|\mathbf{y}, \mathbf{H})}{p(b_i = -1|\mathbf{y}, \mathbf{H})}, \quad i = 1, \dots, BN_t. \tag{2.7}$$

By using Bayes' theorem, equation (2.7) can be written as

$$\lambda_i^D = \log \frac{\sum_{\mathcal{L}_{i,+1}} p(\mathbf{y}|\mathbf{b}, \mathbf{H}) p(\mathbf{b})}{\sum_{\mathcal{L}_{i,-1}} p(\mathbf{y}|\mathbf{b}, \mathbf{H}) p(\mathbf{b})}, \quad i = 1, \dots, BN_t, \tag{2.8}$$

where if $\mathcal{L}$ denotes the list of all 2^{BN_t} bit vectors, then $\mathcal{L}_{i,\pm 1} \triangleq \{\mathbf{b} \in \mathcal{L} \mid b_i = \pm 1\}$. Under the assumed model of additive white Gaussian noise at the receiver,

$$p(\mathbf{y}|\mathbf{b}, \mathbf{H}) = \frac{1}{(2\pi)^{N_t/2}} \exp\Big(-\frac{\|\mathbf{y} - \mathbf{H}\mathcal{M}(\mathbf{b})\|^2}{2\sigma^2}\Big). \tag{2.9}$$

If we let $\breve{D}(\mathbf{b})$ be defined as $\breve{D}(\mathbf{b}) = \|\mathbf{y} - \mathbf{H}\mathcal{M}(\mathbf{b})\|^2 - 2\sigma^2 \log p(\mathbf{b})$, then (2.8) can be written as

$$\lambda_i^D = \log \frac{\sum_{\mathcal{L}_{i,+1}} \exp(-\breve{D}(\mathbf{b})/(2\sigma^2))}{\sum_{\mathcal{L}_{i,-1}} \exp(-\breve{D}(\mathbf{b})/(2\sigma^2))}, \quad i = 1, \dots, BN_t. \tag{2.10}$$

In the scheme considered in this thesis, $p(\mathbf{b})$ is not available directly, it is approximated using the information received from the decoder in previous iteration. Using the fact that the transmitted bits are interleaved randomly after encoding, they can be approximated as being independent from each other. Therefore, we can estimate

$p(\mathbf{b})$ using

$$p(\mathbf{b}) \simeq \prod_{i=1}^{BN_t} p(b_i), \tag{2.11}$$

where, with a mild abuse of notation, $p(b_i)$ denotes the estimate of the probability that the i^{th} bit is equal to b_i and it was determined by the previous iteration of the decoder.

Using this approximation, we can approximate λ_i^D by

$$\lambda_i^D \simeq \log \frac{\sum_{\mathcal{L}_{i,+1}} \exp(-D(\mathbf{b})/(2\sigma^2))}{\sum_{\mathcal{L}_{i,-1}} \exp(-D(\mathbf{b})/(2\sigma^2))}, \quad i = 1, \ldots, BN_t, \tag{2.12}$$

where

$$D(\mathbf{b}) \triangleq \|\mathbf{y} - \mathbf{H}\mathcal{M}(\mathbf{b})\|^2 - 2\sigma^2 \sum_{i=1}^{BN_t} \log p(b_i). \tag{2.13}$$

Let us define

$$\lambda_i^A = \log \frac{p(b_i = +1)}{p(b_i = -1)}, \quad i = 1, \ldots, BN_t, \tag{2.14}$$

as the i^{th} element of $\boldsymbol{\lambda}^A$, where $\boldsymbol{\lambda}^A$ denotes the vector that represents (2.11) in LLR format; i.e., $\prod_i p(b_i) \propto \exp((\boldsymbol{\lambda}^A)^T\mathbf{b}/2)$. Then equation (2.13) can be written as

$$D(\mathbf{b}) \triangleq \|\mathbf{y} - \mathbf{H}\mathcal{M}(\mathbf{b})\|^2 - \sigma^2(\boldsymbol{\lambda}^A)^T\mathbf{b}. \tag{2.15}$$

Many computationally efficient approximate soft demodulation algorithms are based on the “max-log” approximation of (2.12) under which the logarithm of a summation of exponentials of functions is approximated by the function that has maximum value. Using that approximation we have

$$\lambda_i^D \simeq \log \frac{\sum_{\mathcal{L}_{i,+1}} \exp(-D(\mathbf{b})/(2\sigma^2))}{\sum_{\mathcal{L}_{i,-1}} \exp(-D(\mathbf{b})/(2\sigma^2))} \tag{2.16a}$$

$$\simeq \frac{1}{2\sigma^2}\big(\min_{\mathbf{b}\in\mathcal{L}_{i,-1}} D(\mathbf{b}) - \min_{\mathbf{b}\in\mathcal{L}_{i,+1}} D(\mathbf{b})\big). \tag{2.16b}$$

By making this approximation, approximate soft MIMO demodulation can be achieved by solving the two binary optimization problems in (2.16b) for each of the BN_t bits, b_i in $\mathbf{b}$. In essence, approximate soft demodulation is achieved by solving two hard demodulation problems. Those hard demodulation problems are still “hard” in the computational sense, but approximate solutions can be obtained using various forms of tree search, through estimation, or using semidefinite relaxation techniques. However, many of those techniques remain rather computationally expensive.

Another class of soft demodulation methods is based on approximating (2.12) by taking the summations, or solving the max-log approximation thereof, over a list of candidate bit vectors that is significantly shorter than the list of all bit vectors with $+1$ or -1 in the i^{th} position, which we have denoted by $\mathcal{L}_{i,\pm1}$; cf. (2.8). These demodulators are based on generating a list of bit-vectors, $\hat{\mathcal{L}}$, that contains a number of vectors with small values for $D(\mathbf{b})$ and approximating the LLRs by using marginalization over $\hat{\mathcal{L}}_{i,\pm1}$, rather than $\mathcal{L}_{i,\pm1}$. That is,

$$\lambda_i^D \simeq \log \frac{\sum_{\hat{\mathcal{L}}_{i,+1}} \exp(-D(\mathbf{b})/(2\sigma^2))}{\sum_{\hat{\mathcal{L}}_{i,-1}} \exp(-D(\mathbf{b})/(2\sigma^2))} \tag{2.17a}$$

$$\simeq \frac{1}{2\sigma^2}\big(\min_{\mathbf{b}\in\hat{\mathcal{L}}_{i,-1}} D(\mathbf{b}) - \min_{\mathbf{b}\in\hat{\mathcal{L}}_{i,+1}} D(\mathbf{b})\big). \tag{2.17b}$$

In list-based methods, there are several approaches that can be taken to the construction of the list. The list may be generated once, at the first iteration, and then used for rest of the iterations. In that case, no *a priori* information about the transmitted bits is used in the list generation. Another option is to update the list at each iteration according to the received soft information regarding the transmitted bits. The main difference between list-based methods is in the way in which they generate the list members. This can have a significant impact on the performance and computational cost of the demodulator.

There are some other soft MIMO demodulation methods that are neither list-based nor hard-demodulation-based group. One of these is the MMSE-SIC demodulator, which is based on minimum mean square error equalization with parallel soft interference cancellation. In this scheme estimates of the transmitted symbols are computed and, accordingly, interference can be cancelled (in a probabilistic sense). That enables the soft information for each symbol to be computed as if the symbol were transmitted over a single-input single-output AWGN channel.

Having established a coarse taxonomy for soft MIMO demodulators, we will describe in detail some existing demodulators in the next chapter.

Chapter 3

A Survey of Methods for Soft MIMO Demodulation

In Chapter 2, we described the principles of soft MIMO demodulation and stated that some soft demodulation methods are based on hard demodulation, some of them are based on list generation, and some, such as MMSE-SIC demodulator, do not belong to either of these two classes. In this chapter, we will describe in more detail a variety of different approaches to soft MIMO demodulation that have previously been proposed.

3.1 Tree Searching Method

A number of approaches to soft demodulation are based on a simple transformation of the received signal that reveals that the information of interest can be obtained using a tree search (e.g., Murugan *et al.*, 2006; Larsson, 2009). Since this tree search is a common feature of several methods, we will first describe how the tree search structure is revealed.

Recall from (2.6) that we model the received signal as

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{v}, \tag{3.1}$$

where $\mathbf{v}$ is a zero-mean circular complex random variable with covariance $\mathbf{R}_v = \sigma^2\mathbf{I}$. By applying QR decomposition to the channel matrix $\mathbf{H}$ we can write

$$\mathbf{y} = \mathbf{Q}\mathbf{R}\mathbf{s} + \mathbf{v}, \tag{3.2}$$

where the matrix $\mathbf{Q}$ is a unitary matrix and the matrix $\mathbf{R}$ is upper-triangular. By left multiplying (3.2) by $\mathbf{Q}^H$ we obtain

$$\breve{\mathbf{y}} = \mathbf{R}\mathbf{s} + \breve{\mathbf{v}}, \tag{3.3}$$

where $\breve{\mathbf{y}} = \mathbf{Q}^H\mathbf{y}$, and $\breve{\mathbf{v}} = \mathbf{Q}^H\mathbf{v}$ is, like $\mathbf{v}$, a zero-mean circular complex Gaussian random variable with $\mathbf{R}_{\breve{v}} = \sigma^2\mathbf{I}$.

In terms of describing the algorithm, we will find it more convenient to consider the QL decomposition of $\mathbf{H}$; i.e., $\mathbf{H} = \tilde{\mathbf{Q}}\mathbf{L}$, where $\mathbf{L}$ is a lower-triangular matrix. If we define $\tilde{\mathbf{y}} = \tilde{\mathbf{Q}}\mathbf{y}$ and $\tilde{\mathbf{v}} = \tilde{\mathbf{Q}}\mathbf{v}$, we can write

$$\tilde{\mathbf{y}} = \mathbf{L}\mathbf{s} + \tilde{\mathbf{v}}. \tag{3.4}$$

Expanding the equation in terms of the elements we have

$$\begin{bmatrix} \tilde{y}_1 \\ \tilde{y}_2 \\ \vdots \\ \tilde{y}_{N_t} \end{bmatrix} = \begin{bmatrix} \ell_{1,1} & 0 & \dots & 0 \\ \ell_{2,1} & \ell_{2,2} & \dots & 0 \\ \vdots & \ddots & \dots & \vdots \\ \ell_{N_t,1} & \ell_{N_t,2} & \dots & \ell_{N_t,N_t} \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_{N_t} \end{bmatrix} + \begin{bmatrix} \tilde{v}_1 \\ \tilde{v}_2 \\ \vdots \\ \tilde{v}_{N_t} \end{bmatrix}. \tag{3.5}$$

In some algorithms it can be advantageous to have the flexibility to adjust the ordering of the symbols (e.g., Murugan *et al.*, 2006; Nekuii and Davidson, 2009a). This can be achieved using a permutation matrix $\mathbf{P}$, so that

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{v} = \mathbf{H}\mathbf{P}\mathbf{P}^T\mathbf{s} + \mathbf{v}. \tag{3.6}$$

We can then apply the QL decomposition to the matrix $\mathbf{HP}$. If $\mathbf{s}' = \mathbf{P}^T\mathbf{s}$ and $\mathbf{v}' = (\mathbf{Q}')^H\mathbf{v}$ then we have

$$\mathbf{y} = \mathbf{Q}'\mathbf{L}'\mathbf{s}' + \mathbf{v} \Rightarrow (\mathbf{Q}')^H\mathbf{y} = \mathbf{L}'\mathbf{s}' + \mathbf{v}'. \tag{3.7}$$

In the remainder of the discussion we will stick to the natural ordering in (3.5). However, the techniques we will discuss extend directly to the reordered vector $\mathbf{s}'$ in (3.7).

The search tree obtained from (3.5) has the possible values for s_1 as the nodes in the first level of the tree. Each node at each of the subsequent levels is associated with a vector containing the values of the symbols of all nodes along the path from the root of the tree to that node (e.g., Murugan *et al.*, 2006; Larsson, 2009). In the k^{th} level that vector takes the form $\mathbf{s}_k = (s_1, s_2, \dots s_k)$. Given an overall cost function of the form $\|\tilde{\mathbf{y}} - \mathbf{Ls}\|^2 - \sum_{k=1}^{N_t} g_k(s_k)$, of which the cost function in (2.13) is a special case, the metric for the branch from the root of the tree to a node at the k^{th} level is (e.g., Larsson, 2009)

$$f_k(s_1, \ldots, s_k) = \left|\tilde{y}_k - \sum_{j=1}^{k} \ell_{n,j} s_j\right|^2 - g_k(s_k). \tag{3.8}$$

In the case of *a priori* information of the form in (2.11), $g_k(s_k) = \log p(s_k)$, where, again with some abuse of notation, $p(s_k)$ denotes the probability that the k^{th} scalar symbol is equal to s_k. The contribution of a given node to the cost function is represented by the cumulative metric which is the summation of metrics of the node and all its "parents", $f_1(s_1) + f_2(s_1, s_2) + \cdots + f_k(s_1, \ldots, s_k)$.

Tree searching can be used to solve the optimal hard demodulation problem in (2.16) by searching the tree to find a node at level N_t that has the least cumulative metric. As searching all nodes of the tree is computationally costly, the tree can be reduced by using the principles that underlie the branch and bound algorithm (e.g., Murugan *et al.*, 2006), in which a node represented by $\mathbf{s}_k$ survives in the tree if it is possible that one of its descendants is the desired node, otherwise the descendants of that node are not explored.

In list-based soft demodulators, we are looking for leaf nodes that have a small cumulative metric. These nodes will be the corresponding list members. In the case of a simple sphere decoding approach the leaf nodes that make the list have metrics that are smaller than the radius of the defined sphere. One simple form of that approach is the sphere decoding method (e.g. Larsson, 2009), which only searches the nodes that lie within a sphere around the received signal, $\mathbf{y}$. Therefore, the standard pruning criteria for each node in this method can be written as

$$f_1(s_1) + f_2(s_1, s_2) + \cdots + f_k(s_1, \ldots, s_k) \geq r^2, \tag{3.9}$$

which means that if the cumulative metric of a node is greater than the defined radius, that node and all its descendants will be discarded from the tree.

There are different ways of searching the tree according to the order in which the nodes are explored: depth-first, breadth-first and best-first. In depth-first, tree searching starts from the root and explores as far as possible along each branch. It means that for each node, the first child and then the grandchildren are searched before the second child. On the other hand, breadth-first starts from root nodes and inspects all neighbouring nodes and then goes to the immediate children of these nodes. In this tree traversal for each node after searching the first child, the second child is searched before the grandchild. Another method for tree searching is "best-first" in which the tree is explored by expanding the exposed node with the smallest cost. Generally, a tree search algorithm will be terminated when all the available nodes are explored and so the desired node is found. However, in the case of demodulation, there are other criteria that may be used to terminate the searching, including the list size.

3.2 Tree Searching with Max-Log Approximation

In Chapter 2, we described the max-log approximation and list generation method to compute the LLRs. By applying the QL decomposition to the channel matrix, the max-log approximation of the LLR (c.f. (2.13) and (2.16)) can be written as

$$\lambda^D_{n,i} = \min_{\mathbf{s}\in\mathcal{X}^{(-1)}_{n,i}} \big(\frac{1}{2\sigma^2}\|\tilde{\mathbf{y}} - \mathbf{L}\mathbf{s}\|^2 - \log p(\mathbf{s})\big) \\ - \min_{\mathbf{s}\in\mathcal{X}^{(+1)}_{n,i}} \big(\frac{1}{2\sigma^2}\|\tilde{\mathbf{y}} - \mathbf{L}\mathbf{s}\|^2 - \log p(\mathbf{s})\big), \quad (3.10)$$

in which i and n refer to the i^{th} bit in the constellation, $i = 1, \dots, B$, corresponding to the n^{th} entry of the transmitted symbol vector, $n = 1, \dots, N_t$, and $\mathcal{X}^{(-1)}_{n,i}$ and $\mathcal{X}^{(+1)}_{n,i}$ are the sets of symbol vectors for which the i^{th} bit in the n^{th} entry is equal to -1 and $+1$, respectively. With some abuse of notation, $p(\mathbf{s})$ denotes the probability that the symbol vector is equal to $\mathbf{s}$ that is obtained using (2.11) and the previous iteration of the decoder.

In order to determine $\lambda^D_{n,i}$, for all i and n, N_t minimization problems over $\mathcal{X}^{(-1)}_{n,i}$ and N_t minimization problems over $\mathcal{X}^{(+1)}_{n,i}$ need to be solved. As these problems are over a discrete feasible set of size $|\mathcal{C}|^{N_t}/2$, where $\mathcal{C}$ is the constellation of symbols and $|\mathcal{C}|$ is the number of symbols in the constellation, the worst case computational cost grows exponentially in the number of antennas and the number of bits in each symbol.

To reduce the number of problems that need to be solved it can be observed, (Studer and Bolcskei, 2010), that if $\mathbf{s}^{MAP}$ denotes the maximum *a posteriori* probability solution of MIMO symbol detection problem, which is

$$\mathbf{s}^{MAP} = \arg\min_{\mathbf{s}} \Big(\frac{1}{2\sigma^2}\|\tilde{\mathbf{y}} - \mathbf{L}\mathbf{s}\|^2 - \log p(\mathbf{s})\Big), \quad (3.11)$$

then one of the minima in (3.10) is given by

$$\lambda^{MAP} = \frac{1}{2\sigma^2}\|\tilde{\mathbf{y}} - \mathbf{L}\mathbf{s}^{MAP}\|^2 - \log p(\mathbf{s}^{MAP}), \tag{3.12}$$

and the other minima can be written in terms of an optimization over all bits, except the i^{th} bit in n^{th} entry, which is fixed as the binary complement of the corresponding bit in $\mathbf{s}^{MAP}$

$$\lambda_{n,i}^{\overline{MAP}} = \min_{\mathbf{s}\in\mathcal{X}_{n,i}^{\overline{(b_{n,i})}}} \big(\frac{1}{2\sigma^2}\|\tilde{\mathbf{y}} - \mathbf{L}\mathbf{s}\|^2 - \log p(\mathbf{s})\big). \tag{3.13}$$

As a result, the LLR equation in (3.10) can be written as

$$\lambda_{n,i}^{D} = \begin{cases} \lambda^{MAP} - \lambda_{n,i}^{\overline{MAP}}, & b_{n,i}^{MAP} = -1 \\ \lambda_{n,i}^{\overline{MAP}} - \lambda^{MAP}, & b_{n,i}^{MAP} = 1 \end{cases}. \tag{3.14}$$

That means $\lambda_{n,i}^{D}$ can be computed for all n and i by solving the problem in (3.11), which has a worst case complexity of $|\mathcal{C}|^{N_t}$ functional evaluations, plus N_t instances of the problem in (3.13), each of which has a worst case complexity of $|\mathcal{C}|^{N_t/2}$ functional evaluations. This is in contrast to solving the N_t problems of worst case complexity $|\mathcal{C}|^{N_t}$ that would be equal to implementing (3.10) directly.

To solve (3.14) using a tree search method, we can use the metric of (3.8) and compute the cumulative metric of each node, which is $d_k = f_1(s_1) + f_2(s_1, s_2) + \cdots + f_k(s_1, \ldots, s_k)$ as

$$d_k = d_{k-1} + f_k, \tag{3.15}$$

with the initial value of d equal to $d_1 = 0$. Now $d(\mathbf{s}) = \frac{1}{2\sigma^2}\|\tilde{\mathbf{y}} - \mathbf{L}\mathbf{s}\|^2 - \log p(\mathbf{s})$ in (3.11) and (3.13) can be found recursively such that $d_{N_t} = d(\mathbf{s})$. By these definitions, the MAP detection and LLR computation can be constructed as a tree with its root above the level $i = 1$ and leaves on level $n = N_t$. Each path from the root down to a leaf corresponds to a symbol vector. As was stated above, in sphere decoding tree searching is done only for nodes which are within a specified radius around the received signal, $\mathbf{y}$. This radius can also be reduced based on the reached leaf node in the tree, in this way the radius can be initialized with $r \leftarrow \infty$ and whenever a leaf node $\mathbf{s}$ is found the radius is updated as $r \leftarrow d(\mathbf{s})$.

3.2.1 Repeated Tree Search - Sphere Decoding (RST-SD)

A straightforward implementation of the discussion above would involve separate tree searches for the problem in (3.12) and the problems in (3.13). The repeated tree search algorithm (Wang and Giannakis, 2006; Shieh *et al.*, 2013) does exactly that, repeating the tree searching algorithm to compute the LLRs in (3.10) for each bit in the symbol vector, using the sphere decoding idea to control the size of tree search. The main drawback in this method is that some nodes may be searched for more than once, which results in significant computational cost.

3.2.2 Single Tree Search - Sphere Decoding (STS-SD)

The single tree search method (Studer and Bolcskei, 2010), is a more efficient way of searching the tree than RTS, as it ensures that each node in the tree will be searched at most once. This is done by searching for the solution to (3.11) and all instances of the solutions to (3.13) simultaneously. The tree search is performed within a radius analogous to that used in sphere decoding, although in this case the radius must be large enough to include good counter-hypothesis, as well as the MAP solution, without being so large that the complexity becomes problematic.

To describe the algorithm, we will redefine λ^{MAP} and $\lambda_{n,i}^{\overline{MAP}}$ to be the best estimates of the likelihoods in (3.12) and (3.13), respectively, that the algorithm has found so far, rather than the optimal values. The algorithm is initialized with $\lambda^{MAP} = \lambda_{n,i}^{\overline{MAP}} = \infty$ and as the algorithm progresses, whenever a leaf node is revealed the values for λ^{MAP} and $\lambda_{n,i}^{\overline{MAP}}$ are updated. The algorithm also stores $\mathbf{b}^{MAP}$, the current hypothesis of the a MAP bit vector. If we let $\mathbf{b}$ denote the bit vector corresponding to a leaf node that has been revealed, then one of the following two actions is taken.

- If $d(\mathbf{b})$ is less than the current value for λ^{MAP}, then an improved MAP hypothesis has been found. That means that we need to update λ^{MAP} and $\mathbf{b}^{MAP}$; i.e., $\lambda^{MAP} \leftarrow d(\mathbf{b})$, and $\mathbf{b}^{MAP} \leftarrow \mathbf{b}$. The current values for $\lambda_{n,i}^{\overline{MAP}}$ also need to be updated for all those (n, i) pairs for which the $(n, i)^{th}$ bit of $\mathbf{b}$ is different from the $(n, i)^{th}$ bit of the previous $\mathbf{b}^{MAP}$, so that each $\lambda_{n,i}^{\overline{MAP}}$ corresponds to a valid counter-hypothesis to the new MAP hypothesis. (In an implementation, these values for $\lambda_{n,i}^{\overline{MAP}}$ would be updated prior to the update $\lambda^{MAP} \leftarrow d(\mathbf{b})$, and $\mathbf{b}^{MAP} \leftarrow \mathbf{b}$).

- If $d(\mathbf{b})$ is greater than the current value for λ^{MAP}, then we do not have a new MAP hypothesis and hence λ^{MAP} and $\mathbf{b}^{MAP}$ are unchanged. The values of $\lambda_{n,i}^{\overline{MAP}}$ should be checked for all (n, i) pairs for which $b_{n,i}$ is the complement of the current $b_{n,i}^{MAP}$. For those n and i, if $d(\mathbf{b}) < \lambda_{n,i}^{\overline{MAP}}$, then $\lambda_{n,i}^{\overline{MAP}}$ should be updated, $\lambda_{n,i}^{\overline{MAP}} \leftarrow d(\mathbf{b})$.

As in some variants of the sphere decoding algorithm, in the STS-SD method the pruning radius is also updated whenever a leaf is reached. The pruning criteria in this method ensures that a node is explored only if it can lead to a change either in λ^{MAP} or $\lambda_{n,i}^{\overline{MAP}}$ for some n and i.

3.2.3 List Sphere Decoding

It was stated in the previous chapter that one class of soft demodulators consists of those that are list-based. These techniques are based on identifying a list of candidate bit vectors that correspond to small values of $\|\mathbf{y} - \mathbf{Hs}\| - \log p(\mathbf{s})$. Hochwald and ten Brink (2003) proposed a list version of sphere decoding in which the list consists of L leaf nodes that make $\|\mathbf{y} - \mathbf{Hs}\|$ the smallest among all other nodes in a determined sphere with radius r. In this method, when a leaf node is found in that sphere, the radius is not updated (decreased) according to the leaf node that has been found, but the node is added to the list if the list is not full. If the list is full, this point is compared with the point with the largest radius in the list, if the new point has smaller radius it will be stored in the list and the other point will be discarded. In this scheme, the list is generated only in the first iteration of demodulation and decoding and *a priori* information provided by the decoder is not used in list generation. (That *a priori* information is used in the calculation of the outputs of the demodulator.)

Vikalo *et al.* (2004) modified the list generation method of Hochwald and ten Brink (2003) by considering the *a priori* information provided by the decoder in the list generation step. In this method, the generated list is updated in each iteration based on the *a priori* information, received from the decoder, and the tree search seeks list members with small values of $\|\mathbf{y} - \mathbf{Hs}\| - \log p(\mathbf{s})$.

3.2.4 List Sequential Decoding

Another list-based soft demodulator in which the list is constructed using a tree search is the list sequential algorithm (LISS, Hagenauer and Kuhn, 2007). The LISS method provides an updated list at each demodulation decoding iteration, and hence, like the modified sphere decoder (Vikalo *et al.*, 2004), the cost function on the tree includes the information from the previous iteration of the decoder. The LISS method uses the stack algorithm (e.g., Jelinek, 1969; Murugan *et al.*, 2006) to search the tree in a best-first manner. The main stack contains the partial paths which have been explored by the algorithm, and the size of stack is one of the parameters that control the trade-off between performance and complexity. The actual metric used in the tree search can be the conventional metric in (3.8) or can include a metric length bias which is the pdf of channel observations, $\ln p(\mathbf{y})$. Although this term does not actually appear in the expression for the LLR, it helps to compare the length of explored path and to control the branch extension. Due to the "best-first" nature of the tree search, the LISS algorithm produces leaf nodes in increasing order of the metric, and hence the algorithm is terminated once a target number of leaf nodes have been found. (One can also choose to terminate if the stack reaches a given size. An alternative is to discard the "least interesting" node when the stack reaches its capacity.) Once the algorithm terminates, the information in the partially explored paths can either be ignored (if the max-log approximation is used), or can be exploited using an extension technique based on the mean of the current symbol estimates.

3.2.5 Some Other Tree Search Approaches

Two other tree search approaches that have received considerable attention in the literature are the K-best approach (Guo and Nilsson, 2006), and the fixed-complexity sphere decoding (Barbero and Thompson, 2008b,a; Liu *et al.*, 2012). Although we will not provide direct comparisons with those approaches, they are briefly described here for completeness.

The K-best algorithm (Guo and Nilsson, 2006) performs a constrained breath-first search of the tree to find leaf nodes with small metrics. The constraint is that only the K best paths are retained at each level. The principles of this algorithm can be applied to any cost function on the tree and hence it can be applied in hard-demodulation-based and list-based demodulators. The parameter K controls the trade-off between performance and complexity. An advantage of the K-best approach is that the tree is searched only in the forward direction, and this can simplify the implementation.

In the tree search algorithms considered so far, the structure of the tree that is searched is determined by the instance of the problem; it is just the principles of the search that are specified. An alternative approach is to partially pre-specified the structure of the tree. This is what is done in "fixed-complexity sphere decoding", (Barbero and Thompson, 2008b,a; Liu *et al.*, 2012). In that approach, the first few layers of the tree are explored in full, and from that point only a single path is taken from each node in the last fully explored layer to a leaf node. The choice of that path is typically made using a simple criterion. This approach has the advantage that its computational cost is fixed by the structure of the tree, rather than depending on the instance of the problem. However, given the fixed structure of the search, its performance depends on the ordering of the symbols in the tree search; c.f. (3.7).

3.3 MMSE-SIC

One of the low-complexity methods for soft MIMO demodulation systems is minimum mean square error with (parallel) soft interference cancellation (MMSE-SIC) algorithm, which was initially proposed by Wang and Poor (1999). The first step of this algorithm is to estimate the transmitted symbols based on the information provided by the decoder in previous iteration. (In the first iteration all symbols are considered equally likely.) According to the estimated values, an estimate of the interference caused by other symbols is calculated for each symbol and it is removed. In order to suppress the remaining interference and noise the MMSE equalizer is used. Finally the LLR of each bit in each symbol is computed as if the symbol were transmitted over an AWGN channel.

The estimation of each transmitted symbols s_n can be written as

$$\hat{s}_n = \mathbb{E}[s_n] = \sum_{a \in \mathcal{C}} p[s_n = a]a, \quad n = 1, \ldots, N_t, \tag{3.16}$$

where

$$p(s_n = a) = \prod_{i=1}^{B} p(b_{n,i} = k) \tag{3.17}$$

denotes the *a priori* probability of the symbol $a \in \mathcal{C}$ with k equal to the i^{th} bit of symbol a. So for each transmitted symbol we can compute the variance which is

$$E_n = \mathrm{Var}[s_n] = \mathbb{E}[|e_n|^2], \tag{3.18}$$

where $e_n = s_n - \hat{s}_n$.

The probability of each associated binary value $b_{n,i}$ in (3.17) is calculated according to the LLR delivered by the channel decoder as follows

$$p(b_{n,i}=k)=\frac{1}{2}\Big(1+(2k-1)\tanh\big(\tfrac{1}{2}\lambda^{A}_{n,i}\big)\Big), \tag{3.19}$$

where $\lambda^{A}_{n,i}=\log\frac{p(b_{n,i}=+1)}{p(b_{n,i}=-1)}$.

The interference is canceled in this method which means according to the estimated symbols, the effect of other symbols on each transmitted symbol can be removed, so the estimation of the received signal can be written as

$$\hat{\mathbf{y}}_n=\mathbf{y}-\sum_{j,j\neq n}\mathbf{h}_j\hat{s}_j=\mathbf{h}_n\hat{s}_n+\hat{\mathbf{v}}_n, \tag{3.20}$$

where $\hat{\mathbf{v}}_n=\sum\limits_{j,j\neq n}\mathbf{h}_je_j+\mathbf{v}$ is noise-plus-interference (NPI). In order to reduce the NPI in each $\hat{\mathbf{y}}_n$ a linear MMSE filter is used.

In (Tuchler *et al.*, 2002), the MMSE filter vector for the n^{th} symbol is computed as

$$\tilde{\mathbf{w}}^H_n=E_s\mathbf{h}^H_n\tilde{\mathbf{A}}^{-1}_n, \tag{3.21}$$

where

$$\tilde{\mathbf{A}}^{-1}_n=\mathbf{H}\tilde{\mathbf{\Lambda}}_n\mathbf{H}^H+N_0\mathbf{I}_{N_r}, \tag{3.22}$$

in which $\tilde{\mathbf{\Lambda}}$ is defined as

$$\tilde{\mathbf{\Lambda}}_{j,j}=\begin{cases}E_j & j\neq n\\ E_s & j=n\end{cases}, \tag{3.23}$$

where E_n was defined in (3.18) and $\mathbb{E}[\mathbf{s}\mathbf{s}^H]=E_s\mathbf{I}_{N_t}$.

Then the MMSE filter vectors in (3.21) are used to reduce the NPI of the $\mathbf{y}_n$, the output of the MMSE filter is

$$\tilde{z}_n = \tilde{\mathbf{w}}_n^H \hat{\mathbf{y}}_n = \tilde{\mathbf{w}}_n^H \mathbf{h}_n s_n + \tilde{\mathbf{w}}_n^H \tilde{\mathbf{v}}_n. \tag{3.24}$$

Let $\tilde{\nu}_n^2$ denotes the variance of $\tilde{z}_n$,

$$\tilde{\nu}_n^2 = \mathrm{Var}\{\tilde{z}_n\} = \tilde{\mathbf{w}}_n^H \Big(\sum_{j, j \neq n} E_j \mathbf{h}_j \mathbf{h}_j^H + N_0 \mathbf{I}_{N_r} \Big) \tilde{\mathbf{w}}_n, \tag{3.25}$$

Then, assuming that NPI, $\tilde{\mathbf{w}}_n^H \tilde{\mathbf{v}}_n$, is Gaussian distributed, the LLR can be computed as

$$\begin{aligned} \lambda_{n,i}^D = &\log\Big(\sum_{a \in \mathcal{X}_{n,i}^{(1)}} \exp\Big(-\frac{|\tilde{z}_n - \tilde{\mu}_n a|^2}{\tilde{\nu}_n^2} + \sum_{i=1}^{B} \frac{(2[a]_n - 1)}{2} \lambda_{n,i}^A \Big) \Big) \\ &- \log\Big(\sum_{a \in \mathcal{X}_{n,i}^{(-1)}} \exp\Big(-\frac{|\tilde{z}_n - \tilde{\mu}_n a|^2}{\tilde{\nu}_n^2} + \sum_{i=1}^{B} \frac{(2[a]_n - 1)}{2} \lambda_{n,i}^A \Big) \Big), \end{aligned} \tag{3.26}$$

where $\mathcal{X}_{n,i}^{(0)}$ and $\mathcal{X}_{n,i}^{(1)}$ are the sets of symbols for which the i^{th} bit is 1 or 0 and $\tilde{\mu}_n$ is the mean of $\tilde{z}_n$.

This MMSE filter computation contains multiple matrix inversions per symbol vector for each iteration, which entails a significant computational cost. Studer *et al.* (2011) proposed an approximate version of MMSE-PIC in which the number of required matrix inversion is reduced to one per symbol vector instead of N_t. In this method, the MMSE filter vectors are computed as

$$\mathbf{W}^H = \mathbf{A}^{-1} \mathbf{H}^H, \tag{3.27}$$

where $\mathbf{A} = \mathbf{H}^H\mathbf{H}\mathbf{\Lambda} + N_0\mathbf{I}_{N_t}, (N_t = N_r)$ and $\mathbf{\Lambda}$ is a diagonal matrix that $\Lambda_{n,n} = E_n$, the MMSE filter vectors are given by the rows of the matrix $\mathbf{W}^H$.

Studer *et al.* (2011) showed that the computational cost of computing LLR in (3.26) can be reduced by simplifying the computation of $\tilde{\nu}_n^2$. It was shown that if we consider $\text{Var}\{\tilde{z}_n\} = \mu_n - E_n\mu_n^2$, where $\mu_n = \mathbf{w}_n^H\mathbf{h}_n$, we can write $\frac{|\tilde{z}_n - \tilde{\mu}_n a|^2}{\text{Var}\{\tilde{z}_n\}}$ in (3.26) as $\rho_n|z_n - a|^2$, where $\rho_n = \frac{\mu_n}{1-E_n\mu_n}$. Now the output of MMSE filter can be shown to be $z_n = \frac{\mathbf{w}_n^H\hat{\mathbf{y}}_n}{\mu_n}$.

So by using the above equations and applying the max-log approximation in (3.26), the LLR can be approximated as

$$\begin{aligned}\lambda_{n,i}^D \simeq \; & \min_{a\in\mathcal{X}_{n,i}^{(-1)}} \Big\{\rho_n|z_n - a|^2 - \sum_{i=1}^{B}\frac{(2[a]_i - 1)}{2}\lambda_{n,i}^A\Big\} \\ & - \min_{a\in\mathcal{X}_{n,i}^{(1)}} \Big\{\rho_n|z_n - a|^2 - \sum_{i=1}^{B}\frac{(2[a]_i - 1)}{2}\lambda_{n,i}^A\Big\}. \end{aligned} \tag{3.28}$$

Another complexity reduction is performed by omitting the prior term in (3.28).This results in no loss for BPSK and 4-QAM, and according to (Studer *et al.*, 2011) only small loss for higher order constellations. Omitting the *a priori* information, we have

$$\lambda_{n,i}^E = \rho_n\big(\min_{a\in\mathcal{X}_{n,i}^{(-1)}} |z_n - a|^2 - \min_{a\in\mathcal{X}_{n,i}^{(1)}} |z_n - a|^2\big). \tag{3.29}$$

So the extrinsic LLR can be calculated directly. We will compare the performance of the full MMSE-SIC (Wang and Poor, 1999) and the approximated one (Studer *et al.*, 2011) in Chapter 6.

3.4 Partial Marginalization

Another reduced-complexity soft MIMO demodulation method is the partial marginalization method introduced by Larsson and Jalden (2008). The main idea in this method is to partition the symbols (bits) set into two subsets and marginalize the posteriori density of received symbols (bits) in two steps. The first step is exact marginalization over a subset of the transmitted symbols (bits), typically those with low signal to noise ratio (SNR), and the second step is an approximate marginalization over the remaining symbols (bits). The complexity of this method is fixed and it has a fully parallel structure. Hence it can be implemented in parallel hardware.

To describe this approach in more detail, we follow Larsson and Jalden (2008) and consider the case in which the transmitted bits are assumed to be equally likely. By using (2.9), (2.8) can be written as

$$\lambda_i^D = \log \frac{\sum_{\mathbf{s}\in\mathcal{X}_i^{(1)}} \exp(-\frac{1}{N_0}\|\mathbf{y}-\mathbf{H}\mathbf{s}\|^2)}{\sum_{\mathbf{s}\in\mathcal{X}_i^{(-1)}} \exp(-\frac{1}{N_0}\|\mathbf{y}-\mathbf{H}\mathbf{s}\|^2)}, \quad i = 1,\ldots,BN_t. \tag{3.30}$$

If we were to marginalize over all the transmitted bits we would have

$$\lambda_i^D = \log \frac{\sum_{b_1=-1}^{1}\cdots\sum_{b_{i-1}=-1}^{1}\sum_{b_{i+1}=-1}^{1}\cdots\sum_{b_{BN_t}=-1}^{1}\mu(b_1,\cdots,b_{i-1},1,b_{i+1},\cdots,b_{BN_t})}{\sum_{b_1=-1}^{1}\cdots\sum_{b_{i-1}=-1}^{1}\sum_{b_{i+1}=-1}^{1}\cdots\sum_{b_{BN_t}=-1}^{1}\mu(b_1,\cdots,b_{i-1},-1,b_{i+1},\cdots,b_{BN_t})}, \tag{3.31}$$

where

$$\mu(b_1,\ldots,b_{BN_t}) = \exp\bigl(-\frac{1}{N_0}\|\mathbf{y}-\mathbf{H}\mathbf{s}(b_1,\ldots,b_{BN_t})\|\bigr), \tag{3.32}$$

and $\mathbf{s}(b_1,\ldots,b_{BN_t})$ is the symbol vector that corresponds to the bits $[b_1,\ldots,b_{BN_t}]$.

In the partial marginalization method there is a fixed number of bits, r, over which exact marginalization is performed. The other $BN_t - r$ bits are only approximately marginalized. The parameter r offers a trade-off between exact and approximate computation of (3.30). If $r = 0$, there are no explicit sums and only approximate marginalization takes place. Let $\mathcal{I}$ be a bit permutation, exact and approximate marginalization is done over $[b_{\mathcal{I}_1}, \cdots b_{\mathcal{I}_r}]$ and $[b_{\mathcal{I}_{r+1}}, \cdots b_{\mathcal{I}_{BN_T}}]$, respectively. So for $i = \mathcal{I}_l$, we have exact marginalization as follows

$$\lambda_i^D \simeq \log\left(\frac{\sum_{b_{\mathcal{I}_1}=-1}^{1} \cdots \sum_{b_{\mathcal{I}_{l-1}}=-1}^{1} \sum_{b_{\mathcal{I}_{l+1}}=-1}^{1} \cdots \sum_{b_{\mathcal{I}_r}=-1}^{1} (\max_{b_{\mathcal{I}_{r+1}},\cdots,b_{\mathcal{I}_{BN_t}}} \mu(b_1, \cdots, b_{i-1}, 1, b_{i+1}, \cdots, b_{BN_t}))}{\sum_{b_{\mathcal{I}_1}=-1}^{1} \cdots \sum_{b_{\mathcal{I}_{l-1}}=-1}^{1} \sum_{b_{\mathcal{I}_{l+1}}=-1}^{1} \cdots \sum_{b_{\mathcal{I}_r}=-1}^{1} (\max_{b_{\mathcal{I}_{r+1}},\cdots,b_{\mathcal{I}_{BN_t}}} \mu(b_1, \cdots, b_{i-1}, -1, b_{i+1}, \cdots, b_{BN_t}))}\right), \tag{3.33}$$

where the value of bit $b_{\mathcal{I}_i}$ is determined in the numerator and denominator and other bits are marginalized exactly if they belong to $\mathcal{I}$, otherwise they are marginalized approximately.

For LLR computation for bits which do not belong to $\mathcal{I}$, we have

$$\lambda_i^D \simeq \log\left(\frac{\sum_{b_{\mathcal{I}_1}=-1}^{1} \cdots \sum_{b_{\mathcal{I}_r}=-1}^{1} (\max_{b_{\mathcal{I}_{r+1}},\cdots,b_{\mathcal{I}_{l-1}},b_{\mathcal{I}_{l+1}}\cdots b_{\mathcal{I}_{BN_t}}} \mu(b_1, \cdots, b_{i-1}, 1, b_{i+1}, \cdots, b_{BN_t}))}{\sum_{b_{\mathcal{I}_1}=-1}^{1} \cdots \sum_{b_{\mathcal{I}_r}=-1}^{1} (\max_{b_{\mathcal{I}_{r+1}},\cdots,b_{\mathcal{I}_{l-1}},b_{\mathcal{I}_{l+1}}\cdots b_{\mathcal{I}_{BN_t}}} \mu(b_1, \cdots, b_{i-1}, -1, b_{i+1}, \cdots, b_{BN_t}))}\right). \tag{3.34}$$

According to bit index permutation $\mathcal{I}$, symbol index permutation, $\mathcal{J}$ on $[1, \cdots, N_t]$ can be defined. Since r shows the number of bits that are exactly marginalized, $p = \lfloor r/B \rfloor$ is the number of symbols with exact marginalization. So if $\lfloor r/B \rfloor$ is an integer, the channel matrix and received signal can be partitioned according to p such that

$$\mathbf{H}_a \triangleq [\mathbf{h}_{\mathcal{J}_1}, \cdots, \mathbf{h}_{\mathcal{J}_p}], \quad \mathbf{H}_b \triangleq [\mathbf{h}_{\mathcal{J}_{p+1}}, \cdots, \mathbf{h}_{\mathcal{J}_{N_t}}] \tag{3.35a}$$

$$\mathbf{s}_a \triangleq [\mathbf{s}_{\mathcal{J}_1}, \cdots, \mathbf{s}_{\mathcal{J}_p}]^T, \quad \mathbf{s}_b \triangleq [\mathbf{s}_{\mathcal{J}_{p+1}}, \cdots, \mathbf{s}_{\mathcal{J}_{N_t}}]^T \tag{3.35b}$$

Now (2.4) can be written as $\mathbf{y} = \mathbf{H}_a\mathbf{s}_a + \mathbf{H}_b\mathbf{s}_b + \mathbf{v}$ and the maxima in (3.33) can be computed by

$$\min_{\hat{\mathbf{s}}_b \in \mathcal{S}^{N_t - p}} \|\mathbf{y} - \mathbf{H}_a\mathbf{s}_a - \mathbf{H}_b\hat{\mathbf{s}}_b\|^2 \approx \|\mathbf{y} - \mathbf{H}_a\mathbf{s}_a - \mathbf{H}_b\hat{\mathbf{s}}_{b,ZF}\|^2, \tag{3.36}$$

where $\hat{\mathbf{s}}_{b,ZF}$ is the zero forcing estimate of $\mathbf{s}_b$ given $\mathbf{y}$ and $\mathbf{s}_a$,

$$\hat{\mathbf{s}}_{b,ZF} \triangleq \arg \min_{\hat{\mathbf{s}}_b \in \mathcal{S}^{n_t - p}} \|(\mathbf{H}_b^T\mathbf{H}_b)^{-1}\mathbf{H}_b^T(\mathbf{y} - \mathbf{H}_a\mathbf{s}_a) - \mathbf{s}_b\|^2. \tag{3.37}$$

Typically the sets $\mathcal{I}$ and $\mathcal{J}$ contain indices corresponding to the bits or symbols with low SNR. The original partial marginalization approach to soft MIMO detection has been extended to higher-order constellation and to cases where prior information is available (Persson and Larsson, 2011).

3.4.1 SUMIS

Another method based on partial marginalization has been proposed by Cirkic and Larsson (2012). This method is referred to as subspace marginalization with interference suppression (SUMIS). There are two stages in this method, the first stage is to approximate posterior probability for each bit and the second one is using approximated LLR for interference suppression. After interference cancellation, the LLR values are calculated based on the resulting model which does not include the interference. The partition model here is

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{v} = [\bar{\mathbf{H}} \quad \tilde{\mathbf{H}}][\bar{\mathbf{s}}^T \quad \tilde{\mathbf{s}}^T] + \mathbf{v} = \bar{\mathbf{H}}\bar{\mathbf{s}} + \underbrace{\tilde{\mathbf{H}}\tilde{\mathbf{s}} + \mathbf{v}}_{\text{interference+noise}} \tag{3.38}$$

where $\bar{\mathbf{s}}$ contains the i^{th} bit, s_i, in the original vector $\mathbf{s}$. The approximate model is defined as

$$\mathbf{y} = \bar{\mathbf{H}}\bar{\mathbf{s}} + \underbrace{\tilde{\mathbf{H}}\tilde{\mathbf{s}} + \mathbf{v}}_{\text{interference+noise}} \rightarrow \mathbf{y} \simeq \bar{\mathbf{y}} \triangleq \bar{\mathbf{H}}\bar{\mathbf{s}} + \tilde{\mathbf{v}}, \tag{3.39}$$

where $\tilde{\mathbf{v}}$ is a Gaussian stochastic vector $\mathcal{N}(0, \mathbf{Q})$ with $\mathbf{Q} = \tilde{\mathbf{H}}\tilde{\mathbf{\Psi}}\tilde{\mathbf{H}}^T + \frac{N_0}{2}\mathbf{I}$ and $\tilde{\mathbf{\Psi}}$ is the covariance matrix of $\tilde{\mathbf{s}}$. The main purpose of the first stage is to reduce the impact of interfering term $\tilde{\mathbf{H}}\tilde{\mathbf{s}}$. This interfering term is considered to be Gaussian as it consists a sum of variables that are independent.

In second stage the interfering vector $\tilde{\mathbf{s}}$ is suppressed and the model is purified as

$$\mathbf{y}' = \bar{\mathbf{H}}\bar{\mathbf{s}} + \mathbf{v}', \tag{3.40}$$

so the LLRs can be computed according to

$$\lambda_i^D = \log \frac{\sum_{\mathbf{s}\in\mathcal{X}_i^{(1)}} \exp(-\frac{1}{2}\|\mathbf{y}' - \bar{\mathbf{H}}\bar{\mathbf{s}}\|^2)}{\sum_{\mathbf{s}\in\mathcal{X}_i^{(-1)}} \exp(-\frac{1}{2}\|\mathbf{y}' - \bar{\mathbf{H}}\bar{\mathbf{s}}\|^2)}, \tag{3.41}$$

where a uniform *a priori* probability is assumed. The non-uniform *a priori* case was also considered by Cirkic and Larsson (2012). In that case, the definitions of $\tilde{\mathbf{\Psi}}$ and $\mathbf{Q}$ are different.

3.5 Semidefinite Programming

One other approach to reduce the complexity of soft MIMO demodulation is semidefinite relaxation (SDR). A simple SDR method from the hard decision class can be obtained by using the max-log approximation of (2.12), which results in

$$\lambda_i^D \simeq \frac{1}{2\sigma^2}\big(\min_{\mathbf{b}\in\hat{\mathcal{L}}_{i,-1}} D(\mathbf{b}) - \min_{\mathbf{b}\in\hat{\mathcal{L}}_{i,+1}} D(\mathbf{b})\big), \tag{3.42}$$

and then finding approximate solutions to the minimization problems using the semidefinite relaxation techniques.

That approach requires solving $2BN_t$ SDR problems each involving (BN_t-1) bits. Using the idea in (3.14), we actually only have to solve (BN_t+1) SDR problems, one involving BN_t bits, the others involving (BN_t-1) bits (Steingrimsson *et al.*, 2003). An advantage of this method is that each SDR problem can be solved in polynomial time.

The SDR techniques can also be used as a basis for list demodulation techniques (Nekuii *et al.*, 2011). Those techniques are the focus of this thesis and will be reviewed in the next chapter.

Chapter 4

Approaches to Soft Demodulation Based on Semidefinite Programming

As described in the previous chapters, one approach to soft demodulation involves employing the max-log approximation to compute an approximation of the LLRs. As it can be seen in (2.16b), when the max-log approximation is applied, the LLR of each bit is approximated by the difference between the optimal values of two hard demodulation problems (e.g., Steingrimsson *et al.*, 2003; Wang and Giannakis, 2006). One class of soft demodulators can then be constructed by (approximately) solving the hard demodulation problems. Another class is based on generating a list of “good” bit vectors and then finding the best solutions to the problems among the members of the list.

As discussed in Chapter 3, in the worst case, finding the optimal solution to the hard demodulation problems incurs a significant computational cost. Tree search methods, such as sphere decoding, can reshape the distribution of that computational cost (which is a function of the channel and noise realization), but these methods remain expensive in the worst case. In contrast, semidefinite relaxation offers the opportunity to obtain good approximations to the optimal value of the hard demodulation problem with a computational cost that is bounded by a low-order polynomial of the problem size. Initially, SDR was proposed for hard demodulation of symbols from binary phase-shift keying (BPSK) or QPSK constellation, (e.g., Tan and Rasmussen, 2001; Ma *et al.*, 2002), but it has been extended to M-PSK and general QAM constellations (e.g., Ma *et al.*, 2004; Sidiropoulos and Luo, 2006; Kisialiou *et al.*, 2009; Ma *et al.*, 2009).

In this chapter, we first describe semidefinite relaxation and explain its application to hard demodulation. Then, we will show how semidefinite relaxation can be applied in two classes of soft demodulation: the hard-demodulation-based soft demodulators and the list-based soft demodulators. In the case of list-based demodulators, we will describe two distinct applications of SDR. The resulting demodulators differ in the way in which the list is generated. The first demodulator solves a semidefinite program in each demodulation decoding iteration and that SDP incorporates the extrinsic information obtained from the previous iteration of the decoder. This demodulator generates the list for each iteration by applying the conventional randomization procedure to the solution of the SDP. The second demodulator generates the list based on an approximation of the randomization procedure that results in reduction in the number of semidefinite program problems to one per channel use, as distinct from one per demodulation decoding iteration.

4.1 System Model

The system model is the same as what we have in (2.4), a MIMO system with N_t transmit antennas and N_r receive antennas. The vector $\mathbf{y}$, the received signal is

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{v}, \tag{4.1}$$

where $\mathbf{s} \in \mathcal{C}^{N_t}$ is the vector of transmitted symbol (i.e., the mapped version of the output bits of the binary source) and $\mathbf{v}$ represents the additive white Gaussian noise. In some cases, and in particular for systems that employ rectangular QAM, it is sometimes convenient to construct a real-valued equivalent representation of (4.1), namely

$$\tilde{\mathbf{y}} = \tilde{\mathbf{H}}\tilde{\mathbf{s}} + \tilde{\mathbf{v}} \tag{4.2}$$

where

$$\tilde{\mathbf{y}} = \begin{bmatrix} \mathrm{Re}\{\mathbf{y}\} \\ \mathrm{Im}\{\mathbf{y}\} \end{bmatrix}; \quad \tilde{\mathbf{s}} = \begin{bmatrix} \mathrm{Re}\{\mathbf{s}\} \\ \mathrm{Im}\{\mathbf{s}\} \end{bmatrix}; \quad \tilde{\mathbf{v}} = \begin{bmatrix} \mathrm{Re}\{\mathbf{v}\} \\ \mathrm{Im}\{\mathbf{v}\} \end{bmatrix}$$

$$\text{and} \quad \tilde{\mathbf{H}} = \begin{bmatrix} \mathrm{Re}\{\mathbf{H}\} & -\mathrm{Im}\{\mathbf{H}\} \\ \mathrm{Im}\{\mathbf{H}\} & \mathrm{Re}\{\mathbf{H}\} \end{bmatrix}. \tag{4.3}$$

4.2 Hard Demodulation Using SDR

For a "per channel use" received signal model of the form in (4.1), the maximum-likelihood detector solves the optimization problem,

$$\min_{\tilde{s}_i \in \tilde{\mathcal{C}}} \quad \|\tilde{\mathbf{y}} - \tilde{\mathbf{H}}\tilde{\mathbf{s}}\|^2 = \min_{\tilde{s}_i \in \tilde{\mathcal{C}}} D(\tilde{\mathbf{s}}), \tag{4.4}$$

in which $\tilde{\mathcal{C}}$ is the set to which the (real-valued) symbols belong. The expression $D(\tilde{\mathbf{s}})$ can be thought of the $D(\tilde{\mathcal{M}}(\tilde{\mathbf{s}}))$ where $\tilde{\mathcal{M}}$ is a variant of the mapper $\mathcal{M}$ in which the real and imaginary parts are separated. For example, for a QAM constellation with $(2K)^2$ points $\tilde{\mathcal{C}}$ can be represented by $\tilde{\mathcal{C}} = \{\pm 1, \pm 3, \dots, \pm(2K-3), \pm(2K-1)\}$. In the case of 16-QAM, we would have $\tilde{\mathcal{C}} = \{\pm 1, \pm 3\}$.

If we define

$$\tilde{\mathbf{x}} = \begin{bmatrix} \tilde{\mathbf{s}} \\ 1 \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{Q}} = \begin{bmatrix} \tilde{\mathbf{H}}^T\tilde{\mathbf{H}} & -\tilde{\mathbf{H}}^T\tilde{\mathbf{y}} \\ -\tilde{\mathbf{y}}^T\tilde{\mathbf{H}} & 0 \end{bmatrix}, \tag{4.5}$$

the problem in (4.4) can be written as

$$\begin{aligned} \min_{\tilde{\mathbf{x}}} \quad & \tilde{\mathbf{x}}^T\tilde{\mathbf{Q}}\tilde{\mathbf{x}} && \text{(4.6a)} \\ \text{s.t.} \quad & \tilde{\mathbf{x}} = [\tilde{x}_1, \dots, \tilde{x}_{2N_t+1}]^T, && \text{(4.6b)} \\ & \tilde{x}_i \in \tilde{\mathcal{C}}, \quad i = 1, \dots, 2N_t, && \text{(4.6c)} \\ & \tilde{x}_{2Nt+1} = +1. && \text{(4.6d)} \end{aligned}$$

By considering $\mathbf{X} = \tilde{\mathbf{x}}\tilde{\mathbf{x}}^T$, the optimization problem can be written as

$$\begin{aligned} \min_{\mathbf{X}} \quad & \mathrm{Tr}(\mathbf{X}\tilde{\mathbf{Q}}) && \text{(4.7a)} \\ \text{s.t.} \quad & \mathbf{X} \succeq 0, && \text{(4.7b)} \\ & [\mathbf{X}]_{ii} \in \mathcal{B}, \quad i = 1, \dots, 2N_t && \text{(4.7c)} \\ & [\mathbf{X}]_{ii} = 1, \quad i = 2N_t + 1, && \text{(4.7d)} \\ & \mathrm{rank}(\mathbf{X}) = 1, && \text{(4.7e)} \end{aligned}$$

where $\mathcal{B} = \{1, 9, \ldots, (2K-3)^2, (2K-1)^2\}$. There are two difficulties that are encountered in trying to solve this problem, namely the constraint that $\mathbf{X}$ is rank-1 and the constraint that the diagonal elements of $\mathbf{X}$ must take on one of the discrete values in the set $\mathcal{B}$. There are different approaches to address the second difficulty. For a generic rectangular QAM constellation, Sidiropoulos and Luo (2006) replace this constraint by two inequalities that determine a lower and an upper bound for each element. The lower bound is $L = 1$ and the upper bound is $U = (2K-1)^2$. This relaxation of the constraint in (4.7c) results in the following optimization problem.

$$
\begin{aligned}
\min_{\mathbf{X}} \quad & \operatorname{Tr}(\mathbf{X}\tilde{\mathbf{Q}}) && \text{(4.8a)}\\
\text{s.t.} \quad & \mathbf{X} \succeq 0, && \text{(4.8b)}\\
& L \le [\mathbf{X}]_{ii} \le U, \quad i = 1, \ldots, 2N_t, && \text{(4.8c)}\\
& [\mathbf{X}]_{ii} = 1, \quad i = 2N_t + 1, && \text{(4.8d)}\\
& \operatorname{rank}(\mathbf{X}) = 1. && \text{(4.8e)}
\end{aligned}
$$

In the case of 16-QAM the constraint in (4.7c) can be represented precisely in a linear fashion, but doing so requires doubling the size of the matrix variable $\mathbf{X}$ (Wiesel *et al.*, 2005). However, in the case of hard demodulation the solutions of the semidefinite relaxations of both problems are equivalent (Ma *et al.*, 2009; Nekuii, 2008), and hence we will focus on the approach of (Sidiropoulos and Luo, 2006).

The difficulty that remains in solving (4.8) is the rank-1 constraint in (4.8e). The semidefinite relaxation approach to obtaining an approximate solution involves relaxing the rank-1 constraint and solving the following problem,

$$\min_{\mathbf{X}} \quad \mathrm{Tr}\big(\mathbf{X}\tilde{\mathbf{Q}}\big) \tag{4.9a}$$

$$\text{s.t.} \quad \mathbf{X} \succeq 0, \tag{4.9b}$$

$$L \leq [\mathbf{X}]_{ii} \leq U, \quad i = 1, \ldots, 2N_t, \tag{4.9c}$$

$$[\mathbf{X}]_{ii} = 1, \quad i = 2N_t + 1. \tag{4.9d}$$

After solving that problem and finding optimal solution, $\mathbf{X}_{opt}$, it should be checked if $\mathbf{X}_{opt}$ is rank-1 or not. If the solution is rank-1, $\mathbf{X}_{opt}$ is also the optimal solution of (4.8) and it will satisfy the equality $\mathbf{X}_{opt} = \tilde{\mathbf{x}}_{opt}\tilde{\mathbf{x}}_{opt}^T$, and hence an optimal solution to (4.6) is obtained. If the solution is not rank-1, a randomization procedure is used to generate good solutions to (4.6) from $\mathbf{X}_{opt}$. The randomization procedure uses Cholesky factor of the optimal solution, ($\mathbf{X}_{opt} = \mathbf{V}^T\mathbf{V}$), and a sequence of random vectors $\mathbf{u}$ from a uniform distribution on the unit hypersphere. For each vector $\mathbf{u}$ the vector $\bar{\mathbf{x}} = (\mathbf{V}^T\mathbf{u})$ is computed and then it is quantized to the constellation. So we have

$$\mathbf{x} = \varrho\Big(\frac{\mathbf{V}^T\mathbf{u}}{\mathbf{v}_{2N_t+1}^T\mathbf{u}}\Big), \tag{4.10}$$

where $\varrho(.)$ quantizes the elements of its vector argument to the constellation, $\tilde{\mathcal{C}}$ and $\mathbf{v}_{2N_t+1}$ is the right most column of the matrix $\mathbf{V}$. The expression $\mathbf{x}^T\tilde{\mathbf{Q}}\mathbf{x}$ is computed for each constructed vector, and if the value is smaller than the smallest value in previous steps then we set the current value for $\tilde{\mathbf{x}}$ to be equal to $\mathbf{x}$. The value for $\tilde{\mathbf{x}}$ that is obtained after the chosen number of randomizations has been completed is selected as the approximate solution to (4.6).

4.3 Soft Demodulation Using SDR

In soft demodulators, rather than solving the maximum likelihood detection problem in (4.4), we look at variations on the bit wise maximum *a posteriori* probability detection problem

$$\tilde{\mathbf{s}}_{\text{MAP}} = \arg\max_{\tilde{s}_i \in \tilde{\mathcal{C}}} p(\tilde{\mathbf{y}}|\tilde{\mathbf{s}})p(\tilde{\mathbf{s}}), \tag{4.11}$$

which can be written as

$$\tilde{\mathbf{s}}_{\text{MAP}} = \arg\min_{\tilde{s}_i \in \tilde{\mathcal{C}}} D(\tilde{\mathbf{s}}), \tag{4.12}$$

where, according to (2.13), we have

$$D(\tilde{\mathbf{s}}) \triangleq \|\mathbf{y} - \mathbf{H}\tilde{\mathbf{s}}\|^2 - 2\sigma^2 \log p(\tilde{\mathbf{s}}). \tag{4.13}$$

This problem can be solved using tree search algorithm in which $D(\tilde{\mathbf{s}})$ is considered as the metric. However, semidefinite relaxation cannot be applied to this problem, because $\log p(\tilde{\mathbf{s}})$ is a non-polynomial term and we cannot structure the problem as a semidefinite program. In order to use semidefinite relaxation, a polynomial approximation of $\log p(\tilde{\mathbf{s}})$ will need to be obtained.

Although there are various ways to do that (e.g., Nekuii, 2008), we will focus on approximations that will result in a problem with the same structure as (4.8). That means that we should approximate the values that $\log p(\tilde{\mathbf{s}})$ takes on in such a way that we can approximate the problem of minimizing $D(\tilde{\mathbf{s}})$ by a problem that involves minimizing a quadratic form $\tilde{\mathbf{x}}^T\tilde{\mathbf{Q}}\tilde{\mathbf{x}}$. That way we can solve the approximated problem using semidefinite programming. Hence, we seek to approximate the *a priori* term with a second order polynomial. The approximation described below also exploits the assumed independence of the elements of $\tilde{\mathbf{s}}$.

Second Order Polynomial Approximation of $\log p(\tilde{\mathbf{s}})$

As long as there are more than three possible values for $\tilde{s}_i$ to take, we cannot precisely express the log function in terms of a second order polynomial of $\tilde{s}_i$. Instead, we take advantage of the assumed independence of the elements of $\tilde{\mathbf{s}}$ and for the i^{th} element we choose $a_i\tilde{s}_i^2 + b_i\tilde{s}_i + d_i$ as the approximation polynomial. We select the coefficients by minimizing the sum of squared errors between $\log p_i(\tilde{s}_i)$ and $a_i\tilde{s}_i^2 + b_i\tilde{s}_i + c_i$ for all possible values of $\tilde{s}_i$,

$$\min_{a_i,b_i,c_i} \sum_{k=1}^{|\tilde{\mathcal{C}}|} \left|\log p_i(\tilde{s}_i = s^k) - (a_i(s^k)^2 + b_i(s^k) + c_i)\right|^2, \tag{4.14}$$

where i denotes the i^{th} symbol of the symbol-vector and k denotes the k^{th} possible value in the constellation set $\tilde{\mathcal{C}}$, which is of size $|\tilde{\mathcal{C}}|$. In order to minimize the function with respect to a_i, its derivative with respect to a_i should be set to zero. That is

$$\frac{\partial}{\partial a_i} \sum_{k=1}^{|\tilde{\mathcal{C}}|} \left|\log p_i(\tilde{s}_i = s^k) - (a_i(s^k)^2 + b_i(s^k) + c_i)\right|^2 = 0 \tag{4.15a}$$

$$\Rightarrow \sum_{k=1}^{|\tilde{\mathcal{C}}|} \left(2(s^k)^2(a_i(s^k)^2 + b_i(s^k) + c_i) - 2(s^k)^2 \log p_i(\tilde{s}_i = s^k)\right) = 0 \tag{4.15b}$$

$$\Rightarrow a_i(\sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k)^4) + b_i(\sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k)^3) + c_i(\sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k)^2) = \sum_{k=1}^{|\tilde{\mathcal{C}}|} (s^k)^2 \log p_i(\tilde{s}_i = s^k). \tag{4.15c}$$

The same procedure can be performed with respect to b_i and c_i and the result can be shown in a matrix format, namely

$$\begin{bmatrix} \sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k)^4 & \sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k)^3 & \sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k)^2 \\ \sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k)^3 & \sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k)^2 & \sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k) \\ \sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k)^2 & \sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k) & |\tilde{\mathcal{C}}| \end{bmatrix} \begin{bmatrix} a_i \\ b_i \\ c_i \end{bmatrix} = \begin{bmatrix} \sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k)^2 \log p_i(\tilde{s}_i = s^k) \\ \sum_{k=1}^{|\tilde{\mathcal{C}}|}(s^k) \log p_i(\tilde{s}_i = s^k) \\ \sum_{k=1}^{|\tilde{\mathcal{C}}|} \log p_i(\tilde{s}_i = s^k) \end{bmatrix}. \quad (4.16)$$

So if we define vectors $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ with elements a_i, b_i and c_i satisfying (4.16), respectively, we can approximate $\log p(\tilde{\mathbf{s}})$ with $\tilde{\mathbf{s}}^T \operatorname{Diag}(\mathbf{a})\tilde{\mathbf{s}} + \mathbf{b}^T\tilde{\mathbf{s}} + \mathbf{c}^T\mathbf{1}$ and so

$$D(\tilde{\mathbf{s}}) \simeq \hat{D}(\tilde{\mathbf{s}}) = \|\tilde{\mathbf{y}} - \tilde{\mathbf{H}}\tilde{\mathbf{s}}\|_2^2 - 2\sigma^2(\tilde{\mathbf{s}}^T \operatorname{Diag}(\mathbf{a})\tilde{\mathbf{s}} + \mathbf{b}^T\tilde{\mathbf{s}} + \mathbf{c}^T\mathbf{1}), \quad (4.17)$$

where $\operatorname{Diag}(\mathbf{a})$ is the diagonal matrix with the vector $\mathbf{a}$ on the diagonal.

Now we can write the approximated $D(\tilde{\mathbf{s}})$ as a quadratic form. In particular, if we define the vector $\tilde{\mathbf{x}}$ to be $\tilde{\mathbf{x}}^T = [\tilde{\mathbf{s}}^T \quad 1]$, the problem of minimizing $D(\tilde{\mathbf{s}})$ is equivalent to minimizing $\tilde{\mathbf{x}}^T\tilde{\mathbf{Q}}\tilde{\mathbf{x}}$ where

$$\tilde{\mathbf{Q}} = \begin{bmatrix} \tilde{\mathbf{H}}^T\tilde{\mathbf{H}} - 2\sigma^2\operatorname{Diag}(\mathbf{a}) & -\tilde{\mathbf{H}}^T\tilde{\mathbf{y}} - 2\sigma^2\mathbf{b} \\ -\tilde{\mathbf{y}}^T\tilde{\mathbf{H}} - 2\sigma^2\mathbf{b} & 0 \end{bmatrix}. \quad (4.18)$$

Therefore, the SDR problem can be written as

$$\min_{\mathbf{X}} \quad \operatorname{Tr}\big(\mathbf{X}\tilde{\mathbf{Q}}\big) \quad (4.19a)$$

$$\text{s.t.} \quad \mathbf{X} \succeq 0, \quad (4.19b)$$

$$L \le [\mathbf{X}]_{ii} \le U, \quad i = 1, \ldots, 2N_t, \quad (4.19c)$$

$$[\mathbf{X}]_{ii} = 1, \quad i = 2N_t + 1. \quad (4.19d)$$

It can be seen that the difference between the $\tilde{\mathbf{Q}}$ in (4.5) and the one in (4.18) is that the *a priori* information is contained in the structure of $\tilde{\mathbf{Q}}$ in (4.18).

In Chapter 2, we described different classes of soft demodulators, one class is based on solving two hard demodulations for each bit and another class is based on generating a list instead of checking all bits. In next sections, we will describe SDR approached to both classes.

4.3.1 Soft Demodulators Based on Hard Demodulation

We have shown that in soft demodulation based on semidefinite relaxation, we can use max-log approximation to approximate the LLRs of transmitted bits,

$$\lambda_i^D \simeq \log \frac{\sum_{\mathcal{L}_{i,+1}} \exp(-\hat{D}(\tilde{\mathbf{s}})/(2\sigma^2))}{\sum_{\mathcal{L}_{i,-1}} \exp(-\hat{D}(\tilde{\mathbf{s}})/(2\sigma^2))} \tag{4.20a}$$

$$\simeq \frac{1}{2\sigma^2}\big(\min_{\mathcal{L}_{i,-1}} \hat{D}(\tilde{\mathbf{s}}) - \min_{\mathcal{L}_{i,+1}} \hat{D}(\tilde{\mathbf{s}})\big), \tag{4.20b}$$

where $\hat{D}(\tilde{\mathbf{s}})$ is computed as (4.17).

In demodulators that are based on hard-demodulation, in each iteration of exchanging soft information between demodulator and decoder, based on *a priori* information provided by decoder, two hard demodulations are solved for each bit and the smallest value of $\hat{D}(\tilde{\mathbf{s}})$ is reached after checking all bits. These hard demodulation problems can be solved using SDR (Steingrimsson *et al.*, 2003). Then, the updated soft information is sent to the decoder to be processed for next iteration.

4.3.2 Soft Demodulators Based on List Generation

Another class of soft demodulators is based on list generation. In this class a list, $\hat{\mathcal{L}}$, is generated and the LLR is approximated using

$$\lambda_i^D \simeq \log \frac{\sum_{\hat{\mathcal{L}}_{i,+1}} \exp(-\hat{D}(\tilde{\mathbf{s}})/(2\sigma^2))}{\sum_{\hat{\mathcal{L}}_{i,-1}} \exp(-\hat{D}(\tilde{\mathbf{s}})/(2\sigma^2))} \tag{4.21a}$$

$$\simeq \frac{1}{2\sigma^2}\big(\min_{\hat{\mathcal{L}}_{i,-1}} \hat{D}(\tilde{\mathbf{s}}) - \min_{\hat{\mathcal{L}}_{i,+1}} \hat{D}(\tilde{\mathbf{s}})\big). \tag{4.21b}$$

Nekuii *et al.* (2011) introduced two semidefinite relaxation demodulators which are list-based. The way the list is generated is different in these demodulators. In the first demodulator, one semidefinite program is solved in each demodulation decoding iteration and the list is generated using the randomization procedure. This demodulator is called the “list-SDR” demodulator. However, in the second demodulator, by using an analysis of the randomization procedure, and making a prudent approximation, the number of SDPs to be solved can be reduced to one per channel use. This kind of demodulator is called the “single-SDR” demodulator. In the following sections, we will explain both demodulators in detail.

List-SDR Method

As it was shown in Section 4.2, in the case that the optimal solution of (4.19) is not rank-1, a randomization procedure is carried to find a solution for $\tilde{\mathbf{x}}$ in (4.6). The vector generated in randomization procedure often results in small values for $\hat{D}(\tilde{\mathbf{s}})$ which suggests that the generated symbol-vectors are good candidates for being a list member. Based on this observation, the initial members of the list in (4.21b) are the bit vectors that arise from randomization procedure. The list is then enriched by ensuring that the list contains all bit vectors that are one “bit flip” away from an initial member of the list. As in the tree searching method, bit-flipping in list generation is important to LLR computation, because for each i we should ensure that the list contains at least one vector for which $b_i = 1$ and one for which $b_i = -1$. At each demodulation decoding iteration, the soft information provided by decoder is updated and the list-SDR demodulator updates its list by solving the SDP with the updated prior information and applying bit flipping. As such, in the list-SDR approach, a semidefinite program should be solved at each demodulation decoding iteration.

Single-SDR Method

Solving one SDP per demodulation decoding iteration to generate the list incurs considerable computational cost. If we can reduce the number of SDPs that should be solved, the computational cost will be decreased. The approach that was taken in the development of the single-SDR demodulator, which was developed for the case of QPSK modulation by Nekuii *et al.* (2011), was based on deriving an approximation for the probability that the randomization procedure generate each symbol. That expression is then used to approximate the randomization procedure by independently generating each symbol using the obtained probability mass function. That probability mass function can be updated in a straight forward way in response to an update in the prior information from the previous iteration of the decoder. Therefore, only one SDP needs to be solved in each channel use.

We now explain how to derive the probability that each randomization output symbol takes each symbol value in $\tilde{\mathcal{C}}$. Let $\mathbf{X}_{opt} = \mathbf{V}^T\mathbf{V}$ be the Cholesky factor of optimal solution of SDR and suppose that $\mathbf{v}_i$ is the i^{th} column of $\mathbf{V}$. Then $\tilde{s}_i$ can be written as

$$\tilde{s}_i = \varrho\Big(\frac{\mathbf{v}_i^T\mathbf{u}}{\mathbf{v}_{2N_t+1}^T\mathbf{u}}\Big), \tag{4.22}$$

where $\mathbf{u}$ is a random vector from a uniform distribution on the unit sphere and $\varrho(.)$ denotes the quantization process.

In order to compute the probabilities, the regions that the symbols may belong to, should be partitioned and each partition results in one specific value for the symbol. These Voronoi regions are defined according to the constellation and the distance between each real symbol. In order to find the probability of each randomized symbol, we should find the probability that each symbol is placed in each defined region. In (square) QAM constellations with $2^B = (2K)^2$ points, $\tilde{\mathcal{C}} = \{\pm 1, \ldots, \pm 2K-1\}$, and these regions can be determined by $-2(K-n) \leq \tilde{s}_i \leq -2(K-n-1)$, $\forall n = 1, \ldots, 2K-1$.

We know that the inner product of two vectors can be written as $\mathbf{v}^T\mathbf{u} = \|\mathbf{v}\|\|\mathbf{u}\|\cos(\theta)$, where θ is the angle between two vectors. In the randomization procedure, the vector $\mathbf{u}$ is distributed uniformly over the unit sphere, so $\|\mathbf{u}\| = 1$. We also know that $\|\mathbf{v}_{2N_t+1}\| = 1$, (according to (4.9d)) so the probabilities depend on norm of $\|\mathbf{v}_i\|$ and the angles between $\mathbf{v}_i$ and $\mathbf{u}$ and between $\mathbf{v}_{2N_t+1}$ and $\mathbf{u}$. Let θ_i be the angle between $\mathbf{v}_{2N_t+1}$ and $\mathbf{v}_i$ and ϕ be the angel between $\mathbf{v}_{2N_t+1}$, and $\mathbf{u}$. Using (4.22) and considering the regions of the constellation we can derive the probabilities as follows

$$\begin{aligned} & -2(K-n) \leq \frac{\mathbf{v}_i^T \mathbf{u}}{\mathbf{v}_{2N_t+1}^T \mathbf{u}} \leq -2(K-n-1) \qquad (4.23) \\ & \Leftrightarrow -2(K-n) \leq \|\mathbf{v}_i\| \frac{\big(\cos(\theta_i)\cos(\phi) + \sin(\theta_i)\sin(\phi)\big)}{\cos(\phi)} \leq 2(K-n-1) \\ & \Leftrightarrow \frac{-2(K-n) - \|\mathbf{v}_i\|(\cos(\theta_i))}{\|\mathbf{v}_i\| \sin(\theta_i)} \leq \tan(\phi) \leq \frac{-2(K-n-1) - \|\mathbf{v}_i\|(\cos(\theta_i))}{\|\mathbf{v}_i\| \sin(\theta_i)}. \end{aligned}$$

Let us define $\{\gamma_{i,n}\}_{n=1}^{2K-1}$ as

$$\gamma_{i,n} = \tan^{-1}\Big(\frac{-2(K-n) - \|\mathbf{v}_i\|(\cos(\theta_i))}{\|\mathbf{v}_i\| \sin(\theta_i)}\Big). \qquad (4.24)$$

Then for $1 \leq n \leq 2K-2$, from (4.23) and (4.24) we have

$$\gamma_{i,n} \leq \phi \leq \gamma_{i,n+1}, \qquad (4.25)$$

and the desired probabilities can be written as

$$p\big(\tilde{s}_i = -2(K-n)+1\big) = \frac{\gamma_{i,n+1} - \gamma_{i,n}}{\pi}, \quad 1 \leq n \leq 2K-2. \qquad (4.26)$$

For the first and the last point of the (real-valued) constellation, we have

$$\tilde{s}_i = 2K-1 \Rightarrow \gamma_{i,2K-1} \leq \phi \leq \pi/2, \qquad (4.27)$$

$$\tilde{s}_i = -(2K-1) \Rightarrow -\pi/2 \leq \phi \leq \gamma_{i,1}. \qquad (4.28)$$

Hence, the probabilities can be written as

$$p(\tilde{s}_i = 2K-1) = \frac{\pi/2 - \gamma_{i,2K-1}}{\pi}, \tag{4.29}$$

$$p(\tilde{s}_i = -2K+1) = \frac{\gamma_{i,1} + \pi/2}{\pi}. \tag{4.30}$$

In summary, we have

$$p(\tilde{s}_i = -2K+1) = \frac{\gamma_{i,1} + \pi/2}{\pi}, \tag{4.31a}$$

$$p\big(\tilde{s}_i = -2(K-n)+1\big) = \frac{\gamma_{i,n+1} - \gamma_{i,n}}{\pi}, \quad n = 1, \dots, 2K-2, \tag{4.31b}$$

$$p(\tilde{s}_i = 2K-1) = \frac{\pi/2 - \gamma_{i,2K-1}}{\pi}. \tag{4.31c}$$

Now that we have obtained expressions for the probabilities with which the standard randomization procedure would generate each symbol, we can approximate that randomization procedure by independently generating realizations for each symbol from the probability mass function in (4.31). As we will explain below, this approximation that each symbol is independent not only saves the matrix-vector multiplication that is implicit in each step of the standard randomization procedure, it also enables us to decouple the processing of the channel output from the processing of the prior information, and hence allows us to reduce the number of SDPs to be solved from one per demodulation decoding iteration to one per channel use.

In the first demodulation iteration, the approximation of the standard randomization procedure described above involves generating random symbol vectors by randomly generating each symbol independently, from the distribution described in

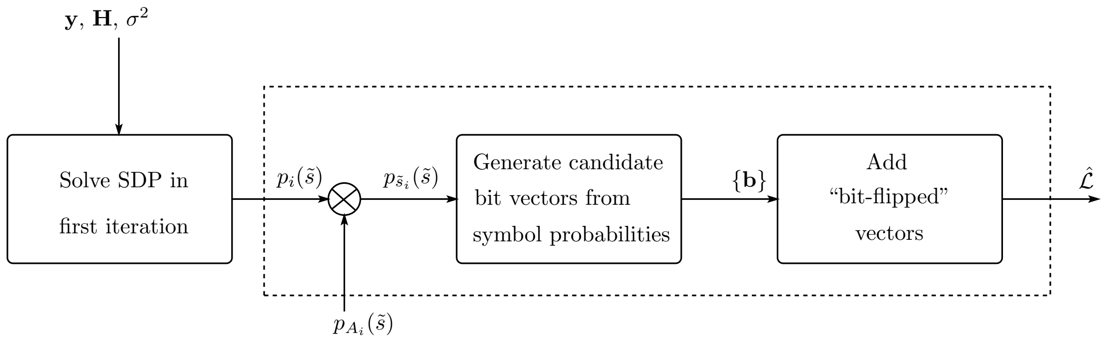
$\mathbf{y}$, $\mathbf{H}$, σ^2
Solve SDP in first iteration
$p_i(\tilde{s})$
$p_{\tilde{s}_i}(\tilde{s})$
Generate candidate bit vectors from symbol probabilities
$\{\mathbf{b}\}$
Add "bit-flipped" vectors
$\hat{\mathcal{L}}$
$p_{A_i}(\tilde{s})$

Figure 4.1: List generation scheme using Single-SDR. The activities in the dashed box are performed at each demodulation iteration.

(4.31). In the second demodulation iteration, we seek to exploit the prior information provided by the decoder. Based on the assumption of ideal interleaving, that information from the decoder is independent for each symbol that we are seeking to demodulate. It is also, by construction (Hagenauer, 1997), independent from the information provided by the channel. Therefore, at the second and subsequent demodulation iterations, rather than solving a new SDP that incorporates the soft information, these approximations allow us to generate an updated list simply by modifying the probability mass function for each symbol to

$$p_{\tilde{s}_i}(\tilde{s}_i = \bar{s}) = \kappa_i p_i(\tilde{s}_i = \bar{s}) p_{A_i}(\tilde{s}_i = \bar{s}), \tag{4.32}$$

where $\bar{s} \in \tilde{\mathcal{C}}$, $p_i(\tilde{s}_i = \bar{s})$ is obtained using (4.31) in the first demodulation iteration, and $p_{A_i}(\tilde{s}_i)$ is the probability mass function for the i^{th} symbol, which is computed using the corresponding bit probabilities provided by the previous iteration of the decoder. The constant κ_i is computed such that $\sum_{\bar{\mathbf{s}}\in\tilde{\mathcal{C}}} p_{\tilde{s}_i}(\tilde{s}_i = \bar{s}) = 1$. A block-diagram representation of the process is provided in Fig. 4.1.

At each demodulation decoding iteration, single-SDR generates M symbol-vectors using scalar random number generator with the computed probabilities in (4.32). The same as list-SDR case, in single-SDR case the generated list is enriched by adding all the single bit-flipping of the list. An advantage of the single-SDR approach over schemes that generate lists of candidate bit-vectors in each demodulation decoding iterations, such as the tree search scheme of Hochwald and ten Brink (2003), is that we do not need to store the whole list, we need only store the probabilities computed in (4.31). These probabilities and the updated information that decoder sends to the demodulator at each iteration are used to generate a new list.

4.4 Interior Point Method to Solve SDP

In the previous sections, we explained semidefinite relaxation and we showed how the SDP can be used in soft demodulation. In this section, we will discuss two methods by which the SDP can be solved.

Many of the methods that have been proposed for solving semidefinite programs fall into the class of interior point methods; e.g., (Boyd and Vandenberghe, 2004). Three classes of interior point methods are the primal-scaling methods, in which the primal variables are updated at each iteration, dual-scaling methods in which the dual variables or primal value are updated, and primal-dual methods in which both the primal and dual variables are updated at each iteration. Based on the insights of (Helmberg *et al.*, 1996), Nekuii (2008) developed a customized primal-dual algorithm for the problem in (4.19). That algorithm is described below.

4.4.1 Customized Primal-Dual Interior Point Method

The customized primal-dual interior point method derived by Nekuii (2008) was obtained using the following steps: First, the problem in (4.19) is written as

$$\begin{aligned} \min_{\mathbf{X}} \quad & \mathrm{Tr}\big(\mathbf{X}\tilde{\mathbf{Q}}\big) && \text{(4.33a)}\\ \text{s.t.} \quad & L \le \mathrm{diag}(\mathbf{X}_{11}) \le U, && \text{(4.33b)}\\ & x_{22} = 1, && \text{(4.33c)}\\ & \mathbf{X} \succeq 0, && \text{(4.33d)} \end{aligned}$$

where

$$\mathbf{X} \triangleq \begin{bmatrix} \mathbf{X}_{11} & \mathbf{x}_{12} \\ \mathbf{x}_{21} & x_{22} \end{bmatrix}. \tag{4.34}$$

Then, the dual variables are defined as $\mathbf{p}_u \in \mathbb{R}^{2N_t}$, $\mathbf{p}_\ell \in \mathbb{R}^{2N_t}$, $v \in \mathbb{R}$ and $\mathbf{Z} \in \mathbb{S}_{2N_t+1}$, where $\mathbb{S}^n$ is the set of symmetric $n \times n$ matrices. The dual problem of (4.33) is

$$\begin{aligned} \max_{\mathbf{p}_u,\mathbf{p}_\ell,v,\mathbf{Z}} \quad & U\mathbf{1}^T\mathbf{p}_u - L\mathbf{1}^T\mathbf{p}_\ell + v && \text{(4.35a)}\\ \text{s.t.} \quad & \tilde{\mathbf{Q}} - \mathrm{Diag}([\mathbf{p}_\ell^T - \mathbf{p}_u^T, -v]^T) = \mathbf{Z}, && \text{(4.35b)}\\ & \mathbf{p}_u \ge 0, \mathbf{p}_\ell \ge 0, \mathbf{Z} \succeq 0. && \text{(4.35c)} \end{aligned}$$

The primal-dual method is based on solving the following perturbed version of the KKT conditions,

$$\mathbf{Z} - \tilde{\mathbf{Q}} + \mathrm{Diag}\{[\mathbf{p}_\ell^T - \mathbf{p}_u^T, -v^T]\} = \mathbf{0} \tag{4.36a}$$

$$1 - x_{22} = 0 \tag{4.36b}$$

$$U\mathbf{1} - \mu\mathbf{p}_u^{-1} - \mathrm{diag}(\mathbf{X}_{11}) = 0 \tag{4.36c}$$

$$\mathrm{diag}(\mathbf{X}_{11}) - L\mathbf{1} - \mu\mathbf{p}_\ell^{-1} = 0 \tag{4.36d}$$

$$\mathbf{Z}\mathbf{X} - \mu\mathbf{I} = 0, \tag{4.36e}$$

where as $\mu > 0$ denotes the perturbation, and is proportional to the duality gap.

At each step of the algorithm, $\mathbf{X}$, $\mathbf{p}_u$, $\mathbf{p}_\ell$, v, $\mathbf{Z}$ are updated in a way that decreases μ and hence the iterations approach values that satisfy the optimality conditions in (4.19). The algorithm is terminated once the duality gap falls below a specified tolerance. Nekuii (2008, Table 5.1) provided an explicit statement of how the generic update algorithm of (Helmberg *et al.*, 1996) could be customized to the problem at hand. In each iteration of that algorithm the update can be computed using a number of operations that grows at most cubically in N_t. In particular, the dominant operations are an inversion and a Cholesky factorization of matrices of size $(2N_t + 1) \times (2N_t + 1)$, and finding the solution of a set of $4N_t + 3$ linear equations.

4.4.2 Dual Scaling Interior Point Method

In the primal-dual interior point method, both the primal and dual variables are updated at each step. In this thesis we develop a customized dual-scaling interior point method in which at each step either the dual variables or the primal value are updated. The dual scaling approach exposes some of the structure of the underlying problem, and by exploiting this structure we obtain an algorithm that requires only about half the computational effort in our intended application.

Chapter 5

A Dual-Scaling Algorithm for SDR-Based Soft Demodulation

In the previous chapter, we described semidefinite relaxation and the ways in which it can be used for soft MIMO demodulation, and we described a primal-dual interior point method to solve the structured semidefinite optimization problem that arises in that application. In this chapter, we will develop an alternative method to solve that problem— a customized dual-scaling interior point method. Unlike primal-dual methods, in which both the primal and dual variables are updated at each iteration, in dual-scaling methods either the dual variables or the value of primal function are updated at each iteration. By carefully exploiting this difference, we are able to construct an implementation with lower computational cost.

The focus of this chapter is on the development of a customized dual-scaling algorithm for the structured SDP in (4.9). In order to place that development in context, we will first describe each step of the dual-scaling algorithm for the generic case and then we will explain the way in which those steps are taken in the customized dual-scaling algorithm.

5.1 Formulation of Dual-Scaling Algorithm for SDP

5.1.1 Generic Formulation

Let us consider the class of semidefinite programming problems of the form

$$\min_{\mathbf{X}} \quad \mathrm{Tr}(\mathbf{C}\mathbf{X}) \tag{5.1a}$$

$$\text{s.t} \quad \mathrm{Tr}(\mathbf{A}_i\mathbf{X}) = b_i, \quad i = 1, \ldots, m, \tag{5.1b}$$

$$\mathbf{X} \succeq 0, \tag{5.1c}$$

where $\mathbf{A}_i, \mathbf{C} \in \mathbb{S}^n$, the set of symmetric matrices of size $n \times n$. Many semidefinite relaxations of quadratic optimization problems can be written this way (Kisialiou *et al.*, 2009). In Appendix A, we will show that the dual of problem (5.1) can be written as

$$\max_{\mathbf{y}} \quad \mathbf{b}^T\mathbf{y} \tag{5.2a}$$

$$\text{s.t} \quad \sum_{i=1}^{m} y_i\mathbf{A}_i + \mathbf{S} = \mathbf{C}, \quad i = 1, \ldots, m, \tag{5.2b}$$

$$\mathbf{S} \succeq 0, \tag{5.2c}$$

where $\mathbf{y}$ and $\mathbf{S}$ are the dual variables and $\mathbf{b} = [b_1, b_2, \ldots, b_m]^T$.

As explained in Section 4.4, problems of the form in (5.1) can be tackled in variety of ways (e.g., Boyd and Vandenberghe, 2004; Ye, 2011), including primal potential methods, primal dual methods, and dual scaling methods. The relative computational costs of those methods are dependent on the features of the matrices $\mathbf{A}_i, \mathbf{C}$ and the vector $\mathbf{b}$, the extent to which the algorithm exposes these features, and the extent to which the implementation exploits these features. In many applications, the structure of the SDPs that arise from semidefinite relaxation of quadratic problems is amenable to dual-scaling algorithms and hence we will pursue the development of a customized version of such an algorithm that exploits the structure of the SDP of interest, c.f. (4.19).

5.1.2 Structured Formulation of (4.19)

According to (4.19), the optimization problem that should be solved is

$$\begin{aligned} \min_{\mathbf{X}} \quad & \operatorname{Tr}\big(\tilde{\mathbf{Q}}\mathbf{X}\big) && (5.3a)\\ \text{s.t.} \quad & \mathbf{X} \succeq 0, && (5.3b)\\ & L \leq [\mathbf{X}]_{ii} \leq U, \quad i = 1, \dots, 2N_t, && (5.3c)\\ & [\mathbf{X}]_{2N_t+1, 2N_t+1} = 1. && (5.3d) \end{aligned}$$

In order to develop a convenient dual formulation, we first reformulate the primal problem in (5.3) so that it only involves equality constraints; c.f. (5.1). To do so we add non-negative slack variables u_i and t_i for the lower and upper bound inequality constraints in (5.3c). That is, we replace (5.3c) by

$$\begin{aligned} & [\mathbf{X}]_{ii} - u_i = L, \quad u_i \geq 0, \quad i = 1, \dots, 2N_t, && (5.4a)\\ & [\mathbf{X}]_{ii} + t_i = U, \quad t_i \geq 0, \quad i = 1, \dots, 2N_t. && (5.4b) \end{aligned}$$

In order to write the problem in the form in (5.1), we define the block structured matrix $\tilde{\mathbf{X}} \in \mathbb{R}^{(6N_t+1)\times(6N_t+1)}$ as

$$\tilde{\mathbf{X}} = \begin{bmatrix} \mathbf{X} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{U} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{T} \end{bmatrix}, \tag{5.5}$$

where $\mathbf{U}$ and $\mathbf{T}$ are diagonal matrices of size $2N_t$ whose i^{th} diagonal elements are u_i and t_i, respectively. Since $\tilde{\mathbf{X}}$ is block diagonal, the constraints that $\mathbf{X}$ is positive semidefinite and that u_i and t_i are non-negative are equivalent to $\tilde{\mathbf{X}}$ being positive semidefinite. (That said, our algorithm will be based on exploiting the structure of (5.5).) In order to write the objective as a function of $\tilde{\mathbf{X}}$, we observe that if we define $\mathbf{C} \in \mathbb{R}^{(6N_t+1)\times(6N_t+1)}$ to be

$$\mathbf{C} = \begin{bmatrix} \tilde{\mathbf{Q}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}, \tag{5.6}$$

then $\mathrm{Tr}(\mathbf{C}\tilde{\mathbf{X}}) = \mathrm{Tr}(\tilde{\mathbf{Q}}\mathbf{X})$.

By performing analogous operations on the constraints, the optimization problem in (5.3) can be written in the form of (5.1) as

$$\min_{\tilde{\mathbf{X}}} \quad \mathrm{Tr}(\mathbf{C}\tilde{\mathbf{X}}) \tag{5.7a}$$

$$\text{s.t.} \quad \mathrm{Tr}(\mathbf{A}_i\tilde{\mathbf{X}}) = e_i, \quad i = 1, \ldots, 4N_t + 1, \tag{5.7b}$$

$$\tilde{\mathbf{X}} \succeq 0, \tag{5.7c}$$

where the matrices $\mathbf{A}_i \in \mathbb{R}^{(6N_t+1)\times(6N_t+1)}$, which are diagonal matrices with elements of $\{0, +1, -1\}$, and the scalars e_i are chosen to represent (5.4a), (5.4b) and (5.3d) in the form of (5.7b). In particular, if we let $\mathbf{E}_i$ denote the $2N_t \times 2N_t$ matrix with all elements equal to zero except for the i^{th} diagonal element which is equal to one, then we can represent (5.4a) in the form of (5.7b) using

$$\mathbf{A}_i = \begin{bmatrix} \mathbf{E}_i & 0 & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 0 & 0 \\ \mathbf{0} & 0 & -\mathbf{E}_i & \mathbf{0} \\ \mathbf{0} & 0 & \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad i = 1, \dots, 2N_t, \tag{5.8}$$

and $e_i = 1$. Analogously, we can represent (5.4b) in the form of (5.7b) using

$$\mathbf{A}_i = \begin{bmatrix} \mathbf{E}_{i-2N_t} & 0 & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 0 & 0 \\ \mathbf{0} & 0 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 0 & \mathbf{0} & \mathbf{E}_{i-2N_t} \end{bmatrix}, \quad i = 2N_t + 1, \dots, 4N_t, \tag{5.9}$$

and $e_i = 9$. Finally, we can represent (5.3d) in the form of (5.7b) using

$$\mathbf{A}_{4N_t+1} = \begin{bmatrix} \mathbf{0} & 0 & \mathbf{0} & \mathbf{0} \\ 0 & 1 & 0 & 0 \\ \mathbf{0} & 0 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 0 & \mathbf{0} & \mathbf{0} \end{bmatrix}, \tag{5.10}$$

and $e_{4N_t+1} = 1$. With these definitions in place, the dual of (5.7) takes the generic form in (5.2). In Appendix B, we explicitly connect the dual variables of that formulation, namely $\mathbf{y}$ and $\mathbf{S}$, to the variables of the dual of the formulation in (4.33), namely, $\mathbf{p}_u$, $\mathbf{p}_\ell$, and ν; cf. (4.35).

It can be seen that the matrices $\tilde{\mathbf{X}}$, $\mathbf{C}$ and $\mathbf{A}_i$ have sparse structure. This sparsity will be exploited in the proposed customized dual-scaling algorithm.

5.2 Principles of Dual-Scaling Algorithm

In general, the optimal value of the dual problem is a lower bound for the primal optimal value. If $g(\mathbf{y}, \mathbf{S})$ denotes the dual objective function, then for any feasible $\tilde{\mathbf{X}}$ and any feasible $\mathbf{y}, \mathbf{S}$, we have $g(\mathbf{y}, \mathbf{S}) \leq f(\tilde{\mathbf{X}})$. As this holds for any feasible $\tilde{\mathbf{X}}$, $\mathbf{y}$ and $\mathbf{S}$, it also holds for the optimal values, which we denote by a star, i.e., $g(\mathbf{y}^\star, \mathbf{S}^\star) \leq f(\tilde{\mathbf{X}}^\star)$. The difference $f(\tilde{\mathbf{X}}) - g(\mathbf{y}, \mathbf{S})$ is called the duality gap.

There are conditions under which this inequality turns to equality, and the optimal dual value is equal to optimal primal value; a condition known as strong duality. Slater's condition is one of the conditions that is sufficient for strong duality (Boyd and Vandenberghe, 2004, pages 226-227). For the problems in the form (5.1), if there exists a strictly positive semidefinite $\mathbf{X}$, i.e., $\mathbf{X} \succ 0$, such that $\mathrm{Tr}(\mathbf{A}_i\mathbf{X}) = b_i$ for $i = 1, \ldots, m$, then Slater's condition is satisfied and strong duality holds. For an optimization problem with a differentiable objective and constraint functions for which strong duality is satisfied, the primal and dual optimal points satisfy conditions known as the "KKT conditions" (Boyd and Vandenberghe, 2004, pages 243-244). Based on these conditions, once a solution to the dual problem has been found, a solution to the primal can be found by solving the KKT equations, (4.36).

Based on the above discussion, an approach to developing dual optimization algorithm is to seek to iteratively reduce the duality gap, while maintaining the feasibility of the pair $(\mathbf{y}, \mathbf{S})$. That reduction can be guided by the following dual potential function (Ye, 2011),

$$\psi(\mathbf{y}, \bar{z}) = \rho \ln(\bar{z} - \mathbf{e}^T \mathbf{y}) - \ln \det \mathbf{S}, \tag{5.11}$$

where $\bar{z} = \text{Tr}(\mathbf{C}\tilde{\mathbf{X}})$ for some feasible $\tilde{\mathbf{X}}$, and $\mathbf{e} = [e_1, e_2, \dots, e_{4N_t+1}]^T$, and ρ is a parameter of the algorithm which is positive. Note that since $\tilde{\mathbf{X}}$ is feasible, $\bar{z}$ is an upper bound on the optimal value of the primal problem. Also recall from (5.2b), that $\mathbf{S} = \mathbf{C} - \sum_{i=1}^{4N_t+1} y_i \mathbf{A}_i$ and hence $\mathbf{S}$ is a function of $\mathbf{y}$. The first term in (5.11) depends on the duality gap, it decreases to $-\infty$ when the duality gap is reduced towards zero. The second term is a barrier function for the cone of positive definite matrices and it guarantees that the matrix $\mathbf{S}$ remains positive semidefinite during the algorithm.

When ρ is chosen to be greater than the size of the problem, $\rho > n$, when n is the size of matrix $\mathbf{A}$ in (5.1), the infimum of the dual potential function is at the optimal solution to the dual problem in (5.2); (Benson *et al.*, 2000). Given feasible starting value for $\mathbf{y}$ and $\mathbf{S}$, the goal of the dual-scaling algorithm is to sequentially choose $\mathbf{y}$ and $\mathbf{S}$ to reduce $\psi(\mathbf{y}, \bar{z})$ until the duality gap is decreased to the point at which it is deemed to be small enough.

5.3 Implementation Principles

5.3.1 Update to the Dual Variables

As explained above, the dual-scaling algorithm seeks to iteratively reduce the dual potential function in (5.11) by updating the dual variables, $\mathbf{y}$ and $\mathbf{S}$. If we denote the values of $\mathbf{y}$ and $\mathbf{S}$ at the k^{th} iteration by $\mathbf{y}^k$ and $\mathbf{S}^k$ and if for simplicity of notation, we denote $\mathbf{y}^{k+1}$ and $\mathbf{S}^{k+1}$ simply by $\mathbf{y}$ and $\mathbf{S}$, then the reduction in the potential function at iteration k is

$$\psi(\mathbf{y}, \bar{z}^k) - \psi(\mathbf{y}^k, \bar{z}^k) \tag{5.12a}$$

$$= \rho \ln(\bar{z}^k - \mathbf{e}^T\mathbf{y}) - \rho \ln(\bar{z}^k - \mathbf{e}^T\mathbf{y}^k) - \ln\det \mathbf{S} + \ln\det \mathbf{S}^k \tag{5.12b}$$

$$= \rho \ln(\bar{z}^k - \mathbf{e}^T\mathbf{y}) - \rho \ln(\bar{z}^k - \mathbf{e}^T\mathbf{y}^k) - \ln\det((\mathbf{S}^k)^{-.5}\mathbf{S}(\mathbf{S}^k)^{-.5}). \tag{5.12c}$$

In order to bound this reduction, the third term in (5.12c) can be bounded using the following lemma.

Lemma 1 (Ye (2011)). *Let $\mathbf{X}$ be a symmetric matrix of size n, $\mathbf{X} \in \mathbb{S}^n$, and $\|\mathbf{X} - \mathbf{I}\|_\infty < 1$. Then*

$$\ln\det(\mathbf{X}) \geq \mathrm{Tr}(\mathbf{X} - \mathbf{I}) - \frac{\|\mathbf{X} - \mathbf{I}\|_F}{2(1 - \|\mathbf{X} - \mathbf{I}\|_\infty)}, \tag{5.13}$$

where $\|.\|_F$ and $\|.\|_\infty$ denote the Frobenius and infinity norm, respectively and

$$\|\mathbf{X}\|_\infty = \max_{i=1,2,\ldots,n} \{|\zeta_i(\mathbf{X})|\} \leq \|\mathbf{X}\|_F, \tag{5.14}$$

where $\zeta_i(\mathbf{X})$ is the i^{th} eigenvalue of matrix $\mathbf{X} \in \mathbb{S}^n$.

In order to use the lemma, we control the update so that $\|(\mathbf{S}^k)^{-.5}\mathbf{S}(\mathbf{S}^k)^{-.5} - \mathbf{I}\|_\infty <$

1. In that case, the reduction can be bounded by the following inequality

$$\begin{aligned}&\psi(\mathbf{y},\bar{z}^k)-\psi(\mathbf{y}^k,\bar{z}^k)\\&\qquad\le\rho\ln(\bar{z}^k-\mathbf{e}^T\mathbf{y})-\rho\ln(\bar{z}^k-\mathbf{e}^T\mathbf{y}^k)-\mathrm{Tr}((\mathbf{S}^k)^{-.5}\mathbf{S}(\mathbf{S}^k)^{-.5}-\mathbf{I})+\Gamma,\end{aligned}\tag{5.15}$$

where $\Gamma=\dfrac{\|(\mathbf{S}^k)^{-.5}\mathbf{S}(\mathbf{S}^k)^{-.5}-\mathbf{I}\|_F}{2(1-\|(\mathbf{S}^k)^{-.5}\mathbf{S}(\mathbf{S}^k)^{-.5}-\mathbf{I}\|_\infty)}$.

In order to simplify the notation, let us define the operator $\mathcal{A}(\mathbf{Z})$ for any symmetric matrix $\mathbf{Z}$ as

$$\mathcal{A}(\mathbf{Z})=\begin{pmatrix}\mathrm{Tr}(\mathbf{A}_1\mathbf{Z})\\\mathrm{Tr}(\mathbf{A}_2\mathbf{Z})\\\vdots\\\mathrm{Tr}(\mathbf{A}_{4N_t+1}\mathbf{Z})\end{pmatrix},\tag{5.16}$$

and the operator $\mathcal{A}^T$ for any vector $\mathbf{y}$ as

$$\mathcal{A}^T(\mathbf{y})=\sum_{i=1}^{4N_t+1}y_i\mathbf{A}_i.\tag{5.17}$$

It can be shown

$$\mathcal{A}(\mathbf{Z})^T\mathbf{y}=\mathrm{Tr}\big(\mathcal{A}^T(\mathbf{y})\mathbf{Z}\big).\tag{5.18}$$

According to the definition of $\mathbf{S}$ and by using (5.17), (A.4b) can be written as $\mathbf{S}=\mathbf{C}-\mathcal{A}^T(\mathbf{y})$. As a result

$$(\mathbf{S}^k)^{-.5}\mathbf{S}(\mathbf{S}^k)^{-.5}-\mathbf{I}=(\mathbf{S}^k)^{-.5}(\mathbf{S}-\mathbf{S}^k)(\mathbf{S}^k)^{-.5}=-(\mathbf{S}^k)^{-.5}(\mathcal{A}^T(\mathbf{y}-\mathbf{y}^k))(\mathbf{S}^k)^{-.5}.\tag{5.19}$$

By using (5.18) and (5.19), the reduction of dual potential function at iteration k can be bounded by

$$\psi(\mathbf{y},\bar{z}^k)-\psi(\mathbf{y}^k,\bar{z}^k) \\ \leq \rho\ln\big(1-\mathbf{e}^T\frac{\mathbf{y}-\mathbf{y}^k}{\bar{z}^k-\mathbf{e}^T\mathbf{y}^k}\big)+\mathcal{A}((\mathbf{S}^k)^{-1})^T(\mathbf{y}-\mathbf{y}^k)+\Gamma. \quad (5.20)$$

The goal now is to further bound (5.20) in terms of the gradient of the dual potential function, which is

$$\bigtriangledown_{\mathbf{y}}\psi(\mathbf{y},\bar{z})=-\frac{\rho}{\bar{z}-\mathbf{e}^T\mathbf{y}}\mathbf{e}+\mathcal{A}(\mathbf{S}^{-1}). \quad (5.21)$$

Using the inequality $\ln(1-x^\theta)\leq\theta x$ and (5.21), the equation in (5.20) can be bounded by

$$\psi(\mathbf{y},\bar{z}^k)-\psi(\mathbf{y}^k,\bar{z}^k)\leq\bigtriangledown_{\mathbf{y}}\psi^T(\mathbf{y}^k,\bar{z}^k)(\mathbf{y}-\mathbf{y}^k)+\Gamma. \quad (5.22)$$

Based on the above analysis, we can guarantee a decrease in the potential function at iteration k by choosing $\mathbf{y}$ so that the right hand side of (5.22) is negative. In fact a good reduction can be obtained by solving the following problem

$$\min_{\mathbf{y}} \quad \bigtriangledown_{\mathbf{y}}\,\psi^T(\mathbf{y}^k,\bar{z}^k)(\mathbf{y}-\mathbf{y}^k) \quad (5.23a)$$

$$\text{s.t.} \quad \|(\mathbf{S}^k)^{-.5}(\mathcal{A}^T(\mathbf{y}-\mathbf{y}^k))(\mathbf{S}^k)^{-.5}\|_F^2\leq\alpha^2, \quad (5.23b)$$

where α is a positive constant less than one. In this case, using Lemma 1, we have

$$\|(\mathbf{S}^k)^{-.5}(\mathcal{A}^T(\mathbf{y}-\mathbf{y}^k))(\mathbf{S}^k)^{-.5}\|_F^2\leq\alpha^2 \quad (5.24)$$

$$\Rightarrow\|(\mathbf{S}^k)^{-.5}(\mathcal{A}^T(\mathbf{y}-\mathbf{y}^k))(\mathbf{S}^k)^{-.5}\|_\infty^2\leq\alpha^2, \quad (5.25)$$

which results in a bounded Γ, $\Gamma\leq\frac{\alpha}{2(1-\alpha)}$. Hence, Γ can be ignored in (5.22) and we can obtain a good reduction in the duality gap by solving (5.23).

By solving the problem in (5.23), we are able to find the update to the dual variables in a way that the dual potential function is reduced. Note that, since $\mathbf{S} = \mathbf{C} - \sum_{i=1}^{4N_t+1} y_i \mathbf{A}_i$, the updated $\mathbf{y}$ determines the updated $\mathbf{S}$. Therefore, finding the new $\mathbf{y}$ is enough to update all the dual variables.

5.3.2 Generic Approach to Solving Dual Potential Reduction Problem

In order to solve the problem in (5.23), we can write the Lagrangian function and the dual problem of (5.23). The Lagrangian function is written as

$$\begin{aligned}&\mathcal{L}(\mathbf{y}, \mathbf{S}, \lambda)\\ &= \nabla\psi^T(\mathbf{y}^k, \bar{z})(\mathbf{y} - \mathbf{y}^k) - \lambda\Big(\mathrm{Tr}\Big[(\mathbf{S}^k)^{-1}\mathcal{A}^T(\mathbf{y} - \mathbf{y}^k)(\mathbf{S}^k)^{-1}\mathcal{A}^T(\mathbf{y} - \mathbf{y}^k)\Big] - \alpha^2\Big). \quad (5.26)\end{aligned}$$

To find the dual function of the problem, we should find the derivative of Lagrangian function with respect to $\mathbf{y}$, $\frac{\partial \mathcal{L}}{\partial \mathbf{y}} = \mathbf{0}$. As seen in (5.26), the Lagrangian is a quadratic function of $\mathbf{y}$ and hence solving $\frac{\partial \mathcal{L}}{\partial \mathbf{y}} = \mathbf{0}$ involves solving a set of linear equations such as what we will have in (5.35). In order to reduce the cost of solving this problem and to find an explicit equation to update the dual variables, we will exploit the structure of matrices used in Lagrangian function in the next section.

5.3.3 Exploiting the Structure of Matrices in Dual-Scaling Algorithm to Update the Dual Variables

As stated in previous sections, the matrices such as $\mathbf{A}_i$ and $\mathbf{S}$ are sparse matrices and this feature can help us to solve SDP with a lower computational cost. We can also exploit this structure of the matrices in solving the problem on (5.26).

If we define $\mathbf{P}_1 \in \mathbb{R}^{(2N_t+1)\times(2N_t+1)}$, and $\mathbf{P}_2, \mathbf{P}_3 \in \mathbb{R}^{(2N_t)\times(2N_t)}$ as

$$\mathbf{P}_1 = \begin{bmatrix} y_1 + y_{2N_t+1} & 0 & 0 & 0 & 0 \\ 0 & y_2 + y_{2Nt+2} & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & y_{2N_t} + y_{4N_t} & 0 \\ 0 & 0 & 0 & 0 & y_{4N_t+1} \end{bmatrix}, \tag{5.27}$$

$$\mathbf{P}_2 = \begin{bmatrix} y_1 & 0 & 0 & 0 \\ 0 & y_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & y_{2N_t} \end{bmatrix}, \text{and } \mathbf{P}_3 = \begin{bmatrix} -y_{2N_t+1} & 0 & 0 & 0 \\ 0 & -y_{2N_t+2} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & -y_{4N_t} \end{bmatrix}, \tag{5.28}$$

we can show that

$$\sum_{i=1}^{4N_t+1} y_i \mathbf{A}_i = \begin{bmatrix} \mathbf{P}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{P}_3 \end{bmatrix}. \tag{5.29}$$

Therefore, since $\mathbf{S} = \mathbf{C} - \sum_{i=1}^{4N_t+1} y_i \mathbf{A}_i$, we can write

$$\mathbf{S} = \begin{bmatrix} \tilde{\mathbf{Q}} - \mathbf{P}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{P}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\mathbf{P}_3 \end{bmatrix}, \quad \mathbf{S}^{-1} = \begin{bmatrix} (\tilde{\mathbf{Q}} - \mathbf{P}_1)^{-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{P}_2^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\mathbf{P}_3^{-1} \end{bmatrix}, \tag{5.30}$$

where the diagonal structure of $\mathbf{P}_2$ and $\mathbf{P}_3$ enables simple construction of the inverse. So the first step to compute (5.26) is to compute $(\mathbf{S}^k)^{-1}\mathcal{A}^T(\mathbf{y} - \mathbf{y}^k)$. To do so we define $(\mathbf{S}_1^k)^{-1}$, $(\mathbf{S}_2^k)^{-1}$ and $(\mathbf{S}_3^k)^{-1}$ as

$$(\mathbf{S}_1^k)^{-1} = \begin{bmatrix} [(\mathbf{S}^k)^{-1}]_{11} & [(\mathbf{S}^k)^{-1}]_{12} & \dots & [(\mathbf{S}^k)^{-1}]_{1(2N_t+1)} \\ [(\mathbf{S}^k)^{-1}]_{21} & [(\mathbf{S}^k)^{-1}]_{22} & \dots & [(\mathbf{S}^k)^{-1}]_{2(2N_t+1)} \\ \vdots & \dots & \ddots & \vdots \\ [(\mathbf{S}^k)^{-1}]_{(2N_t+1)1} & [(\mathbf{S}^k)^{-1}]_{(2N_t+1)2} & \dots & [(\mathbf{S}^k)^{-1}]_{(2N_t+1)(2N_t+1)} \end{bmatrix}, \tag{5.31}$$

$$(\mathbf{S}_2^k)^{-1} = \begin{bmatrix} [(\mathbf{S}^k)^{-1}]_{(2N_t+2)(2N_t+2)} & 0 & \dots & 0 \\ 0 & [(\mathbf{S}^k)^{-1}]_{(2N_t+3)(2N_t+3)} & \dots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \dots & [(\mathbf{S}^k)^{-1}]_{(4N_t+1)(4N_t+1)} \end{bmatrix}, \tag{5.32}$$

$$(\mathbf{S}_3^k)^{-1} = \begin{bmatrix} [(\mathbf{S}^k)^{-1}]_{(4N_t+2)(4N_t+2)} & 0 & \dots & 0 \\ 0 & [(\mathbf{S}^k)^{-1}]_{(4N_t+3)(4N_t+3)} & \dots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \dots & [(\mathbf{S}^k)^{-1}]_{(6N_t+1)(6N_t+1)} \end{bmatrix}. \tag{5.33}$$

So we have

$$(\mathbf{S}^k)^{-1}\mathcal{A}^T(\mathbf{y}-\mathbf{y}^k) = \begin{bmatrix} (\mathbf{S}_1^k)^{-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & (\mathbf{S}_2^k)^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & (\mathbf{S}_3^k)^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{P}_1 - \mathbf{P}_1^k & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{P}_2 - \mathbf{P}_2^k & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{P}_3 - \mathbf{P}_3^k \end{bmatrix}. \tag{5.34}$$

By employing these expressions, we can obtain an expression for the trace in (5.26), and hence for the Lagrangian function. In order to find the dual function of (5.23) to solve the dual problem, we should compute the derivative of the Lagrangian function,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{y}} = \mathbf{0} \tag{5.35a}$$

$$\Rightarrow \mathbf{M}^k(\mathbf{y}^{k+1} - \mathbf{y}^k) + \beta \bigtriangledown_{\mathbf{y}} \psi^T(\mathbf{y}^k, \bar{z}^k) = \mathbf{0} \tag{5.35b}$$

$$\Rightarrow \mathbf{M}^k(\mathbf{y}^{k+1} - \mathbf{y}^k) + \beta(-\frac{\rho}{\bar{z}^k - \mathbf{e}^T\mathbf{y}^k}\mathbf{e} + \mathcal{A}((\mathbf{S}^k)^{-1})) = \mathbf{0}, \tag{5.35c}$$

where $\beta = \frac{\alpha}{\sqrt{-\bigtriangledown_{\mathbf{y}}\psi^T(\mathbf{y}^k,\bar{z}^k)\mathbf{d}(\bar{z}^k)_y}}$. If we define the matrix $\mathbf{N}^k$ as

$$[\mathbf{N}^k]_{ij} = \left([(\mathbf{S}^k)^{-1}]_{ij}\right)^2, \quad 1 \le i,j \le 2N_t, \tag{5.36}$$

the matrix $\mathbf{M}^k$ can be written as

$$\mathbf{M}^k = \begin{bmatrix} \mathbf{N}^k + ((\mathbf{S}_2^k)^{-1})^2 & \mathbf{N}^k & \mathbf{p} \\ \mathbf{N}^k & \mathbf{N}^k + ((\mathbf{S}_3^k)^{-1})^2 & \mathbf{p} \\ \mathbf{p}^T & \mathbf{p}^T & \left([(\mathbf{S}^k)^{-1}]_{(2N_t+1)(2N_t+1)}\right)^2 \end{bmatrix}, \tag{5.37}$$

where the i^{th} element of $\mathbf{p}$ is equal to $\left([(\mathbf{S}^k)^{-1}]_{(i)(2N_t+1)}\right)^2$.

According to (5.35), we have

$$\mathbf{y}^{k+1} - \mathbf{y}^{k} = -\beta(\mathbf{M}^{k})^{-1} \bigtriangledown_{\mathbf{y}} \psi^{T}(\mathbf{y}^{k}, \bar{z}^{k}). \tag{5.38}$$

Hence, we can define the update for the dual variables as

$$\mathbf{y}^{k+1} = \mathbf{y}^{k} + \beta\mathbf{d}(\bar{z}^{k}), \tag{5.39}$$

where

$$\mathbf{d}(\bar{z}^{k}) = -(\mathbf{M}^{k})^{-1} \bigtriangledown_{\mathbf{y}} \psi^{T}(\mathbf{y}^{k}, \bar{z}^{k}). \tag{5.40}$$

Actually it is typically more efficient to consider $\mathbf{d}(\bar{z}^{k})$ to be the solution of the set of linear equations $\mathbf{M}^{k}\mathbf{d}(\bar{z}^{k}) = - \bigtriangledown_{\mathbf{y}} \psi^{T}(\mathbf{y}^{k}, \bar{z}^{k})$ and to exploit the fact that $\mathbf{M}^{k}$ is symmetric and positive semidefinite in the solution of that set of equations.

5.3.4 Generic Update to the Primal Function Value

In a dual-scaling algorithm, the main focus is on updating the dual variable rather than the primal variable. Indeed, we do not find any direction to update the primal variable. However, updating the value of primal function during the algorithm can be helpful as it may result in duality gap reduction that can help the algorithm to find the solution faster. In order to find a feasible primal variable, the KKT conditions (e.g., (4.36)) can be used such that the feasible dual variables which are available can help us to find a feasible primal value at each iteration. Benson *et al.* (2000) solve the following least square problem to find a feasible $\tilde{\mathbf{X}}$

$$\min_{\tilde{\mathbf{X}}} \quad \left\|(\mathbf{S}^k)^{.5}\tilde{\mathbf{X}}(\mathbf{S}^k)^{.5} - \tfrac{\triangle^k}{\rho}\mathbf{I}\right\| \tag{5.41a}$$

$$\text{s.t} \quad \mathcal{A}(\tilde{\mathbf{X}}) = \mathbf{e}, \tag{5.41b}$$

where $\Delta^k = \bar{z}^k - \mathbf{e}^T\mathbf{y}^k$.

The explicit solution of this problem, which is

$$\tilde{\mathbf{X}}(\bar{z}^k) = \frac{\triangle^k}{\rho}(\mathbf{S}^k)^{-1}\big(\mathcal{A}^T(\mathbf{d}(\bar{z}^k)) + \mathbf{S}^k\big)(\mathbf{S}^k)^{-1}, \tag{5.42}$$

can be used to update the primal function value using $\bar{z}^{k+1} = \mathrm{Tr}(\mathbf{C}\tilde{\mathbf{X}}(\bar{z}^k))$. However, computing this matrix can be costly, so we update $\bar{z} = \mathrm{Tr}(\mathbf{C}\tilde{\mathbf{X}})$ directly from the dual variables using

$$\mathrm{Tr}(\mathbf{C}\tilde{\mathbf{X}}(\bar{z}^k)) = \mathbf{e}^T\mathbf{y}^k + \mathrm{Tr}(\tilde{\mathbf{X}}(\bar{z}^k)\mathbf{S}^k) \tag{5.43a}$$

$$= \mathbf{e}^T\mathbf{y}^k + \frac{\triangle^k}{\rho}\big(\mathbf{d}(\bar{z}^k)^T\mathcal{A}(\mathbf{S}^k)^{-1}) + n\big), \tag{5.43b}$$

It was stated before that in dual-scaling algorithm, the primal variable is not updated at each iteration, hence to have updated value of primal function, we should define the conditions in which we want to update the primal function value instead of the dual variables. Note that in updating dual or primal variables/function, feasibility is the important issue.

There is a lemma by Benson *et al.* (2000) that defines how to decide whether to update the dual variables or the primal function value in a way that ensures that the primal and dual variables remain feasible.

Lemma 2 (Benson *et al.* (2000)). *Define* $\mathbf{P}$ *as*

$$\mathbf{P}(\bar{z}^k) = \frac{p}{\Delta^k}(\mathbf{S}^k)^{.5}\tilde{\mathbf{X}}(\bar{z}^k)(\mathbf{S}^k)^{.5} - \mathbf{I}, \tag{5.44}$$

and let $\mu^k = \frac{\Delta^k}{n} = \frac{\bar{z}^k - \mathbf{b}^T\mathbf{y}^k}{n}$, $\mu = \frac{\mathrm{Tr}(\tilde{\mathbf{X}}(\bar{z}^k)\mathbf{S}^k)}{n} = \frac{\mathrm{Tr}(\mathbf{C}\tilde{\mathbf{X}}(\bar{z}^k)) - \mathbf{b}^T\mathbf{y}^k}{n}$, $\rho \geq n + \sqrt{n}$ *where* n *is the size of the (square) matrices and* $\alpha < 1$. *If*

$$\|\mathbf{P}(\bar{z}^k)\|_F < \min\Big(\alpha\sqrt{\frac{n}{n+\alpha^2}}, 1-\alpha\Big), \tag{5.45}$$

then the following inequalities hold:

1. $\tilde{\mathbf{X}}(\bar{z}^k) \succ 0$

2. $\|(\mathbf{S}^k)^{.5}\tilde{\mathbf{X}}(\bar{z}^k)(\mathbf{S}^k)^{.5} - \mu\mathbf{I}\|_F \leq \alpha\mu$

3. $\mu \leq (1 - \frac{\alpha}{2\sqrt{n}})\mu^k$

In order to explain this lemma, let us consider that the current iteration is the k^{th} iteration of the algorithm. If (5.45) holds in this iteration, the $\tilde{\mathbf{X}}$ that results from (5.42), using the current $\bar{z}^k$, is guaranteed to be positive semidefinite. The duality gap that arises from the updated $\tilde{\mathbf{X}}$ is also bounded, based on the third inequality in the lemma. Hence, if (5.45) holds, we update the value of primal function and if not the dual variables are updated. It can be seen that at each iteration either the primal function value or the dual variables are updated, so although we don't update $\tilde{\mathbf{X}}$ at each iteration, we can use the reduction of the duality gap which is the result of updating the value of primal function at some iterations.

Using (5.42), $\mathbf{P}(\bar{z}^k)$ can be written as

$$\mathbf{P}(\bar{z}^k) = \frac{\rho}{\Delta^k}(\mathbf{S}^k)^{.5}\Big(\frac{\triangle^k}{\rho}(\mathbf{S}^k)^{-1}\big(\mathcal{A}^T(\mathbf{d}(\bar{z}^k)) + \mathbf{S}^k\big)(\mathbf{S}^k)^{-1}\Big)(\mathbf{S}^k)^{.5} - \mathbf{I} \tag{5.46a}$$

$$= (\mathbf{S}^k)^{-.5}\Big(\mathcal{A}^T(\mathbf{d}(\bar{z}^k))\Big)(\mathbf{S}^k)^{-.5} \tag{5.46b}$$

$$= (\mathbf{S}^k)^{-.5}\Big(\mathcal{A}^T(\frac{\mathbf{y}^{k+1} - \mathbf{y}^k}{\beta})\Big)(\mathbf{S}^k)^{-.5}, \tag{5.46c}$$

which, by using (5.39) and (5.40), will result in

$$\|\mathbf{P}(\bar{z}^k)\|_F = -\bigtriangledown_{\mathbf{y}}\, \psi^T(\mathbf{y}^k, \bar{z}^k)\mathbf{d}(\bar{z}^k). \tag{5.47}$$

In the customized dual-scaling algorithm that we will develop below, the same principles in updating the value of primal function are used. In Section 5.5, we will also explain that by updating the primal function value more frequently, the algorithm is terminated faster as the reduction of duality gap corresponding to the primal function update is typically greater than the reduction resulting from the dual variable update.

To summarize this section, we describe the generic dual-scaling algorithm in Algorithm 1.

5.4 Finding a Feasible Starting Point

In interior point algorithms we need find feasible starting points for the primal and dual variables. The constraints on the primal variables are that $L \leq [\mathbf{X}]_{ii} \leq U$ and that the matrix $\mathbf{X}$ is positive semidefinite. The constraints on the dual variables are that the matrix $\mathbf{S}$ is positive semidefinite and that $\mathbf{C} = \sum_{i=0}^{4N_t+1} y_i\mathbf{A}_i + \mathbf{S}$; cf. (5.2).

Algorithm 1

```
    Given initial values for the dual variables, $(\mathbf{S}^0, \mathbf{y}^0)$, and a primal upper bound, $\bar{z}^0$, set $k = 0$, $\alpha \in (0, 1)$ and the target duality gap equal to $\epsilon$.
2:  while $\bar{z}^k - \mathbf{e}^T\mathbf{y}^k \geq \epsilon$ do
        Compute $\mathcal{A}((\mathbf{S}^k)^{-1})$ using (5.16)
4:      Compute $\bigtriangledown_{\mathbf{y}}\psi^T(\mathbf{y}^k, \bar{z}^k)$ using (5.21)
        Compute $\mathbf{M}^k$ and find the direction for updating $\mathbf{y}$ using (5.40)
6:      Calculate $\|\mathbf{P}(\bar{z}^k)\|_F$ using (5.47)
        if (5.45) is true then, update the primal function value using (5.43b)
8:      else update the dual variables, $\mathbf{y}$ and $\mathbf{S}$, using (5.39) and $\mathbf{S}^{k+1} = \mathbf{C} - \sum_{i=1}^{4N_t+1} y_i^{k+1}\mathbf{A}_i$ respectively, and set $\bar{z}^{k+1} = \bar{z}^k$
        end if
10:     $k = k + 1$
    end while
12: Find the primal variables $\tilde{\mathbf{X}}$ that satisfy target duality gap by using (5.42)
```

One of the contributions of this thesis is to use insight from the MIMO demodulation application to develop efficient techniques for selecting good starting points.

Primal Feasible Starting Point

There are several ways to initialize the primal variables. Perhaps the simplest way is to choose matrix $\mathbf{X}$ to be a diagonal matrix which has diagonal elements equal to mean of the upper bound and lower bound of the elements. As the upper bound is U and the lower bound is L, we have

$$\mathbf{X} = \begin{bmatrix} (\frac{U+L}{2}) \times \mathbf{I}_{2N_t} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix}. \tag{5.48}$$

Then we construct the full matrix $\tilde{\mathbf{X}}$ based on the slack variables in (5.4). This choice guarantees that the initial point is well inside the boundary of the feasible region.

The previous method does not exploit any knowledge of the application to soft demodulation. One way to consider that knowledge is to estimate the transmitted symbols using linear estimation, such as zero-forcing or minimum mean square error (MMSE) methods and then use that estimate to construct the initial primal feasible point. In the following, we will focus on MMSE estimation. Assume that the real representation of channel matrix and the received signal are $\tilde{\mathbf{H}}$ and $\tilde{\mathbf{y}}$, respectively,

$$\tilde{\mathbf{y}} = \tilde{\mathbf{H}}\tilde{\mathbf{s}} + \tilde{\mathbf{v}}. \tag{5.49}$$

The linear MMSE estimate is

$$\begin{aligned} \hat{\mathbf{s}}_{MMSE} &= (\sigma^2\mathbf{I} + \tilde{\mathbf{H}}^H\tilde{\mathbf{H}})^{-1}\tilde{\mathbf{H}}^H\tilde{\mathbf{y}} & (5.50\text{a}) \\ &= (\sigma^2(\tilde{\mathbf{H}}^H\tilde{\mathbf{H}})^{-1} + \mathbf{I})^{-1}\tilde{\mathbf{s}} + (\sigma^2\mathbf{I} + \tilde{\mathbf{H}}^H\tilde{\mathbf{H}})^{-1}\tilde{\mathbf{H}}^H\tilde{\mathbf{v}}. & (5.50\text{b}) \end{aligned}$$

As this estimate is biased, we will use the so called “unbiased” MMSE estimate as

$$\hat{\mathbf{s}} = \hat{\mathbf{s}}_{UB-MMSE} = \mathbf{\Phi}^{-1}\hat{\mathbf{s}}_{MMSE}, \tag{5.51}$$

where $\mathbf{\Phi}$ is a diagonal matrix with $[\mathbf{\Phi}]_{ii} = [(\sigma^2\mathbf{I} + \tilde{\mathbf{H}}^H\tilde{\mathbf{H}})^{-1}\tilde{\mathbf{H}}^H]_{ii}$.

Now that we have the vector $\hat{\mathbf{s}}$ as an estimate of $\tilde{\mathbf{s}}$, we should clip each $\hat{s}_i$ to the feasible interval, $[-\sqrt{U}, -\sqrt{L}] \cup [\sqrt{L}, \sqrt{U}]$. The first block of primal variable matrix, $\mathbf{X}$ can be constructed by knowing $\tilde{\mathbf{s}}$. Therefore, we can use the estimate of $\tilde{\mathbf{s}}$, $\hat{\mathbf{s}}$, to construct an initialization for first block of primal variable matrix as

$$\mathbf{X} = \begin{bmatrix} \hat{\mathbf{s}}\hat{\mathbf{s}}^T & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix}. \tag{5.52}$$

In order to avoid choosing a starting point on the boundary of the primal feasible set, in practice we will choose a positive δ and clip $\hat{s}_i$ to $[-\sqrt{U}+\delta, -\sqrt{L}-\delta] \cup [\sqrt{L}+\delta, \sqrt{U}-\delta]$ to obtain $\breve{s}_i$. Considering that the initial matrix $\mathbf{X}$ should be positive definite, rather than using (5.52) we construct matrix $\mathbf{X}$ using

$$\mathbf{X} = \begin{bmatrix} \breve{\mathbf{s}}\breve{\mathbf{s}}^T + \breve{\epsilon}\mathbf{I}_{2N_t} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{5.53}$$

in which $\breve{\epsilon} > 0$, to make sure that the primal starting point is reasonably far from the boundary of the feasible set. The second and third block of matrix are then constructed using (5.4).

As the above example suggests, starting points for $\mathbf{X}$ can be generated using any standard suboptimal detector. Another example is the (ordered) zero-forcing decision feedback detector. This method involves a QR-decomposition. As the channel matrix is known at the receiver, we can apply the QR-decomposition to the real representation of channel matrix, $\tilde{\mathbf{H}}$. The point here is to order the columns of $\tilde{\mathbf{H}}$ to mitigate the effects of noise. We use the technique of Wubben *et al.* (2001) to find a QR-decomposition with sorted $\mathbf{R}$. That is, we replace $\tilde{\mathbf{y}} = \tilde{\mathbf{H}}\tilde{\mathbf{s}} + \tilde{\mathbf{v}}$ with

$$\tilde{\mathbf{y}} = \tilde{\mathbf{H}}(\mathbf{P}\mathbf{P}^T)\tilde{\mathbf{s}} + \tilde{\mathbf{v}}, \tag{5.54}$$

where $\mathbf{P}$ is a permutation matrix that reorders $\tilde{\mathbf{H}}$ so that the columns are arranged in an ascending order with respect to their norms. We then apply the QR-decomposition to $\tilde{\mathbf{H}}\mathbf{P} = \mathbf{Q}\mathbf{R}$ and construct $\bar{\mathbf{y}} = \mathbf{Q}^H\tilde{\mathbf{y}} = \mathbf{R}\mathbf{s}' + \bar{\mathbf{v}}$, where $\mathbf{s}' = \mathbf{P}^T\tilde{\mathbf{s}}$ and $\bar{\mathbf{v}} = \mathbf{Q}^{-1}\tilde{\mathbf{v}}$. Since $\mathbf{Q}$ is a unitary matrix, $\bar{\mathbf{v}}$ remains Gaussian with the same covariance as $\tilde{\mathbf{v}}$. The matrix format is

$$\begin{bmatrix} \bar{y}_1 \\ \vdots \\ \bar{y}_n \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{21} & \cdots & r_{2n} \\ \vdots & \ddots & \cdots & \vdots \\ 0 & 0 & \cdots & r_{nn} \end{bmatrix} \begin{bmatrix} s'_1 \\ \vdots \\ s'_n \end{bmatrix} + \begin{bmatrix} \bar{v}_1 \\ \vdots \\ \bar{v}_n \end{bmatrix}. \tag{5.55}$$

Estimates of $\mathbf{s}'$ can be obtained by back substitution starting from the last row, i.e.,

$$s'_m = \frac{\bar{y}_m - \sum_{i=m+1}^{n} r_{m,i} s'_i}{r_{m,m}}. \tag{5.56}$$

After finding the estimation of $\mathbf{s}'$, $\mathbf{s}'_{est}$, we can compute the clipped version of $\mathbf{P}\mathbf{s}'_{est}$ that was used in linear MMSE method, and use it as the estimate of $\tilde{\mathbf{s}}$, $\hat{\mathbf{s}}$. By analogy with the MMSE method we can then construct $\mathbf{X}$ and subsequently $\tilde{\mathbf{X}}$.

Dual Feasible Starting Point

Given a primal feasible starting point, $\tilde{\mathbf{X}}$, we consider two different ways to find corresponding dual feasible starting values for $\mathbf{y}$ and $\mathbf{S}$. The first is based on the KKT conditions and the second is derived from the fact that the initial value for $\mathbf{y}$ should be chosen so that $\mathbf{S} = \mathbf{C} - \sum_{i=1}^{4N_t+1} y_i \mathbf{A}_i$ is strictly positive definite.

Our first approach to determining a suitable starting pair $(\mathbf{y}, \mathbf{S})$ from a given starting value for $\tilde{\mathbf{X}}$ is to observe that the modified version of the KKT conditions in (5.2), contain the condition $\mathbf{S}\tilde{\mathbf{X}} = \nu \mathbf{I}$, where ν is proportional to the duality gap. A pair satisfying that condition would correspond to a point on the central path of an interior point method (e.g., Boyd and Vandenberghe, 2004). We do not need to start at a point on the central path, but we would like to start close to the central path.

If we recall that the matrix $\mathbf{S}$ must satisfy the constraint $\mathbf{S} = \mathbf{C} - \sum_{i=1}^{4N_t+1} y_i\mathbf{A}_i$, then given $\tilde{\mathbf{X}}$, we can find an initial feasible starting value for $\mathbf{y}$, and hence $\mathbf{S}$, by choosing a value for ν, and solving the problem

$$\min_{\mathbf{y}} \quad \Big\|\big(\mathbf{C} - \sum_{i=1}^{4N_t+1} y_i\mathbf{A}_i\big)\tilde{\mathbf{X}} - \nu\mathbf{I}\Big\|_F^2 \tag{5.57a}$$

$$\text{s.t.} \quad \mathbf{C} - \sum_{i=1}^{4N_t+1} y_i\mathbf{A}_i \succeq 0. \tag{5.57b}$$

This is a convex optimization problem, and can be efficiently solved. A convenient interface to some powerful solution techniques is provided by the popular parser software CVX (Grant and Boyd, 2008, 2014). The choice of ν involves a trade-off between the duality gap at the starting point, and the distance that the initial points are from the central path.

In cases where $\tilde{\mathbf{X}}$ is invertible, such as in the first of our methods for selecting $\tilde{\mathbf{X}}$, the objective in (5.57a) can be modified to $\big\|(\mathbf{C} - \sum_{i=1}^{4N_t+1} y_i\mathbf{A}_i) - \nu\tilde{\mathbf{X}}^{-1}\big\|_F^2$, if desired. In cases in which the upper left block of $\tilde{\mathbf{X}}$ is rank-1, such as in our other methods, we can first modify $\tilde{\mathbf{X}}$ to

$$\tilde{\mathbf{X}}' = \tilde{\mathbf{X}} + \begin{bmatrix} \tilde{\epsilon}\mathbf{I}_{2N_t+1,2N_t+1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \tag{5.58}$$

for some small $\tilde{\epsilon}$, and then use $(\tilde{\mathbf{X}}')^{-1}$. (Of course, in both cases we could simply solve (5.57) directly.)

To avoid the cost of solving (5.57), we can use the fact that we would like $\mathbf{S} = \mathbf{C} - \sum_{i=1}^{4N_t+1} y_i\mathbf{A}_i$ to be positive definite to find a feasible starting point. In order to do so, we use Gerschorin's Theorem (e.g., Atkinson, 2008), which shows that each eigenvalue of a matrix lies at least in one of the circles with centre equal to diagonal elements of the matrix and radius equal to the summation of the absolute values of the elements in the row corresponding to that diagonal element, i.e. for matrix $\mathbf{S}$ with eigenvalues ζ_ℓ, it means

$$|\zeta_\ell - \mathbf{S}_{ii}| \leq \sum_{i \neq j} |\mathbf{S}_{ij}|, \quad \forall \ell, \tag{5.59}$$

which implies that

$$-\sum_{i \neq j} |\mathbf{S}_{ij}| + \mathbf{S}_{ii} \leq \zeta_\ell \leq \sum_{i \neq j} |\mathbf{S}_{ij}| + \mathbf{S}_{ii}, \quad \forall \ell. \tag{5.60}$$

The constraint that $\mathbf{S}$ is a positive definite matrix is equivalent to requiring that the eigenvalues of $\mathbf{S}$ are positive. Using (5.60), in order to guarantee that each eigenvalue of $\mathbf{S}$ is positive, we can impose the constraint

$$\mathbf{S}_{ii} - \sum_{i \neq j} |\mathbf{S}_{ij}| > 0 \quad \forall i. \tag{5.61}$$

On the other hand, based on the equation in (5.30), we know that $\mathbf{S}$ is block matrix whose second and third blocks are diagonal and hence all diagonal elements of these two blocks should be positive. This results in a determination of the sign of each y_i. To define the value of each y_i, we should consider $\mathbf{S}_{ii} - \sum_{i \neq j} |\mathbf{S}_{ij}| > 0$ for $i = 1, \ldots, 2N_t + 1$, corresponding to the first block of $\mathbf{S}$, which is $\mathbf{S}_1 = \tilde{\mathbf{Q}} - \mathbf{P}_1$. Therefore, we have

$$\tilde{\mathbf{Q}}_{ii} - (y_i + y_{2N_t+i}) - \sum_{i \neq j} |\tilde{\mathbf{Q}}_{ij}| > 0 \Rightarrow -(y_i + y_{2N_t+i}) > \sum_{i \neq j} |\tilde{\mathbf{Q}}_{ij}| - \tilde{\mathbf{Q}}_{ii}. \tag{5.62}$$

To solve this inequality we can define a positive vector, $\boldsymbol{\gamma}_1$, such that

$$-(y_i + y_{2N_t+i}) = \sum_{i \neq j} |\tilde{\mathbf{Q}}_{ij}| - \tilde{\mathbf{Q}}_{ii} + \gamma_{1,i}, \tag{5.63}$$

where $\gamma_{1,i}$ is the i^{th} element of $\boldsymbol{\gamma}_1$. Now by fixing either y_i or y_{2N_t+i}, we can find the other one, considering the previously derived condition on their signs. We will show one of the possible variants of this technique in Appendix C.

5.5 Some Customizations of the Dual-Scaling Algorithm

In addition to what we have described in the previous sections, there are some other ways to make the algorithm faster. In this section, we will describe two different procedures that can reduce the number of iterations required for the duality gap converge to the target value, ϵ. These customizations exploit features of our application, and in particular the fact that the target value ϵ is typically quite large (Nekuii, 2008).

Updating Primal Function Value (Customization 1)

As stated in Section 5.3.4, depending on the outcome of a test based on Lemma 2, either the dual variables or the primal function value is updated. Our numerical experience shows that the duality gap reduction in the iterations in which the primal function value is updated tends to be greater than the iterations in which the dual variables are updated. These observations suggest that we could update the primal function value more frequently. In particular, we customize the dual-scaling algorithm such that the primal function value is updated whenever the duality gap reduction slows. We will base the decision of whether to update primal function value on the relative reduction in the duality gap at the previous iteration. The primal function value is updated at the k^{th} iteration if

$$\frac{\Delta^{k-1} - \Delta^{k}}{\Delta^{k-1}} < \delta, \tag{5.64}$$

where Δ^k is the duality gap in the k^{th} iteration and δ is a positive parameter. If the condition in (5.64) does not hold, the convention test based on Lemma 2 is employed. The effect of this customization is shown in Fig. 5.1, which demonstrates the reduction of the duality gap due to the primal update triggered by (5.64) at iteration 15. As analogous updates are employed, the target duality gap will be reached with fewer iterations.

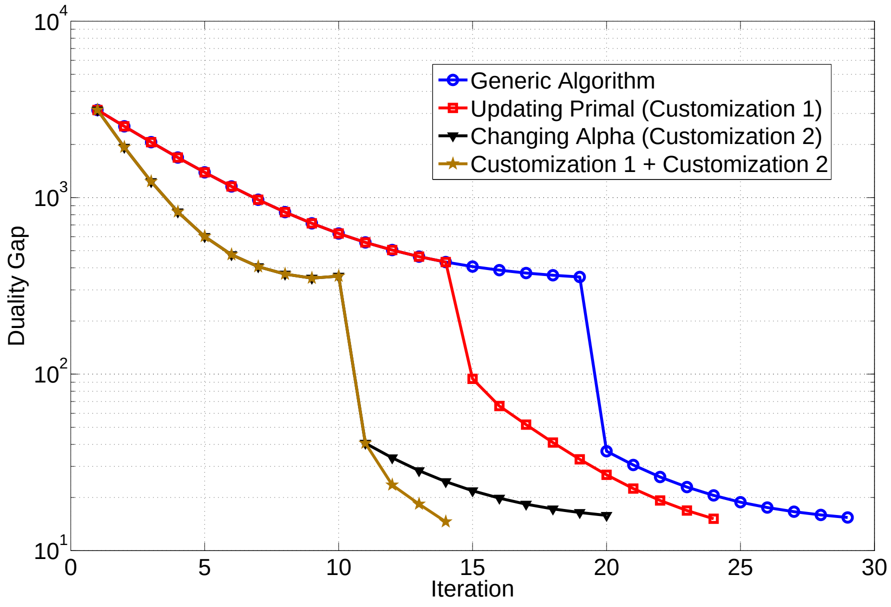

Figure 5.1: Duality gap vs. iteration for different customizations.

Different α for Early Iterations (Customization 2)

In the generic dual-scaling algorithm, the dual variables are updated according to

$$\mathbf{y}^{k+1} = \mathbf{y}^k + \beta \mathbf{d}(\bar{z}^k), \tag{5.65}$$

where $\beta = \frac{\alpha}{\sqrt{-\nabla_{\mathbf{y}} \psi^T(\mathbf{y}^k, \bar{z}^k) \mathbf{d}(\bar{z}^k)_y}}$. Based on Lemma 1 and (5.23), α is restricted to be less than one. Considering (5.65), α can be defined as the step size we take toward the optimal dual variables. Therefore, a bigger α could help us to reach the optimal dual variables more quickly. However, it may cause infeasibility in the updated variables.

In the early iterations we are, most probably, reasonably far from the boundary of the feasible set and from the optimal solution. Therefore, it seems likely that we could take bigger steps while remaining feasible. When we are closer to the optimal solution, we should take smaller steps in order not to generate infeasible updates. Although α is chosen to be greater than one in early iterations of the customized dual-scaling algorithm, we still use Lemma 2 to make a decision on whether to update primal function value or dual variables. If the probability that the algorithm fails to provide semidefinite primal and dual variables, $\tilde{\mathbf{X}}$ and $\mathbf{S}$, is less than the probability of error, this choice does not degrade the performance of the demodulator. (Numerical experience suggests that this is the case.) However, we are reducing the number of iterations and therefore the computational cost. The effect of this customization is shown in the lower curves in Fig. 5.1.

In our simulations, we use both customization 1 and customization 2 which results in a significant reduction in the number of iterations taken by the algorithm. A typical graph for the algorithm using both customization is shown in Fig. 5.1.

The complete customized dual-scaling algorithm is described in Algorithm 2.

Computational Cost

The key computational task in the customized dual-scaling algorithm is computing $\mathbf{S}^{-1}$ in the step 6 of the algorithm, as shown in (5.31). This requires inversion of a matrix of size $(2N_t + 1) \times (2N_t + 1)$. Finding the solution of the set of linear equations in the step 8 of the algorithm based on the equation in (5.40) involves the Cholesky factorization of a matrix of size $(4N_t + 1) \times (4N_t + 1)$. Hence the order of the computational cost of each iteration is cubic in the problem size. Interior-point method analysis yields a bound on the number of iterations of $O(\sqrt{N_t} \log(\epsilon_0/\epsilon))$, where ϵ_0 is the initial duality gap.

Therefore, the computational cost of the customized dual-scaling algorithm is of the same order as the customized primal-dual interior point method. However, computational experiments suggest that the cost of each iteration of the dual-scaling algorithm is about half that of the primal-dual algorithm and, as shown in the next chapter, the overall computational cost of the dual-scaling method is lower than that of the primal-dual algorithm.

Algorithm 2

```
    Find good feasible dual initial points, $\mathbf{S}^0$ and $\mathbf{y}^0$, and a corresponding primal upper bound, $\bar{z}^0$, based on Section 5.4. Set $\epsilon$ equal to the target duality gap, set $\gamma \in [0,1]$, fix the number of early iterations, $k'$, and set $k = 0$
2:  **while** $\bar{z}^k - \mathbf{e}^T\mathbf{y}^k \geq \epsilon$ **do**
        **if** $k \leq k^{'}$ **then** choose $\alpha$ so that bigger step size is resulted (a large value of $\alpha < 2$)
4:      **else**  choose $\alpha$ so that smaller step size is resulted (a value of $\alpha < 1$)
        **end if**
6:      Compute $\mathcal{A}(\mathbf{S}^{-1})$ using (5.16)
        Compute $\bigtriangledown_{\mathbf{y}}\psi^T(\mathbf{y}^k, \bar{z}^k)$ using (5.21)
8:      Compute $\mathbf{M}^k$ and find the direction for updating $\mathbf{y}$ using (5.40)
        Calculate $\|\mathbf{P}(\bar{z}^k)\|_F$ using (5.47)
10:     **if** $\frac{\Delta^{k-1}-\Delta^k}{\Delta^{k-1}} < \gamma$ **then** update the value of primal function using (5.43b)
        **else if** (5.45) is true **then** update the primal function value using (5.43b)
12:     **else**  update the dual variables, $\mathbf{y}$ and $\mathbf{S}$, using (5.39) and $\mathbf{S}^{k+1} = \mathbf{C} - \sum_{i=1}^{4N_t+1} y_i^{k+1}\mathbf{A}_i$ respectively, and $\bar{z}^{k+1} = \bar{z}^k$
        **end if**
14:     $k = k + 1$
    **end while**
16: Find the primal variable $\tilde{\mathbf{X}}$ that satisfies the target duality gap by using (5.42)
```

Chapter 6

Performance Analysis and EXIT Charts

In this section we will compare the performance and complexity of the proposed demodulators to those of the existing soft demodulators that were described in Chapter 3. We will consider both the proposed customized dual-scaling algorithm for the SDR-based demodulators (see Chapter 5) and the existing customized primal-dual algorithm (see Chapter 4). The other demodulators that we will consider are the single tree search demodulator (Studer and Bolcskei, 2010), the LISS demodulator (Hagenauer and Kuhn, 2007), and the full MMSE-SIC method of Wang and Poor (1999) and the approximate MMSE-SIC method of Studer *et al.* (2011).

6.1 Simulation Scenario

We consider a MIMO system with $N_t = 4$ transmitter antennas and $N_r = 4$ receiver antennas. The channel is modeled using an i.i.d. Rayleigh fading model that

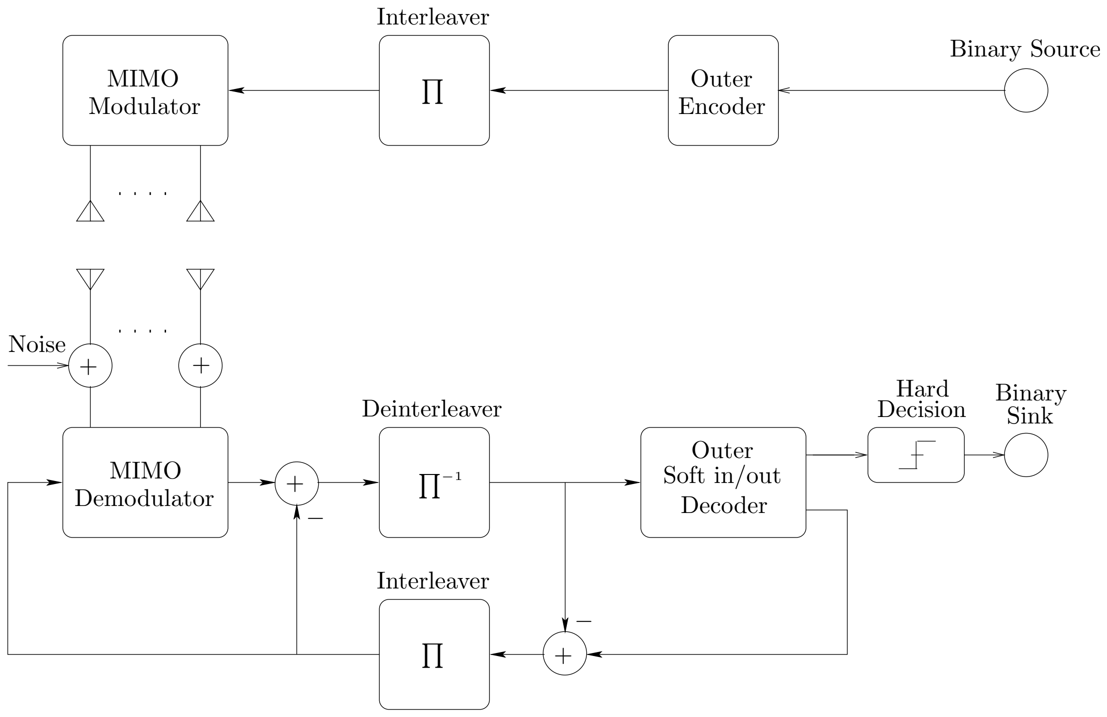

Figure 6.1: MIMO BICM-IDD transceiver

is assumed to be interleaved so that consecutive channel uses are not significantly correlated. The transceiver structure is the BICM-IDD structure described in Chapter 2; see Fig 6.1. The outer encoder is a (punctured) binary parallel-concatenated turbo encoder of rate $R = 1/2$ in which the generator polynomial of the recursive systematic convolutional codes has an octal representation $(5, 7)$. The interleaver is also randomly generated for each different codeword. The modulator block consists of a 16-QAM symbol mapper that employs Gary coding to map the bits to the symbols and a spatial multiplexer to transmit a symbol from each antenna at each channel use.

The BCJR algorithm is used to decode the convolutional codes in the turbo code, and 8 decoding iterations are performed in the decoder before passing the soft information back to the demodulator. We also clip the extrinsic LLR of the decoder to the interval $[-5, +5]$ for list-based demodulators. This value is chosen based on the analysis by de Jong and Willink (2005) and is consistent with the clipping value used by Nekuii (2008). For the STS demodulator the clipping interval is a parameter of the algorithm.

In the customized dual-scaling algorithm method, the starting primal variable, $\tilde{\mathbf{X}}$, is initialized based on the first method in Section 5.4, in which the diagonal elements are set to the mean of the lower bound and the upper bound of the constellation; see (5.48). The dual variables are initialized using variant of the Gerschorin bounding technique described in Appendix C.

For the SDR-based demodulators we choose the number of randomization iterations to be $M = 50$ or $M = 200$ and the accuracy $\epsilon = 10^{-1}$ is considered based on the observations made by Nekuii (2008). For the LISS demodulator we choose the stack size equal to be $S = 500$, and list size equal to be $L = 80$. The important parameter in single tree search method is the clipping parameter, we simulate this method by choosing clipping value to be 5 and 10.

To evaluate the computational cost of each different method, we counted the floating point operation (FLOP) required by each demodulator. Each real-valued mathematical operation for real numbers is counted as one FLOP; as an example, the addition of two complex numbers counts as two FLOPs. In the SDR-based demodulators the FLOP count includes, among other things, the number of FLOPs that are required to solve SDP, to perform the randomization procedure, to generate the list and to compute the LLRs. In both MMSE cases we count the FLOPs required to compute the estimates of symbols, those required to cancel interference and the FLOPs required to compute and implement the MMSE filter. For the STS method, the FLOP count includes the FLOPs expended in the preprocessing step, in which the QR-decomposition is performed on the channel matrix, and FLOPs required to examine each node and to compute the soft information. In the LISS demodulator the FLOPs required to generate the list, compute the probabilities, extend the paths and compute the LLRs are counted.

6.2 Primal-Dual Interior Point vs. Dual Scaling Algorithm

In Fig 6.2, we compare the bit error rate performance of different SDR demodulators using the primal-dual interior point method or the dual-scaling algorithm to solve the semidefinite programming problem. In Fig 6.3 the empirical cumulative distribution of the computational cost of each method is shown.

It can be seen that the performance of demodulators using dual-scaling algorithm is close to the performance of demodulators that use primal-dual interior point. The difference in the performance of these two algorithms is due to the coarse nature of the solution; the accuracy chosen in these simulation is $\epsilon = 10^{-1}$ and that results in a sizeable region of acceptable solutions. If we were to choose ϵ to be smaller, say 10^{-2} or even 10^{-3}, the performance of the primal-dual and dual-scaling algorithms will be closer, but the computational costs will be increased. The choice of $\epsilon = 10^{-1}$ is based on observations by Nekuii (2008). These simulations also show that the approximations that were used to obtain the single-SDR method result in some

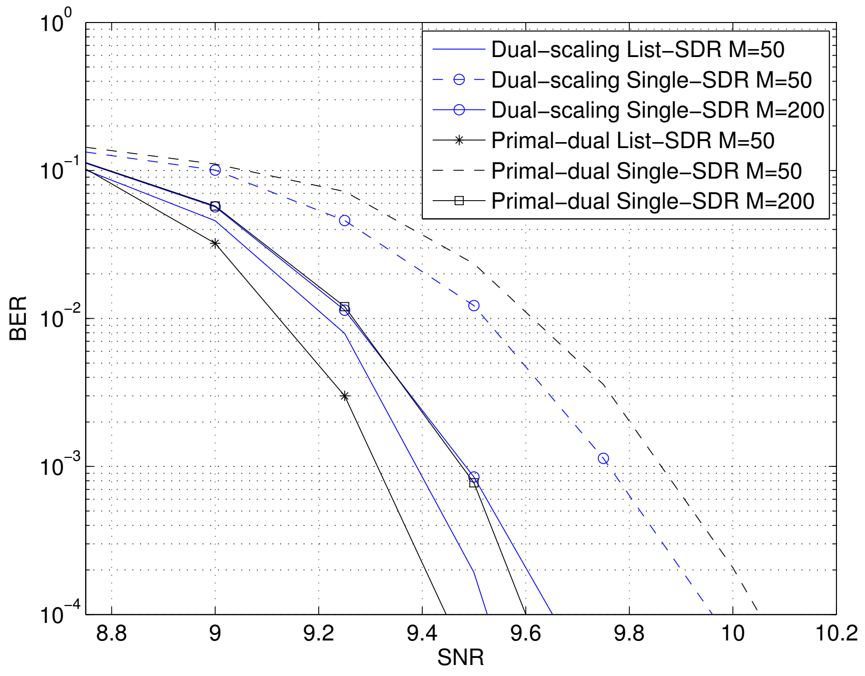

Figure 6.2: BER vs. SNR.

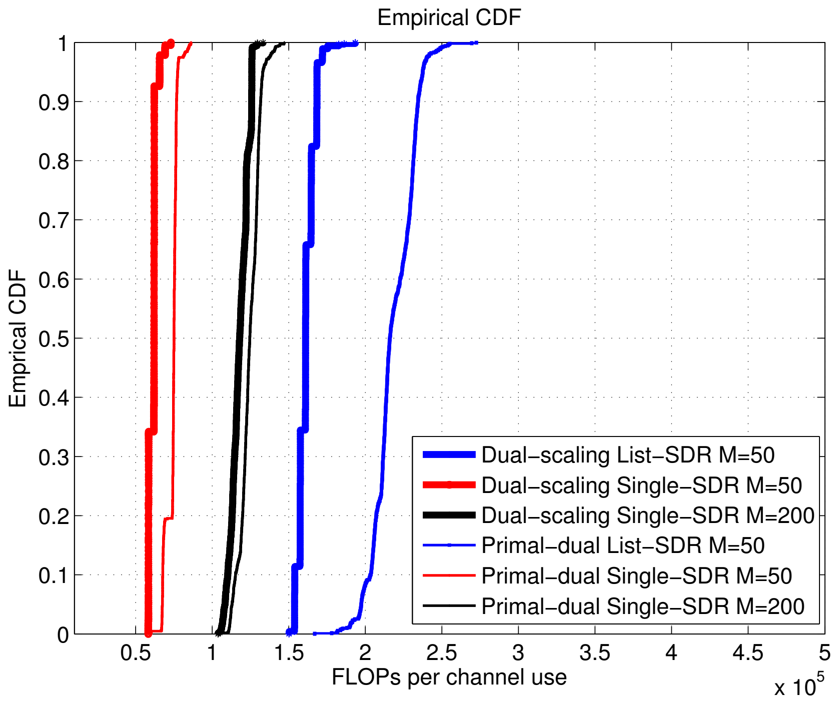

Figure 6.3: Empirical CDF of the number of FLOPs per channel use.

degradation in performance, but that much of that degradation can be recovered by using more randomization steps, each of which is of modest cost.

However as shown in Fig 6.3, the reduced computational cost of the dual-scaling algorithm is an advantage of this method, especially in the case of the list-SDR method in which one SDP is solved in each demodulation-decoding iteration.

By looking at Fig. 6.2 and Fig. 6.3 simultaneously, we can deduce that the single-SDR method with the dual-scaling algorithm and $M = 200$ randomization provides a desirable trade-off between performance and computational cost. In the next section we will compare the performance and cost of the dual single-SDR with other existing methods introduced in Chapter 4.

6.3 Dual Single-SDR vs. Existing Methods

In this section we will compare the performance and complexity of the dual-scaling single-SDR demodulator with $M = 200$ randomizations against those of the existing methods reviewed in Chapter 3. In Fig. 6.4 we compare the performance of these methods in terms of the bit error rate at different SNRs. In Fig. 6.5 the empirical CDF of the computational cost of each of these methods is shown.

The first observation from these figures is that the dual-scaling single-SDR demodulator provides better performance than all except the LISS demodulator, and that under most reasonable measures of computational cost the dual-scaling single-SDR demodulator is cheaper than tree-search methods and is close to that of the "full" MMSE-SIC of Wang and Poor (1999). The performance of the dual-scaling single-SDR demodulator is significantly better than that of the full MMSE-SIC.

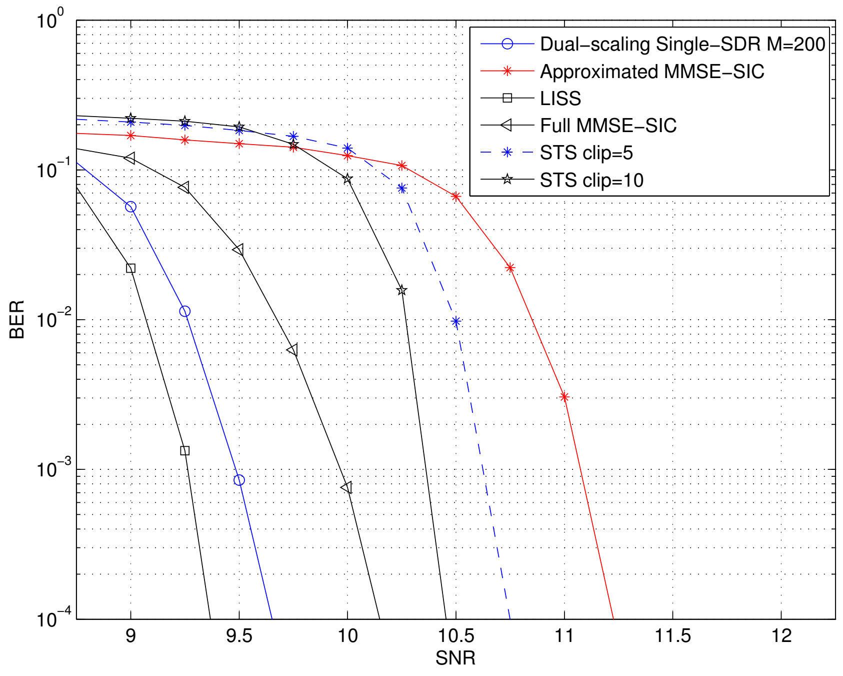

Figure 6.4: BER vs. SNR.

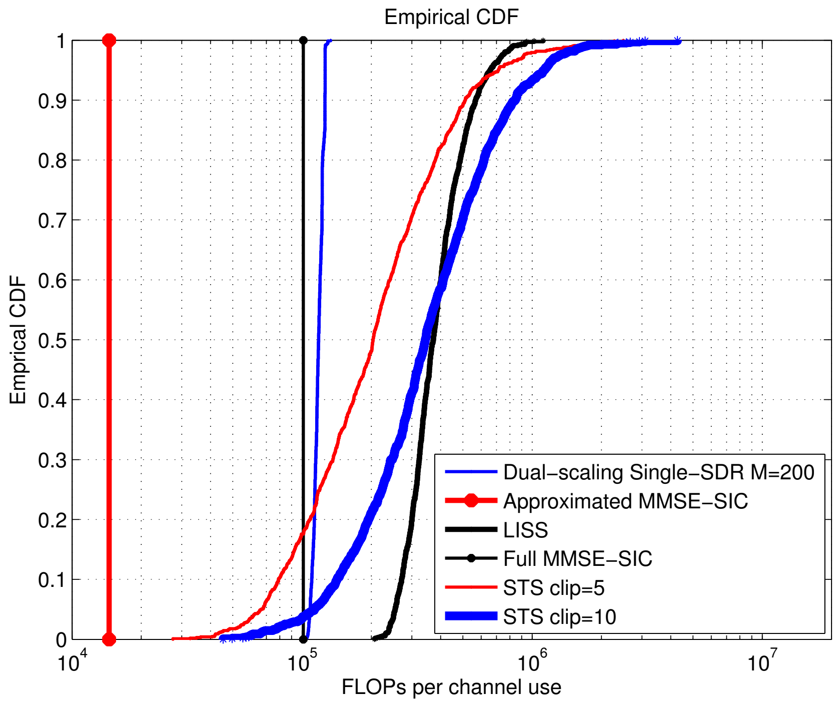

Figure 6.5: Empirical CDF of the number of FLOPs per channel use.

Our results show that, as expected, the LISS method provides the best performance, but as shown in Fig. 6.5 it is among the more computationally expensive methods of those we have examined.

In the STS method, the clipping value plays an important role in the trade-off between performance and computational cost, and this is apparent in our figures, with the STS method with the clipping level equal to 10 providing better performance than the STS method with the clipping level equal to 5, but at greater computational cost. However the dual-scaling single-SDR demodulator provides better performance than both of them, and does so at lower computational cost. Furthermore, the computational cost of the dual-scaling single-SDR demodulator is concentrated around its mean, which can be of considerable benefit when it comes to specifying or managing computational resources.

Finally, we observe that the approximations made in the approximated MMSE-SIC demodulator lead to a significant reduction in the computational cost compared to that of the full MMSE-SIC, but the resulting degradation in performance might not be deemed to be negligible.

Some of the data in Fig. 6.4 and Fig. 6.5 have been summarized in Table 6.1, where we state the additional SNR that each method requires to achieve a BER of 10^{-4} beyond the SNR required by the LISS method. We also state the fraction of the (average) computational cost of the LISS method that each of the demodulators requires.

Some insight into the differences in the performance of these methods can be obtained using the EXIT chart, and we will do that in the next section.

Table 6.1: Comparison of performance and computational cost of different demodulators in comparison with LISS demodulator. The additional SNR is the difference between the SNRs (in dB) required by the method and the LISS method to achieve a BER of 10^{-4}. The fraction of the computational cost is the ratio of the average computational costs of the method and the LISS method when evaluated at 9.75 dB.

Method	Additional SNR (dB)	Fraction of comp. cost
Dual-scaling single-SDR M=200	0.24	0.294
Full MMSE-SIC	0.74	0.252
STS clip=10	1.15	1.107
Approximated MMSE-SIC	1.76	0.036

6.4 EXIT Chart

The extrinsic information transfer chart introduced by ten Brink (2001) is a technique to analyze the behaviour of iterative soft input soft output decoding. Given the "turbo" structure of iterative demodulation and decoding, an analogous analysis can be applied. In an EXIT chart, the mutual information between the input bits and the soft information extracted by the demodulator or decoder is used to observe the progress which is made as the iterations increment. Let B and Ξ denote the random variables representing the input bits and the soft information, respectively. The mutual information is

$$I(B;\Xi) = \frac{1}{2}\sum_{b=\pm 1}\int_{-\infty}^{+\infty} f(\xi|b)\log\frac{f(\xi|b)}{f(\xi)}d\xi, \tag{6.1}$$

where $f(\xi|b)$ is the distribution of the soft information given the input bits.

Let I_A denote the mutual information between the input bits and the soft information that forms the input to the demodulator (or the decoder) and I_E denote the mutual information between the input bits and the extrinsic soft information (at the output) of the demodulator (decoder). The EXIT chart technique depicts the relation between I_A and I_E. In computing I_A, it is usual to model the input $f_A(\xi|b)$ via a Gaussian distribution, (e.g., ten Brink, 2001; Hagenauer, 2004; Lee *et al.*, 2005), but the distribution of $f_E(\xi|b)$ is determined by Monte Carlo simulation and no Gaussian assumption is assumed (ten Brink, 2001). The EXIT chart characteristic can be shown as a plot of $I_E = T(I_A)$.

In Fig. 6.6, we have plotted the EXIT charts of the SDR-based demodulators at an SNR of 9.75 dB. Based on this EXIT chart these methods are expected to have similar performance.

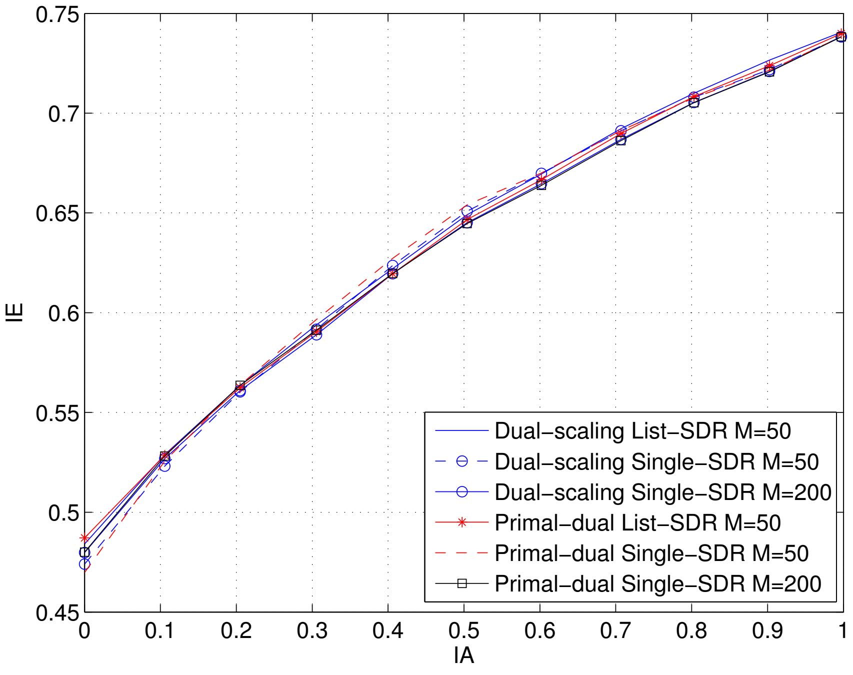

Figure 6.6: EXIT chart of primal-dual interior point and dual-scaling algorithm in both list-SDR and single-SDR cases.

In Fig. 6.7 we have plotted the EXIT chart of the dual-scaling single-SDR demodulator, and the EXIT charts of the other demodulators considered in Fig. 6.4. Fig. 6.7 suggests that the performance of the dual-scaling single-SDR demodulator and that of the LISS method should be significantly better than that of the other methods, and that is indeed the case in Fig. 6.4. The EXIT chart also correctly predicts that the full MMSE-SIC demodulator and the STS demodulator with clipping value equal to 10 will provide better performance than the approximate MMSE-SIC and the STS demodulator with clipping value equal to 5.

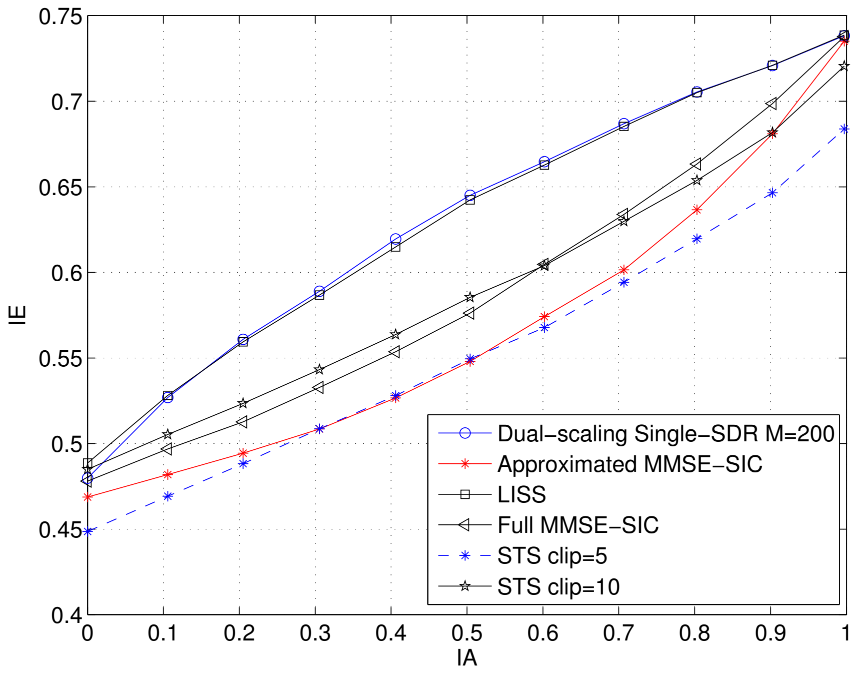

Figure 6.7: EXIT chart of dual-scaling single-SDR demodulator in comparison with some existing methods.

Chapter 7

Conclusions and Future Work

7.1 Conclusion

The main focus of this thesis was on developing a low-complexity high-performance soft demodulator for MIMO communication systems that is based on ideas that are quite distinct from current approaches. Most of the existing approaches are based on either tree search ideas, or the principles of minimum mean square error estimation. The demodulator developed in this thesis is distinct because it is based on the principles of semidefinite relaxation, in which a hard detection problem is approximated by a semidefinite program and a randomization procedure.

For binary signalling schemes, list-based soft demodulators based on semidefinite relaxation were developed by Nekuii *et al.* (2011), and some preliminary extensions to higher order QAM systems have been previously explored (Nekuii and Davidson, 2009b; Nekuii, 2008). In that work, the semidefinite program was solved using a customized primal-dual interior point method. That method is based on solving the primal and the dual problem of semidefinite programming jointly, and at each iteration of the demodulation process both primal and dual variables are updated. In this thesis we developed a lower-complexity method, with the same level of performance, to solve the semidefinite program. The development involved the customization of a dual-scaling algorithm for solving semidefinite program. In this method either the primal value or the dual variables are updated at each iteration. By carefully exploiting the structure exposed by the dual-scaling approach, a customized algorithm that results in significantly reduced computational cost at each iteration, and reduced overall computational cost, was obtained. Further customization and computational cost reduction was obtained by using communication insights to generate effective starting points for the algorithm.

In Chapter 6, we compared the performance and computational cost of different soft demodulation methods. Among the list-SDR and single-SDR classes, it was shown that the proposed dual-scaling algorithm provides a tangible reduction in the computational cost. In the case of the list-SDR demodulator, in which one SDP is solved per iteration, the reduction is quite significant. That said, the single-SDR demodulator, in which an SDP is solved only once per channel use, is significantly cheaper than the list-SDR approach. When 50 randomizations are used, the degradation in the performance of the single-SDR method related to the list-SDR is not necessarily negligible, but much of that degradation can be recovered, without giving up too much of the computational advantage, by employing 200 randomizations.

We also compared the performance and computational cost of the dual-scaling single-SDR demodulator with some other existing methods. It was seen that the dual-scaling single-SDR demodulator provides better performance than all of the other methods we considered, except the LISS method, and does so at a computational cost that is about 1/4 of the median cost of the LISS method, is cheaper than the single-tree search (STS) method, and is about the same as that of the (full) MMSE-SIC method.

Insight into the differences in performance of these different approaches was obtained from an EXIT chart analysis of the considered methods. The (dual-scaling) single-SDR demodulator was found to have an EXIT characteristic that is similar to that of the LISS method, and superior to those of the two instances of the STS method that we considered and those of the full and approximate MMSE-SIC.

7.2 Future Work

This thesis developed a customized low-complexity algorithm to solve the semidefinite program that arises in semidefinite relaxation approach to soft MIMO demodulation. The keys to the customization were to exploit the structure of the same matrices in the generic algorithm, and to use some communication insights to develop effective initialization strategies. Although several aspects of the structure of the matrices involved in dual algorithm are exploited in this thesis, there are still some unexplored points that might be able to be used to reduce the computational cost or increase the performance of the algorithm. We may consider the following points as future research:

- As can be seen in (5.37), the matrix $\mathbf{M}$ is a structured matrix. One possible point for future work could be the development of techniques that can exploit this structure in the calculation of its inverse, in order to lower the computational cost.

- In order to reduce the computational cost, we can also evaluate approximations to some of the expensive steps of the algorithm. Calculation of the inverse of the matrix **M**, and the computation of the optimal primal variable at the last iteration of the algorithm are the examples of the expensive steps in the algorithm. We can explore the trade-off between the performance degradation and computational cost reduction of the algorithm in the case of using approximations in different steps of the algorithm.

- The customized dual-scaling algorithm proposed in this thesis is in the general format, which means that by changing the constellation parameters we can apply it directly to the higher order constellations. One direction to future work would be to explore the performance and computational cost of the algorithm in the context of higher order constellation (i.e, 64-QAM). Based on the structure of matrix **S** in (5.30) and the structure of matrix **M** in (5.37) and also diagonal matrices in these structures, there may be some additional advantages of using the dual-scaling algorithm in comparison to the primal-dual interior point method. As the cost of each iteration of dual-scaling algorithm is lower than the primal-dual and this difference in computational cost may be more significant in higher order constellations, a larger gap in computational cost graph between primal-dual and dual-scaling algorithm might be observed.

- It was explained that for the first few iteration we use a step size α that is greater than one. In this case Lemma 2 cannot guarantee that the updated primal variable is feasible and we have already accepted the risk of infeasibility. It may be a direction for future work to consider the option of bigger step sizes, especially if the structure of the SDP can be exploited to guarantee that the updated primal and dual variables would be feasible.

- One other important point for future work is the channel matrix structure. We don't explore the structure of the channel to take an advantages of cases in which the channel is constant over several channel uses. Finding a way to take advantage of this could have a substantial impact on the computational cost of the algorithm.

- One other direction of the further work is the implementation of these methods either in a general purpose processor or in dedicated hardware. The comparison of the performance and computational cost of these implementations, and their comparison with existing implementations would be of considerable interests.

Appendix A

Dual Problem

The derivation of the dual problem associated with the problem in (5.1) begins with the Lagrangian, which can be written as

$$\mathcal{L}(\mathbf{X}, \mathbf{S}, \mathbf{y}) = \mathrm{Tr}(\mathbf{C}\mathbf{X}) - \sum_i (\mathrm{Tr}(\mathbf{A}_i\mathbf{X}) - e_i)y_i - \mathrm{Tr}(\mathbf{S}\mathbf{X}). \tag{A.1}$$

The Lagrangian dual function is defined as the infimum of (A.1) with respect to the primal variable; i.e, $g(\mathbf{y}, \mathbf{S}) = \inf_{\mathbf{X}} \mathcal{L}(\mathbf{X}, \mathbf{S}, \mathbf{y})$. Since the Lagrangian in (A.1) is a linear function of $\mathbf{X}$, we can determine the minimizing (if there are any finite minimizers) by finding the derivative and setting it equal to zero; i.e., $\frac{\partial \mathcal{L}(\mathbf{X},\mathbf{S},\mathbf{y})}{\partial \mathbf{X}} = \mathbf{0}$. Using standard expressions from matrix calculus (e.g., Brewer, 1978), that results in the equation

$$\mathbf{C} - \sum_i y_i\mathbf{A}_i - \mathbf{S} = \mathbf{0}. \tag{A.2}$$

Therefore, dual function can be written as

$$g(\mathbf{y}, \mathbf{S}) = \inf_{\mathbf{X}} \mathcal{L}(\mathbf{X}, \mathbf{S}, \mathbf{y}) = \begin{cases} \mathbf{e}^T\mathbf{y} & \text{if } \mathbf{C} - \sum_i y_i \mathbf{A}_i - \mathbf{S} = \mathbf{0} \\ -\infty & \text{otherwise,} \end{cases} \tag{A.3}$$

where $\mathbf{e} = [e_1, e_2, \ldots, e_m]$.

Hence, the dual problem can be written as

$$\max_{\mathbf{y}} \quad \mathbf{e}^T\mathbf{y} \tag{A.4a}$$

$$\text{s.t} \quad \sum_{i=1}^{m} y_i \mathbf{A}_i + \mathbf{S} = \mathbf{C}, \tag{A.4b}$$

$$\mathbf{S} \succeq 0. \tag{A.4c}$$

Appendix B

Connection of Different Representations of Dual Variables

The KKT conditions of the problem in (4.33) are given in (4.36). These conditions involve the variables of the dual of (4.33), which appears in (4.35). In order to connect the variables of that dual problem (4.35) to the variables of the dual problem of the problem in generic form in (5.7), we have

$$\mathbf{y}^T = [\mathbf{p}_\ell, -\mathbf{p}_u, -\nu]^T. \tag{B.1}$$

which results in

$$\mathbf{S}_1 = \mathbf{Z}, \tag{B.2}$$

where $\mathbf{S}_1$ is the upper left block of matrix $\mathbf{S}$.

Appendix C

Dual Initial Point Using Positive Semidefinite Constraint on S

There are a variety of ways in which the positive semidefinite constraint on **S** and Gerschorin's Theorem can be used to generate a feasible initial point for the dual-scaling algorithm. Here we describe the approach used in our simulations.

As defined in Chapter 5, **S** is a block matrix of the form

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{S}_3 \end{bmatrix}, \tag{C.1}$$

where $\mathbf{S}_2$ and $\mathbf{S}_3$ are diagonal with elements dependent on $\{y_i\}$, as stated in (5.28) and (5.30). Gerschorin's Theorem shows that each eigenvalue of a matrix lies at least in one of the circles with center equal to diagonal elements of the matrix and radius equal to summation of the row corresponding to that diagonal element. As our goal is to choose a strictly positive definite $\mathbf{S}$, we would like to ensure that all of its eigenvalues are positive. Therefore, the first result of the positive definite constraint is to define the sign of each y_i. This arises from the fact that the elements of the diagonal matrices $\mathbf{S}_2$ and $\mathbf{S}_3$ are also eigenvalues of $\mathbf{S}$. Considering (5.28) we have

$$\begin{cases} y_i > 0 & \text{for } 1 \leq i \leq 2N_t, \\ y_i < 0 & \text{for } 2N_t + 1 \leq i \leq 4N_t. \end{cases}$$

The second result is provided by applying Gerschorin's Theorem to the first block of $\mathbf{S}$. Considering (5.30), we require

$$-(y_i + y_{2N_t+i}) > \sum_{i \neq j} |\tilde{\mathbf{Q}}_{ij}| - \tilde{\mathbf{Q}}_{ii}. \tag{C.2}$$

By adding positive slack variables, $\gamma_{1,i} > 0$, that expression can be written as an equality,

$$-(y_i + y_{2N_t+i}) = \sum_{i \neq j} |\tilde{\mathbf{Q}}_{ij}| - \tilde{\mathbf{Q}}_{ii} + \gamma_{1,i}. \tag{C.3}$$

One way to find values for $\{y_i\}$ that satisfy (C.3) is to choose a unique constant γ_1 for all i. To do so we should find the i for which the $\tilde{\mathbf{Q}}_{ii}$ is the smallest. If we let ℓ denote the i for which $\tilde{\mathbf{Q}}_{ii}$ is the smallest, then we can write

$$-(y_i + y_{2N_t+i}) \geq \sum_{i \neq j} |\tilde{\mathbf{Q}}_{ij}| - \tilde{\mathbf{Q}}_{\ell\ell} \geq \sum_{i \neq j} |\tilde{\mathbf{Q}}_{ij}| - \tilde{\mathbf{Q}}_{ii}. \tag{C.4}$$

Then we choose γ_1 so that $\sum_{i \neq j} |\tilde{\mathbf{Q}}_{ij}| - \tilde{\mathbf{Q}}_{\ell\ell} + \gamma_1 > 0, \forall j$ and impose the equality constraint

$$-(y_i + y_{2N_t+i}) = \sum_{i \neq j} |\tilde{\mathbf{Q}}_{ij}| - \tilde{\mathbf{Q}}_{\ell\ell} + \gamma_1. \tag{C.5}$$

If we constrain the value for $\{y_i\}$ in this way, we can be sure that (C.2) holds for all i. To find the specific values for y_i and y_{2N_t+i} that satisfy (C.5), we choose a non-negative constant which we denote by γ_2 and define $\gamma_{3,i}$ as $\gamma_{3,i} = \gamma_2 + (\sum_{i\neq j} |\tilde{\mathbf{Q}}_{ij}| - \tilde{\mathbf{Q}}_{\ell\ell} + \gamma_1)$. Then we choose y_i according to

$$\begin{cases} y_i = \sum_{i\neq j} |\tilde{\mathbf{Q}}_{ij}| + \gamma_2, & \text{for } 1 \leq i \leq 2N_t, \\ y_i = -(\sum_{i\neq j} |\tilde{\mathbf{Q}}_{ij}| + \gamma_{3,i}), & \text{for } 2N_t + 1 \leq i \leq 4N_t, \\ y_i = -(\sum_{i\neq j} |\tilde{\mathbf{Q}}_{i,j}| - \tilde{\mathbf{Q}}_{i,i} + \gamma_2), & \text{for } i = 4N_t + 1. \end{cases}$$

In our simulations we choose $\gamma_1 = 1$ and $\gamma_2 = 1$.

Bibliography

Atkinson, K. E. (2008). *An introduction to numerical analysis.* John Wiley & Sons.

Bahl, L., Cocke, J., Jelinek, F., and Raviv, J. (1974). Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Trans. Inf. Theory*, **20**(2), 284–287.

Barbero, L. G. and Thompson, J. S. (2008a). Extending a fixed-complexity sphere decoder to obtain likelihood information for turbo-MIMO systems. *IEEE Trans. Veh. Technol.*, **57**(5), 2804–2814.

Barbero, L. G. and Thompson, J. S. (2008b). Fixing the complexity of the sphere decoder for MIMO detection. *IEEE Trans. Wireless Commun.*, **7**(6), 2131–2142.

Benson, S. J., Ye, Y., and Zhang, X. (2000). Solving large-scale sparse semidefinite programs for combinatorial optimization. *SIAM J. Optim.*, **10**(2), 443–461.

Boutros, J. and Caire, G. (2002). Iterative multiuser joint decoding: Unified framework and asymptotic analysis. *IEEE Trans. Inf. Theory*, **48**(7), 1772–1793.

Boyd, S. P. and Vandenberghe, L. (2004). *Convex optimization.* Cambridge University Press.

Brewer, J. (1978). Kronecker products and matrix calculus in system theory. *IEEE Trans. Circuits Syst. I*, **25**(9), 772–781.

Caire, G., Taricco, G., and Biglieri, E. (1998). Bit-interleaved coded modulation. *IEEE Trans. Inf. Theory*, **44**(3), 927–946.

Cirkic, M. and Larsson, E. G. (2012). SUMIS: A near-optimal soft-ouput MIMO detector at low and fixed complexity. Available online at: `http://arxiv.org/abs/1207.3316`.

Costello, D. J. and Forney Jr, G. D. (2007). Channel coding: The road to channel capacity. *Proc. IEEE*, **95**(6), 1150–1177.

de Jong, Y. L. and Willink, T. J. (2005). Iterative tree search detection for MIMO wireless systems. *IEEE Trans. Commun.*, **53**(6), 930–935.

Goldsmith, A. (2005). *Wireless communications*. Cambridge University Press.

Grant, M. and Boyd, S. (2008). Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited. `http://stanford.edu/~boyd/graph_dcp.html`.

Grant, M. and Boyd, S. (2014). CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`.

Guo, Z. and Nilsson, P. (2006). Algorithm and implementation of the K-best sphere decoding for MIMO detection. *IEEE J. Sel. Areas Commun.*, **24**(3), 491–503.

Hagenauer, J. (1997). The turbo principle: Tutorial introduction and state of the art. In *Proc. Int. Symp. Turbo Codes and Related Topics*, pages 1–11, Brest, France.

Hagenauer, J. (2004). The EXIT chart—Introduction to extrinsic information transfer in iterative processing. In *Proc. Eur. Signal Process. Conf.*, pages 1541–1548, Vienna.

Hagenauer, J. and Kuhn, C. (2007). The list-sequential (LISS) algorithm and its application. *IEEE Trans. Commun.*, **55**(5), 918–928.

Hanzo, L., Liew, T. H., and Yeap, B. L. (2002). *Turbo coding, turbo equalisation and space-time coding*. John Wiley.

Hassibi, B. and Hochwald, B. M. (2002). High-rate codes that are linear in space and time. *IEEE Trans. Inf. Theory*, **48**(7), 1804–1824.

Helmberg, C., Rendl, F., Vanderbei, R. J., and Wolkowicz, H. (1996). An interior-point method for semidefinite programming. *SIAM J. Optim.*, **6**(2), 342–361.

Hochwald, B. M. and ten Brink, S. (2003). Achieving near-capacity on a multiple-antenna channel. *IEEE Trans. Commun.*, **51**(3), 389–399.

Hu, J. and Duman, T. M. (2008). Graph-based detection algorithms for layered space-time architectures. *IEEE J. Sel. Areas Commun.*, **26**(2), 269–280.

i Fàbregas, A. G., Martinez, A., and Caire, G. (2008). Bit-interleaved coded modulation. *Found. Trends Commun. Inf. Theory*, **5**(1–2), 1–153.

Jelinek, F. (1969). Fast sequential decoding algorithm using a stack. *IBM J. Res. Dev*, **13**(6), 675–685.

Kisialiou, M., Luo, X., and Luo, Z.-Q. (2009). Efficient implementation of quasi-maximum-likelihood detection based on semidefinite relaxation. *IEEE Trans. Signal Process.*, **57**(12), 4811–4822.

Kschischang, F. R., Frey, B. J., and Loeliger, H.-A. (2001). Factor graphs and the sum-product algorithm. *IEEE Trans. Inf. Theory*, **47**(2), 498–519.

Larsson, E. G. (2009). MIMO detection methods: How they work. *IEEE Signal Process. Mag.*, **26**(3), 91–95.

Larsson, E. G. and Jalden, J. (2008). Fixed-complexity soft MIMO detection via partial marginalization. *IEEE Trans. Signal Process.*, **56**(8), 3397–3407.

Lee, S.-J., Singer, A. C., and Shanbhag, N. R. (2005). Linear turbo equalization analysis via BER transfer and EXIT charts. *IEEE Trans. Signal Process.*, **53**(8), 2883–2897.

Lin, S. and Costello, D. J. (2004). *Error control coding*. Pearson Education.

Liu, L., Lofgren, J., and Nilsson, P. (2012). Low-complexity likelihood information generation for spatial-multiplexing MIMO signal detection. *IEEE Trans. Veh. Technol.*, **61**(2), 607–617.

Lozano, A. and Jindal, N. (2010). Transmit diversity vs. spatial multiplexing in modern MIMO systems. *IEEE Trans. Wireless Commun.*, **9**(1), 186–197.

Luo, Z.-Q., Ma, W.-K., So, A.-C., Ye, Y., and Zhang, S. (2010). Semidefinite relaxation of quadratic optimization problems. *IEEE Signal Process. Mag.*, **27**(3), 20–34.

Ma, W.-K., Davidson, T. N., Wong, K. M., Luo, Z.-Q., and Ching, P.-C. (2002). Quasi-maximum-likelihood multiuser detection using semi-definite relaxation with application to synchronous CDMA. *IEEE Trans. Signal Process.*, **50**(4), 912–922.

Ma, W.-K., Ching, P.-C., and Ding, Z. (2004). Semidefinite relaxation based multiuser detection for M-ary PSK multiuser systems. *IEEE Trans. Signal Process.*, **52**(10), 2862–2872.

Ma, W.-K., Su, C.-C., Jaldén, J., Chang, T.-H., and Chi, C.-Y. (2009). The equivalence of semidefinite relaxation MIMO detectors for higher-order QAM. *IEEE J. Sel. Topics Signal Process.*, **3**(6), 1038–1052.

Murugan, A. D., El Gamal, H., Damen, M. O., and Caire, G. (2006). A unified framework for tree search decoding: Rediscovering the sequential decoder. *IEEE Trans. Inf. Theory*, **52**(3), 933–953.

Nekuii, M. (2008). *Soft demodulation schemes for MIMO communication systems.* Ph.D. thesis, McMaster University.

Nekuii, M. and Davidson, T. N. (2009a). A multistack algorithm for soft MIMO demodulation. *IEEE Trans. Veh. Technol.*, **58**(5), 2592–2597.

Nekuii, M. and Davidson, T. N. (2009b). A semidefinite relaxation approach to efficient soft demodulation of MIMO 16-QAM. In *Proc. IEEE Int. Conf. Commun.*, Dresden.

Nekuii, M., Kisialiou, M., Davidson, T. N., and Luo, Z.-Q. (2011). Efficient soft-output demodulation of MIMO QPSK via semidefinite relaxation. *IEEE J. Sel. Topics Signal Process.*, **5**(8), 1426–1437.

Persson, D. and Larsson, E. G. (2011). Partial marginalization soft MIMO detection with higher order constellations. *IEEE Trans. Signal Process.*, **59**(1), 453–458.

Richardson, T. and Urbanke, R. L. (2008). *Modern coding theory*. Cambridge University Press.

Shieh, S.-L., Chiu, R.-D., Feng, S.-L., and Chen, P.-N. (2013). Low-complexity soft-output sphere decoding with modified repeated tree search strategy. *IEEE Commun. Lett.*, **17**(1), 51–54.

Sidiropoulos, N. D. and Luo, Z.-Q. (2006). A semidefinite relaxation approach to MIMO detection for high-order QAM constellations. *IEEE Signal Process. Lett.*, **13**(9), 525–528.

Steingrimsson, B., Luo, Z.-Q., and Wong, K. M. (2003). Soft quasi-maximum-likelihood detection for multiple-antenna channels. *IEEE Trans. Signal Process.*, **51**(11), 2710–2719.

Studer, C. and Bolcskei, H. (2010). Soft–input soft–output single tree-search sphere decoding. *IEEE Trans. Inf. Theory*, **56**(10), 4827–4842.

Studer, C., Fateh, S., and Seethaler, D. (2011). ASIC implementation of soft-input soft-output MIMO detection using MMSE parallel interference cancellation. *IEEE J. Solid-State Circuits*, **46**(7), 1754–1765.

Tan, P. H. and Rasmussen, L. K. (2001). The application of semidefinite programming for detection in CDMA. *IEEE J. Sel. Areas Commun.*, **19**(8), 1442–1449.

ten Brink, S. (2001). Convergence behavior of iteratively decoded parallel concatenated codes. *IEEE Trans. Commun.*, **49**(10), 1727–1737.

Tse, D. and Viswanath, P. (2005). *Fundamentals of wireless communication*. Cambridge University Press.

Tuchler, M., Singer, A. C., and Koetter, R. (2002). Minimum mean squared error equalization using a priori information. *IEEE Trans. Signal Process.*, **50**(3), 673–683.

Vikalo, H., Hassibi, B., and Kailath, T. (2004). Iterative decoding for MIMO channels via modified sphere decoding. *IEEE Trans. Wireless Commun.*, **3**(6), 2299–2311.

Wang, R. and Giannakis, G. B. (2006). Approaching MIMO channel capacity with soft detection based on hard sphere decoding. *IEEE Trans. Commun.*, **54**(4), 587–590.

Wang, X. and Poor, H. V. (1999). Iterative (turbo) soft interference cancellation and decoding for coded CDMA. *IEEE Trans. Commun.*, **47**(7), 1046–1061.

Wiesel, A., Eldar, Y. C., and Shitz, S. (2005). Semidefinite relaxation for detection of 16-QAM signaling in MIMO channels. *IEEE Signal Process. Lett.*, **12**(9), 653–656.

Wubben, D., Bohnke, R., Rinas, J., Kuhn, V., and Kammeyer, K. (2001). Efficient algorithm for decoding layered space-time codes. *Electron. Lett.*, **37**(22), 1348–1350.

Ye, Y. (2011). *Interior point algorithms: Theory and analysis*. John Wiley & Sons.

Zheng, L. and Tse, D. N. C. (2003). Diversity and multiplexing: A fundamental trade-off in multiple-antenna channels. *IEEE Trans. Inf. Theory*, **49**(5), 1073–1096.