V-BLAST/MAP: A NEW SYMBOL DETECTION ALGORITHM FOR MIMO CHANNELS

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND

ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Yavuz Yapıcı

2005, January

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Erdal Arıkan (Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Tolga Duman

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Defne Aktaş

Approved for the Institute of Engineering and Science:

Prof. Dr. Mehmet Baray
Director of the Institute

ABSTRACT

V-BLAST/MAP: A NEW SYMBOL DETECTION ALGORITHM FOR MIMO CHANNELS

Yavuz Yapıcı
M.S. in Electrical and Electronics Engineering
Supervisor: Prof. Dr. Erdal Arıkan
2005, January

In this thesis, we introduce a new symbol detection algorithm for MIMO channels. This algorithm is an extension of the well-known V-BLAST algorithm. The new algorithm, V-BLAST/MAP, combines elements of the V-BLAST algorithm and the maximum a-posteriori (MAP) rule.

The original V-BLAST algorithm is a multi-layer symbol detection scheme which detects symbols transmitted at different transmit antennas successively in a certain data-independent order. The proposed V-BLAST/MAP algorithm differs from V-BLAST only in the ordering strategy of the symbols detected. The complexity of the V-BLAST/MAP is higher than that of V-BLAST; however, the performance improvement is also significant. Simulations show that V-BLAST/MAP achieves symbol error rates close to the optimal maximum likelihood (ML) scheme while retaining the low-complexity nature of the V-BLAST.

Keywords: MIMO, ML, V-BLAST, MAP .

ÖZET

V-BLAST/MAP: MIMO KANALLARI İÇİN YENİ BİR SEMBOL ALGILAMA ALGORİTMASI

Yavuz Yapıcı
Elektronik Mühendisliği, Yüksek Lisans
Tez Yöneticisi: Prof. Dr. Erdal Arıkan
2005, Ocak

Bu tezde, MIMO kanalları için yeni bir sembol algılama algoritması tanıtılacaktır. Aslında bu algoritma, çok iyi bilinen V-BLAST algoritmasına bir ilavedir. V-BLAST/MAP ismi verilen yeni algoritma, V-BLAST algoritması ve maksimum a-posteriori (MAP) kuralının öğelerini birleştirmektedir.

Orjinal V-BLAST algoritması, çok katmanlı bir sembol algılama algoritması olup birbirinden farklı iletici antenlerden iletilen sembolleri veriye dayanmayan belli bir sıralamayla algılamaktadır. Sunulan V-BLAST/MAP algoritması, V-BLAST'tan yalnızca sembollerin algılama sırasıyla farklılaşmaktadır. V-BLAST/MAP algoritmasının kompleksitesi, V-BLAST'ınkinden daha yüksektir; fakat, performans iyileşmesi de kayda değerdir. Simülasyonlar, V-BLAST/MAP'in sembol hata oranlarının optimal çalışan maksimum şanslılık (ML) kuralının sonuçlarına yakın olduğunu, bununla birlikte V-BLAST algoritmasının düşük kompleksi-tedeki doğasının da korunduğunu göstermektedir.

Anahtar sözcükler: MIMO, ML, V-BLAST, MAP.

Acknowledgement

I would like to express my thanks to my supervisor Prof.Dr. Erdal Arıkan for his invaluable advice and comments, especially in revising my writing. His guidance allowed me to complete this work. I am deeply grateful to Dr. Tolga Duman and Dr. Defne Aktaş for accepting to be on my thesis committee and providing useful advice. I appreciate other faculty members, staffs and colleagues. Finally, I thank my parents for their supports and endless love through my life.

Contents

Glossary of Terms

M: Number of transmitter antennas

N: Number of receiver antennas

a: Sent symbol vector

r: Received symbol vector

H: Channel transfer matrix

v: Noise vector

N_0: Noise power

$\mathcal{CN}(\mu, \sigma^2)$: Complex Gaussian distribution with mean μ and variance σ^2

CSI: Channel State Information

P_e: Probability of decision error

SER: Symbol Error Rate

$\mathcal{A}$: Modulation alphabet

ρ/M: Average energy of a constellation point

ρ: Average received energy for each symbol

E_b: Average bit energy

SNR: Signal to interference ratio

W: Weighting matrix

$\mathbf{H}^+$: Moore-Penrose pseudo-inverse of **H**

$Q(\cdot)$: Quantizer

$\mathbf{H}^\dagger$: Hermitian transpose of **H**

p_{ij}: Reliability probability of j'th subchannel at the i'th layer

List of Figures

Chapter 1

Introduction

Recent research on wireless communication systems has shown that using multiple antennas at both transmitter and receiver offers the possibility of wireless communication at higher data rates compared to single antenna systems. The information-theoretic capacity of these multiple-input multiple-output (MIMO) channels was shown to grow linearly with the smaller of the numbers of transmit and receiver antennas in rich scattering environments, and at sufficiently high signal-to-noise (SNR) ratios [1].

Some special detection algorithms have been proposed in order to exploit the high spectral capacity offered by MIMO channels. One of them is the V-BLAST (Vertical Bell-Labs Layered Space-Time) algorithm which uses a layered structure [2]. This algorithm offers highly better error performance than conventional linear receivers and still has low complexity. In this thesis, we offer a new symbol detection algorithm called V-BLAST/MAP that has a layered structure as V-BLAST, but uses a modified detecting algorithm that yields a better error performance than V-BLAST at slightly higher complexity.

In this chapter, we state the MIMO channel model that will be used throughout this thesis, state the MIMO symbol detection problem, present some brief description of previous detection algorithms and briefly compare their error performance with that of V-BLAST/MAP. These topics are considered in detail in

the following chapters.

1.1 The MIMO Channel Model

Throughout this thesis, we use the MIMO channel model depicted in Fig. 1.1 with M transmitter and N receiver antennas.

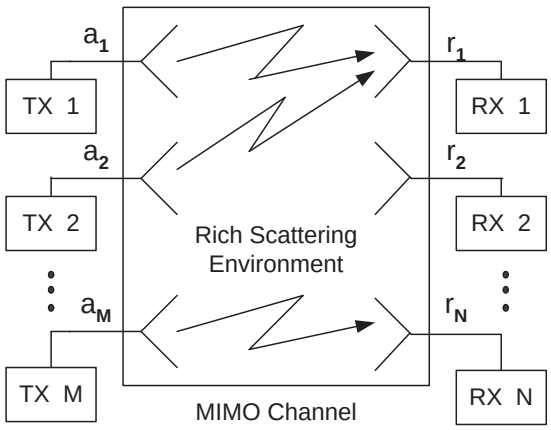

Figure 1.1: Multiple Input Multiple Output (MIMO) channel model. TX and RX stand for transmitter and receiver antennas, respectively.

In each use of the MIMO channel, a vector $\mathbf{a} = (a_1, a_2, \ldots, a_M)^T$ of complex numbers is sent and a vector $\mathbf{r} = (r_1, r_2, \ldots, r_N)^T$ of complex numbers is received. We assume an input-output relationship of the form

$$\mathbf{r} = \mathbf{H}\mathbf{a} + \mathbf{v} \tag{1.1}$$

where $\mathbf{H}$ is a $N \times M$ matrix representing the scattering effects of the channel and $\mathbf{v} = (v_1, v_2, \ldots, v_N)^T$ is the noise vector. Throughout, we assume that $\mathbf{H}$ is a random matrix with independent complex Gaussian elements $\{h_{ij}\}$ with mean 0 and unit variance, denoted $h_{ij} \sim \mathcal{CN}(0, 1)$. We also assume throughout that $\mathbf{v}$ is a complex Gaussian random vector with i.i.d. elements $v_i \sim \mathcal{CN}(0, N_0)$. It is assumed that $\mathbf{H}$ and $\mathbf{v}$ are independent of each other and of the data vector $\mathbf{a}$.

We will assume that the receiver has perfect knowledge of the channel realization **H**, while the transmitter has no such channel state information (CSI). Receiver's possession of CSI is justified in cases where the channel is a relatively slowly time-varying random process; see [3] for a discussion of this point.

1.2 The Symbol Detection Problem

The symbol detection problem considered in this thesis is the problem of estimating the MIMO channel input vector **a** given the received vector **r** assuming that the receiver has perfect knowledge of **H**. This decision is made on a symbol by symbol basis without taking into account any statistical dependencies that may be present in the sequence of vectors **a**. In other words, we exclude coding across the time dimension and consider only the modulation-demodulation problem as depicted in Fig. 1.2. The goal is to minimize the probability of decision error

$$P_e = Pr\{\hat{\mathbf{a}} \neq \mathbf{a}\} \tag{1.2}$$

where $\hat{\mathbf{a}} = (\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_M)^T$ is the demodulator's estimate of **a**.

Figure 1.2: Modulation, transmission and decision in MIMO wireless systems.

We study the above detection problem under the additional assumptions on the input vector **a** that:

(i) Each element of **a** belongs to a common modulation alphabet $\mathcal{A}$, $a_i \in \mathcal{A}$, $i = 1, \ldots, M$, $\mathbf{a} \in \mathcal{A}^M$. Typically, $\mathcal{A}$ will be a QAM alphabet such as $\mathcal{A} = \{\pm A \pm j\, A\}$ as in the case of 4-QAM.

(ii) We will assume that symbols in $\mathcal{A}$ have equal a priori probabilities.

(iii) The vector $\mathbf{a}$ is a random vector over $\mathcal{A}^M$ such that

$$\mathcal{E}\{\mathbf{a}\mathbf{a}^\dagger\} = \frac{\rho}{M}\,\mathbf{I}_M \tag{1.3}$$

where ρ is a constant, $\mathbf{I}_M$ is the identity matrix of size M, $\mathcal{E}\{.\}$ is the expectation operator and $\mathbf{a}^\dagger$ denotes Hermitian transpose of $\mathbf{a}$. Assumption (1.3) implies that the elements of $\mathbf{a}$ are uncorrelated and each has *energy*

$$\mathcal{E}\{\,|\,a_i\,|^2\,\} \;=\; \rho/M \tag{1.4}$$

yielding a total average transmitted energy of ρ per symbol, combined over all antennas.

The parameter ρ also has the significance of being the average received energy per symbol E_s at each receiver antenna, as can be seen by computing the energy at receiver antenna i:

$$\begin{aligned} E_s &= \mathcal{E}\Big\{\Big|\sum_j h_{ij}\,a_j\Big|^2\Big\} \\ &= \mathcal{E}\Big\{\sum_j\sum_k h_{ij}\,h_{ik}^*\,a_j\,a_k^*\Big\} \\ &= \sum_j\sum_k \mathcal{E}\big(h_{ij}\,h_{ik}^*\big)\mathcal{E}\big(a_j\,a_k^*\big) \\ &= \sum_j \mathcal{E}\big(|\,a_j|^2\big) \\ &= \rho \end{aligned} \tag{1.5}$$

Using above equation, the average received energy per bit at each receiver antenna may be computed as

$$E_b = \frac{E_s}{\log_2|\mathcal{A}|} \tag{1.6}$$

and receiver signal-to-noise ratio (SNR) is defined as

$$SNR = \frac{E_b}{N_0} = \frac{\rho/\log_2|\mathcal{A}|}{N_0} \tag{1.7}$$

While designing a receiver structure for this MIMO system, two main considerations that should be taken into account are the error performance and the implementation complexity. The aim of this thesis is to design a receiver structure that is powerful in terms of error performance and is practical to implement.

1.3 Some Detection Algorithms

For the signal detection problem defined in the previous section, one decision rule is the *maximum a posteriori probability* (MAP) *rule* defined as

$$\hat{\mathbf{a}} = \arg\max_{\mathbf{a}' \,\epsilon\, \mathcal{A}^M} \big\{ Pr\big(\mathbf{a}' \,|\, \mathbf{r} \text{ is received} \big) \big\} \tag{1.8}$$

It is well-known that the MAP rule minimizes the probability of error P_e (see, e.g., [4, p. 324]).

Another decision rule is the *maximum likelihood* (ML) *rule* defined as

$$\begin{aligned} &\text{Set } \hat{\mathbf{a}} = \mathbf{a}' \,\epsilon\, \mathcal{A}^M \text{ for some } \mathbf{a}' \text{ so that} \\ &f\big(\mathbf{r} \,|\, \mathbf{a}' \big) \geq f\big(\mathbf{r} \,|\, \mathbf{a}'' \big) \text{ for all } \mathbf{a}'' \,\epsilon\, \mathcal{A}^M. \end{aligned} \tag{1.9}$$

where

$$f\big(\mathbf{r} \,|\, \mathbf{a} \big) = \frac{1}{(2\pi)^N N_0^N} \exp\big\{ -\frac{1}{N_0} \left\| \mathbf{Ha} - \mathbf{r} \right\|^2 \big\} \tag{1.10}$$

since $\mathbf{v} \sim \mathcal{CN}(\mathbf{0}, N_0 \mathbf{I}_N)$. Thus, the ML rule here reduces to

$$\hat{\mathbf{a}} = \arg\min_{\mathbf{a}' \,\epsilon\, \mathcal{A}^M} \big\{ \left\| \mathbf{Ha} - \mathbf{r} \right\|^2 \big\} \tag{1.11}$$

In fact, ML rule is equivalent to MAP rule if all the source symbols are equally likely to be transmitted a-priori.

Although MAP rule offers optimal error performance, it suffers from complexity issues. It has exponential complexity in the sense that the receiver has to consider $|\mathcal{A}|^M$ possible symbols for an M transmitter antenna system. For example, if 64-QAM is used with 4 transmit antennas, then a straightforward

implementation of the MAP detector needs to search over $64^4 = 16,777,216$ symbols. Similar complexity problems apply to ML detectors. Note that the difficulty of detection is caused by entanglement of the elements of $\mathbf{a}$ through multiplication by $\mathbf{H}$.

In order to solve the detection problem in MIMO systems, research has focused on sub-optimal receiver models which are powerful in terms of error performance and are practical for implementation purposes, as well. One such receiver is the V-BLAST (Vertical Bell-Labs Layered Space-Time) receiver which utilizes a layered architecture and applies *successive cancellation* by splitting the channel vertically [5]. This algorithm will be described in detail in Chapter 2.

1.4 Thesis Contribution

In this thesis, we propose a new symbol detection algorithm for MIMO systems which combines features of MAP and V-BLAST rules. We call this algorithm V-BLAST/MAP. This new algorithm offers better error performance than V-BLAST at the expense of increased but still practical level of complexity. V-BLAST/MAP has a layered structure as V-BLAST, but also incorporates features of the MAP rule.

Fig. 1.3 depicts the error performance of V-BLAST/ZF/MAP versus those of V-BLAST/ZF and ML for the case of (M,N)=(4,12) and 4-QAM modulation with alphabet $\{\pm A \pm jA\}$. The vertical axis is symbol error rate (SER) which equals P_e as defined in (1.2). The horizontal axis is marked by the SNR or E_b/N_0 as defined in (1.7).

1.5 Thesis Outline

The remainder of this thesis organized as follows. A brief review of previous detection algorithms for MIMO channels are presented in Chapter 2. The new

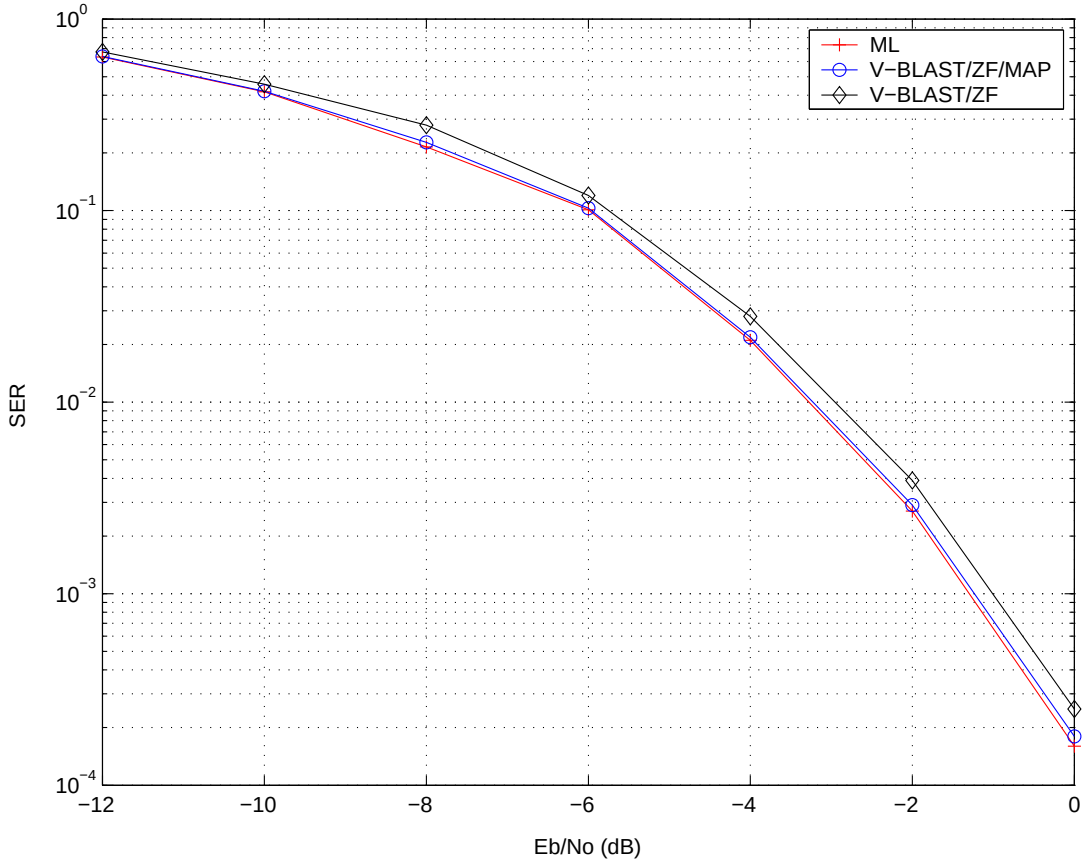

Figure 1.3: Symbol error rates (SER) of V-BLAST/ZF/MAP receiver, V-BLAST/ZF receiver and ML receiver. The simulation is for (M,N)=(4,12) and 4-QAM modulation.

symbol detection algorithm called V-BLAST/MAP is studied in detail in Chapter 3. Some concluding remarks are made also in Chapter 3.

Chapter 2

Previous Detection Algorithms for MIMO Channels

Throughout this chapter, we use the MIMO channel model and the notation of Chapter 1 with the assumptions *(i)-(iii)* of Section 1.2 on the channel input.

As pointed out in Section 1.3, the decision rule that minimizes the probability of symbol error P_e, which is defined in Eqn. (1.2), is the ML rule given by Eqn. (1.11). However, since the ML rule requires searching over $|\mathcal{A}|^M$ symbols, it is not practical when this number is large. In this chapter, we review a number of suboptimal symbol detection rules that have been proposed as practical alternatives to the ML rule.

2.1 Linear Receivers

Linear receivers are the class of receivers for which the symbol estimate $\hat{\mathbf{a}}$ is given by a transformation of the received vector $\mathbf{r}$ of the form

$$\hat{\mathbf{a}} = Q(\mathbf{W}\mathbf{r}) \tag{2.1}$$

where $\mathbf{W}$ is a matrix that may depend on $\mathbf{H}$ and Q is a quantizer (also called slicer) that maps its argument to the nearest signal point in $\mathcal{A}^M$ (using Euclidian

distance). Our review of linear receivers follows the presentation of [6].

2.1.1 Zero-Forcing (ZF) Receiver

Zero-Forcing (ZF)receiver is a low-complexity linear detection algorithm that outputs

$$\hat{\mathbf{a}} = Q(\hat{\mathbf{a}}_{ZF}) \tag{2.2}$$

where

$$\hat{\mathbf{a}}_{ZF} = \mathbf{H}^{+}\mathbf{r} \tag{2.3}$$

and $\mathbf{H}^{+}$ denotes the Moore-Penrose pseudoinverse [7] of $\mathbf{H}$, which is a generalized inverse that exists even when $\mathbf{H}$ is rank-deficient (See Appendix A.1).

Example 1:

Consider a MIMO channel with (M,N)=(3,4) with 16-QAM constellation $\mathcal{A} = \{\pm 1 \pm j, \pm 1 \pm j3, \pm 3 \pm j, \pm 3 \pm j3\}$ and noise variance $N_0 = 2.5$ with resulting $E_b/N_0 = 1$. Suppose the realization of channel transfer matrix is

$$\mathbf{H} = \begin{bmatrix} -0.7i & 0.3 - 0.3i & -0.5 - 0.4i \\ 0.8 - 0.6i & 0.7 - 1.1i & -0.8 - 1.1i \\ -0.8 & 0.2 + 0.3i & 0.2i \\ -0.1 - 0.2i & 1.2 - 0.3i & -1.7 - 0.6i \end{bmatrix}$$

and that we send $\mathbf{a} = \begin{pmatrix} 1+i, & -1-i, & 1+3i \end{pmatrix}^T$, and that the channel adds the noise vector $\mathbf{v} = \begin{pmatrix} 0.6+0.4i, & 0.4-0.1i, & 0.7+0.5i, & 0.2-0.2i \end{pmatrix}^T$. Then the received vector $\mathbf{r}$ is

$$\mathbf{r} = \mathbf{Ha} + \mathbf{v} = \begin{bmatrix} 1.5 - 2.2i \\ 2.4 - 2.8i \\ -0.4 - 0.6i \\ -1.1 - 7i \end{bmatrix}$$

The pseudo-inverse $\mathbf{H}^{+}$ is computed as

$$\mathbf{H}^{+} = \begin{bmatrix} -0.2+0.7i & -0.2+0.1i & -0.8+0.4i & 0.2-0.2i \\ 0.5+0.2i & 0.7i & 0.5+0.3i & 0.2-0.5i \\ 0.4-0.1i & 0.3+0.4i & 0.5+0.0i & -0.7-0.2i \end{bmatrix}$$

and the matrix $\mathbf{H}^{+}\mathbf{H}$ is

$$\mathbf{H}^{+}\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Using Eqn. (2.3), the ZF estimator output $\hat{\mathbf{a}}_{ZF}$ is computed as

$$\hat{\mathbf{a}}_{ZF} = \mathbf{H}^{+}\mathbf{r} = \begin{bmatrix} -0.2+1.1i \\ -0.6-0.1i \\ 1.6+3.6i \end{bmatrix}$$

When we slice $\hat{\mathbf{a}}_{ZF}$ to the nearest 16-QAM symbol, we get the following decision vector

$$\hat{\mathbf{a}} = \begin{bmatrix} -1+i \\ -1-i \\ 1+3i \end{bmatrix}$$

which turns out not to be a correct estimate of $\mathbf{a}$.

The ZF receiver eliminates co-channel interference entirely in the above example since $\mathbf{H}^{+}\mathbf{H} = \mathbf{I}$. On the other hand, ZF receivers are known to have the drawback of enhancing noise power [6]. Indeed, this may be seen in the above example as follows. The signal-to-noise ratio at the ZF estimator input is

$$SNR_{ZF,\,in} = \frac{\left\|\mathbf{Ha}\right\|^2}{\left\|\mathbf{v}\right\|^2} = 5.03$$

while the SNR at the ZF estimator output is

$$SNR_{ZF,\,out} = \frac{\left\|\mathbf{H}^{+}\mathbf{Ha}\right\|^2}{\left\|\mathbf{H}^{+}\mathbf{v}\right\|^2} = \frac{\left\|\mathbf{a}\right\|^2}{\left\|\mathbf{H}^{+}\mathbf{v}\right\|^2} = 4.41$$

We see that there is a deterioration of the SNR attributable to noise enhancement, as seen from the ratio

$$\frac{SNR_{ZF,\,out}}{SNR_{ZF,\,in}} = 0.88 < 1.$$

For a more realistic performance estimation of the ZF receiver, we show in Fig. 2.1 the simulation results for a (M,N)= (8,12) system with 16-QAM modulation. The E_b/N_0, defined by Eqn. (1.7), ranges between -10 dB and 0 dB in steps of 1 dB. The symbol error rate SER is calculated by performing 10,000 trials at each E_b/N_0 point. A new realization of $\mathbf{H}$ was chosen in each trial and for each E_b/N_0 value.

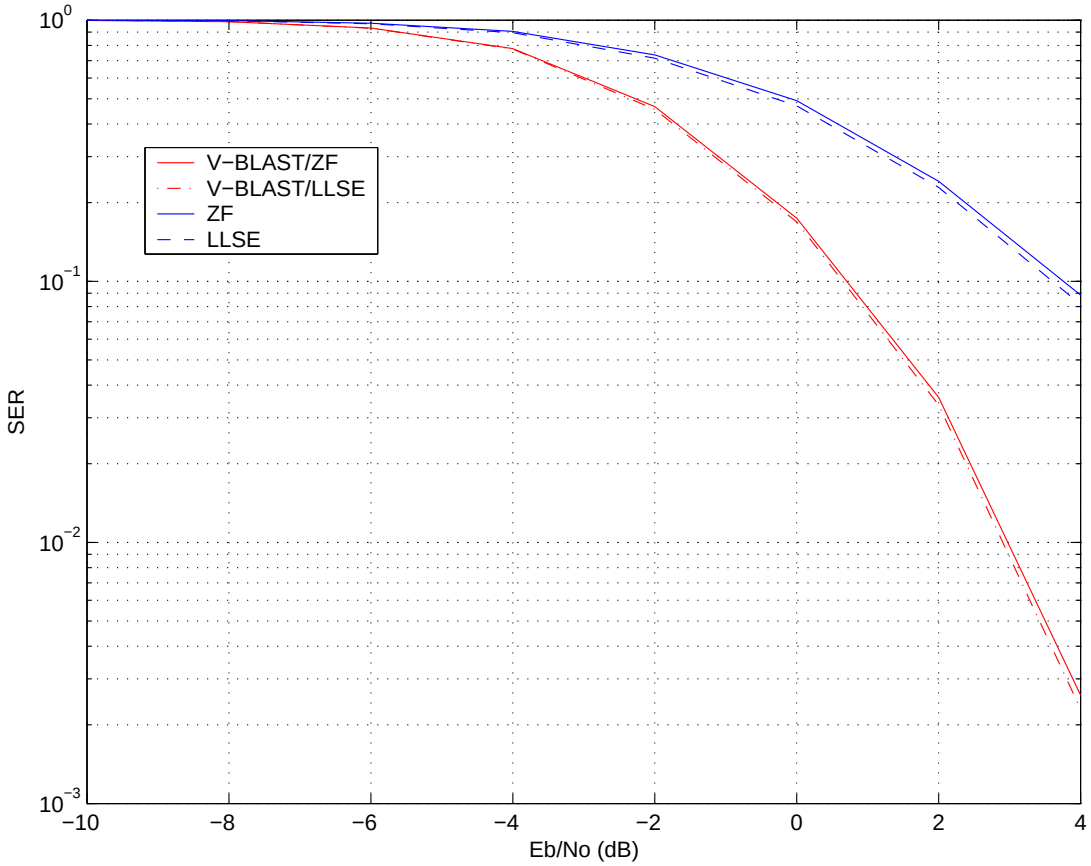

Figure 2.1: Symbol Error Rates (SER) of ZF receiver, LLSE receiver, V-BLAST/ZF receiver and V-BLAST/LLSE receiver. Simulations are for (M,N)=(8,12) and QAM-16 modulation.

2.1.2 Linear Least Squares Estimation (LLSE) Receiver

The LLSE receiver is a receiver that outputs the estimate

$$\hat{\mathbf{a}} = Q\big(\hat{\mathbf{a}}_{LLSE}\big) \tag{2.4}$$

where $\hat{\mathbf{a}}_{LLSE}$ is a linear estimator given by

$$\hat{\mathbf{a}}_{LLSE} = \mathbf{W}\mathbf{r} \tag{2.5}$$

where $\mathbf{W}$ is chosen to minimize

$$\mathcal{E}\big\{\,\|\,\mathbf{W}\mathbf{r} - \mathbf{a}\,\|^2\big\}.$$

For the model here, where $\mathbf{H}$ and $\mathbf{v}$ are Gaussian, the LLSE estimator matrix is given by [6]

$$\mathbf{W} = \frac{\rho}{M}\,\mathbf{H}^{\dagger}\big(\,\frac{\rho}{M}\,\mathbf{H}\mathbf{H}^{\dagger} \;+\; N_0\mathbf{I}_N\,\big)^{-1} \tag{2.6}$$

Example 2

Assume that we use the same channel of Example 1 with the same $\mathbf{H}$, $\mathbf{a}$, $\mathbf{v}$ and N_0 values. Here, $\rho/M = 10$ and the LLSE matrix is given by

$$\mathbf{W} = \begin{bmatrix} -0.2+0.5i & -0.1+0.1i & -0.6+0.3i & 0.1-0.2i \\ 0.3+0.1i & 0.0+0.6i & 0.4+0.2i & 0.2-0.4i \\ 0.2-0.1i & 0.2+0.3i & 3+0.0i & -0.5-0.1i \end{bmatrix}$$

Using Eqn. (2.5), estimator output $\hat{\mathbf{a}}_{LLSE}$ is computed to be

$$\hat{\mathbf{a}}_{LLSE} = \mathbf{W}\mathbf{r} = \begin{bmatrix} 0.1+1.1i \\ -0.4-0.4i \\ 1.4+3.2i \end{bmatrix}$$

Therefore, the decision vector of LLSE estimator is

$$\hat{\mathbf{a}} = \begin{bmatrix} 1.0+1.0i \\ -1.0-1.0i \\ 1.0+3.0i \end{bmatrix}$$

which is a correct estimate of $\mathbf{a}$.

The LLSE estimator does not eliminate the co-channel interference entirely since

$$\mathbf{WH} = \begin{bmatrix} 0.6 & -0.2i & 0.1-0.1i \\ 0.2i & 0.6 & -0.1-0.2i \\ 0.1+0.1i & -0.1+0.2i & 0.7 \end{bmatrix}$$

does not equal the identity matrix, unlike the case for the ZF estimator. On the other hand, the LLSE estimator has the desirable property of not enhancing noise as much as the ZF estimator. This may be seen in the above example by calculating the SNR's at the input and output of the LLSE estimator:

$$\begin{aligned} SNR_{LLSE,\,in} &= \frac{\left\|\mathbf{Ha}\right\|^2}{\left\|\mathbf{v}\right\|^2} = 5.03 \\ SNR_{ZF,\,out} &= \frac{\left\|\mathbf{WHa}\right\|^2}{\left\|\mathbf{Wv}\right\|^2} = 27.56 \end{aligned}$$

The ratio of the SNR's is now given by

$$\frac{SNR_{LLSE,\,out}}{SNR_{LLSE,\,in}} = 5.48$$

which is better than the corresponding quantity for the ZF estimator.

For a more realistic performance estimation of the LLSE receiver, we show in Fig. 2.1 the simulation results for a (M,N)= (8,12) system with 16-QAM modulation. The E_b/N_0, defined by Eqn. (1.7), ranges between -10 dB and 0 dB in steps of 1 dB. The symbol error rate SER is calculated by performing 10,000 trials at each E_b/N_0 point. A new realization of $\mathbf{H}$ was chosen in each trial and for each E_b/N_0 value. We observe that LLSE performs slightly better than ZF for this example.

2.2 V-BLAST Receiver

The V-BLAST detection algorithm [8] is a recursive procedure that exctracts the components of the transmitted vector $\mathbf{a}$ according to a certain ordering $(k_1, k_2, ..., k_M)$ of the indices of the elements of $\mathbf{a}$. Thus, $(k_1, k_2, ..., k_M)$ is a permutation of $(1, 2, ..., M)$. In V-BLAST, this permutation depends on $\mathbf{H}$ (which is known at the receiver by assumption) but not on the received vector $\mathbf{r}$.

2.2.1 V-BLAST/ZF Detection Algorithm

The V-BLAST/ZF algorithm is a variant of V-BLAST derived from ZF rule.

V-BLAST/ZF Detection Algorithm [5]

Initialization :

$$\mathbf{W}_1 = \mathbf{H}^+ \tag{2.7a}$$

$$i = 1 \tag{2.7b}$$

Recursion :

$$k_i = \underset{j \notin \{k_1 ... k_{i-1}\}}{\arg\min} \|(\mathbf{W}_i)_j\|^2 \tag{2.7c}$$

$$y_{k_i} = (\mathbf{W}_i)_{k_i} \mathbf{r}_i \tag{2.7d}$$

$$\hat{a}_{k_i} = Q(y_{k_i}) \tag{2.7e}$$

$$\mathbf{r}_{i+1} = \mathbf{r}_i - \hat{a}_{k_i} (\mathbf{H})_{k_i} \tag{2.7f}$$

$$\mathbf{W}_{i+1} = \mathbf{H}^+_{\overline{k_i}} \tag{2.7g}$$

$$i = i + 1 \tag{2.7h}$$

where $\mathbf{H}^+$ denotes the Moore-Penrose pseudoinverse [7] of $\mathbf{H}$, $(\mathbf{W}_i)_j$ is the j'th row of W_i, $Q(\cdot)$ is a quantizer to the nearest constellation point, $(\mathbf{H})_{k_i}$ denotes the k_i'th column of $\mathbf{H}$, $\mathbf{H}_{\overline{k_i}}$ denotes the matrix obtained by zeroing the columns $k_1, k_2, ..., k_i$ of $\mathbf{H}$, and $\mathbf{H}^+_{\overline{k_i}}$ denotes the pseudo-inverse of $\mathbf{H}_{\overline{k_i}}$.

In the above algorithm, Eqn. (2.7c) determines the order of channels to be detected; Eqn. (2.7d) performs *nulling* and computes the decision statistic; Eqn. (2.7e) slices computed decision statistic and yields the decision; Eqn. (2.7f) performs *cancellation* by decision feedback, and Eqn. (2.7g) computes the new pseudo-inverse for the next iteration.

V-BLAST/ZF may be seen as a successive-cancellation scheme derived from the ZF scheme discussed in Section 2.1.1. The ZF rule creates a set of sub-channels by forming $\hat{\mathbf{a}}_{ZF} = (\mathbf{H}^{+}\mathbf{H})\mathbf{a} + \mathbf{H}^{+}\mathbf{v}$, as in Eqn. 2.3. The j'th such sub-channel has noise variance $\left\|(\mathbf{H}^{+})_j\right\|^2 N_0$. The order selection rule prioritizes the sub-channel with the smallest noise variance.

Example 3:

In this part, we demonstrate the simulation of V-BLAST/ZF algorithm numerically. We again use the same channel of Example 1 with the same $\mathbf{H}$, $\mathbf{a}$ and $\mathbf{v}$ values. After initialization, we have the pseudoinverse matrix

$$\mathbf{W}_1 = \mathbf{H}^{+} = \begin{bmatrix} -0.2+0.7i & -0.2+0.1i & -0.8+0.4i & 0.2-0.2i \\ 0.5+0.2i & 0.0+0.7i & 0.5+0.3i & 0.2-0.5i \\ 0.4-0.1i & 0.3+0.4i & 0.5+0.0i & -0.7-0.2i \end{bmatrix}$$

Since there are 3 components of $\mathbf{a}$, V-BLAST/ZF algorithm completes the decision process after 3 iterations as follows:

Step 1: In the first layer, k_1 is computed to be 3 since

$$\begin{aligned} \|(\mathbf{W}_1)_1\|^2 &= 1.43 \\ \|(\mathbf{W}_2)_1\|^2 &= 1.36 \\ \|(\mathbf{W}_3)_1\|^2 &= 1.12 \end{aligned}$$

Therefore, algorithm chooses 3^{rd} sub-channel to process in this step, and the first component of estimate is calculated according to Eqn. (2.7d) as

$$y_{k_1} = \begin{bmatrix} 0.4-0.1i & 0.3+0.4i & 0.5 & -0.7-0.2i \end{bmatrix} \begin{bmatrix} 1.5-2.2i \\ 2.4-2.8i \\ -0.4-0.6i \\ -1.1-7.0i \end{bmatrix} = 1.6+3.6i$$

When this estimate is sliced by Eqn. (2.7e), we get the decision for the first component of the transmitted vector as

$$\hat{a}_{k_1} = 1.0+3.0i$$

After symbol cancellation described in Eqn. (2.7f), we get the following modified received vector

$$\mathbf{r}_2 = \mathbf{r}_1 - \hat{a}_{k_1}(\mathbf{H})_{k_1} = \begin{bmatrix} 0.8-0.3i \\ 0.6i \\ -0.8i \\ -1.2-1.3i \end{bmatrix}$$

Here, the matrix $\mathbf{H}_{\overline{k_1}}$ equals

$$\mathbf{H}_{\overline{k_1}} = \begin{bmatrix} -0.7i & 0.3-0.3i & 0 \\ 0.8-0.6i & 0.7-1.1i & 0 \\ -0.8 & 0.2+0.3i & 0 \\ -0.1-0.2i & 1.2-0.3i & 0 \end{bmatrix}$$

The new pseudoinverse $\mathbf{W}_2$ for next iteration is computed to be

$$\mathbf{W}_2 = \begin{bmatrix} -0.1+0.4i & 0.3+0.1i & -0.5+0.1i & -0.3+0.1i \\ 0.1 & 0.1+0.3i & 0.2-0.1i & 0.4+0.1i \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 2: In the second layer, algorithm chooses the 2^{nd} sub-channel to process, since

$$\begin{aligned} \|(\mathbf{W}_1)_1\|^2 &= 0.55 \\ \|(\mathbf{W}_2)_1\|^2 &= 0.36 \\ \|(\mathbf{W}_3)_1\|^2 &= 0 \end{aligned}$$

Therefore, estimate y_{k_2} is

$$y_{k_2} = \begin{bmatrix} 0.1 & 0.1+0.3i & 0.2-0.1i & 0.4+0.1i \end{bmatrix} \begin{bmatrix} 0.8-0.3i \\ -0.0+0.6i \\ -0.0-0.8i \\ -1.2-1.3i \end{bmatrix} = -0.5-0.9i$$

and the decision $\hat{a}_{k_2}$ is

$$\hat{a}_{k_2} = -1.0-1.0i$$

After cancellation, the modified received vector is calculated to be

$$\mathbf{r}_3 = \mathbf{r}_2 - \hat{a}_{k_2}(\mathbf{H})_{k_2} = \begin{bmatrix} 1.4-0.3i \\ 1.8+0.2i \\ -0.2-0.3i \\ 0.3-0.5i \end{bmatrix}$$

where the matrix $\mathbf{H}_{\overline{k_2}}$ equals

$$\mathbf{H}_{\overline{k_1}} = \begin{bmatrix} -0.7i & 0 & 0 \\ 0.8-0.6i & 0 & 0 \\ -0.8 & 0 & 0 \\ -0.1-0.2i & 0 & 0 \end{bmatrix}$$

and the new pseudoinverse $\mathbf{W}_3$ is

$$\mathbf{W}_3 = \begin{bmatrix} 0.3i & 0.4+0.2i & -0.4 & 0.1i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 3: In the last layer, k_3 is set to 1. The estimate y_{k_3} is calculated as

$$y_{k_3} = \begin{bmatrix} 0.3i & 0.4+0.2i & -0.4 & 0.1i \end{bmatrix} \begin{bmatrix} 1.0-1.3i \\ 0.3-0.5i \\ -0.2-1.3i \\ -1.5-5.5i \end{bmatrix} = 0.8+1.1i$$

and the corresponding decision $\hat{a}_{k_3}$ is

$$\hat{a}_{k_3} = 1+i$$

We may combine the components of decision vector according to the order of indices (k_1, k_2, k_3), which yields

$$\hat{\mathbf{a}} = \begin{bmatrix} 1+i \\ -1-i \\ 1+3i \end{bmatrix}$$

which is the correct estimate of $\mathbf{a}$.

For a more realistic performance estimation of the V-BLAST/ZF receiver, we show in Fig. 2.1 the simulation results for a (M,N)= (8,12) system with 16-QAM modulation. The E_b/N_0, defined by Eqn. (1.7), ranges between -10 dB and 0 dB in steps of 1 dB. The symbol error rate SER is calculated by performing 10,000 trials at each E_b/N_0 point. A new realization of $\mathbf{H}$ was chosen in each trial and for each E_b/N_0 value. Result of this simulation is very similar to an experiment performed in a real laboratory environment which is reported in [5]. We observe that V-BLAST/ZF performs significantly better than both ZF and LLSE receivers.

2.2.1.1 V-BLAST/LLSE Detection Algorithm

The V-BLAST/LLSE algorithm is a variant of V-BLAST where the weigthing matrix is chosen according to the LLSE rule [9].

V-BLAST/LLSE Detection Algorithm:

$$
\begin{aligned}
&\textit{Initialization}: \\
&\qquad \mathbf{W}_1 = \frac{\rho}{M}\,\mathbf{H}^{\dagger}\big(\frac{\rho}{M}\,\mathbf{H}\mathbf{H}^{\dagger} + N_0\,\mathbf{I}_N\big) \\
&\qquad i = 1 \\
&\textit{Recursion}: \\
&\qquad k_i = \underset{j\notin\{k_1,\ldots,k_{i-1}\}}{\arg\min} \|(\mathbf{W}_i)_j\|^2 \\
&\qquad y_{k_i} = (\mathbf{W}_i)_{k_i}\mathbf{r}_i \\
&\qquad \hat{a}_{k_i} = Q\,(\,y_{k_i}\,) \\
&\qquad \mathbf{r}_{i+1} = \mathbf{r}_i - \hat{a}_{k_i}(\mathbf{H})_{\,k_i} \\
&\qquad \mathbf{W}_{i+1} = \frac{\rho}{M}\,\mathbf{H}^{\dagger}_{\bar{k}_i}\big(\frac{\rho}{M}\,\mathbf{H}_{\bar{k}_i}\mathbf{H}^{\dagger}_{\bar{k}_i} + N_0\,\mathbf{I}_N\big) \\
&\qquad i = i+1
\end{aligned}
$$

Example 4:

For the numerical simulation results of V-BLAST/LLSE algorithm, we use the same channel as in the previous examples with the same $\mathbf{H}$, $\mathbf{a}$ and $\mathbf{v}$ values. After initialization, we have the LLSE matrix

$$
\mathbf{W}_1 = \begin{bmatrix}
-0.2+0.5i & -0.1+0.1i & -0.6+0.3i & 0.1-0.2i \\
0.3+0.1i & 0.6i & 0.4+0.2i & 0.2-0.4i \\
0.2-0.1i & 0.2+0.3i & 0.3 & -0.5-0.1i
\end{bmatrix}
$$

There will be again 3 steps in the algorithm:

Step 1: In the first layer, k_1 is computed to be 3 since

$$
\begin{aligned}
\|(\mathbf{W}_1)_1\|^2 &= 0.82 \\
\|(\mathbf{W}_2)_1\|^2 &= 0.76 \\
\|(\mathbf{W}_3)_1\|^2 &= 0.61
\end{aligned}
$$

Therefore, algorithm chooses 3^{rd} sub-channel to process in this step, and the first component of estimate is calculated to be

$$y_{k_1} = \begin{bmatrix} -0.2+0.5i & -0.1+0.1i & -0.6+0.3i & 0.1-0.2i \end{bmatrix} \begin{bmatrix} .5-2.2i \\ 2.4-2.8i \\ -0.4-0.6i \\ -1.1-7.0i \end{bmatrix} = 1.4+3.2i$$

When this estimate is sliced to the nearest point from the alphabet $\mathcal{A}$, we get the decision for the first component of the transmitted vector as

$$\hat{a}_{k_1} = 1.0+3.0i$$

After symbol cancellation, we get the following modified received vector

$$\mathbf{r}_2 = \mathbf{r}_1 - \hat{a}_{k_1}(\mathbf{H})_{k_1} = \begin{bmatrix} 0.8-0.3i \\ 0.6i \\ -0.8i \\ -1.2-1.3i \end{bmatrix}$$

The LLSE matrix $\mathbf{W}_2$ for next iteration is computed to be

$$\mathbf{W}_2 = \begin{bmatrix} -0.1+0.3i & 0.2+0.1i & -0.4+0.1i & -0.3+0.1i \\ 0.1 & 0.1+0.2i & 0.2-0.1i & 0.4+0.1i \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 2: In the second layer, algorithm chooses the 2^{nd} sub-channel to process, since

$$\begin{aligned} \|(\mathbf{W}_1)_1\|^2 &= 0.47 \\ \|(\mathbf{W}_2)_1\|^2 &= 0.32 \\ \|(\mathbf{W}_3)_1\|^2 &= 0 \end{aligned}$$

Therefore, estimate y_{k_2} is

$$y_{k_2} = \begin{bmatrix} 0.1-0.0i & 0.1+0.2i & 0.2-0.1i & 0.4+0.1i \end{bmatrix} \begin{bmatrix} 0.8-0.3i \\ 0.6i \\ -0.8i \\ -1.2-1.3i \end{bmatrix} = -0.5-0.8i$$

and the decision $\hat{a}_{k_2}$ is

$$\hat{a}_{k_2} = -1.0-1.0i$$

After cancellation, the modified received vector is calculated to be

$$\mathbf{r}_3 = \mathbf{r}_2 - \hat{a}_{k_2}(\mathbf{H})_{k_2} = \begin{bmatrix} 1.4-0.3i \\ 1.8+0.2i \\ -0.2-0.3i \\ 0.3-0.5i \end{bmatrix} \tag{2.9}$$

and the new LLSE matrix $\mathbf{W}_3$ is

$$\mathbf{W}_3 = \begin{bmatrix} 0.3i & 0.3+0.2i & -0.3 & 0.1i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 3: In the last layer, k_3 is set to 1. The estimate y_{k_3} for this layer is calculated to be

$$y_{k_3} = \begin{bmatrix} 0.3i & 0.3+0.2i & -0.3 & 0.1i \end{bmatrix} \begin{bmatrix} 1.4-0.3i \\ 1.8+0.2i \\ -0.2-0.3i \\ 0.3-0.5i \end{bmatrix} = 0.7+i$$

and the corresponding decision $\hat{a}_{k_3}$ is

$$\hat{a}_{k_3} = 1+i$$

We may combine the components of decision vector according to the order of indices (k_1, k_2, k_3) to have

$$\hat{\mathbf{a}} = \begin{bmatrix} 1+i \\ -1-i \\ 1+3i \end{bmatrix}$$

which is the correct estimate of $\mathbf{a}$.

For a more realistic performance estimation of the V-BLAST/LLSE receiver, we show in Fig. 2.1 the simulation results for a (M,N)= (8,12) system with 16-QAM modulation. The E_b/N_0, defined by Eqn. (1.7), ranges between -10 dB and 0 dB in steps of 1 dB. The symbol error rate SER is calculated by performing 10,000 trials at each E_b/N_0 point. A new realization of $\mathbf{H}$ was chosen in each trial and for each E_b/N_0 value. We observe a slight improvement compared to the performance of V-BLAST/ZF.

Chapter 3

V-BLAST/MAP Detection Algorithm

Throughout this chapter, we again use the MIMO channel model and the notation of Chapter 1 with the assumptions *(i)-(iii)* of Section 1.2 on the channel input.

In this chapter, we propose a new symbol detection algorithm for MIMO channels, which is called V-BLAST/MAP, that combines the features of V-BLAST and MAP rules. This algorithm uses the layered structure of V-BLAST, but uses a different strategy for channel processing order, inspired by the MAP rule.

3.1 V-BLAST/ZF/MAP Detection Algorithm

Using the same notation of V-BLAST algorithm, V-BLAST/ZF/MAP algorithm may be described as follows:

V-BLAST/ZF/MAP Detection Algorithm :

Initialization :

$$\mathbf{W}_1 = \mathbf{H}^+ \tag{3.1a}$$

$$i = 1 \tag{3.1b}$$

Recursion:

$$\mathbf{y}_i = \mathbf{W}_i \, \mathbf{r}_i \tag{3.1c}$$

$$\mathbf{s}_i = Q\left(\mathbf{y}_i\right) \tag{3.1d}$$

$$p_{ij} = \frac{f_{ij}\left(\, y_{ij} \,|\, s_{ij} \,\right)}{\sum\limits_{s' \epsilon \mathcal{A}} f_{ij}\left(\, y_{ij} \,|\, s' \,\right)}, \quad j \notin \{k_1, ..., k_{i-1}\} \tag{3.1e}$$

$$k_i = \underset{j \notin \{k_1, ..., k_{i-1}\}}{\arg\max} \{p_{ij}\} \tag{3.1f}$$

$$\hat{a}_{k_i} = s_{i\,k_i} \tag{3.1g}$$

$$\mathbf{r}_{i+1} = \mathbf{r}_i - \hat{a}_{k_i} (\mathbf{H})_{k_i} \tag{3.1h}$$

$$\mathbf{W}_{i+1} = \mathbf{H}^+_{\bar{k}_i} \tag{3.1i}$$

$$i = i + 1 \tag{3.1j}$$

Here the vectors $\mathbf{y}_i = (y_{i1}, y_{i2}, ..., y_{iM})^T$ and $\mathbf{s}_i = (s_{i1}, s_{i2}, ..., s_{iM})^T$ are the counterparts of those in Eqn.'s (2.2) and (2.3) in the ZF detector. In (3.1e), f_{ij} is a density function given by

$$f_{ij}(\, y_{ij} \,|\, s_{ij}) = \frac{1}{\pi \sigma_j^2} \exp\{-\frac{1}{\sigma_j^2} \left\| y_{ij} - s_{ij} \right\|^2\} \tag{3.2}$$

where $\sigma_j^2 = N_0 \left\| (\mathbf{W}_i)_j \right\|^2$. In (3.1e) and (3.1f), the index j ranges over all elements of $\{1, 2, ..., M\}$ excluding those in $\{k_1, ..., k_{i-1}\}$, i.e., $j \,\epsilon\, \{1, ..., M\} \backslash \{k_1, ..., k_{i-1}\}$.

V-BLAST/ZF/MAP algorithm is identical to V-BLAST/ZF except for the ordering in which symbols are detected. Instead of selecting the next symbol to be detected according to the rule (2.7c), here the set of all potential symbol decisions are ranked with respect to their a-posteriori probabilities of being correct, as estimated by p_{ij}. Thus, it is important to emphasize that p_{ij}'s are not true MAP probabilities but approximations to how probable it is that $s_{ij} = a_j$. The

approximation is due to the omission in calculations of the crosscorrelations between the noise terms $z_{ij} \triangleq y_{ij} - s_{ij}$ on the component subchannels. Notice that the index permutation $(k_1, k_2, ..., k_M)$ produced by V-BLAST/ZF/MAP depends on both $\mathbf{H}$ and $\mathbf{r}$, unlike V-BLAST/ZF where the permutation depends only on $\mathbf{H}$.

The complexity of V-BLAST/ZF/MAP is increased with respect to that of V-BLAST/ZF by the computation done in step (3.1e). The order of complexity of computing p_{ij} is roughly $O(|\mathcal{A}|)$ for any fixed j, and upperbounded by $O(M|\mathcal{A}|)$ when considered as a whole. This computation can be further simplified by approximating the denominator of (3.1e) but that issue is not explored in this thesis.

One major point about complexities of V-BLAST/ZF and V-BLAST/ZF/MAP is that in the former allows pre-computation of all weighting vectors (which can be use repeatedly as long as $\mathbf{H}$ is fixed) whereas in the latter the weighting vector must be computed in real-time since it also depends on $\mathbf{r}$. This increased complexity of V-BLAST/ZF/MAP is justified by performance improvements as illustrated later in this section.

Example 5:

In this example, we examine the numerical simulation results of V-BLAST/ZF/MAP algorithm. We use the same channel as in Example 1 of Section 2.1.1 with the same $\mathbf{H}$, $\mathbf{a}$ and $\mathbf{v}$ values. After initialization, we have the pseudoinverse matrix

$$\mathbf{W}_1 = \begin{bmatrix} -0.2 + 0.7i & -0.2 + 0.1i & -0.8 + 0.4i & 0.2 - 0.2i \\ 0.5 + 0.2i & 0.7i & 0.5 + 0.3i & 0.2 - 0.5i \\ 0.4 - 0.1i & 0.3 + 0.4i & 0.5 & -0.7 - 0.2i \end{bmatrix}$$

There will be again 3 steps in the algorithm as follows:

Step 1: We compute

$$\mathbf{y}_1 = \begin{bmatrix} -0.2 + 1.1i \\ -0.6 - 0.1i \\ 1.6 + 3.6i \end{bmatrix}$$

and

$$\mathbf{s}_1 = \begin{bmatrix} -1.0 + 1.0i \\ -1.0 - 1.0i \\ 1.0 + 3.0i \end{bmatrix}$$

The reliability estimates are computed as

$$\begin{aligned} p_{11} &= 0.546 \\ p_{12} &= 0.454 \\ p_{13} &= 0.785 \end{aligned}$$

Therefore, the algorithm sets k_1 to 3 and the third component of the decision is chosen to be

$$\hat{a}_{k_1} = \mathbf{s}_{13} = 1.0 + 3.0i$$

After symbol cancellation, we get the following modified received vector

$$\mathbf{r}_2 = \mathbf{r}_1 - \hat{a}_{k_1}(\mathbf{H})_{k_1} = \begin{bmatrix} 0.8 - 0.3i \\ 0.6i \\ -0.8i \\ -1.2 - 1.3i \end{bmatrix}$$

and new pseudoinverse $\mathbf{W}_2$ for next iteration is computed to be

$$\mathbf{W}_2 = \begin{bmatrix} -0.1 + 0.4i & 0.3 + 0.1i & -0.5 + 0.1i & -0.3 + 0.1i \\ 0.1 & 0.1 + 0.3i & 0.2 - 0.1i & 0.4 + 0.1i \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 2: Now, we have

$$\mathbf{y}_2 = \begin{bmatrix} 0.5 + 1.1i \\ -0.5 - 0.9i \\ 0 \end{bmatrix}$$

and

$$\mathbf{s}_2 = \begin{bmatrix} 1.0 + 1.0i \\ -1.0 - 1.0i \\ -1.0 - 1.0i \end{bmatrix}$$

The reliabilities are computed for the unprocessed subchannels as

$$\begin{aligned} p_{21} &= 0.967 \\ p_{22} &= 0.996 \end{aligned}$$

Therefore, the algorithm sets k_2 to 2 and the second component of decision is chosen to be

$$\hat{a}_{k_2} = \mathbf{s}_{22} = -1.0 - 1.0i$$

After symbol cancellation, we get the following modified received vector

$$\mathbf{r}_3 = \mathbf{r}_2 - \hat{a}_{k_2}(\mathbf{H})_{k_2} = \begin{bmatrix} 1.4 - 0.3i \\ 1.8 + 0.2i \\ -0.2 - 0.3i \\ 0.3 - 0.5i \end{bmatrix}$$

and new pseudoinverse $\mathbf{W}_3$ for next iteration is computed to be

$$\mathbf{W}_3 = \begin{bmatrix} 0.3i & 0.4 + 0.2i & -0.4 & 0.1i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 3: Now

$$\mathbf{y}_3 = \begin{bmatrix} 0.8 + 1.1i \\ 0 \\ 0 \end{bmatrix}$$

and

$$\mathbf{s}_3 = \begin{bmatrix} 1.0 + 1.0i \\ -1.0 - i \\ -1.0 - i \end{bmatrix}$$

Since there is a single subchannel to be detected, the algorithm sets k_3 to 1. Therefore, the first component of decision is

$$\hat{a}_{k_3} = \mathbf{s}_{31} = 1.0 + 1.0i$$

We may combine the components of decision vector according to the order of indices (k_1, k_2, k_3), and obtain

$$\hat{\mathbf{a}} = \begin{bmatrix} 1 + i \\ -1 - i \\ 1 + 3i \end{bmatrix}$$

which is the correct estimate of $\mathbf{a}$.

For a more realistic performance estimation of the V-BLAST/ZF/MAP receiver, we show in Fig. 3.1 the simulation results for a (M,N)= (8,12) system with 16-QAM modulation. The E_b/N_0, defined by Eqn. (1.7), ranges between -10 dB and 0 dB in steps of 1 dB. The symbol error rate SER is calculated by performing 10,000 trials at each E_b/N_0 point. A new realization of $\mathbf{H}$ was chosen in each trial and for each E_b/N_0 value. We observe that V-BLAST/ZF/MAP performs significantly better than both both V-BLAST/ZF and V-BLAST/LLSE receivers.

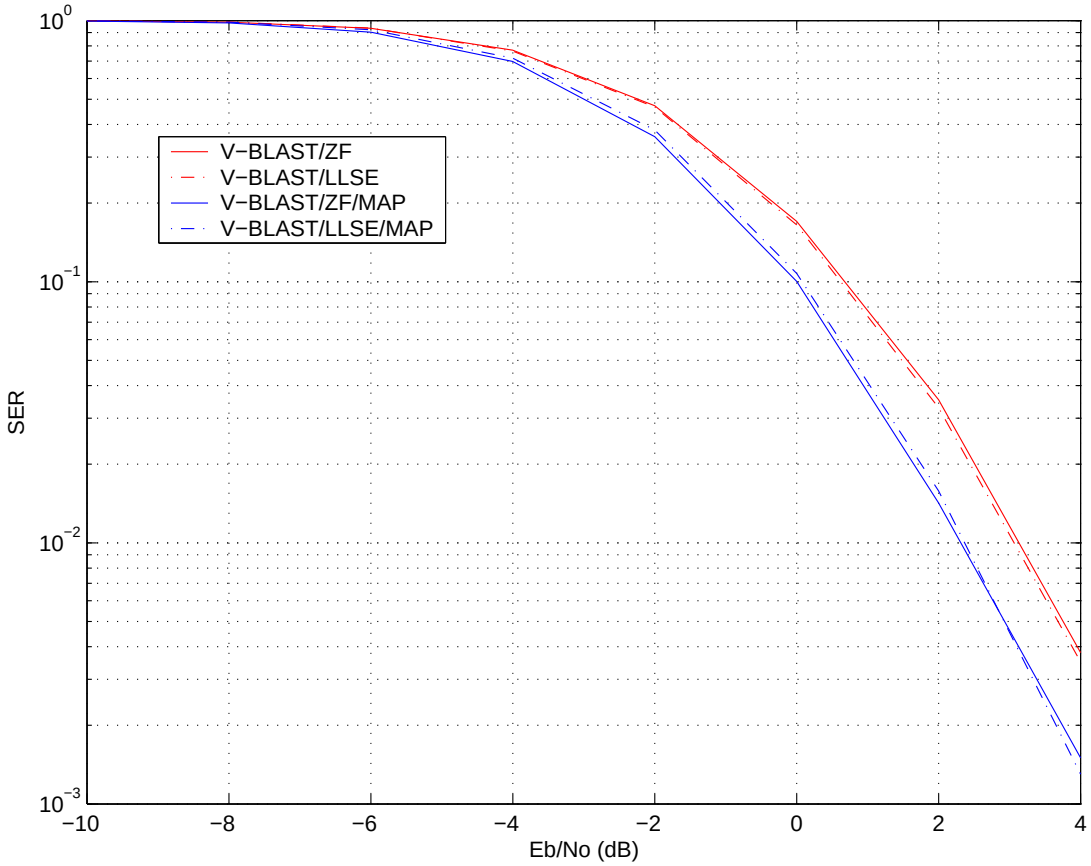

Figure 3.1: Symbol error rates (SER) of V-BLAST/ZF/MAP receiver, V-BLAST/LLSE/MAP receiver, V-BLAST/ZF receiver and V-BLAST/LLSE receivers. Simulations are for (M,N)= (8,12) and QAM-16 modulation.

3.2 V-BLAST/LLSE/MAP Detection Algorithm

In this section, we use the LLSE technique in order to compute weighting matrix. Then, V-BLAST/LLSE/MAP algorithm may be described as follows:

V-BLAST/LLSE/MAP Detection Algorithm :

Initialization :

$$i = 1$$

$$\mathbf{W}_i = \frac{\rho}{M}\,\mathbf{H}_i^{\dagger}\,(\frac{\rho}{M}\,\mathbf{H}_i\mathbf{H}_i^{\dagger}\ +\ N_0\,\mathbf{I}_N\,)$$

Recursion :

$$\mathbf{y}_i = \mathbf{W}_i\,\mathbf{r}_i$$

$$\mathbf{s}_i = Q\,(\mathbf{y}_i)$$

$$p_{ij} = \frac{f_{ij}\,(\,y_{ij}\,|\,s_{ij}\,)}{\sum\limits_{s'\epsilon\mathcal{A}} f_{ij}\,(\,y_{ij}\,|\,s'\,)}\,,\quad j \notin \{k_1, ..., k_{i-1}\}$$

$$k_i = \underset{j\notin\{k_1,...,k_{i-1}\}}{\arg\max}\ \{p_{ij}\}$$

$$\hat{a}_{k_i} = \mathbf{s}_{i\,k_i}$$

$$\mathbf{r}_{i+1} = \mathbf{r}_i - \hat{a}_{k_i}(\mathbf{H}_i)_{k_i}$$

$$\mathbf{W}_{i+1} = \frac{\rho}{M}\,\mathbf{H}_{\bar{k}_i}^{\dagger}\,(\frac{\rho}{M}\,\mathbf{H}_{\bar{k}_i}\mathbf{H}_{\bar{k}_i}^{\dagger}\ +\ N_0\,\mathbf{I}_N\,)$$

$$i = i + 1$$

Example 6:

In this example, we examine the numerical simulation results of V-BLAST/LLSE/MAP algorithm. We use the same channel as in Example 1 of Section 2.1.1 with the same $\mathbf{H}$, $\mathbf{a}$, $\mathbf{v}$ and N_0 values with $\rho/M = 10$. After initialization, we have the LLSE matrix

$$\mathbf{W}_1 = \begin{bmatrix} -0.2+0.5i & -0.1+0.1i & -0.6+0.3i & 0.1-0.2i \\ 0.3+0.1i & 0.6i & 0.4+0.2i & 0.2-0.4i \\ 0.2-0.1i & 0.2+0.3i & 0.3 & -0.5-0.1i \end{bmatrix}$$

There will be again 3 steps in the algorithm as follows:

Step 1: The algorithm starts by computing

$$\mathbf{y}_1 = \begin{bmatrix} 0.1 + 1.1i \\ -0.4 - 0.4i \\ 1.4 + 3.2i \end{bmatrix}$$

and

$$\mathbf{s}_1 = \begin{bmatrix} 1.0 + 1.0i \\ -1.0 - 1.0i \\ 1.0 + 3.0i \end{bmatrix}$$

The reliabilities are computed for each subchannel as

$$\begin{aligned} p_{11} &= 0.167 \\ p_{12} &= 0.166 \\ p_{13} &= 0.264 \end{aligned}$$

Therefore, the algorithm sets k_1 to 3 and form the decision

$$\hat{a}_{k_1} = \mathbf{s}_{13} = 1.0 + 3.0i$$

After symbol cancellation, we get the following modified received vector

$$\mathbf{r}_2 = \mathbf{r}_1 - \hat{a}_{k_1}(\mathbf{H})_{k_1} = \begin{bmatrix} 0.8 - 0.3i \\ 0.6i \\ -0.8i \\ -1.2 - 1.3i \end{bmatrix}$$

and LLSE matrix $\mathbf{W}_2$ for next iteration is computed to be

$$\mathbf{W}_2 = \begin{bmatrix} -0.1 + 0.3i & 0.2 + 0.1i & -0.4 + 0.1i & -0.3 + 0.1i \\ 0.1 & 0.1 + 0.2i & 0.2 - 0.1i & 0.4 + 0.1i \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 2: Now,

$$\mathbf{y}_2 = \begin{bmatrix} 0.5 + 1.0i \\ -0.5 - 0.8i \\ 0 \end{bmatrix}$$

and

$$\mathbf{s}_2 = \begin{bmatrix} 1.0 + 1.0i \\ -1.0 - 1.0i \\ -1.0 - 1.0i \end{bmatrix}$$

The reliabilities are

$$\begin{aligned} p_{21} &= 0.169 \\ p_{22} &= 0.165 \end{aligned}$$

Therefore, the algorithm sets k_2 to 1 and forms the decision

$$\hat{a}_{k_2} = \mathbf{s}_{21} = 1.0 + 1.0i$$

After symbol cancellation, we get the following modified received vector

$$\mathbf{r}_3 = \mathbf{r}_2 - \hat{a}_{k_2}(\mathbf{H})_{k_2} = \begin{bmatrix} 0.4i \\ -1.4 + 0.3i \\ 0.8 \\ -1.3 - 1.0i \end{bmatrix}$$

and the LLSE matrix $\mathbf{W}_3$ for next iteration is computed to be

$$\mathbf{W}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.1 + 0.1i & 0.2 + 0.3i & 0.1 - 0.1i & 0.3 + 0.1i \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Step 3: Finally,

$$\mathbf{y}_3 = \begin{bmatrix} 0 \\ -0.7 - 0.8i \\ 0 \end{bmatrix}$$

and

$$\mathbf{s}_3 = \begin{bmatrix} -1.0 - i \\ -1.0 - i \\ -1.0 - i \end{bmatrix}$$

Since there is a single subchannel to be processed, the algorithm sets k_3 to 2, and

$$\hat{a}_{k_3} = \mathbf{s}_{32} = -1.0 - i$$

We may combine the components of decision vector according to the order of indices (k_1, k_2, k_3), and obtain

$$\hat{\mathbf{a}} = \begin{bmatrix} 1 + i \\ -1 - i \\ 1 + 3i \end{bmatrix}$$

which is the correct estimate of $\mathbf{a}$.

For a more realistic performance estimation of the V-BLAST/LLSE/MAP algorithm, we show in Fig. 3.1 the simulation results for a (M,N)= (8,12) system with 16-QAM modulation. The E_b/N_0 ranges between -10 dB and 0 dB in steps of 1 dB. The symbol error rate SER is calculated by performing 10,000 trials at each E_b/N_0 point. A new realization of $\mathbf{H}$ was chosen in each trial and for each E_b/N_0 value.

3.3 Discussion of Simulation Results

The performance curves in Fig. 3.1 show that V-BLAT/MAP provides significant improvement in SER compared to ordinary V-BLAST, both for the ZF and LLSE versions. In Fig. 3.1, we are unable to provide a performance curve for the optimal ML detection algorithm because, for the case considered in that figure, the ML algorithm would require 16^8 likelihood ratios, for each simulation run. In Chapter 1, in Fig. 1.3 we already provided simulation results for the (M,N)=(4,12) and 4-QAM case, where a comparison between V-BLAST/MAP and ML algorithms

was possible. Whether V-BLAST/MAP bridges the performance gap between V-BLAST and ML for the (M,N)=(8,12), 16-QAM case as much as it does for the (M,N)=(4,12) and 4-QAM case is an open question. However, we may state as the main conclusion of this thesis that V-BLAST/MAP offers significantly better SER performance than V-BLAST at a modest increase in complexity.

Bibliography

[1] İ. E. Telatar, "Capacity of multi-antenna gaussian channels," *Eur. Trans. Tel.*, vol. 10, pp. 585–595, November-December 1999.

[2] P. W. Wolniansky, G. J. Foschini, G. D. Golden, and R. A. Valenzuela, "V-blast: An architecture for realizing very high date rates over the rich-scattering wireless channel," *Proc. URSI ISSSE*, pp. 295–300, 1998.

[3] G. J. Foschini, "Layered space-time architecture for wireless communication in a fading environment when using multiple antennas," *Bell Labs Tech. J.*, vol. 1, pp. 41–59, Autumn 1996.

[4] S. Haykin, *Communication Systems*. John Wiley & Sons, Inc., 2001.

[5] G. D. Golden, G. J. Foschini, R. A. Valenzuela, and P. W. Wolniansky, "Detection algorithm and initial laboratory results using v-blast space-time communication architecture," *Electronic Letters*, vol. 35, pp. 14–16, January 1999.

[6] H. Bölcskei and A. J. Paulraj, *Multiple-input multiple-output (MIMO) wireless systems*, pp. 90.1 – 90.14. CRC Press, 2nd ed., 2002.

[7] G. H. Golub and C. F. V. Loan, *Matrix computations*. John Hopkins University Press, Baltimore, 1983.

[8] G. J. Foschini and M. J. Gans, "On limits of wireless communications in a fading environment when using multiple antennas," *Wireless Pers. Commun.*, vol. 6, no. 3, pp. 311–335, 1998.

[9] R. L. Cupo., G. D. Golden, C. Martin, K. L. Sherman, N. R. Sollenbergen, J. H. Winters, and P. W. Wolniansky, "A four-element adaptive antenna array for is-136 pcs base stations," (Phoenix), pp. 1577–1581, IEEE Vehicular Technology Conference, May 1997.

Appendix A

A.1 Singular Value Decomposition and Moore-Penrose Pseudoinverse

In this section, we follow the presentation and notation in Telatar [1]. Let $\mathcal{C}^{N\times M}$ ($\mathcal{R}^{N\times M}$) denotes the set of all complex-valued (real-valued) matrices with N rows and M columns. Any matrix $\mathbf{H} \in \mathcal{C}^{N\times M}$, can be expressed as

$$\mathbf{H} = \mathbf{U}\mathbf{D}\mathbf{V}^{\dagger}$$

where $\mathbf{U}$ is an $N \times N$ unitary matrix, $\mathbf{V}$ is an $M \times M$ unitary matrix, and $\mathbf{D}$ is an $N \times M$ matrix whose diagonal elements D_{ii} equal the non-negative square roots of the eigenvalues of $\mathbf{H}\mathbf{H}^{\dagger}$ (which are non-negative since $\mathbf{H}\mathbf{H}^{\dagger}$ is positive semi-definite), and off diagonal elements D_{ij} $(i \neq j)$ are 0 (see [1]). The pseudo-inverse is then given by

$$\mathbf{H}^{+} = \mathbf{V}\mathbf{D}^{+}\mathbf{U}^{\dagger}$$

where $\mathbf{D}^{+}$ is the matrix obtained by taking the transpose of $\mathbf{D}$ and setting $D_{ii}^{+} = D_{ii}^{-1}$ if $D_{ii} > 0$ and $D_{ii}^{+} = 0$ otherwise.

Appendix B

Code

```
% SER of V-BLAST, ZF and LLSE receivers for (M,N)=(8,12) and 16-QAM modulation.
% For each Eb/N0 value, we perform 10.000 iteration.

clear all;
close all;
% 16 point QAM is used
partition=[-2,0,2];
xcodebook=[-3,-1,1,3];
ycodebook=xcodebook;
M=8; % no of transmitter antennas
N=12; % no of receiver antennas

Es = 2*sum(xcodebook * xcodebook')/size(xcodebook,2); % average symbol energy per antenna
Eb = Es/(2*log2(size(xcodebook,2))); % transmitted bit energy per antenna
EbN0 = -10:2:4;
N0 = Eb./10.^(EbN0/10); % Noise power

F=1000; % no of trials at a given noise level
for T=1:length(EbN0) % T loop; choose SNR level
tic
    % V-BLAST/ZF algorithm
    er=0; % block error event counter
    bler(T)=0; % block error rate
```

```
24:
25:         % V-BLAST/LLSE algorithm
26:         mer=0; % block error event counter
27:         mbler(T)=0; % block error rate
28:
29:         % ZF algorithm
30:         zer=0; % block error event counter
31:         zbler(T)=0; % block error rate
32:
33:         % LLSE algorithm
34:         ser=0; % block error event counter
35:         sbler(T)=0; % block error rate
36:
37:
38:         for f=1:F
39:             a=randsrc(M,1,xcodebook)+j*(randsrc(M,1,ycodebook));
40:             H=(randn(N,M)+j*randn(N,M)) / sqrt(2);
41:             v=(randn(N,1)+j*randn(N,1))* sqrt(N0(T)/2);
42:             r=H*a+v;
43:
44:             %Copy r and H for ZF Receiver
45:             zH=H;
46:             zr=r;
47:
48:             %Copy r and H for LLSE Receiver
49:             sH=H;
50:             sr=r;
51:
52:             %Copy r and H for VBLAST/LLSE
53:             mH=H;
54:             mr=r;
55:
56:             % V-BLAST/ZF algorithm begins
57:             k=zeros(1,M);
58:             G=pinv(H);
59:             for i=1:M %i loop
60:                 for J=1:M
61:                     n(J)=(norm(G(J,:)))^2;
62:                 end
63:                 for t=1:i-1
```

```
64:                     n(k(t))= Inf;
65:                 end
66:                 [Y,I]=min(n);
67:                 k(i)=I;
68:                 w=G(I,:);
69:                 y=w*r;
70:                 [o,n1]=quantiz(real(y),partition,xcodebook);
71:                 [o,n2]=quantiz(imag(y),partition,ycodebook);
72:                 b(I)=n1+j*n2;
73:                 r=r-b(I)*H(:,I);
74:                 H(:,I)=0;
75:                 G=pinv(H);
76:             end % end i loop
77:             if sum(abs(a-b.')) ~=0
78:                 er=er+1;
79:             end % V-BLAST with ZF algorithm ends
80:
81:             % V-BLAST/LLSE Algorithm starts
82:             k=zeros(1,M);
83:             W=Es*mH'*pinv(Es*mH*mH'+N0(T)*eye(N)); % Es=ro/M
84:             for i=1:M
85:                 for J=1:M
86:                     n(J)=(norm(W(J,:)))^2;
87:                 end
88:                 for t=1:i-1
89:                     n(k(t))= Inf;
90:                 end
91:                 [Y,I]=min(n);
92:                 k(i)=I;
93:                 my=W(I,:)*mr;
94:                 [o,n1]=quantiz(real(my),partition,xcodebook);
95:                 [o,n2]=quantiz(imag(my),partition,ycodebook);
96:                 mb(I)=n1+j*n2;
97:                 mr=mr-mb(I)*mH(:,I);
98:                 mH(:,I)=0;
99:                 W=Es*mH'*pinv(Es*mH*mH'+N0(T)*eye(N));
100:            end % end i loop
101:
102:            if sum(abs(a-mb.')) ~=0
103:                mer=mer+1;
```

```
104:            end % V-BLAST/LLSE Algorithm ends
105:
106:            % ZF Algorithm begins
107:            zb=zeros(1,M);
108:            zG=pinv(zH);
109:            zy=zG*zr;
110:            for J=1:M
111:                [o,n1(J)]=quantiz(real(zy(J)),partition,xcodebook);
112:                [o,n2(J)]=quantiz(imag(zy(J)),partition,ycodebook);
113:            end
114:
115:            zb(1:M)=n1(1:M)+j*n2(1:M);
116:            if sum(abs(a-zb.')) ~=0
117:                zer=zer+1;
118:            end % ZF Algorithm ends
119:
120:
121:            % LLSE Receiver
122:            sy=Es*sH'*pinv(Es*sH*sH'+N0(T)*eye(N))*sr;
123:            for J=1:M
124:                [o,n1(J)]=quantiz(real(sy(J)),partition,xcodebook);
125:                [o,n2(J)]=quantiz(imag(sy(J)),partition,ycodebook);
126:            end
127:            sb=n1+j*n2;
128:            if sum(abs(a-sb.')) ~=0
129:                ser=ser+1;
130:            end
131:
132:        end % end f loop
133:        bler(T)=(er) / F;
134:        mbler(T)=(mer) / F;
135:        zbler(T)=(zer) / F;
136:        sbler(T)=(ser) / F;
137:    toc
138:    end % end of T loop
139:
140:
141:    figure
142:    semilogy(EbN0,bler,'-dr')
143:    xlabel('Eb/No (dB)'); ylabel('SER');
```

```
144:  hold on;
145:  semilogy(EbN0,mbler,'-+b')
146:  semilogy(EbN0,zbler,'-*r')
147:  semilogy(EbN0,sbler,'-ob')
148:  hold off;
149:  legend('V-BLAST/ZF','V-BLAST/LLSE','ZF','LLSE');
150:  grid
```

```
1:  % Test of V-BLAST/ZF, V-BLAST/LLSE, V-BLAST/ZF/MAP and V-BLAST/ZF/MAP
2:  % algorithms with M transmitter and N receiver antennas. The modulation is 16-QAM.
3:  % At each iteration, a new realization of H is used. T stands for SNR level and there are
4:  % totally F iteration for each choice of T.
5:
6:
7:  clear all;
8:  close all;
9:  % 16 point QAM is used
10: partition=[-2,0,2];
11: xcodebook=[-3,-1,1,3];
12: ycodebook=xcodebook;
13: for i1=1:size(xcodebook,2)
14: for i2=1:size(xcodebook,2)
15: constellation((i1-1)*4+i2)=xcodebook(i1)+j*ycodebook(i2);
16: end
17: end
18:
19: M=8; % no of transmitter antennas
20: N=12; % no of receiver antennas
21:
22: Es = 2*sum(xcodebook * xcodebook')/size(xcodebook,2); % average symbol energy per antenna
23: Eb = Es/(2*log2(size(xcodebook,2))); % transmitted bit energy per antenna
24: EbN0 = -10:2:4;
25: N0 = Eb./10.^(EbN0/10); % Noise power
26:
27: randn('state',12); % initialize state of function for repeatability
28: rand('state',12); % initialize state of function for repeatability
29:
30:
```

```
31:   F=10000; % no of trials at a given noise level
32:
33:   for T=1:length(EbN0) % T loop; choose SNR level
34:       % V-BLAST/ZF algorithm
35:       er=0; % block error event counter
36:       bler(T)=0; % block error rate
37:
38:       % V-BLAST/LLSE algorithm
39:       mer=0; % block error event counter
40:       mbler(T)=0; % block error rate
41:
42:       % V-BLAST/ZF/MAP algorithm
43:       zer=0; % block error event counter
44:       zbler(T)=0; % block error rate
45:
46:       % V-BLAST/LLSE/MAP algorithm
47:       ser=0; % block error event counter
48:       sbler(T)=0; % block error rate
49:
50:       for f=1:F
51:           a=randsrc(M,1,xcodebook)+j*(randsrc(M,1,ycodebook));
52:           H=(randn(N,M)+j*randn(N,M)) / sqrt(2);
53:           v=(randn(N,1)+j*randn(N,1))* sqrt((N0(T)/2));
54:           r=H*a+v;
55:
56:           %Copy r and H for V-BLAST/LLSE Receiver
57:           mH=H;
58:           mr=r;
59:
60:           %Copy r and H for V-BLAST/ZF/MAP Receiver
61:           zH=H;
62:           zr=r;
63:
64:           %Copy r and H for V-BLAST/LLSE/MAP Receiver
65:           sH=H;
66:           sr=r;
67:
68:           % V-BLAST/ZF algorithm begins
69:           k=zeros(1,M);
70:           G=pinv(H);
```

```
71:             for i=1:M %i loop
72:                 for J=1:M
73:                     n(J)=(norm(G(J,:)))^2;
74:                 end
75:                 for t=1:i-1
76:                     n(k(t))= Inf;
77:                 end
78:                 [Y,I]=min(n);
79:                 k(i)=I;
80:                 w=G(I,:);
81:                 y=w*r;
82:                 [o,n1]=quantiz(real(y),partition,xcodebook);
83:                 [o,n2]=quantiz(imag(y),partition,ycodebook);
84:                 b(I)=n1+j*n2;
85:                 r=r-b(I)*H(:,I);
86:                 H(:,I)=0;
87:                 G=pinv(H);
88:             end % end i loop
89:             if sum(abs(a-b.')) ~=0
90:                 er=er+1;
91:             end % V-BLAST with ZF algorithm ends
92:
93:
94:             % V-BLAST/LLSE Algorithm starts
95:             k=zeros(1,M);
96:             W=Es*mH'*pinv(Es*mH*mH'+N0(T)*eye(N)); % Es=ro/M
97:             for i=1:M
98:                 for J=1:M
99:                     n(J)=(norm(W(J,:)))^2;
100:                end
101:                for t=1:i-1
102:                    n(k(t))= Inf;
103:                end
104:                [Y,I]=min(n);
105:                k(i)=I;
106:                my=W(I,:)*mr;
107:                [o,n1]=quantiz(real(my),partition,xcodebook);
108:                [o,n2]=quantiz(imag(my),partition,ycodebook);
109:                mb(I)=n1+j*n2;
110:                mr=mr-mb(I)*mH(:,I);
```

```
111:            mH(:,I)=0;
112:            W=Es*mH'*pinv(Es*mH*mH'+N0(T)*eye(N));
113:        end % end i loop
114:
115:        if sum(abs(a-mb.')) ~=0
116:            mer=mer+1;
117:        end % V-BLAST/LLSE Algorithm ends
118:
119:
120:        % V-BLAST/ZF/MAP algorithm starts
121:        G=pinv(zH);
122:        u=zeros(M,1); % outputs in each channel
123:        p=zeros(1,M);
124:        zk=zeros(1,M);
125:        ahat=zeros(1,M); % decisions in each channel
126:
127:        for i=1:M % i loop
128:            u = G*zr;
129:            for J=1:M
130:                if size(find(zk==J),2) == 0 % exclude J that have been decided earlier
131:                [o,n1]=quantiz(real(u(J)),partition,xcodebook);
132:                [o,n2]=quantiz(imag(u(J)),partition,ycodebook);
133:                n(J)=N0(T)*(norm(G(J,:)))^2;
134:                % decision for J'th channel
135:                ahat(J)=n1+j*n2;
136:                % calculate decision reliability probabilities
137:                numerat = exp(-(1/n(J))*(abs(ahat(J)-u(J)))^2); % numerator of pij
138:                denom =0; % denominator of pij
139:                for i1=1:size(constellation,2)
140:                    denom = denom + exp(-(1/n(J))*(abs(constellation(i1)-u(J)))^2);
141:                end
142:                p(J)=numerat/denom;
143:            else % if J has already been processed
144:                p(J)=-1;
145:            end
146:            end
147:            [Y,I]=max(p);
148:            zk(i)=I;
149:            zb(I) = ahat(I);
150:            zr=zr-zb(I)*zH(:,I);
```

```
151:            zH(:,I)=0;
152:            G=pinv(zH);
153:
154:        end % end i loop
155:
156:        if sum(abs(a-zb.')) ~=0
157:            zer=zer+1;
158:        end
159:
160:        % V-BLAST/LLSE/MAP algorithm
161:        G=Es*sH'*pinv(Es*sH*sH'+N0(T)*eye(N));
162:        u=zeros(M,1); % outputs in each channel
163:        p=zeros(1,M);
164:        k=zeros(1,M);
165:        ahat=zeros(1,M); % decisions in each channel
166:
167:        for i=1:M % i loop
168:            u = G*sr;
169:            for J=1:M
170:                if size(find(k==J),2) == 0 % exclude J that have been decided earlier
171:                [o,n1]=quantiz(real(u(J)),partition,xcodebook);
172:                [o,n2]=quantiz(imag(u(J)),partition,ycodebook);
173:                n(J)=N0(T)*(norm(G(J,:)))^2;
174:                % decision for J'th channel
175:                ahat(J)=n1+j*n2;
176:
177:                % calculate reliability probabilities
178:                % calculate decision reliability probabilities
179:                numerat = exp(-(1/n(J))*(abs(ahat(J)-u(J)))^2); % numerator of pij
180:                denom =0; % denominator of pij
181:                for i1=1:size(constellation,2)
182:                    denom = denom + exp(-(1/n(J))*(abs(constellation(i1)-u(J)))^2);
183:                end
184:                p(J)=numerat/denom;
185:            else % if J has already been processed
186:                p(J)=-1;
187:            end
188:            end
189:            [Y,I]=max(p);
190:            k(i)=I;
```

```
191:                    sb(I) = ahat(I);
192:                    sr=sr-sb(I)*sH(:,I);
193:                    sH(:,I)=0;
194:                    G=Es*sH'*pinv(Es*sH*sH'+N0(T)*eye(N));
195:
196:            end % end i loop
197:
198:            if sum(abs(a-sb.')) ~=0
199:                ser=ser+1;
200:            end
201:
202:        end % end f loop
203:        bler(T) = (er) / F;
204:        mbler(T) = (mer) / F;
205:        zbler(T) = (zer) / F;
206:        sbler(T) = (ser) / F;
207:
208:
209:    end % end of T loop
210:
211:
212:    d=8;
213:    figure
214:    semilogy(EbN0(1:d),bler(1:d),'-+r')
215:    xlabel('Eb/No (dB)'); ylabel('SER');
216:    hold on;
217:    semilogy(EbN0(1:d),mbler(1:d),'-db')
218:    semilogy(EbN0(1:d),zbler(1:d),'-or')
219:    semilogy(EbN0(1:d),sbler(1:d),'-*b')
220:    hold off;
221:    legend('V-BLAST/ZF','V-BLAST/LLSE','V-BLAST/ZF/MAP','V-BLAST/LLSE/MAP');
222:    grid
```

```
1:    % This file compares V-BLAST/ZF/MAP, V-BLAST/ZF and ML.
2:    % (M,N)=(4,12) and 4-QAM modulation.
3:    % For each Eb/N0 value, we perform F iteration.
4:
5:    clear all;
```

```
close all;
% 4 point QAM is used
partition=[0];
xcodebook=[-1,1];
ycodebook=xcodebook;
for i1=1:size(xcodebook,2)
for i2=1:size(xcodebook,2)
constellation((i1-1)*2+i2)=xcodebook(i1)+j*ycodebook(i2); % constellation
end
end
Eb = 1; % bit energy for this constellation

F=[1000, 1000, 1000, 1000, 5000, 10000, 100000]; % no of trials at a given noise level
M=4; % No of transmitter antennas
N=12; % No of receiver antennas

Es = 2*sum(xcodebook * xcodebook')/size(xcodebook,2); % average symbol energy per antenna
Eb = Es/(2*log2(size(xcodebook,2))); % transmitted bit energy per antenna
EbN0 = -12:2:0;
N0 = Eb./10.^(EbN0/10); % Noise power
randn('state',12); % initialize state of function for repeatability
rand('state',12); % initialize state of function for repeatability

for T=1:length(EbN0) % T loop; choose SNR level

    % V-BLAST/ZF algorithm
    er=0; % block error event counter
    bler(T)=0; % block error rate

    % V-BLAST/ZF/MAP algorithm
    zer=0; % block error event counter
    zbler(T)=0; % block error rate

    % ML rule
    mler=0;
    mlbler(T)=0;


    for f=1:F(T) % f loop

```

```
46:         a=randsrc(M,1,xcodebook)+j*(randsrc(M,1,ycodebook));
47:         H=(randn(N,M)+j*randn(N,M)) / sqrt(2);
48:         v=(randn(N,1)+j*randn(N,1))* sqrt(N0(T)/2);
49:         r=H*a+v;
50:
51:         %Copy r and H for ML Receiver
52:         mlH = H;
53:         mlr = r;
54:
55:         %Copy r and H for V-BLAST/ZF/MAP Receiver
56:         zH = H;
57:         zr = r;
58:
59:         % V-BLAST/ZF algorithm begins
60:         k=zeros(1,M);
61:         G=pinv(H);
62:         for i=1:M %i loop
63:             for J=1:M
64:                 n(J)=(norm(G(J,:)))^2;
65:             end
66:             for t=1:i-1
67:                 n(k(t))= Inf;
68:             end
69:             [Y,I]=min(n);
70:             k(i)=I;
71:             w=G(I,:);
72:             y=w*r;
73:             [o,n1]=quantiz(real(y),partition,xcodebook);
74:             [o,n2]=quantiz(imag(y),partition,ycodebook);
75:             b(I)=n1+j*n2;
76:             r=r-b(I)*H(:,I);
77:             H(:,I)=0;
78:             G=pinv(H);
79:         end % end i loop
80:         if sum(abs(a-b.')) ~=0
81:             er=er+1;
82:         end % V-BLAST with ZF algorithm ends
83:
84:         % V-BLAST/ZF/MAP algorithm starts
85:         G=pinv(zH);
```

```
86:         u=zeros(M,1); % outputs in each channel
87:         p=zeros(1,M);
88:         zk=zeros(1,M);
89:         ahat=zeros(1,M); % decisions in each channel
90:
91:         for i=1:M % i loop
92:             u = G*zr;
93:             for J=1:M
94:                 if size(find(zk==J),2) == 0 % exclude J that have been decided earlier
95:                 [o,n1]=quantiz(real(u(J)),partition,xcodebook);
96:                 [o,n2]=quantiz(imag(u(J)),partition,ycodebook);
97:                 n(J)=N0(T)*(norm(G(J,:)))^2;
98:                 % decision for J'th channel
99:                 ahat(J)=n1+j*n2;
100:                % calculate decision reliability probabilities
101:                numerat = exp(-(1/n(J))*(abs(ahat(J)-u(J)))^2); % numerator of pij
102:                denom =0; % denominator of pij
103:                for i1=1:size(constellation,2)
104:                    denom = denom + exp(-(1/n(J))*(abs(constellation(i1)-u(J)))^2);
105:                end
106:                p(J)=numerat/denom;
107:            else % if J has already been processed
108:                p(J)=-1;
109:            end
110:            end
111:            [Y,I]=max(p);
112:            zk(i)=I;
113:            zb(I) = ahat(I);
114:            zr=zr-zb(I)*zH(:,I);
115:            zH(:,I)=0;
116:            G=pinv(zH);
117:
118:        end % end i loop
119:
120:        if sum(abs(a-zb.')) ~=0
121:            zer=zer+1;
122:        end
123:
124:        % ML algorithm begins
125:        c = constellation;
```

```
126:            val=Inf;
127:            for n=1:4
128:                for m=1:4
129:                    for k=1:4
130:                        for g=1:4
131:                            d=[c(n),c(m),c(k),c(g)]';
132:                            if norm(mlr-mlH*d)<val
133:                                mla=[c(n),c(m),c(k),c(g)]';
134:                                val=norm(mlr-mlH*d);
135:                            end
136:                        end
137:                    end
138:                end
139:            end
140:
141:            if sum(abs(a-mla)) ~=0
142:                mler=mler+1;
143:            end
144:
145:
146:        end % end f loop
147:
148:        bler(T)=(er) / F(T); % for V-BLAST/ZF
149:        zbler(T)=(zer) / F(T); % for V-BLAST/ZF/MAP
150:        mlbler(T)=(mler) / F(T); % for ML
151:
152:    end % end T loop
153:
154:    figure
155:    semilogy(EbN0,mlbler)
156:    xlabel('Eb/No (dB)'); ylabel('SER');
157:    hold on;
158:    semilogy(EbN0,zbler,'r')
159:    semilogy(EbN0,bler,'–')
160:    hold off;
161:    legend('ML','V-BLAST/ZF/MAP','V-BLAST/ZF');
162:    grid
```