Low Complexity Iterative Receiver for Frequency Selective Channels and MIMO Systems

A Project Report

submitted by

RAKESH MALLADI

in partial fulfilment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

AND

MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

May 2011

THESIS CERTIFICATE

This is to certify that the thesis titled **Low Complexity Iterative Receiver for Frequency-Selective Channels and MIMO Systems**, submitted by **Rakesh Malladi**, to the Indian Institute of Technology Madras, for the award of the degree of **Bachelor of Technology** and **Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. R David Koilpillai,
Project Guide,
Professor,
Dept. of Electrical Engineering,
IIT-Madras, 600 036

Dr. Kiran Kuchi,
Project Guide,
Adjunct Professor,
Dept. of Electrical Engineering,
IIT-Madras, 600 036

Place: Chennai

Date: 27th May 2011

ACKNOWLEDGEMENTS

First and foremost I am deeply indebted to my guides Prof.R David Koilpillai and Dr. Kiran Kuchi for their guidance and encouragement throughout the course of this project. I would like to thank them profusely for their patient guidance throughout the project. I would like to acknowledge the contributions of Nokia Modem Algorithm Design group, Copenhagen for giving a good direction to this project. They have been highly inspiring and motivating in all the meetings with them about the project. Next I would like to express my gratitude to Arvind K R, for introducing me to this exciting problem and for his guidance during the initial stages of this project.

I would like to thank the Department of Electrical Engineering for providing an excellent laboratory and computational resources for making this project possible. I would like to acknowledge Prof Harishankar Ramachandran for his guidance when I was uncertain about converting from B.Tech program to Dual Degree program. I would like to acknowledge the role of my friends Santhosh, Rakesh Pokala, Amar Sagar, Sravan Kumar Ayila, my senior Sumith and others for encouraging me to follow my heart and convert to Dual Degree Program. I would like to acknowledge and thank Prof Srikrishna Bhashyam for making all the 5 communications courses I took under him interesting.

It is a pleasure to thank all my lab mates - Vinay Praneeth, Mahesh, Sashidharan, Rakesh Pokala, Karthik, Siddharth, Bala, Karthik, Murali and others for their wonderful company throughout the year. I would like thank all my friends for making my five years of stay at IIT Madras enjoyable.

Finally, since a word of thanks would be insufficient to express my gratitude to my parents, I would like to dedicate my work to them.

ABSTRACT

KEYWORDS: Iterative receiver; Turbo Equalization; MAP algorithm ; RSMAP

An optimal receiver should detect the transmitted data directly from the received symbols using the complex statistical relationship between the two. Such a receiver is not feasible in practice. So we develop an iterative receiver based on turbo principle, which achieves near optimal performance. Also inner decoder should pass soft information to maximize the performance from the outer decoder. These soft output algorithms are generally complex to implement. In this thesis we present low-complexity inner decoding algorithms that produce soft information for two types of inner codes, trellis based (for frequency-selective fading channels) and tree-based (MIMO channels in flat fading). Turbo-equalizer with low-complexity equalization algorithm is developed for ISI channels. Finally the performance of the widely-linear low-complexity equalizer with a widely-linear MMSE DFE prefilter to suppress co-channel interference from a single interferer is investigated.

A reduced complexity, near MAP-optimal soft detector that outputs a posteriori probabilities (APP's) for multiple-input multiple-output (MIMO) systems in flat fading channels is proposed. This detection algorithm is based on BCJR algorithm and also uses ideas from reduced state sequence estimation (RSSE) and set partitioning. This algorithm is shown to be near-optimal. Exact complexity of implementing this algorithm is computed in terms of number of computations required. Realizing its high complexity, we propose two approximate algorithms, one with only forward recursion and soft decision failure prevention mechanism and the other based on the famous max log approximation. We analyze these algorithms from complexity-performance trade-off point of view.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

APP	A Posteriori Probability
LLR	Log-likelihood Ratio
ISI	Inter Symbol Interference
ML	Maximum Likelihood
LMMSE	Linear Minimum Mean Square Equalizer
GSM	Global System for Mobile communication
EDGE	Enhanced Data rates for GSM Evolution
CCI	Co-Channel Interference
SNR	Signal to Noise Ratio
GMSK	Gaussian Minimum Shift Keying
PAM	Pulse Amplitude modulation
SAIC	Single Antenna Interference Cancellation
MUROS	Multiple Users Reusing One Slot
ISI	Inter Symbol Interference
MLSE	Maximum Likelihood Sequence Estimation
RSSE	Reduced State Sequence Estimation
DDFSE	Delayed Decision Feedback Sequence Estimation
DFE	Decision Feedback Equalizer
WL	Widely Linear
VAMOS	Voice services over Adaptive Multi-user channel in One Slot
SCPIR	Sub Channel Power Imbalance Ratio
OSC	Orthogonal Subchannel

NOTATION

$\underline{\alpha}$	Forward state metrics vector
$\underline{\beta}$	Backward state metrics vector
γ	Transition Probability
$\|S\|$	Cardinality of the set S
A'	Conjugate Transpose of matrix A
$A^\star$	Conjugate of matrix A
$[A]_{i,j}$	Refers to the $(i,j)^{th}$ element of the matrix A
$A \circ B$	Hadamard product (entry wise matrix multiplication) of two matrices A and B of same dimension
$< \mathbf{a}, \mathbf{b} >$	Inner product between vectors **a** and **b**

CHAPTER 1

Introduction

Reliable data transmission over wireless channels faces two main challenges:

- Wireless channel propagation effects (multi-path fading and time dispersion).
- Noise added due to radio (RF) front end at receiver.

The noise at the RF end of a receiver is usually modeled as white Gaussian noise. Efficient channel coding schemes are designed to deal with this noise. The channel encoder at the transmitter introduces necessary redundancy to protect the data from errors. This is referred to as the outer code.

The effects due to a wireless channel are classified into large scale and small scale fading effects. While large scale fading determines the power of signal received at the receiver, it is the small scale fading that causes significant variation of signal strength over a short period of time or distance. The channel in many typical environments like Typical Urban (TU), Hilly Terrain (HT) environments has been well characterized and all these environments offer a frequency-selective channel for data transmission. This effect is very similar to the tapped delay line model that represents channel coding schemes like convolutional codes. Hence the effect of channel on transmitted symbols is referred to as an inner code. Many communication systems can be viewed as a concatenation of inner code with an outer code (usually channel code).

An optimal receiver detects the transmitted data bits, using the complex statistical relationship between transmitted data bits and the received symbols, after being subjected to both inner and outer codes. Implementing such a receiver is very complex in practice. Hence most practical receivers today separate the detection process at receiver into two separate sequential tasks: first decode the inner code and then decode the outer code. This approach is suboptimal, since we are separately implementing two inherently dependent tasks. Clearly, there is scope for better performance, if we

can improve the receiver architecture, by exploiting the dependencies between the two decoding steps.

The idea of an iterative receiver, based on the **turbo principle**, is proposed to improve the performance. Turbo Receiver is an iterative receiver architecture, which achieves the performance of an optimal Maximum A Posteriori (MAP) detector via iterative message passing between a soft-input soft-output (SISO) inner decoder and a SISO outer decoder. A general framework for implementing the turbo receiver is developed in this thesis. This general frame work is developed using the specific example of convolutional coded data transmitted over a frequency selective channel.

Next we focus on the inner code decoder and try to develop general low-complexity, near-optimal decoding algorithms for the inner code. Specifically, we focus on two classes of communication systems, one class where the inner code can be depicted on a trellis and the other class where it is depicted on a tree. Equalization of inter-symbol interference (ISI) channels belong to the first class, whereas Multiple Input Multiple Output (MIMO) systems in flat fading channels belong to the later category. We develop low complexity soft equalization algorithms for frequency-selective channels. Next we apply the low-complexity iterative receiver developed for frequency selective channels to suppress interference in an interference limited scenario. Single Antenna Interference Cancellation (SAIC) algorithms in conjunction with iterative receivers already developed, are used to suppress the interference from a co-channel interferer. Specifically, we are interested in the case of a VAMOS (Voice services over Adaptive Multi-user channels on One Slot) with a GMSK interferer and EDGE (8-PSK) with GMSK interferer scenarios.

For MIMO systems, we also develop low complexity forward only algorithms that achieve excellent complexity-performance trade-off. Exact number of computations necessary for implementing this algorithm for a general MIMO system is also estimated.

Simulations results of our proposed receiver are presented in these two cases.

The organization of thesis is as follows

Chapter 2 introduces the need for turbo equalization, using a frequency-selective fading channel as an example. The general framework for a turbo-equalizer is presented in this chapter. The gains from turbo equalization applied to coded BPSK modulation in a frequency selective channel are presented.

Chapter 3 focuses on low-complexity equalization algorithms for frequency-selective channels. Hard and soft equalization algorithms are presented in this chapter. Performance of these algorithms for a coded 8-PSK modulated system are presented. Also, the performance of the turbo equalizer in conjunction with the inner-code decoder developed are presented.

Chapter 4 focuses on interference suppression receivers, by combining SAIC filter with iterative receiver developed for frequency selective channels. Exact performance of this receiver for specific cases of VAMOS with GMSK interference and EDGE with GMSK interference are presented.

Chapter 5 focuses on low-complexity soft-detection algorithms for MIMO systems in flat fading channels. A forward-only soft-detection algorithm that gives excellent complexity-performance trade-off is presented. Also explicit number of computations required to implement all the variations of the basic algorithm presented are calculated.

Chapter 6 contains concluding remarks on this work. Also possible avenues for future work are identified.

CHAPTER 2

Turbo Equalization

2.1 Conventional Receiver

A typical communication system consists of a transmitter, a channel and a receiver. The transmitter usually contains many blocks, which convert the binary data to an analog signal to transmit over the channel. The receiver then processes the received signal to detect the transmitted binary data. A typical communication system is shown in Fig. 2.1 below.

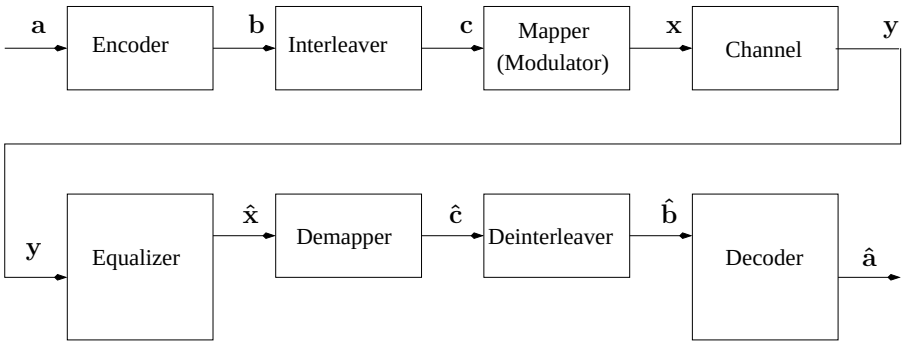

Figure 2.1: Typical Block Diagram of a Communication System

The functions of each block are explained below:

- Encoder
  - Introduces redundancy in data $\mathbf{a}$ by using error control coding.
  - Error control coding protects the data from additive noise.
- Interleaver
  - To insulate information $\mathbf{b}$ from bursty noise.
- Mapper/Modulator

  - Maps binary information bits $\mathbf{c}$ to complex symbols $\mathbf{x}$

- Complex symbols are then passed through an A/D converter, upconverted to generate a bandpass signal and transmitted through the channel.

- The receiver then filters out the out-of-band noise and downconverts the received signal to baseband.

- Equalizer

  - It attempts to remove the ISI (inter-symbol interference) to recover the transmitted symbols.

  - It could either pass hard estimates or soft information about the transmitted symbols.

- Demapper

  - Maps complex transmitted symbols $\hat{\mathbf{x}}$ to binary information bits $\hat{\mathbf{c}}$

- Deinterleaver

  - Performs a transformation that undoes the one done by the interleaver.

- Decoder

  - This block estimates the transmitted data bits, $\hat{\mathbf{a}}$ from the received code bits $\hat{\mathbf{b}}$.

2.1.1 Practical Receiver

The optimal receiver, which achieves the minimum probability of bit error, is the one that maximizes the a posteriori probability (APP) given the observed sequence $\mathbf{y}$, i.e,

$$\hat{a}\left[k\right] = \underset{a \in \{0,1\}}{\text{argmax}} \; \mathcal{P}\left(a\left[k\right] = a|\mathbf{y}\right) \tag{2.1}$$

Such an algorithm is called the 'Maximum A Posteriori (MAP) Algorithm'.

Figure 2.2: Optimal Receiver

$$\begin{aligned}\mathcal{P}\left(a\left[k\right]=a|\mathbf{y}\right) &= \sum_{\forall \mathbf{a}:a[k]=a} \mathcal{P}\left(\mathbf{a}|\mathbf{y}\right) \\ &= \sum_{\forall \mathbf{a}:a[k]=a} \frac{\mathcal{P}\left(\mathbf{y}|\mathbf{a}\right)\mathcal{P}\left(\mathbf{a}\right)}{\mathcal{P}\left(\mathbf{y}\right)} \end{aligned} \tag{2.2}$$

Computing the $\mathcal{P}\left(\mathbf{y}|\mathbf{a}\right)$ is extremely difficult, since the received symbols depend on entire vector of transmitted data bits, $\mathbf{a} = (a[1], a[2], \ldots, a[K])$. Implementing such a receiver requires the knowledge of complex statistical relationship between observations and transmitted bits, which is practically infeasible.

Hence most practical receivers today split the process of detection of data into two separate tasks:

- Equalization: It processes the received signal to take care of ISI introduced by the channel. It is further classified as:
  (a) Algorithmic Classification
    · Trellis Based Equalizer - Examples include MAP symbol detection [2], [15] and ML sequence detection [9].
    · Linear Filtering Based Equalizers - Examples include zero forcing equalizer and LMMSE.
  (b) Based on Output
    · Hard Output Equalizers - They output estimated complex symbols.
    · Soft Output Equalizers - They output APP's or Log-likelihood Ratios (LLR's) for trellis based equalizers and estimation error, $e_k = \hat{x_k} - x_k$ for linear filter based equalizers.
- Decoding: It recovers the transmitted data bits from equalized symbols, taking care of additive noise at receiver.

The easiest way of implementing separate equalization and decoding is for the equalizer to pass hard estimates of the transmitted symbols, which are then mapped to corresponding code bits. These estimates are then passed to decoder to estimate the data bits. Passing hard estimates between equalizer and decoder destroys some information on how likely the estimated symbol might have been. This additional 'soft' information on transmitted bit estimates, in terms of probability of each code bit taking the value 0 or 1, can be exploited by the channel decoding algorithm. In fact many practical receivers today implement this. Our simulations results indicate that passing soft information will provide gains of about 2 dB over passing hard information between equalizer and decoder for data transmission in frequency selective channels.

It is very important to note that equalization and decoding are inherently dependent tasks. Performance degradation occurs due to separation of these dependent tasks. Main motivation behind Turbo Equalization is to enable feasible approaches to jointly solving the equalization and decoding tasks.

2.2 Turbo Equalization

Turbo Equalization is an iterative equalization and decoding technique that approaches the performance of a MAP detector via iterative message passing between Soft-Input Soft-Output (SISO) equalizer and SISO decoder.

This work is based on the ideas developed in turbo codes. The remarkable performance of turbo codes suggests that performance will be improved if soft information is not restricted to flow in only one direction. Hence we create a feedback loop between the equalizer and the decoder. The equalizer computes soft information on transmitted symbols from the received symbols, $\mathbf{y}$, which is mapped to soft information on coded bits. This is fed as input to the decoder. The channel decoder in addition to estimating the data bits $\mathbf{a}$ also estimates information about the coded bits $\mathbf{b}$. This new information is the likelihood of certain code bits being transmitted, which is not derived from already known information which was input to the decoder. This soft information can be mapped back to information on transmitted symbols, which is given as APP's to the equalizer. The equalizer then utilizes this a priori information and the received complex

symbols to output information on transmitted symbols. This creates a feedback loop between the equalizer and decoder, in which each block communicates its belief on each bit/symbol taking a particular value. This process is called as 'message passing' or 'belief propagation'. This process is continued till a stopping criterion (either the number of iterations or achieving certain probability of error is reached).

Since we are interested in trellis-based equalization algorithms (they outperform linear-filter-based algorithms, but are more complex), soft information is usually represented by a posteriori probabilities. This soft information, when dealing with binary variables, is more conveniently represented by 'Log Likelihood Ratio (LLR)' rather than by probabilities. The LLR for a binary random variable a is defined as

$$L(a) = \ln\left[\frac{\mathcal{P}(a=0)}{\mathcal{P}(a=1)}\right]$$

Conditional LLR of each data bit $a\,[k]$ is given by

$$L(a\,[k] \mid \mathbf{y}) = \ln\left[\frac{\mathcal{P}(a\,[k]=0 \mid \mathbf{y})}{\mathcal{P}(a\,[k]=1 \mid \mathbf{y})}\right]$$

From (2.2), we have

$$\begin{aligned} L(a\,[k] \mid \mathbf{y}) &= \ln \frac{\sum\limits_{\forall \mathbf{a}: a[k]=0} \mathcal{P}\left(\mathbf{y}|\mathbf{a}\right) \prod\limits_{i=1}^{K} \mathcal{P}\left(a_i\right)}{\sum\limits_{\forall \mathbf{a}: a[k]=1} \mathcal{P}\left(\mathbf{y}|\mathbf{a}\right) \prod\limits_{i=1}^{K} \mathcal{P}\left(a_i\right)} \\ &= \underbrace{\ln \frac{\sum\limits_{\forall \mathbf{a}: a[k]=0} \mathcal{P}\left(\mathbf{y}|\mathbf{a}\right) \prod\limits_{i=1:i\neq k}^{K} \mathcal{P}\left(a_i\right)}{\sum\limits_{\forall \mathbf{a}: a[k]=1} \mathcal{P}\left(\mathbf{y}|\mathbf{a}\right) \prod\limits_{i=1:i\neq k}^{K} \mathcal{P}\left(a_i\right)}}_{} + L(a\,[k]) \\ &= L_{ext}\left(a\,[k]\,|\mathbf{y}\right) + L(a\,[k]) \end{aligned} \tag{2.3}$$

The term $L_{ext}\left(a\,[k]\,|\mathbf{y}\right)$ is called the 'Extrinsic LLR' and $L(a\,[k])$ is called the 'Intrinsic LLR' of $a\,[k]$ given the received sequence $\mathbf{y}$.

Simple and efficient algorithms for equalization and decoding assume soft information about each bit (or symbol) is independent of the information about other bits (or

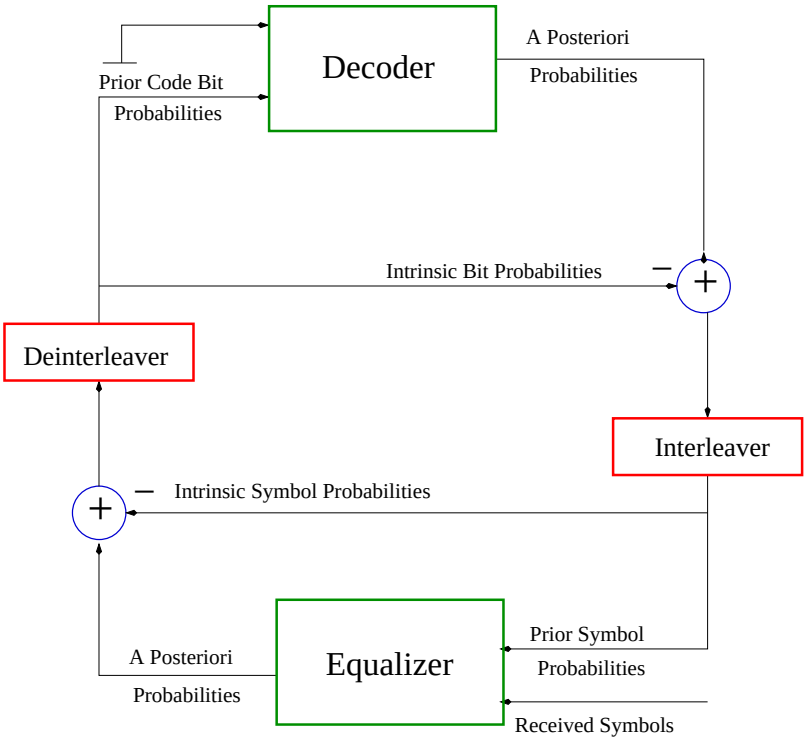

Figure 2.3: Block Diagram of a Turbo Equalizer

symbols). However when the equalizer (or decoder) outputs soft information about a particular bit using the input they received from decoder (or equalizer) about the same bit, then this independence between soft a priori information and observations is lost. It creates a feedback loop of length 2. In fact passing intrinsic information (included in $L(a\,[k]\mid \mathbf{y})$) will lead to faster convergence to a local optimum rather than global optimum. Hence when feeding soft information between equalizer and decoder, only the 'Extrinsic information, $L_{ext}\,(a\,[k]\,|\mathbf{y})$' should be passed between equalizer and decoder.

The Fig. 2.3 represents the entire turbo equalization process.

The notation $L\,(\mathbf{b}|\mathbf{p})$ represents the LLR's at the output of decoder, which are fed to equalizer after removing the intrinsic LLR's, $L_{ext}\,(\mathbf{b}|\mathbf{y})$. The LLR's $L_{ext}\,(\mathbf{c}|\mathbf{y})$ are deinterleaved to get $L_{ext}\,(\mathbf{b}|\mathbf{y})$. The LLR's $L\,(\mathbf{c}|\mathbf{y})$ are the outputs from the equalizer which takes received observations $\mathbf{y}$ and the prior LLR's $L_{ext}\,(\mathbf{c}|\mathbf{p})$, which are obtained by interleaving $L_{ext}\,(\mathbf{b}|\mathbf{p})$, as inputs.
All these operations are summarized below[15].

Input

- Channel Coefficients $h[l]$ for $l = 0, 1, ..., L$
- Observation Sequence $\mathbf{y}$
- A sequence of LLRs $\mathrm{L}_{ext}(\mathbf{c} \mid \mathbf{p})$ initialized to 0.
- A predetermined number of iterations l.

Recursively compute for l iterations

- $\mathrm{L}(\mathbf{c} \mid \mathbf{y})$ = Forward/Backward ($L_{ext}(\mathbf{c} \mid \mathbf{p})$).
- $\mathrm{L}_{ext}(\mathbf{c} \mid \mathbf{y}) = \mathrm{L}(\mathbf{c} \mid \mathbf{y}) - \mathrm{L}_{ext}(\mathbf{c} \mid \mathbf{p})$
- $\mathrm{L}(\mathbf{b} \mid \mathbf{y})$ = Forward/Backward ($L_{ext}(\mathbf{b} \mid \mathbf{p})$).
- $\mathrm{L}_{ext}(\mathbf{b} \mid \mathbf{y}) = \mathrm{L}(\mathbf{b} \mid \mathbf{y}) - \mathrm{L}_{ext}(\mathbf{b} \mid \mathbf{p})$ ‘

Output

- Compute data bit estimates $\hat{a}\,[k]$ from $\mathrm{L}(a\,[k] \mid \mathbf{y})$

It is important to note that the above framework developed for Turbo Equalization is independent of the specific algorithm used for equalization and channel decoding. This framework is suited for all equalization algorithms that take in complex received observations and also prior probabilities of the transmitted symbols and output LLR’s of the coded bits. Similarly it is suited for all channel decoding algorithms that take in prior LLR’s of coded bits and outputs the LLR’s of data bits and the coded bits given the input prior probability vector for the coded bits. Also note that the framework developed for turbo equalization is independent of the modulation used. This framework can be used for a general communication system, with the equalizer and the decoder being replaced with an inner and outer code decoder, respectively.

We now study the performance of Turbo Equalizer for the communication system described in the following section.

2.3 System Model

The objective of the communication system is to transmit a vector of binary data, $\mathbf{a} = (a[1, a[2], \ldots a[K])$ and decode the information reliably at the receiver. The transmitter protects the data by introducing redundancy via error correcting codes of rate

$R = K/N$. Without loss of generality, a convolutional rate $\frac{1}{2}$ code, whose generator polynomial is
$(1 + D^2, 1 + D + D^2)$ is used. The output code bits for input $a[k]$ are given by

$$\begin{aligned} b[2k-1] &= a[k] \oplus a[k-2] \\ b[2k] &= a[k] \oplus a[k-1] \oplus a[k-2] \end{aligned} \tag{2.4}$$

These code bits, $\mathbf{b} = (b[1], b[2], \ldots, b[N])$ are further interleaved using a block interleaver, which writes data into the interleaver matrix by columns and reads out along the rows of the matrix. These interleaved coded bits, $\mathbf{c} = (c[1], c[2], \ldots, c[N])$ are then mapped to a complex symbols vector, $\mathbf{x} = (x[1], x[2], \ldots, x[N])$ drawn from a BPSK constellation. These symbols are transmitted over a frequency-selective channel with $(L+1)$ taps with impulse response, $\mathbf{h} = (h[0], h[1], \cdots, h[L])$. In complex baseband representation, the received symbols $\mathbf{y}$ are given by

$$y[k] = \sum_{l=0}^{l=L} h[l]\, x[k-l] + n[k],\ k = 1, 2, \cdots N. \tag{2.5}$$

where $n[k]$ is additive white Gaussian noise with zero mean and variance σ^2.

The receiver processes the received data to estimate transmitted data. We are interested in trellis-based equalization algorithms. Specifically we will study the performance of a MAP symbol detector that applies eqn. (2.1), ignoring the effect of error control coding. This equalizer is optimal in the sense of minimizing the symbol error probability. We also implement soft (hard) convolutional decoders based on MAP principle (Viterbi algorithm based approach [10]).

2.4 Maximum *A Posteriori* Symbol Detector Algorithm

For a channel with an impulse response of length $(L+1)$ taps, there are L delay elements. For a binary input alphabet $\{+1, -1\}$, this implies the system can be in anyone of the 2^L possible states. Let $S = \{r_1, r_2, \ldots, r_{2^L}\}$ represent the set of possible states. Let the state of the channel at time k be represented by a random variable $s_k \in S$. Since

we are interested in binary modulation schemes, given a present state s_k, there are two distinct, unique possible future states s_{k+1} corresponding to input being $+1$ or -1. This system can be represented on a trellis, which gives all the possible future states from each given present state at every stage.

Without loss of generality, let us consider a 3-tap channel, i.e, $\mathbf{h} = [h[0], h[1], h[2]]$. The effect of channel and white Gaussian noise at receiver given by eqn. (2.5) is also represented by a tapped delay line model (Fig. 2.4) below. The

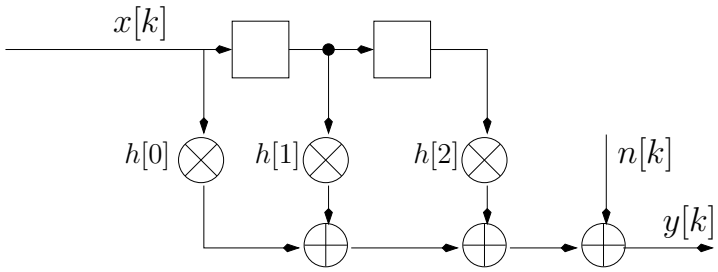

Figure 2.4: tapped delay line model for a 3-tap ISI channel

set of possible states is $S = \{r_0, r_1, r_2, r_3\}$. They are also represented by $\{0, 1, 2, 3\}$. These states at time k are also represented by the ordered pairs $\{(0,0), (0,1), (1,0), (1,1)\}$ respectively, where the first entry corresponds to the memory of channel $x[k-1]$ at time $(k-1)$ and the second entry corresponds to memory $x[k-2]$ at time $(k-2)$.The trellis for this channel is given in Fig. 2.5.

The set $\mathcal{B} = \{(0,0), (0,2), (1,0), (1,2), (2,1), (2,3), (3,1), (3,3)\}$ is the set of all possible transitions in this trellis. A branch in this trellis is denoted by ordered pair $(i, j, x_{i,j}, v_{i,j})$, such that state $s_{k+1} = r_j$ is reached from state $s_k = r_i$ at time k with an input $x[k] = x_{i,j}$ and output $v[k] = v_{i,j}$, where $v[k] = \sum_{l=0}^{l=L} h[l]\, x[k-l]$.
Now let us compute the APP's $\mathcal{P}(x[k] = x|\mathbf{y})$. Let us assume the input random variables $x[k]$ are IID, i.e, $\mathcal{P}(\mathbf{x}) = \prod_{k=1}^{k=N} \mathcal{P}(x[k])$. Now let us compute the probability that transmitted input sequence has a branch $(i, j, x_{i,j}, v_{i,j})$ in the subset state trellis at time k, i.e, $\mathcal{P}(s_{k+1} = r_j, s_k = r_i/\mathbf{y})$. This is computed efficiently based on the forward/backward algorithm [15], [2].

We know

$$\mathcal{P}(s_{k+1} = r_j, s_k = r_i|\mathbf{y}) = \mathcal{P}(s_{k+1} = r_j, s_k = r_i, \mathbf{y}) / \mathcal{P}(\mathbf{y}) \tag{2.6}$$

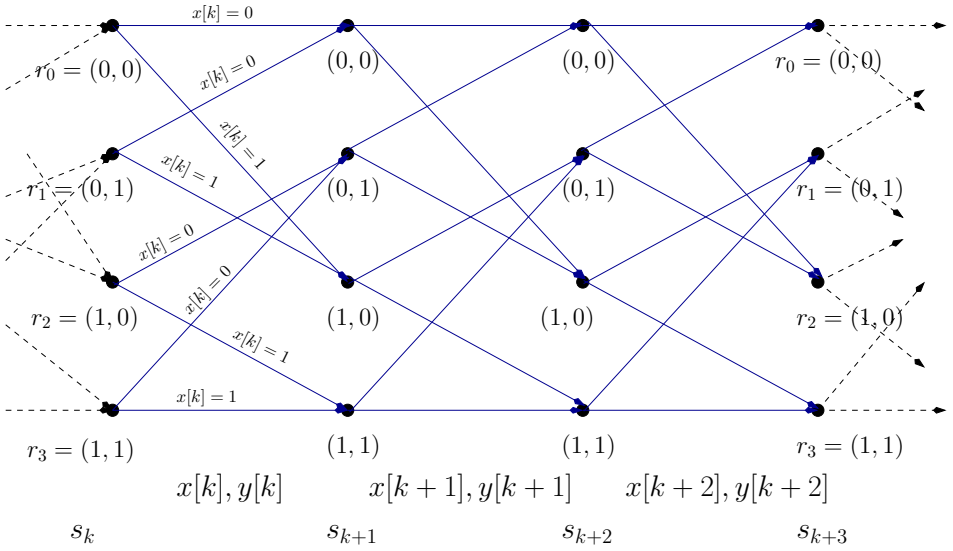

Figure 2.5: 4-state trellis representation for a 3-tap channel with binary modulation

Applying the chain rule, i.e, $\mathcal{P}(a,b) = \mathcal{P}(a)\,\mathcal{P}(b|a)$ to $\mathcal{P}(s_{k+1}=r_j, s_k=r_i, \mathbf{y})$, we have

$$\begin{aligned}
&\mathcal{P}(s_{k+1}, s_k, \mathbf{y}) \\
&= \mathcal{P}(s_{k+1}, s_k, (y[1], \cdots, y[k-1]), y[k], (y[k+1], \cdots, y[N])) \\
&= \underbrace{\mathcal{P}(s_k, y[1], \cdots, y[k-1])}_{\alpha_k(s_k)} \cdot \underbrace{\mathcal{P}(s_{k+1}, y[k]\,|s_k)}_{\gamma_k(s_k,s_{k+1})} \cdot \underbrace{\mathcal{P}(y[k+1], \cdots, y[N]\,|s_{k+1})}_{\beta_{k+1}(s_{k+1})}
\end{aligned} \tag{2.7}$$

The forward metric of state s at stage k, $\alpha_k(s)$, is obtained by the following recursion,

$$\alpha_k(s) = \sum_{\forall s' \in S} \alpha_{k-1}(s') \cdot \gamma_{k-1}(s', s), \quad k = 1, 2, \ldots, K-1 \tag{2.8}$$

The initial condition $\alpha_0(s)$ depends on the state of channel at $t = 0$. If the starting state of the trellis is known, then the forward metric for that state is 1, since the trellis will start from that state with a probability of 1 and 0 for other states. If all states are possible initially, then $\alpha_0(s) = 1, \forall s \in S$

Similarly, $\beta_k(s)$ the backward metric of state s at stage k is obtained by the follow-

ing recursion,

$$\beta_k(s) = \sum_{\forall s' \in S} \beta_{k+1}(s') \cdot \gamma_k(s, s'), \quad k = 1, 2, \ldots, K-1 \tag{2.9}$$

Similarly the initial conditions $\beta_K(s)$ are determined by the knowledge of state of system at the end of current frame.

The transition probability from state $s_k = r_i$ to state $s_{k+1} = r_j$ given by $\gamma_k(r_i, r_j)$ is 0 if the ordered pair $(i, j) \notin B$. The transition probability $\forall (i, j) \in B$ is given by

$$\gamma_k(r_i, r_j) = \begin{cases} \mathcal{P}(x[k] = x_{i.j}) \cdot \mathcal{P}(y[k] | v[k] = v_{i,j}) & \text{if } (i, j) \in \mathcal{B} \\ 0 & \text{if } (i, j) \notin \mathcal{B} \end{cases} \tag{2.10}$$

The term $\mathcal{P}(x[k] = x_{i.j})$ is the prior probability of each transmitted symbol, which is given as input to the equalizer. The other term in the product $\mathcal{P}(y[k] | v[k] = v_{i,j})$ is easily computed by observing from eqn. (2.5) that $y[k] = v[k] + n[k]$ and hence

$$\text{given } v[k] \ , y[k] \sim \mathcal{N}\left(v[k], \sigma^2\right)$$

.

$$\mathcal{P}(y[k] | v[k]) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y[k]-v[k])^2}{2\sigma^2}} \tag{2.11}$$

Now, the APP's for k^{th} transmitted symbol, $x[k]$ taking a value from the set $x \in \{0, 1\}$ is given by

$$\begin{aligned} \mathcal{P}(x[k] = x | \mathbf{y}) &= \sum_{\forall (i,j) \in B : x_{i,j} = x} \mathcal{P}(s_{k+1} = r_j, s_k = r_i | \mathbf{y}) \\ &= \sum_{\forall (i,j) \in B : x_{i,j} = x} \mathcal{P}(s_{k+1} = r_j, s_k = r_i, \mathbf{y}) / \mathcal{P}(\mathbf{y}) \text{ (from eqn. (2.6))} \\ &= \sum_{\forall (i,j) \in B : x_{i,j} = x} \alpha_k(r_i) \gamma_k(r_i, r_j) \beta_{k+1}(r_j) \text{ (from eqn. (2.7))} \end{aligned} \tag{2.12}$$

Note that $\mathcal{P}(\mathbf{y})$ is taken care of during normalization. The transmitted symbols $\mathbf{x}$ are mapped to the coded bit vector $\mathbf{c}$ by the relation $\mathbf{c} = (1 - \mathbf{x})/2$.

The LLR's of the code bits which will be passed to the deinterleaver are given by

$$\begin{aligned} \mathrm{L}\left(c\left[k\right]|\mathbf{y}\right) &= \ln \frac{\mathcal{P}(c\left[k\right]=0 \mid \mathbf{y})}{\mathcal{P}(c\left[k\right]=1 \mid \mathbf{y})} \\ &= \frac{\sum\limits_{\forall(i,j)\in B:x_{i,j}=+1} \alpha_k\left(r_i\right)\gamma_k\left(r_i,r_j\right)\beta_{k+1}\left(r_j\right)}{\sum\limits_{\forall(i,j)\in B:x_{i,j}=-1} \alpha_k\left(r_i\right)\gamma_k\left(r_i,r_j\right)\beta_{k+1}\left(r_j\right)} \text{ (from eqn. (2.12))} \end{aligned} \tag{2.13}$$

To implement this algorithm, first the transition probabilities have to computed for each stage using eqns. (2.10), (2.11). Then using eqn. (2.8) the forward metrics for all states, assuming some initial conditions have to be computed. The trellis has to be traced backwards again to compute backward metrics using eqn. (2.9). These metrics have to be substituted into eqn. (2.13) to compute the LLR for each code bit. All these computations on the trellis can be efficiently described by a sequence of matrix operations [15].

2.4.1 Forward/Backward Algorithm using matrix operations

The number of states in the trellis is $|S|$. The number of stages in this trellis are N. Let us define 2-D matrices $\mathbf{P}_k$ for $k = 1, 2, \cdots, N$ of size $|S| \times |S|$, where

$$\left\{\mathbf{P}_k\right\}_{i,j} = \gamma_k\left(r_i, r_j\right), i, j = 1, 2, \cdots, |S| \text{ and } k = 1, 2, \cdots, N$$

Let us define two matrices $\mathbf{A}\left(x\right)$ for $x \in \{+1, -1\}$ of size $|S| \times |S|$ such that

$$\left[\mathbf{A}\left(x\right)\right]_{i,j} = \begin{cases} 1, & \text{if } (i,j) \in \mathcal{B} \text{ such that } x_{i,j} = x \\ 0, & \text{otherwise} \end{cases} \tag{2.14}$$

Let $\mathbf{B}_k\left(x\right)$ for $x \in \{0, 1\}, k = 1, 2, \cdots, N$ denote a matrix of size $|S| \times |S|$. The matrix $\mathbf{B}_k\left(x\right)$, obtained by the Hadamard product of two matrices $\mathbf{P}_k$ and $\mathbf{A}\left(x\right)$, is given by

$$\mathbf{B}_k\left(x\right) = \mathbf{P}_k \circ \mathbf{A}\left(x\right) \tag{2.15}$$

Let us also define 2 column vectors $\mathbf{f}_k$ and $\mathbf{b}_k$ of length $|S|$ for $k = 1, 2, \ldots, N$. They correspond to forward and backward metrics. Without loss of generality let us initialize

$\mathbf{f}_0 = \mathbf{1}$ and $\mathbf{b}_N = \mathbf{1}$ (this implies trellis could in any one of the $|S|$ possible states both in the beginning and at the end).

The MAP symbol detection algorithm is summarized in Table 2.1 [15].

Forward/Backward Algorithm for Equalizer
Input Matrices $\mathbf{P}_k$ and $\mathbf{B}_k(x)$ for $k = 1, 2, \cdots, N$, column vectors $\mathbf{f}_k$ and $\mathbf{b}_k$ with some initial conditions ($\mathbf{f}_0$ and $\mathbf{b}_N$)
Recursively Compute $\mathbf{f}_k = \mathbf{f}'_{k-1}\mathbf{P}_{k-1}, k = 1, 2, \cdots, N$ $\mathbf{b}_k = \mathbf{P}_k\mathbf{b}_{k+1}, k = 1, 2, \cdots, N$
Output For $k = 1, 2, \cdots, N$ the LLR of code bits is given by $\mathrm{L}(c_k\|\mathbf{y}) = ln\frac{\mathbf{f}'_k \mathbf{B}_k(+1)\mathbf{b}_{k+1}}{\mathbf{f}'_k \mathbf{B}_k(-1)\mathbf{b}_{k+1}}$

Table 2.1: Forward/Backward Algorithm for Equalization

2.5 MAP Convolutional Decoder

The MAP equalizer passes the LLR's computed using eqn. (2.13) to deinterleaver, which deinterleaves and outputs the LLR's of coded bits $\mathrm{L}(b[k]|\mathbf{y})$. These LLR's are then converted to prior bit probabilities using

$$\begin{aligned} \mathcal{P}(b[k] = 0|\mathbf{y}) &= \frac{1}{1 + \exp\{-\mathrm{L}(b[k]|\mathbf{y})\}} \\ \mathcal{P}(b[k] = 1|\mathbf{y}) &= 1 - \mathcal{P}(b[k] = 0|\mathbf{y}) \end{aligned} \tag{2.16}$$

The goal of decoder is to decode the convolutional code with input being

$$\mathbf{P} = (\mathcal{P}(b[1] = 0|\mathbf{y}), \mathcal{P}(b[2] = 0|\mathbf{y}), \cdots, \mathcal{P}(b[N] = 0|\mathbf{y})) \tag{2.17}$$

We are interested in a MAP decoder that outputs the LLR's of data bits $\mathrm{L}(a_k|\mathbf{P})$, $k = 1, 2, \cdots, K$ and also of the code bits $\mathrm{L}(b_k|\mathbf{P}), k = 1, 2, \cdots, N$. We know that convolutional code can be represented on a tapped delay line model similar to that of an ISI channel, and hence an algorithm very similar to the forward/backward algorithm

used for equalization can be used for decoding. Using same notation as in previous section, for the convolutional code given by eqn. (2.4) and the frequency-selective channel represented by the tapped delay model in Fig. 2.4, the differences in both algorithms are

- Each branch in the trellis for convolutional code at stage k is represented by $(i, j, a_{i,j}, b_{1,i,j}, b_{2,i,j})$, where the input $a_k = a_{i,j}$ resulted in a transition from state $s_k = r_i$ to state $s_{k+1} = r_j$ and the two output code bits $b_{2k-1} = b_{1,i,j}$ and $b_{2k} = b_{2,i,j}$.

- The expression eqn. (2.10) has to be modified as

$$\gamma_k(r_i, r_j) = \begin{cases} \mathcal{P}(a_k = a_{i,j})\,\mathcal{P}(b_{2k-1} = b_{1,i,j}|\mathbf{y})\,\mathcal{P}(b_{2k} = b_{2,i,j}|\mathbf{y}), & \text{if } (i,j) \in \mathcal{B} \\ 0, & \text{if } (i,j) \notin \mathcal{B} \end{cases} \tag{2.18}$$

Computing forward and backward metrics is similar for both decoding and equalization. The exact equations are

$$\alpha_k(s) = \sum_{\forall s' \in S} \alpha_{k-1}(s') \cdot \gamma_{k-1}(s', s), \quad k = 1, 2, \ldots, K-1 \tag{2.19}$$

$$\beta_k(s) = \sum_{\forall s' \in S} \beta_{k+1}(s') \cdot \gamma_k(s, s'), \quad k = 1, 2, \ldots, K-1 \tag{2.20}$$

Initial knowledge about the frame structure determine the $\alpha_0(s)$ and $\beta_N(s), \forall s \in \mathcal{S}$. Using very similar expressions, the LLR's for data bits are computed using

$$\begin{aligned} \mathrm{L}(a[k]|\mathbf{P}) &= \ln \frac{\mathcal{P}(a[k] = 0 \mid \mathbf{P})}{\mathcal{P}(a[k] = 1 \mid \mathbf{P})} \\ &= \frac{\sum_{\forall (i,j) \in B: a_{i,j} = 0} \alpha_k(r_i)\,\gamma_k(r_i, r_j)\,\beta_{k+1}(r_j)}{\sum_{\forall (i,j) \in B: a_{i,j} = 1} \alpha_k(r_i)\,\gamma_k(r_i, r_j)\,\beta_{k+1}(r_j)} \end{aligned} \tag{2.21}$$

Similarly LLR's of coded bits, which are fed back to equalizer are computed using

$$\begin{aligned} \mathrm{L}\left(b\left[2k-1\right]|\mathbf{P}\right) &= \frac{\sum\limits_{\forall(i,j)\in B:b_{1,i,j}=0}\alpha_k\left(r_i\right)\gamma_k\left(r_i,r_j\right)\beta_{k+1}\left(r_j\right)}{\sum\limits_{\forall(i,j)\in B:b_{1,i,j}=1}\alpha_k\left(r_i\right)\gamma_k\left(r_i,r_j\right)\beta_{k+1}\left(r_j\right)} \\ \mathrm{L}\left(b\left[2k\right]|\mathbf{P}\right) &= \frac{\sum\limits_{\forall(i,j)\in B:b_{2,i,j}=0}\alpha_k\left(r_i\right)\gamma_k\left(r_i,r_j\right)\beta_{k+1}\left(r_j\right)}{\sum\limits_{\forall(i,j)\in B:b_{2,i,j}=1}\alpha_k\left(r_i\right)\gamma_k\left(r_i,r_j\right)\beta_{k+1}\left(r_j\right)} \end{aligned} \tag{2.22}$$

2.5.1 Matrix Representation

To compute the LLR's of data bits, matrix $\mathbf{A}\left(x\right), x\in\{0,1\}$ is defined as

$$\left[\mathbf{A}\left(x\right)\right]_{i,j}=\begin{cases}1, & \text{if } (i,j)\in\mathcal{B}\,\text{such that}\,a_{i,j}=x\\ 0, & \text{otherwise}\end{cases} \tag{2.23}$$

We use same definitions introduced in Section 2.4.1 for the matrices $\mathbf{P}_k$ and $\mathbf{B}\left(x\right)_k, x\in\{0,1\}$, $k=1,2,\cdots,K$ and the two column vectors $\mathbf{f}_k$ and $\mathbf{b}_k, k=1,2,\cdots,K$. The algorithm is summarized in the Table 2.2 [15].

Forward/Backward Algorithm for Decoder
Input Matrices $\mathbf{P}_k$ and $\mathbf{B}\left(x\right)_k$ for $k=1,2,\cdots,K$, column vectors $\mathbf{f}_k$ and $\mathbf{b}_k$ with some initial conditions ($\mathbf{f}_0$ and $\mathbf{b}_K$)
Recursively Compute $\mathbf{f}_k=\mathbf{f}'_{k-1}\mathbf{P}_{k-1}, k=1,2,\cdots,K$ $\mathbf{b}_k=\mathbf{P}_k\mathbf{b}_{k+1}, k=1,2,\cdots,K$
Output For $k=1,2,\cdots,K$ the LLR of data bits is given by $\mathrm{L}\left(a_k\right)=ln\frac{\mathbf{f}'_k\,\mathbf{B}_k(0)\,\mathbf{b}_{k+1}}{\mathbf{f}'_k\,\mathbf{B}_k(1)\,\mathbf{b}_{k+1}}$

Table 2.2: Forward/Backward Algorithm to compute LLR's of data bits

To compute LLR's of odd indexed codebits, define the matrix $\mathbf{A}(x), x \in \{0,1\}$

$$[\mathbf{A}(x)]_{i,j} = \begin{cases} 1, & \text{if } (i,j) \in \mathcal{B} \text{ such that } b_{1,i,j} = x \\ 0, & \text{otherwise} \end{cases} \tag{2.24}$$

and to find the LLR's of even indexed code bits, define the matrix $\mathbf{A}(x), x \in \{0,1\}$

$$[\mathbf{A}(x)]_{i,j} = \begin{cases} 1, & \text{if } (i,j) \in \mathcal{B} \text{ such that } b_{2,i,j} = x \\ 0, & \text{otherwise} \end{cases} \tag{2.25}$$

Follow the procedure as outlined in Table 2.2, to compute LLR's of code bits.

2.6 Simulation Results

We simulated a communication system as outlined in Section 2.3 to study the performance of turbo-equalization. The input to the rate $R = 1/2$ convolutional encoder is a frame consisting 1000 binary data bits. A block interleaver with 16 rows is used to interleave the coded bits. BPSK modulation is used and the channel is a 3-tap channel $\mathbf{h} = [0.407\, 0.815\, 0.407]$. The performance of this system is presented below.

Fig. 2.6 demonstrates the gain in performance by passing soft information between equalizer and channel decoder over passing hard information. We used a Viterbi algorithm based hard-output equalizer followed by a convolutional decoder that accepts hard inputs to obtain the red (top) curve. The green (bottom) curve is obtained by using a MAP symbol detector as described in Section 2.4 followed by a MAP convolutional decoder (Section 2.5). We can observe a gain of about 2 dB from soft message passing.

Fig. 2.7 demonstrates the gains from turbo-equalization. We can observe a gain of about 1.75 dB between 0^{th} and 1^{st} iterations, about 0.50 dB between 1^{st} and 2^{nd} iterations. The performance saturate after 6 iterations, resulting in a total gain of about 2.75 dB compared to just soft message passing. Compared to hard message passing between equalizer and decoder, receivers employing turbo equalization provide a gain of about 4.75 dB.

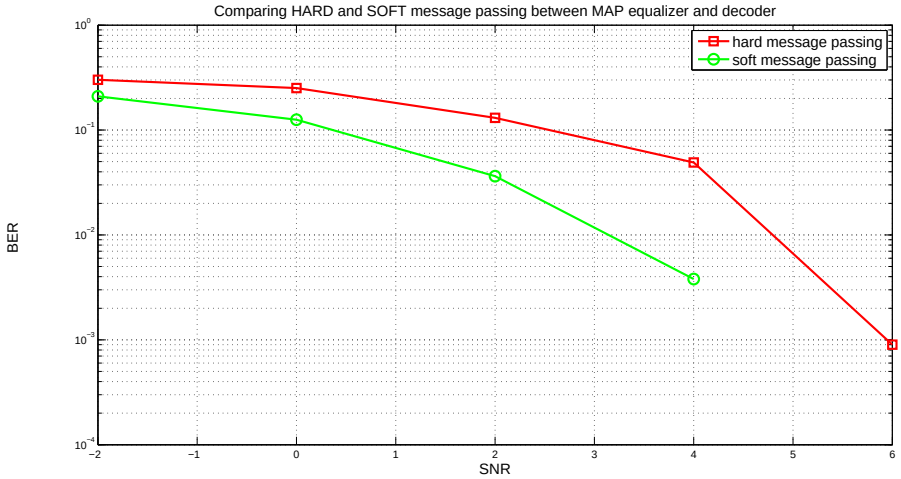

Figure 2.6: Comparing hard and soft message passing receivers for coded BPSK in ISI channels

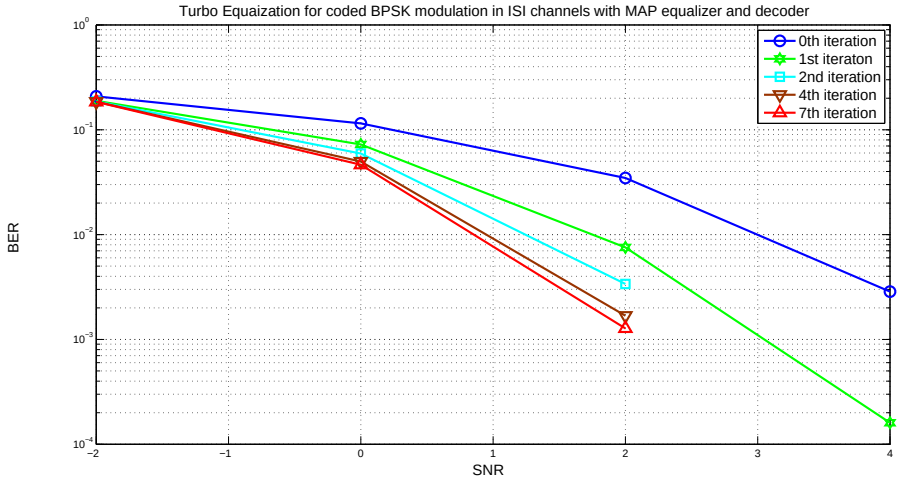

Figure 2.7: Turbo Equalizer performance for coded BPSK in ISI channels

CHAPTER 3

Equalization for Higher Order Modulation

In Section 2.4 we discussed trellis-based MAP equalization algorithm for binary modulation schemes. The same algorithm could be theoretically extended for higher order modulation (HOM) schemes. However the complexity associated with these algorithms is prohibitively high for HOM. Consider a HOM whose signal set has M points. Even for a $(L+1)$-tap frequency-selective channel, the total number of possible states is M^L. This number is extremely large even for typical systems like EDGE in TU channels (in this case it is 8^2) and for HT channels (it is 8^5). Hence implementing a MAP/ML based equalization algorithms is difficult in these cases. There is a need for low complexity algorithms that achieve near optimal performance. These low complexity algorithms are further classified as hard and soft equalization algorithms.

3.1 Hard Output Equalization Algorithm for HOM

The initial focus of work on reducing the complexity of MLSE was to shorten the channel impulse response. Various techniques like use of linear equalizer [25], decision feedback equalizer (DFE)[18] were proposed to truncate the channel response. Ref. [7] introduced what is today known as decision feedback sequence estimator (DFSE). All these techniques don't offer any way to mitigate the higher complexity due to constellation size. Refs. [11] and [6] first proved that performance of MLSE could be achieved by following a lower number of more likely paths. In contrast with various other ad hoc methods in literature, [8] proposed reduced state sequence estimator (RSSE), the first structured way of reducing the complexity, but still achieving near ML performance. This is also applicable to both channels with large memory and higher order modulation schemes. RSSE is based on constructing a trellis with reduced number of states. This trellis is obtained by applying Ungerboeck set partitioning principles [28] on the actual trellis for the system. In the following subsection, we describe the RSSE in detail.

3.1.1 Reduced State Sequence Estimation

Consider the same system model as given by eqn. (2.5). The M-ary modulated symbols (constellation χ) are transmitted over a frequency-selective channel with $(L+1)$ taps whose impulse response, $\mathbf{h} = [h[0], h[1], \cdots, h[L]]$. The received symbols are given by

$$y[k] = \sum_{l=0}^{l=L} h[l]\, x[k-l] + n[k],\ k = 1, 2, \cdots N. \tag{3.1}$$

where $n[k]$ is additive white Gaussian noise with zero mean and variance σ^2.

In the full trellis representation of this system, each state of trellis is given by $s_k = [x[k-1] \quad x[k-2] \cdots x[k-L]]$. Since each element in this L-length vector can take M possible values, we have M^L possible states in the ML trellis. Also from each state s_k, M transitions are possible each corresponding to each input symbol. Thus M branches leave from each state in the ML trellis.

In RSSE, for each element x_{k-l}, we partition the signal set, χ into J_l two dimensional subsets, where $J_l \in [1, \mathrm{M}]$. This two dimensional set partitioning for the l^{th} memory tap is denoted by $\Omega(l)$. The element x_j under partition $\Omega(l)$ is an element of the subset with index denoted by $a_j(l)$, which takes a value between 0 and $J_l - 1$. The set partitioning is constrained by:

- The numbers J_l are non increasing, i.e., $J_1 \geq J_2 \geq \cdots \geq J_L$.
- The two dimensional partition corresponding to l^{th} tap is obtained by a further partition of the subsets corresponding the two dimensional partition for $(l+1)^{th}$ tap, for each l between 1 and $L-1$.

According to this notation, the state s_k in the full MLSE trellis corresponds to the subset state given by

$$t_k = [a_{k-1}(1) \quad a_{k-2}(2) \cdots a_{k-L}(L)]$$

The above two conditions imply that we can uniquely determine the next subset state, provided we know the present subset state and the subset to which input symbol belongs. Therefore the subset states t_l determine a proper trellis, referred to as 'subset

state trellis'. The total number of states in this subset trellis is given by $\prod_{k=1}^{L} J_k$.

Note that, unlike in a full trellis where the input can be any of the M possible values, in the subset trellis input for next stage will definitely be from one of J_1 subset states. Thus only J_1 distinct next states are possible from any state in a subset state trellis. Moreover, when $J_1 < \mathrm{M}$, two different inputs belonging to same subset will be represented by same transition in the subset trellis, if they start in a common subset state. Hence in this case, the trellis will have **parallel transitions**, that start from a common state and end in a common state for all the input symbols in the same subset.

Note that each subset state contains a union of some ML states. Hence certain paths merge earlier in subset state trellis when compared to a ML trellis. So the set partitioning should be such that we can distinguish between these early merging paths easily. It is reported that maximizing the intra subset Euclidean distance when partitioning provides the best performance [8]. The Fig. 3.1 shows the Ungerboeck partitioning [28] for 8-PSK modulation.

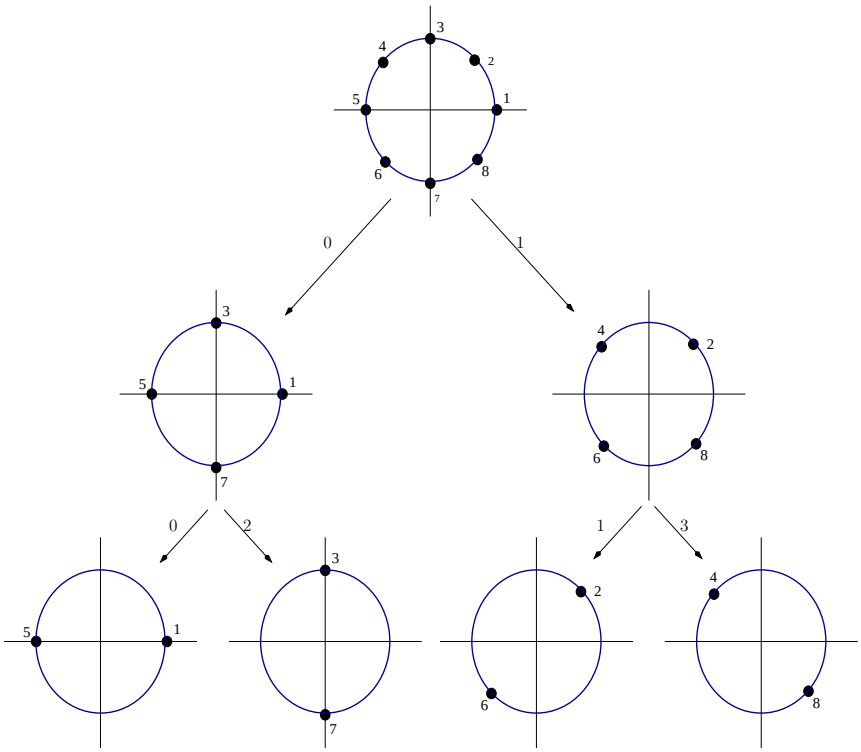

Figure 3.1: Ungerboeck partitioning for 8-PSK modulation

The branch metric for transition from state s_k with an input $x[k] \in \chi$ is computed

by

$$BM_k\left(s_k, x_k\right) = |y[k] - h[0]x[k] - \sum_{l=1}^{l=L} h[l]\hat{x}[k-l]|^2, \tag{3.2}$$

where $\hat{x}[n-l]$, $l = 1, 2, \cdots, L$ is the path history associated with the subset state s_k. We have to store the path histories associated with each subset state, since there isn't a one-one correspondence between the subset states and the path history like in case of the ML trellis. Further when subset trellis has parallel transitions, we can exploit the symmetry of set partitioning to reduce the computational effort. Due to symmetry, we can use slicing operations to determine the winner from among intra-subset symbols. In other words, delay free decisions are made to select the winner among all the elements of a subset. This also reduces the number of branch metric computations to NJ_1 from $N\mathrm{M}$. The state metric for a state is obtained by adding the branch metric to the past states metric and selecting the path with minimum metric. This process is continued till the end of subset trellis. The state with minimum metric in the final stage is selected and we trace back to determine the transmitted symbols. Fig. 3.2 shows the subset trellis of 8-PSK modulation in a 3-tap channel. $\mathbf{J} = [2 \quad 2]$ set partitioning is employed. Each thick line in the Fig. 3.2 represents 4 parallel transitions

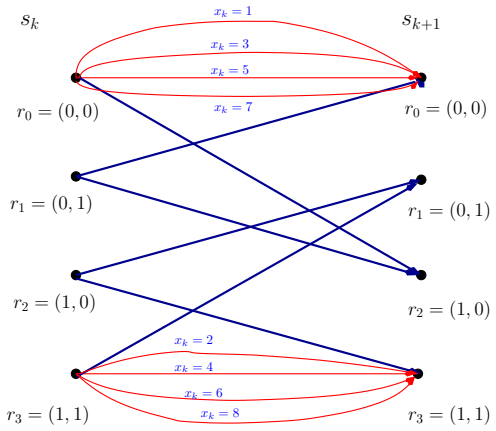

Figure 3.2: Subset trellis for [2 2] partition

Note that when $J_l = 1, \forall l$, RSSE becomes DFE and when $J_l = \mathrm{M}, \forall l$, RSSE becomes MLSE. Thus, choosing **J** carefully, we can get the performance between DFE and MLSE. For systems with longer impulse response of length $(L+1)$, using $J_l = 1$, $l \in [L'+1, L]$ and selecting the first L' taps partition carefully reduces the

complexity. Choosing $J_l = \mathrm{M},\ l \in [1, L']$ and $J_l = 1,\ l \in [L'+1, L]$ results in decision feedback sequence estimator (DFSE).

3.1.2 Simulation Results

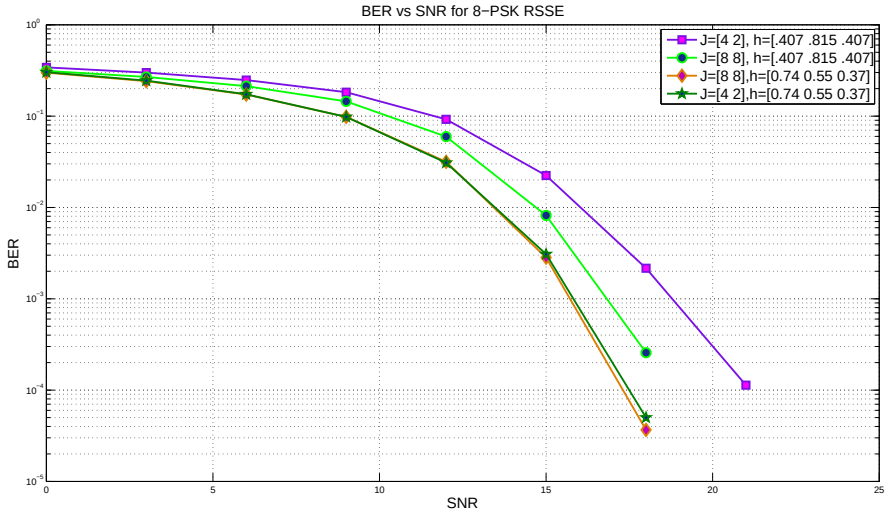

Figure 3.3: Performance of RSSE in minimum-phase and non-minimum phase channels

Fig. 3.3 shows the performance of RSSE for 8-PSK modulation in minimum-phase and non-minimum phase channels. We can observe that 8-state RSSE performs as well as a 64-state MLSE for minimum-phase channels and 8-state RSSE loses performance compared to 64-state for non-minimum phase channels. Also, we observe that MLSE/RSSE perform well in minimum-phase rather than in non-minimum phase channels. Hence, for non-minimum phase channels, we need to increase the number of states of RSSE to get same performance as MLSE. Also, in practice when implementing RSSE for fading channels, a prefilter is implemented to shorten and also shape the effective channel response as minimum-phase [1].

3.2 Soft Output Equalization Algorithms for HOM

In the following Subsection, we describe a soft equalization algorithm for HOM. It is developed by applying the ideas of set partitioning and subset state trellis to the

forward/backward equalization algorithm [2],[15].

3.2.1 Reduced State Maximum *A Posteriori* Algorithm

Consider the same system model as in Subsection 3.1.1. Using same notation as in previous section, the received symbols are given by

$$y[k] = \sum_{l=0}^{l=L} h[l]\, x[k-l] + n[k], k = 1, 2, \cdots N. \tag{3.3}$$

where $n[k]$ is additive white Gaussian noise with zero mean and variance σ^2. A state in the full trellis for this system is denoted by $s_k = [x[k-1]\, x[k-2] \cdots x[k-L]]$, the L most recent symbols. To reduce the number of states, we define a two dimensional set partitioning, $\Omega(l)$, for every $x[n-l]$, such that signal set χ for the l^{th} tap is divided into J_l subsets. We index each subset by $0, 1 \cdots J_l - 1$. The element x_j is an element of the subset state $a_j(l)$ under $\Omega(l)$ partition.
This partitions are constrained by [8]

- The numbers J_k are non-increasing, i.e, $J_1 \geq J_2 \geq \cdots \geq J_L$.
- $\Omega(k)$ is a further partition of the subsets of $\Omega(k+1)$ for each k between 1 and $L-1$.

The trellis formed by the subset states is referred to as the 'subset state trellis'. It will have $\prod_{k=1}^{k=L} J[k]$ states. There will be $J[1]$ transitions from each state in this trellis. Each transition in this subset trellis has as many parallel transitions as the number of elements in the corresponding subset.

The set partitioning is done according to Ungerboeck set partitioning principles described in Subsection 3.1.1. Fig. 3.1 describes the set partitioning for 8-PSK modulation.

Without loss of generality, let us consider a partition scheme with $J[1] = 2,\ J[2] = 2,\ J[l] = 1,$ for $l \in [3, L]$ for an 8-PSK modulated system. The 8-PSK symbols in subset states with indices 0 and 1 are $\chi(0) = \{1, j, -1, -j\}$ and $\chi(1) = \{1+j, -1+$

$j, -1-j, 1-j\}$ respectively for a two subset Ungerboeck set partitioning. The subset trellis for this partition scheme has only 4 states denoted by $\mathrm{S} = \{r_0, r_1, r_2, r_3\}$. The state is also denoted by an ordered pair of 2 binary bits, $(a\,[k-1]\,, a\,[k-2])$ at stage k. The subset state trellis for this system is given in Fig. 3.4

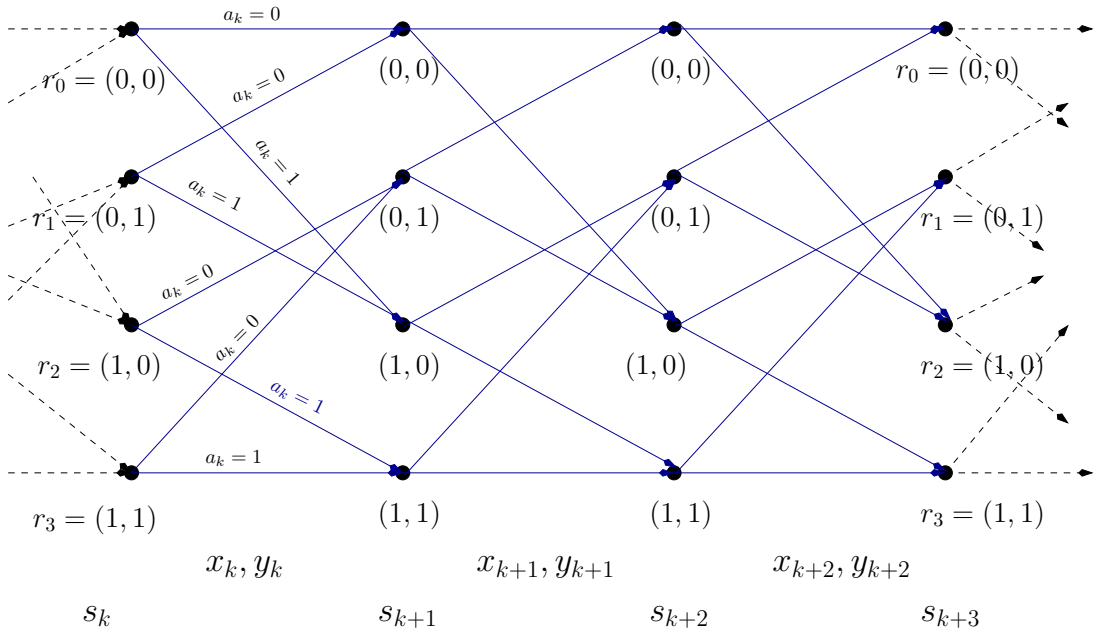

Figure 3.4: Subset trellis with 4 states

The set $\mathrm{B} = \{(0,0), (0,2), (1,0), (1,2), (2,1), (2,3), (3,1), (3,3)\}$ is the set of all possible transitions in this trellis. A branch in this trellis is denoted by an ordered pair $(i, j, a_{i,j})$, such that state $s_{k+1} = r_j$ is reached from state $s_k = r_i$ at time k with an input drawn from the subset state with index $a_{i,j}$.

Let us assume the input random variables $x\,[k]$ are IID, i.e., $\mathrm{P}\,(\mathbf{x}) = \prod_{k=1}^{k=N} \mathrm{P}\,(x\,[k])$. Now let us compute the probability that transmitted input sequence has a branch $(i, j, a_{i,j})$ in the subset state trellis at time k, i.e, $\mathrm{P}\,(s_{k+1} = r_j, s_k = r_i/\mathbf{y})$. This is computed efficiently based on forward/backward algorithm [15].

We know

$$\mathrm{P}\,(s_{k+1} = r_j, s_k = r_i|\mathbf{y}) = \mathrm{P}\,(s_{k+1} = r_j, s_k = r_i, \mathbf{y})\,/\mathrm{P}\,(\mathbf{y})$$

Applying the chain rule, i.e., $\mathrm{P}(a,b) = \mathrm{P}(a)\,\mathrm{P}(b|a)$ to $\mathrm{P}(s_{k+1} = r_j, s_k = r_i, \mathbf{y})$, we have

$$\begin{aligned}
&\mathcal{P}(s_{k+1}, s_k, \mathbf{y}) \\
&= \mathcal{P}(s_{k+1}, s_k, (y[1], \cdots, y[k-1]), y[k], (y[k+1], \cdots, y[N])) \\
&= \underbrace{\mathcal{P}(s_k, y[1], \cdots, y[k-1])}_{\alpha_k(s_k)} \cdot \underbrace{\mathcal{P}(s_{k+1}, y[k]\,|s_k)}_{\gamma_k(s_k, s_{k+1})} \cdot \underbrace{\mathcal{P}(y[k+1], \cdots, y[N]\,|s_{k+1})}_{\beta_{k+1}(s_{k+1})}
\end{aligned} \tag{3.4}$$

$\alpha_k(s)$, the forward metric of state s at stage k is obtained by the following recursion,

$$\alpha_k(s) = \sum_{\forall s' \in S} \alpha_{k-1}(s') \cdot \gamma_{k-1}(s', s), \quad k = 1, 2, \ldots, N-1 \tag{3.5}$$

The initial condition $\alpha_0(s)$ depends on the state of channel at $t = 0$. If the starting state of the channel is known, then the forward metric for that state is 1 and 0 for the other states. If all states are possible initially, then $\alpha_0(s) = 1, \forall s \in S$. Similarly, $\beta_k(s)$ the backward metric of state s at stage k is obtained by the following recursion,

$$\beta_k(s) = \sum_{\forall s' \in S} \beta_{k+1}(s') \cdot \gamma_k(s, s'), \quad k = 1, 2, \ldots, N-1 \tag{3.6}$$

Similarly the initial conditions $\beta_N(s)$ are determined by the knowledge of state of system at the end of current frame.

The transition probability from state $s_k = r_i$ to state $s_{k+1} = r_j$ given by $\gamma_k(r_i, r_j)$ is 0 if the ordered pair $(i,j) \notin B$. For all $(i,j) \in B$, the transition occurs when input takes any of the 4 symbols in $\chi(a_{i,j})$. The transition probability is then computed by conditioning it on input taking a particular value. Since these events are independent, $\gamma_k(r_i, r_j)$ is obtained by summing them. The following equation determines the transition probability.

$$\gamma_k(s_k = r_i, s_{k+1} = r_j) = \begin{cases} \sum\limits_{\forall x \in \chi(a_{i,j})} \mathrm{P}(x[k] = x) \cdot \mathrm{P}(y[k]|s_k = r_i, x[k] = x) & \text{if } (i,j) \in \mathrm{B} \\ 0 & \text{if } (i,j) \notin \mathrm{B} \end{cases} \tag{3.7}$$

The first term in the product on RHS corresponds to the prior input probability, which is known *a priori*. The second term in the product has a Gaussian probability distribution function, provided past history of transmitted symbols up to state $s_k = r_i$ is known.

Determining the Past History for each State

Let us assume we know the path history of every state in stage $(k-1)$. We are interested in the survivor symbol for the transition to state $s_k = r_i$. Let a denote the subset index which results in this transition. The survivor symbol $\hat{x}[k-1]$ to reach state $s_k = r_i$ is given by

$$\hat{x}[k-1] = \arg \max_{x \in \chi(a)} \alpha_{k-1}(r_l) \, \mathrm{P}\left(x[k-1] = x\right) \mathrm{P}\left(y[k-1]/s_{k-1} = r_l, x[k-1] = x\right), \quad \forall l : (l,i) \in \mathcal{B} \tag{3.8}$$

The 3^{rd} term in the product in above expression has a Gaussian pdf and is computed using

$$\mathrm{P}\left(y[k-1] | s_{k-1} = r_l, x[k-1] = x\right) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{|y[k-1] - h[0]x - \sum_{l=1}^{l=L} h[l]\hat{x}[k-l-1]|^2}{2\sigma^2}}, \tag{3.9}$$

where $\hat{x}[k-l-1]$, $l = 1, 2, \cdots, L$ is the path history associated with state $s_{k-1} = r_l$.

Substituting $(k-1)$ in eqn. (3.9) with k, we can compute the 2^{nd} term of product in eqn. (3.7). Using this probability and the *a priori* symbol probability, transition probability is computed from eqn. (3.7).

Using eqns. (3.7), (3.9) and the recursive relations in eqns. (3.5) and (3.6) along with known initial conditions $\alpha_0(s), \forall s$ and $\beta_N(s), \forall s$, we can compute the forward and backward metrics for every state and transition probability for every branch in this subset trellis.

The APP's for each input symbol are computed using

$$\begin{aligned}&\mathrm{P}\left(x\left[k\right]=x/\mathbf{y}\right)\\&=\sum_{\forall(i,j)\in\mathrm{B}:x\in\chi(a_{i,j})}\alpha_k\left(r_i\right)\cdot\mathrm{P}\left(x\left[k\right]=x\right)\cdot\mathrm{P}\left(y\left[k\right]/s_k=r_i,x\left[k\right]=x\right)\cdot\beta_{k+1}\left(r_j\right)\end{aligned}\tag{3.10}$$

Each symbol in a M-ary constellation maps to $\log_2\mathrm{M}$ bits.The symbol level APP's computed using eqn. (3.10) are converted to bit level APP's using

$$\mathrm{P}\left(b_{n-k}\left(j\right)=0|\mathbf{y}\right)=\sum_{\begin{pmatrix}\forall x\in\chi:j^{th}\text{ position in binary}\\\text{mapping of symbol is }x\text{ is }0\end{pmatrix}}\mathrm{P}\left(x[n-k]=x|\mathbf{y}\right),\tag{3.11}$$

where $j=1,2,\cdots,\log_2\mathrm{M}$.

This algorithm could also be represented using matrix notation as done in Subsection 2.4.1. We have to add an extra dimension of input symbol index to $\mathbf{P}$ matrices and modify the equations provided in that subsection. The final equations turn out to be very similar to those, except for changes due to increase in one dimension of matrix $\mathbf{P}$.

3.2.2 Simulation Results

The performance of RSMAP equalization algorithm is evaluated for 8-PSK modulated systems in both minimum and non-minimum phase channels. The impulse response of the 3-tap channel used in simulations is $\mathbf{h}=[0.750.560.37]$ for minimum phase and $\mathbf{h}=[0.407\ 0.815\ 0.407]$ for non-minimum phase channels. Rate $\frac{1}{2}$ convolution encoder with constraint length $K=3$ given by eqn. (2.4) in Subsection 2.3 is used as encoder along with a block interleaver. We also compare the performance of RSMAP/MAP followed by a soft-input channel decoder (denoted by soft RSMAP/soft MAP in the figure) with RSSE followed by a Viterbi decoder (denoted by hard RSSE) for the convolution rate $\frac{1}{2}$ code.

Fig. 3.5 shows that 8-state RSMAP performs very close to full MAP (64 states) at

much lower complexity. RSMAP loses only 0.2 dB compared to MAP. Also we observe a gain of about 1.8 dB between hard RSSE and soft RSMAP.

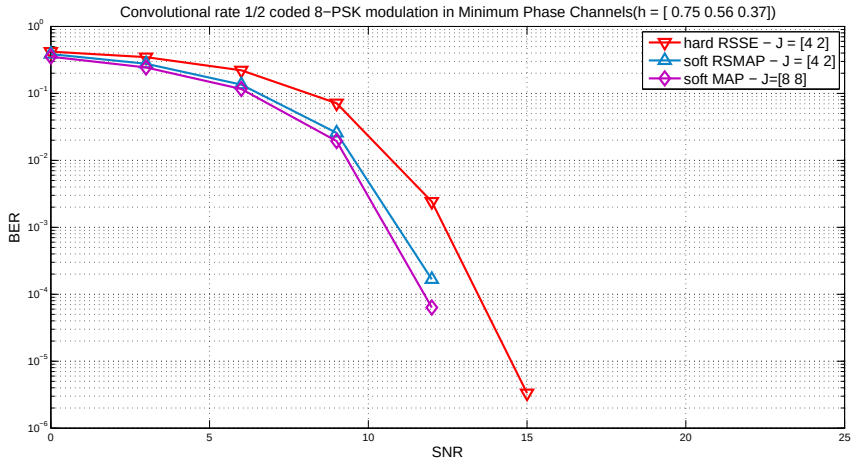

Figure 3.5: 8-state RSMAP vs 64-state MAP equalizer in minimum phase channels

Fig. 3.6 shows that 16-state RSMAP performs very close to full MAP (64-states). RSMAP looses only 0.2 dB compared to MAP. Also we observe a gain of about 1.8 dB between hard RSSE and soft RSMAP.

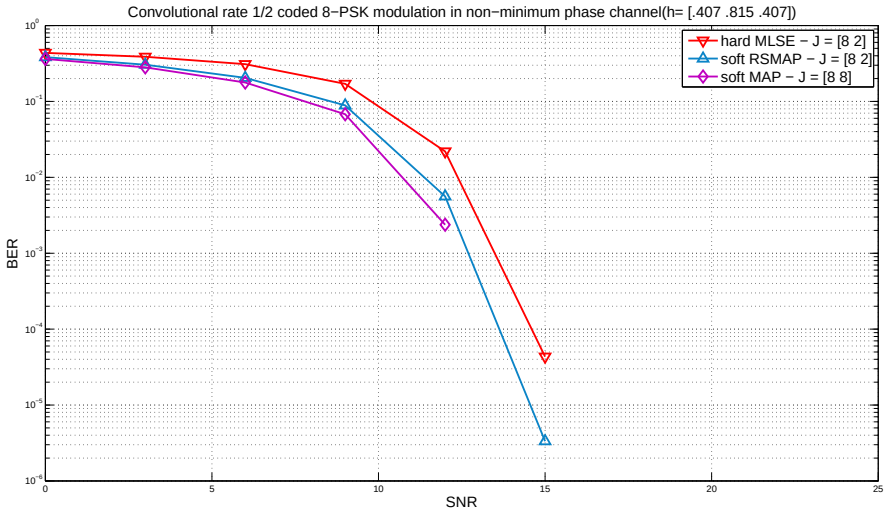

Figure 3.6: 16-state RSMAP vs 64-state MAP equalizer in non-minimum phase channels

This demonstrates that RSMAP is near MAP optimal at significantly lower complexity than MAP algorithm. In the next section we discuss the performance of a turbo equalizer with RSMAP as the inner decoder algorithm.

3.3 Turbo-Equalization for HOM

The general framework of a turbo-equalizer is described in Section 2.2. The low-complexity soft-equalization algorithms described in previous section are now applied in this framework to develop a iterative receiver for HOM in frequency-selective channels. Specifically we investigated the performance of an iterative receiver in frequency-selective channels for 8-PSK modulated system.

We simulated the performance of turbo-equalizer for 8-PSK in the minimum-phase channel $\mathbf{h} = [0.75 \;\; 0.56 \;\; 0.37]$ with convolutional encoder given by eqn. (2.4) and a block interleaver. $[4\ ,\ 2]$ partition scheme was used in RSMAP, since we observed that it is near MAP optimal from Fig. 3.5.

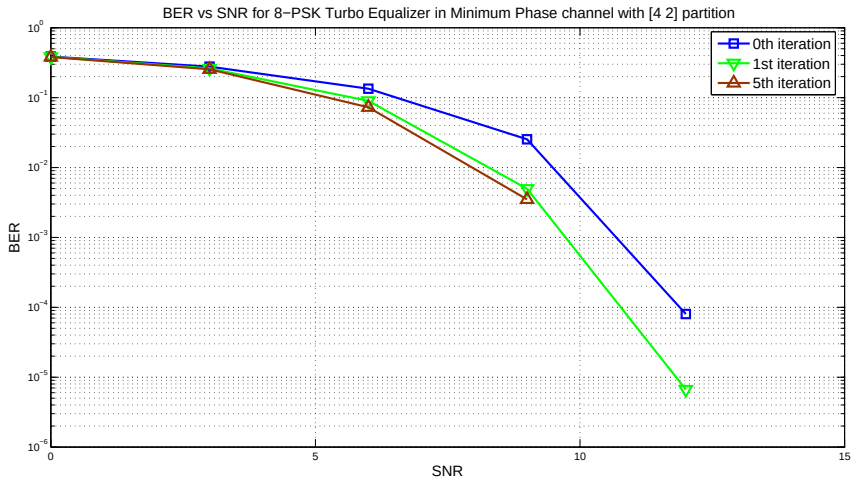

Figure 3.7: 8-PSK Turbo Equalizer in 3-tap ISI channel with 8 state equalizer

The gains from turbo-equalizer saturated by the end of 5^{th} iteration. We can see from Fig. 3.7 that there is a gain of about 2 dB by the end of 5^{th} iteration compared to soft message passing from RSMAP to decoder. Compared to hard RSSE, a gain of about 4 dB is achieved by the turbo-equalizer.

CHAPTER 4

SAIC with Iterative Receiver

A lot of work has been done in literature to suppress co-channel interference (CCI). [23] and [24] focus on co-channel suppression for GSM systems with a single antenna. For GSM users suffering interference from a co-channel GSM user, Single Antenna Interference Cancellation (SAIC) algorithms based on the projection method or the filtering-based techniques give good performance. However when we move to higher order modulation schemes used in present day wireless standards, the receiver architecture has to be modified to suppress CCI. Leveraging the gains from an iterative receiver, using widely linear filtering based suppression techniques is a good architecture to suppress CCI for HOM users. In this chapter, we describe the outline of receiver envisaged to suppress CCI and simulate its performance.

4.1 Receiver Architecture to Suppress CCI

At the receiver, the channel estimates of desired user are computed. A block diagram of the receiver excluding the channel estimator is shown in Fig. 4.1. The received signal is then filtered using widely linear MMSE prefilter (refer to Appendix B). The objective of this filter is to

- Suppress the co-channel interferer.
- Shape the effective channel response as minimum phase.

Widely-linear (WL) filtering converts the white noise at the input of filter to colored noise at the output. This knowledge has to be utilized in the equalizer to get better performance. Also the metric has to be computed using widely-linear principles, since we are employing WL prefilter. More details on WL equalizer are provided in next section. The equalizer is followed by demodulator, deinterleaver and decoder. The

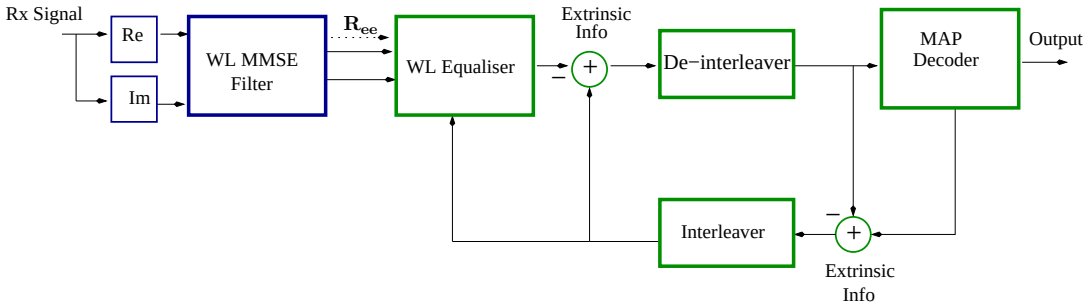

Figure 4.1: Iterative SAIC receiver

decoder then passes the information to equalizer completing the feedback loop. This loop is stopped only after completing a certain number of predetermined iterations or after achieving a desired error rate.

4.2 Widely Linear (WL) Equalizer

The WL MMSE DFE prefilter transforms white noise to a colored impairment at its output. Hence the metric computation has to be changed in the equalizer. This computation should utilize the error covariance matrix $\mathbf{R_{ee}}$ provided by the prefilter. The received complex signal is converted to a 2×1 vector whose 1^{st} element is the real part and 2^{nd} element is the imaginary part of the complex received signal. On passing this 2×1 received signal vector $\mathbf{y}_n$ through the 2-D WL MMSE DFE filter($\mathbf{F}$), the resultant output can be written as

$$\mathbf{V}_n = \sum_{k=0}^{N_b} \mathbf{B}_k \, \mathbf{a}_{n-k} + \mathbf{e}_n. \tag{4.1}$$

where $\mathbf{V}_n$ denotes a 2×1 effective received symbol vector for the turbo equalizer, $\mathbf{e}_n$ denotes 2×1 impairment vector and $\mathbf{B}_n$ is the resultant matrix channel obtained by 2-D filtering. For a hard-decision equalizer the branch metric is computed by

$$M = \mathbf{e}^H \, \mathbf{R}_{ee}^{-1} \, \mathbf{e}, \tag{4.2}$$

$$\mathbf{e} = \mathbf{V}_n - \hat{\mathbf{V}}_n \tag{4.3}$$

$$\hat{\mathbf{V}}_n = \sum_{k=0}^{N_b} \mathbf{B}_k \, \tilde{\mathbf{a}}_{n-k}, \tag{4.4}$$

where $\tilde{\mathbf{a}}$ are the surviving symbols for each state in the trellis.

For soft-output equalizers, we compute $\mathcal{P}\left(y\left[k\right]|\text{a present state}\right)$ using

$$\mathcal{P}\left(y\left[k\right]|\text{a present state}\right) = \frac{1}{\sqrt{2\pi\det(\mathbf{R_{ee}})}} e^{-\frac{|M|^2}{2}} \tag{4.5}$$

4.3 VAMOS in GMSK

In this section, the performance of a SAIC receiver for a VAMOS user capable of suppressing a co-channel GMSK interferer is shown.

4.3.1 System Model

VAMOS uses Adaptive QPSK (AQPSK or α-QPSK) modulation in which each orthogonal sub channel is assigned to a different user within the cell. Sub Channel Power Imbalance Ratio (SCPIR, χ) is defined as the ratio of power of 2^{nd} (quadrature) user ($\sin^2(\alpha)$) to the power of 1^{st} (in-phase) user ($\cos^2(\alpha)$).

$$\text{SCPIR} = \chi = 20\log_{10}(\tan\alpha)\text{dB}.$$

Here we consider the discrete time model of VAMOS receiver which does α-QPSK detection in the presence of a dominant GMSK interferer.

In the considered scenario of VAMOS downlink transmission, the baseband received signal can be written as

$$\begin{aligned} y\left[n\right] = \cos(\alpha) \sum_{k=0}^{L} h(k)\, a_1(n-k) + j\ \sin(\alpha) \sum_{k=0}^{L} h(k) \\ a_2(n-k) + \sum_{k=0}^{L} g(k)\, b_1(n-k) + w(n) \end{aligned} \tag{4.6}$$

Here, the discrete-time channel impulse response $h(k)$ of order L comprises the effects of GMSK modulation, the mobile channel from base station to desired user, receiver filtering and GMSK de-rotation at the receiver. We assume that the channel is constant within a transmission burst (time slot) and varies from burst to burst (block fading).

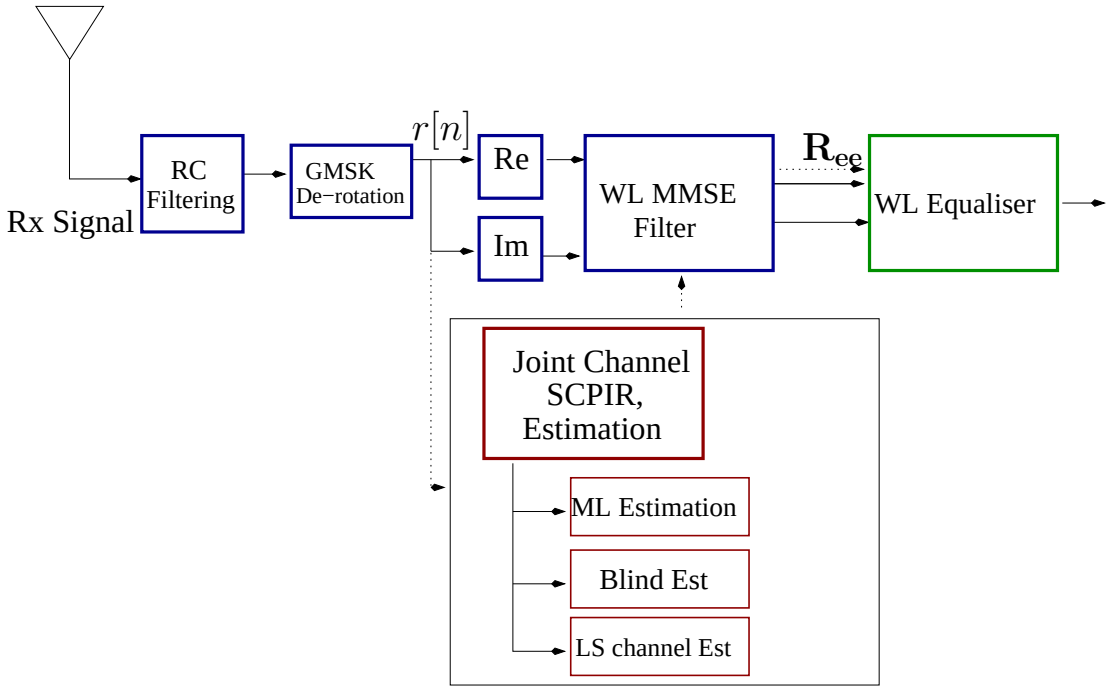

Figure 4.2: VAMOS receiver structure

$a_1(k)$ and $a_2(k)$ denote the GMSK transmit symbols of both users. $b_1(k)$ denotes the GMSK transmit symbols of co-channel interferer and $g(k)$ denotes the channel from the base station of neighboring cell, $w(n)$ denotes the white Gaussian noise of variance σ_n^2. The factor α denotes the constellation points and the power difference between the two orthogonal users. We don't assume the knowledge of α in the receiver. We assume the knowledge of training sequence of both orthogonal users as well as co-channel interferer for the estimation of channel vector. Fig. 4.2 shows the proposed VAMOS receiver structure. Appendix A and Appendix B contain the channel estimation and SAIC algorithms developed for this system.

4.3.2 Simulation Results

In this subsection, we evaluate the performance of VAMOS in GMSK interference with algorithms discussed above. In all the simulations, the radio channel for the desired signal and interferer is assumed to be a Typical Urban channel with a vehicular speed of 3 km/h, i.e., a TU3 channel. The frequency of operation is 900 MHz, and independent fading for each burst is generated to simulate the effect of ideal frequency hopping. The entire receiver uses 2X oversampling of the received signal. All the equations derived

above can be applied even when the receiver does oversampling. We augmented both polyphase channel outputs into a single vector and applied all the equations. We used a single dominant interferer model with the interferer passing through an independent TU3 fading channel through out the analysis. Perfect synchronization between desired signal and interferer is assumed. Received signal is passed through a raised cosine filter of roll off factor 0.5. To simulate interference limited system, we constrained SNR to be 30 dB. The performance has been presented in terms of uncoded BER (raw BER) as a function of average signal to interference power ratio.

In Fig. 4.3, the raw BER of user-1 versus C/I is presented. Here SCPIR is estimated using ML estimation technique described in Appendix A. Iterative solving of coupled equations is started from $\hat{\alpha} = \frac{\pi}{3}$ and 10 iterations are carried out before converging to final estimates of $\hat{\alpha}$.

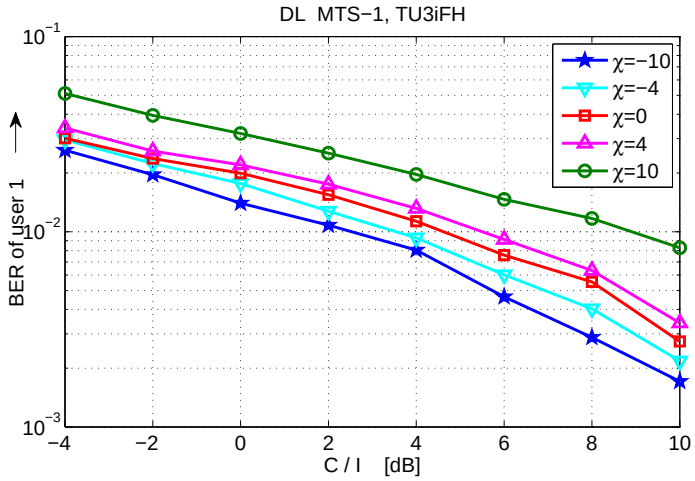

Figure 4.3: Raw BER of user-1 versus C/I ($C_1 + C_2 = C$) for different ML estimated SCPIR values, WL RSSE with [2 2] set partitioning.

In Fig. 4.4, the raw BER of user-1 versus C/I is presented. Here SCPIR is estimated using blind estimation technique described in Appendix A. One dimensional search is implemented using 'golden section search' technique.

As χ increases, power of user-1 decreases and hence the performance decreases. We can see that raw BER even in the worst case meets the requirement suggested in [22]. We can see from figs. 4.3 and 4.4 that the loss in performance is negligible due to blind estimation. Hence blind estimation of SCPIR can be implemented in the receiver.

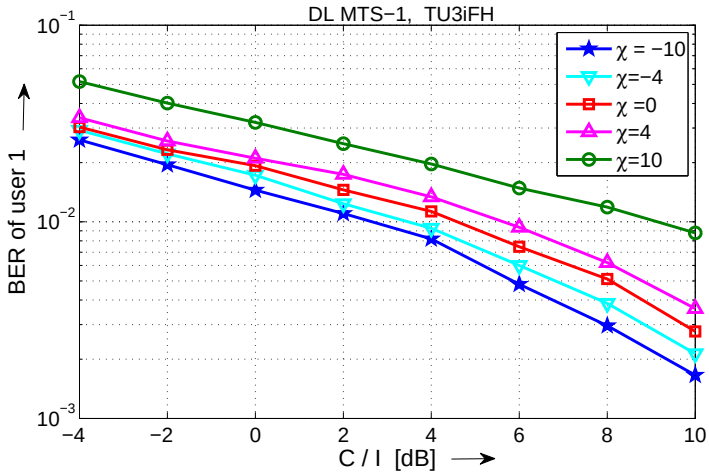

Figure 4.4: Raw BER of user-1 versus C/I ($C_1 + C_2 = C$) for different blind estimated SCPIR values, WL RSSE with [2 2] set partitioning.

CHAPTER 5

Low Complexity Detection of MIMO Systems

5.1 Introduction

Spatial multiplexing over a multiple-input multiple-output (MIMO) channel is a technique that can be used to meet the increasing demand for higher wireless data rates when constrained with limited spectrum. We are interested in V-BLAST [31] transmission scheme, in which data is demultiplexed into n streams to be transmitted from each of the n transmit antennas across a flat-fading channel. The receiver with m receive antennas receives n streams, each experiencing interference from other streams.

Various detection algorithms could be implemented at the receiver. The optimal decoding scheme, from the point of minimizing the symbol error for uncoded transmission, is the maximum likelihood (ML) detection. ML detector which provides a diversity of order m at high SNR has a complexity of the order of M^n, where M is the size of modulation alphabet used. Several low-complexity detection algorithms were proposed in literature. The original V-BLAST detector performs interference cancellation, nulling and optimal ordering. There is also minimum mean square error V-BLAST [13]. The V-BLAST detector can be implemented by QR decomposition followed by a successive interference cancellation receiver. These algorithms are equivalent to a decision feedback equalizer (DFE). The main drawback of these detection algorithms, even after optimal ordering, is the low diversity order for the first stage $m - n + 1$ [20], resulting in the error propagation which limits the performance. To overcome this, authors in [5] proposed to use ML decoding for first p stages, followed by a DFE for the later stages.

Sphere decoding (SD) [30] is known to provide optimal performance for MIMO SM systems. However the problem with SD is its variable complexity depending on channel conditions. Recently, fixed complexity sphere decoder was proposed [4]. These

are hard-detection algorithms. The authors in [17] propose soft-detection algorithms for MIMO systems using two breadth-first algorithms, M algorithm and T algorithm. A fixed complexity sphere decoder which outputs soft information is also developed [3]. The authors in [21] utilize the ideas from reduced state sequence estimation [8] and Ungerboeck set partitioning [28], and propose uniform set partitioning (USP). The performance of this algorithm degrades at high SNR. This approach is referred to as reduced state sequence detector (RSSD). In [14], authors propose the use of non-uniform set partitioning, recognizing that more partitions are required for the initial stages than later stages due to the low diversity order of the initial stages. RSSD based on QR decomposition of channel matrix is a very elegant method that approaches the performance of ML decoding at a much lower complexity [14]. An adaptive M-algorithm in conjunction with set partitioning is used in [16]. In this approach set partitioning for each stage is adapted depending on channel conditions.This algorithm has variable complexity. In this paper, we are interested in soft detection algorithms with deterministic complexity. To the best of our knowledge, this is the first paper to address the computation of symbol APP's using ideas inspired from maximum a posteriori (MAP) equalizer [15] and RSSE [8] for MIMO systems in flat fading channels.

The main contributions from this paper are

- Developed a framework and implemented BCJR algorithm on a reduced state tree, formed by set partitioning and uses ideas from RSSE to output soft information on transmitted bits.
- A forward only soft detection algorithm is developed. We define the soft decision failures that occur with forward only algorithm. We bound the number of soft decision failures and also propose a mechanism to prevent soft decision failure that occurs when backward recursion is not implemented. Reduction in complexity at a negligible loss in performance when max log approximation is applied is also recognized.
- Exact complexity to implement these algorithms is computed and we also analyze the performance of these algorithms.

The outline of the remainder of the paper is as follows. In section II we describe the

system model. In section III the notation used is described. Also RSSD algorithm presented in [14] is briefly described in this section. In section IV the need for non-uniform set partitioning is intuitively explained, formal proof is given in [14]. In section V reduced state maximum a posteriori (RSMAP) detection algorithm is presented. In section VI the approximate versions of RSMAP algorithm are presented. The complexity analysis and the simulation results are presented in sections VII and VIII respectively. Finally conclusions are given in section IX.

5.2 System Model

Let us consider a typical MIMO communication system with a convolutional encoder and a block interleaver. The interleaved coded bits, modulated using a M-ary modulation scheme (signal set denoted by χ), are transmitted from n transmit antennas. The received signals are given by

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{w} \tag{5.1}$$

where $\mathbf{x}$ represents $n \times 1$ vector of transmitted symbols from a n transmit and $\mathbf{y}$ represents the $m \times 1$ vector of received symbols from a m receive antenna system. The $m \times n$ matrix $\mathbf{H}$ represents the block fading channel between the n transmit and m receive antennas. Every element of this matrix is a circularly symmetric complex-Gaussian random variable with zero mean and unit variance, representing the channel between a transmit-receive antenna pair. We assume the receiver knows the channel matrix $\mathbf{H}$. The $m \times 1$ vector $\mathbf{w}$ represents the additive complex white Gaussian noise vector in which each element has zero mean and variance σ^2. The power from each transmitted antenna is normalized to $\frac{1}{n}$, so that the total signal power at transmitter is unity. The signal to noise ratio (SNR) is defined as $\frac{1}{\sigma^2}$. We are interested in the case when $n \leq m \leq n+1$, since when $m \gg n$ low-complexity schemes such as V-BLAST detection perform very well.

5.3 Hard Detection of MIMO Systems

5.3.1 QR decomposition

The first step is to perform QR decomposition of the channel matrix $\mathbf{H}$, resulting in two matrices $\mathbf{Q}$ and $\mathbf{R}$, where $\mathbf{Q}$ is an $m \times m$ unitary matrix and $\mathbf{R}$ is an $m \times n$ upper triangular matrix. We know that matrix $\mathbf{R}$ is independent of matrix $\mathbf{Q}$ and elements of $\mathbf{R}$ are independent of each other [27]. Also the square of the absolute values of the diagonal entries of $\mathbf{R}$ are Chi-square distributed, with $2\,(m-i+1)$ degrees of freedom for the i^{th} diagonal element and the last diagonal element, r_{nn} has the least degrees of freedom. Hence it sees the worst channel statistics. We now define $\mathbf{z}$ as

$$\mathbf{z} = \mathbf{Q}^{\mathcal{H}}\mathbf{y} = \mathbf{R}\mathbf{x} + \tilde{\mathbf{w}} \tag{5.2}$$

where $\mathbf{z}$ is a $m \times 1$ vector, $\tilde{\mathbf{w}}$ is a $m \times 1$ complex white Gaussian noise vector with zero mean and variance σ^2 and $(.)^{\mathcal{H}}$ denotes the Hermitian (conjugate transpose) operation. Since the bottom $(m-n)$ rows of the upper triangular matrix $\mathbf{R}$ are all zeros, the corresponding rows in $\mathbf{z}$ contain only noise. Hence ignoring these $(m-n)$ rows, we have

$$\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} = \begin{bmatrix} r_{1,1} & r_{1,2} & \dots & r_{1,n} \\ 0 & r_{2,2} & \dots & r_{2,n} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & r_{n,n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} \tilde{w}_1 \\ \tilde{w}_2 \\ \vdots \\ \tilde{w}_n \end{bmatrix} \tag{5.3}$$

The vectors $\mathbf{z}$, $\tilde{\mathbf{w}}$ and the matrix $\mathbf{R}$ are redefined to contain only the first n rows, ignoring the last $(m-n)$ rows. We define the vector $\mathbf{r}_i$ as $\mathbf{r}_i = \left[r_{i,i},\, r_{i,(i+1)},\, \dots\, , r_{i,n}\right]$.

5.3.2 Set-Partitioning and Representation on a tree

The optimal ML detector for the system eqn. (5.1) requires a search over M^n possible ordered pairs of transmitted symbols. This receiver has high complexity. To reduce this complexity, but still achieve near ML-performance, we use ideas from set partitioning and RSSE [8]. Since the x_n doesn't suffer from any interference, we start our detection

algorithm here. We define a two-dimensional set-partitioning scheme $\Omega(k)$ for x_k. Specifically, we divide the M-ary constellation into J_k subsets for the k^{th} element x_k. The symbol x_k under partition $\Omega(k)$ is in subset with index $a_k(k) \in [0, J_k - 1]$. The set-partitioning is done such that intra-subset Euclidean distance is maximized. Let us define a $(n+1) \times 1$ vector, $\mathbf{K} = [K_0, K_1, K_2, \ldots, K_n]$, where $K_0 = 1$, $K_i = \prod_{k=0}^{i-1} J_{n-k}$, $i = 1, 2, \ldots, n$, i.e., the last n elements are the cumulative product of vector $\mathbf{J} = [J_n, J_{n-1}, \ldots, J_1]$.

The tree on which the detection algorithms are implemented is described now. We define a parent node indexed by 0, from which $K_1 = J_n$ child nodes originate, each corresponding to a subset state $\mathbf{t}_1 = [a_n(n)]$ at stage 1 as per $\Omega(n)$. Each of these nodes further branch out to J_{n-1} child nodes, each now corresponding to a subset state vector $\mathbf{t}_2 = [a_{n-1}(n-1), a_n(n)]$. The number of child nodes at this 2^{nd} stage is K_2. This process is continued till we reach the last stage. At the final stage n, the total number of child nodes is given by K_n and each corresponds to a subset state vector $\mathbf{t}_n = [a_1(1), \ldots, a_{n-1}(n-1), a_n(n)]$. This tree is referred to as 'subset state tree'.

A node in the tree indexed i at stage k is denoted by r_i^k, $i = 0, 1, \ldots, K_k - 1$ and $k = 0, 1, 2, \ldots, n$ (Note the superscript is dropped when there is no confusion). The starting parent node is denoted by r_0^0. The set of all possible states at stage k is $S_k = \{r_0^k, r_1^k, \ldots, r_{K_k-1}^k\}$. s_k denotes a state from this set at stage k. For a parent node i at a stage k, $i = 0, 1, \ldots, (K_k - 1)$, $k = 0, 1, 2, \ldots, n-1$, the child nodes at stage $k+1$ are indexed by $\{i, i + K_k, \ldots, i + (J_{n-k} - 1) K_k\}$. This representation helps in describing the soft detection algorithms. The set $\chi_k(i, j)$ refers to the set of all possible input symbols from χ that result in transition from r_i^k to r_j^{k+1}. The set of all possible branch transitions from stage k is denoted by $\mathbf{B}_k = \{(i, j) : \text{ a branch exists between } r_i^k \text{ and } r_j^{k+1}\}$. The notation $BM_{i,j}^k$ refers to the branch metric for the transition from state i at stage k to state j at stage $k+1$ in the tree. Fig. 5.1 represents the 4×4 MIMO system with $\mathbf{J} = [4\,2\,1\,1]$ partition on a tree.

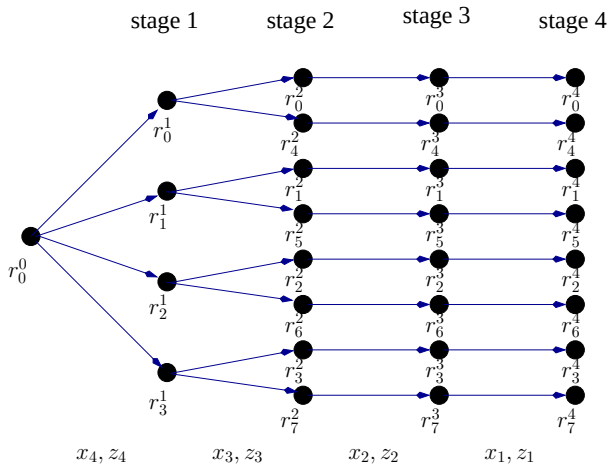

Figure 5.1: Tree diagram representation of a 4X4 MIMO system with $\mathbf{J} = [4\,2\,1\,1]$ partition

5.3.3 Reduced State Sequence Detection

Each edge in the tree of Fig. 5.1 corresponds to a set of parallel transitions when $J_k < M$. The sequence detector starts from x_n since it doesn't suffer from interference due to other streams. We first select the symbol with minimum branch metric from among the parallel transitions for every edge. Due to the symmetry in set partitioning, we can employ slicing to select this winner symbol for each of the J_n child nodes. Let $\mathbf{p}_{1,j} = [\hat{x}_{n,j}]$ denote the survivor symbol vector from among all intra subset symbols for j^{th} node, $j = 0, 1, \ldots, K_1 - 1$ at stage 1. We define branch metric as

$$BM_{0,j}^{0} = |z_n - \langle \mathbf{r}_n, \mathbf{p}_{1,j} \rangle|^2,$$

where $\langle \mathbf{a}, \mathbf{b} \rangle$ denotes the inner product between vectors $\mathbf{a}$ and $\mathbf{b}$. Each $\mathbf{p}_{1,j}$ is fed to pick intra-subset winners for all the child nodes at stage 2 originating from j^{th} node at stage 1. Branch metrics are also computed similarly for all the K_2 nodes at this stage.

The branch metric for transition from state i at stage k to state j at stage $(k+1)$ is given by

$$BM_{i,j}^{k} = |z_{n-k} - \langle \mathbf{r}_{n-k}, \mathbf{p}_{(k+1),j} \rangle|^2, \tag{5.4}$$

where $k = 0, 1, 2, \ldots, n-1$, $j = 0, 1, \ldots, K_{k+1}-1$ and $i = j \pmod{K_k}$. The $(k+1)\times 1$ survivor state vector for node j at stage $(k+1)$ is given by

$$\mathbf{p}_{(k+1),j} = \begin{bmatrix} \hat{x}_{(n-k),j} & \mathbf{p}_{k,i} \end{bmatrix},$$

where $\hat{x}_{(n-k),j}$ is the surviving symbol for the state j at stage $(k+1)$. Also note that $\mathbf{p}_{0,0}$ refers to an empty vector.

At the end of stage n, we have $K_n = \prod_{k=1}^{n} J_k$ paths, referred to as 'Subset Survivor Paths (SSP)'. We obtain a path metric PM_i for each SSP, where $i \in \{0, 1, \ldots, K_n - 1\}$, by summing up all the branch metrics in the path. The optimum path is the one with minimum metric and we trace back on this path to detect the transmitted symbols, i.e.,

$$\hat{\mathbf{x}} = \mathbf{p}_{n,i^\star}, \text{ where } i^\star = \arg\min_i PM_i \tag{5.5}$$

5.4 Non-Uniform Set-Partitioning

We know from ref. [8] that whenever $J_k < M$ there is a SNR gain due to set partitioning and from [20] that k^{th} stage of the MIMO has a diversity of the order $(m-k+1)$. It is also known from [20] that even optimal ordering, doesn't change the diversity order and it only increases the first step SNR by 3 dB. Using the approach in [21], the probability of symbol error is bounded as

$$\mathrm{P}\left(E_s\right) \leq \mathrm{P}\left(F\right) + \mathrm{P}_s\left(\varepsilon_{MLD}\right), \tag{5.6}$$

where the first term corresponds to the probability that intra-subset error occurs in slicing and the second term is the probability that wrong SSP is selected, which is outer-bounded by the same probability over the entire search space of ML decoding. The second term as shown in [32], has a diversity order of m for each sub-stream symbol. Using the approach in [5], the first term is upper-bounded by

$$\mathrm{P}\left(F\right) \leq \sum_{i=n}^{i=1} \frac{L_i}{\left(1 + \frac{d_{min}^2 J_i}{4\sigma^2}\right)^{m-i+1}}, \tag{5.7}$$

where d_{min} is the minimum distance of the signal constellation and L_i denotes the number of nearest neighbors in the subset constellation, which depends on J_i. From eqns. (5.6), (5.7) we know that the error probability is limited by the 1^{st} stage (diversity of $m-n+1$) and if we do not apply set-partitioning for this stage, performance is then

limited by the 2^{nd} stage with diversity $m - n + 2$. For the specific case when $m = n$, the first stage is a real bottleneck, since its diversity is only of order 1.

In [21], authors proposed the use of USP, $J_k = 2^q$, $\forall k \in [1, n], 2 \leq q \leq \log_2 M$. It is observed that with USP the performance is degraded at high SNR. Set partitioning in the 1^{st} stage resulted in low diversity order, which degraded the performance. Realizing that except for the first few stages, the remaining stages, due to better diversity orders are more tolerant to errors introduced by set partitioning, the authors of [14] proposed non-uniform set partitioning. Accordingly the number of set partitions for 1^{st} stage, J_n be M, since it has the least diversity order. The number of set partitions for other stages can be kept low and they should also be in non-increasing order. Hence it is advantageous to restrict to partitions with $\mathbf{J} = [M, J_{n-1}, \ldots, J_1]$, where $M \geq J_{n-1} \geq \ldots \geq J_1 \geq 1$. For the case when $m > n$, the extra receive antennas increase the diversity order for all stages, leading to better performance. In these cases, we can also partition the 1^{st} stage symbol x_n (the extra diversity order offsets any loss due to set partitioning).

5.5 Soft-Detection Algorithms

In this section, soft-detection algorithms for a general MIMO system in flat fading channels are developed. We develop a forward/backward algorithm on the lines of BCJR algorithm [2], [15]. An optimal forward/backward algorithm has high complexity even for modulation schemes such as 8-PSK and moderate number of transmitted streams. To reduce the complexity, but still achieve near MAP-performance, we develop this algorithm on a subset state tree depicted in Fig. 5.1. This algorithm is referred as 'reduced-state maximum-a-posteriori (RSMAP) detection algorithm'.

5.5.1 Reduced State Maximum *A Posteriori* Detector

The objective is to output the symbol level APP's,

$$\mathrm{P}\left(x_{n-k} = x|\mathbf{z}\right), \, x \in \chi, \, k = 0, 1, \ldots, n-1 \tag{5.8}$$

Let us first find the probability that a branch from state $s_k = r_i$ at stage k to state $s_{k+1} = r_j$ at stage $k+1$, given the received observations is in the optimal detected path. This conditional probability is given by

$$\mathrm{P}\left(s_{k+1} = r_j, s_k = r_i | \mathbf{z}\right) = \mathrm{P}\left(s_{k+1} = r_j, s_k = r_i, \mathbf{z}\right) / \mathrm{P}\left(\mathbf{z}\right)$$

Applying chain rule of probabilities to the numerator, results in

$$\mathrm{P}\left(s_{k+1} = r_j, s_k = r_i, \mathbf{z}\right) = \underbrace{\mathrm{P}\left(s_k, z_n, \cdots, z_{n-k+1}\right)}_{\alpha_k(s_k)} \cdot \underbrace{\mathrm{P}\left(s_{k+1}, z_{n-k} | s_k\right)}_{\gamma_k(s_k, s_{k+1})} \cdot \underbrace{\mathrm{P}\left(z_{n-k-1}, \cdots, z_1 | s_{k+1}\right)}_{\beta_{k+1}(s_{k+1})} \tag{5.9}$$

The forward metric of a state r_i at stage k, $\alpha_k\left(r_i\right)$ is calculated using the following recursive relation

$$\alpha_k\left(r_i\right) = \alpha_{k-1}\left(r_l\right) \cdot \gamma_{k-1}\left(r_l, r_i\right),\ l = i \,(\mathrm{mod}) K_{k-1} \tag{5.10}$$

Similarly, the backward metric of state r_i at stage k, $\beta_k\left(r_i\right)$ is given by the recursive relation

$$\beta_k\left(r_i\right) = \sum_{q=0}^{J_{n-k}-1} \beta_{k+1}\left(r_j\right) \cdot \gamma_k\left(r_i, r_j\right),\ j = i + qK_k \tag{5.11}$$

The initial conditions are set to $\alpha_0(r_0) = 1$ and $\beta_n(r_i) = 1,\ \forall i \in [0, K_n - 1]$, since all states are possible in the last stage. The transition probability is computed by

$$\gamma_k\left(s_k = r_i, s_{k+1} = r_j\right) = \mathrm{P}\left(s_{k+1} | s_k\right) \cdot \mathrm{P}\left(z_{n-k} | s_k, s_{k+1}\right)$$

This transition probability is $0, \forall\left(i, j\right) \notin \mathrm{B}_k$ and if $\left(i, j\right) \in \mathrm{B}_k$, this probability is computed by conditioning RHS on input symbols belonging to the set $\chi_k\left(i, j\right)$. Since these events are independent, transition probability, if $\left(i, j\right) \in \mathrm{B}_k$ is given by

$$\gamma_k\left(s_k = r_i, s_{k+1} = r_j\right) = \sum_{\forall x \in \chi_k(i,j)} \mathrm{P}\left(x_{n-k} = x\right) \mathrm{P}\left(z_{n-k} | s_k = r_i, x_{n-k} = x\right) \tag{5.12}$$

The first term in the product on RHS corresponds to the prior input probability, which is known *a priori*. The second term in the product has a Gaussian distribution function, provided past history of transmitted symbols up to the state $s_k = r_i$ is known. To

determine the path history for a state $s_k = r_i$, we compute the product under summation in eqn. (5.12) (substituting k in the product by $k-1$) for all possible input values in $\chi_{k-1}(l,i)$, $l = i \,(\text{mod}) K_{k-1}$, $k = 1, 2, \cdots, n-1$. Let $\hat{x}_{(n-k+1),i}$ be the symbol that maximizes this product. The past history for the state $s_k = r_i$ is given by

$$\mathbf{p}_{k,i} = \begin{bmatrix} \hat{x}_{(n-k+1),i} & \mathbf{p}_{(k-1),l} \end{bmatrix} \tag{5.13}$$

The second term in the product of eqn. (5.12) is calculated using

$$\mathrm{P}\left(z_{n-k} | s_k = r_i, x_{n-k} = x\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{|z_{n-k} - \langle \mathbf{r}_{n-k}, [x \quad \mathbf{p}_{k,i}] \rangle|^2}{2\sigma^2} \right) \tag{5.14}$$

The transition probabilities for all transitions in the tree are evaluated using prior input symbol probabilities and eqns. (5.12), (5.14). The forward and backward metrics for all nodes in the tree are evaluated using the eqns. (5.10), (5.11) and the initial conditions. The desired APP's of the transmitted symbols $\mathrm{P}\left(x_{n-k} = x | \mathbf{z}\right)$, calculated by summing the fraction of the branch APP's $\mathrm{P}\left(s_{k+1} = r_j, s_k = r_i | \mathbf{z}\right)$ that correspond to the input symbol $x_{n-k} = x$, is given by

$$\mathrm{P}\left(x_{n-k} = x | \mathbf{z}\right) = \sum_{\forall (i,j) \in \mathrm{B_k} : x \in \chi_k(i,j)} \alpha_k\left(r_i\right) \mathrm{P}\left(x_{n-k} = x\right) \mathrm{P}\left(z_{n-k} | s_k = r_i, x_{n-k} = x\right) \beta_{k+1}\left(r_j\right) \tag{5.15}$$

These *a posteriori* symbol probabilities are converted to *a posteriori* bit-level probabilities using

$$\mathrm{P}\left(b_{n-k}[j] = 0 | \mathbf{z}\right) = \sum_{\begin{pmatrix} \forall x \in \chi : j^{th} \text{ position in binary} \\ \text{mapping of symbol } x \text{ is } 0 \end{pmatrix}} \mathrm{P}\left(x_{n-k} = x | \mathbf{z}\right), \tag{5.16}$$

where $j = 0, 1, \cdots, \log_2 M,$ which are then fed to the channel decoder after getting deinterleaved.

This algorithm differs from BCJR implemented on a trellis for transmission of binary modulation over frequency selective channels [15] in

- Unlike in a trellis where lookup table is required to relate present and past/future states, tree structure in this problem helped us define an exact mathematical relation between parent and child nodes.
- Lack of summation in eqn. (5.10) because of tree structure instead of trellis.
- Summation in eqn. (5.12) because of the parallel transitions in the tree due to set partitioning.
- The path history vector in the exponent of exponential in eqn. (5.14) gradually increases in length as we move from stage 1 to stage n, whereas the path history in case of ISI channels is constant in length.
- Only the part of transition probability of a branch, which corresponds to a particular symbol is used in the computation of APP for that symbol using eqn. (5.15), since a single branch maps to a set of parallel transitions.

5.6 Approximate Soft Detection Algorithms

The RSMAP algorithm described in the previous section is the optimal algorithm for a given set partitioning scheme, in terms of minimizing the error rate. The downside is its complexity of implementation, especially for large number of transmitted data streams, n. The complexity of RSMAP algorithm is mainly due to

- High memory storage requirement to implement backward recursion.
- Huge number of computations (significant number of them involve exponentiation).

In this section we develop two variations of RSMAP to tackle the two issues mentioned above.

5.6.1 Forward RSMAP Algorithm

We can reduce the storage using approaches in [29]. Instead **we now present a forward only algorithm that doesn't require a backward recursion**. The forward recursion in this algorithm is exactly same as in RSMAP detection algorithm (Subsection 5.5.1). This algorithm involves two stages:

- Soft decision failure correction.
- Computing the symbol APP's.

This algorithm also starts from x_n and proceeds till x_1. The path history of each state is found exactly the same way in both RSMAP and forward RSMAP algorithms. At a given stage $(k+1)$, the forward metrics of all states and all branch transition probabilities are computed using eqns. (5.10), (5.12) and (5.14). The surviving symbol vectors of length $(k+1)$ for each state in this stage are mapped to their corresponding binary bit sequences. Whenever $J_{n-k} < M$, sometimes some of the bit positions amongst $(k+1)\log_2 M$ are either 0 or 1 in all the surviving symbol bit vectors at stage $(k+1)$. These bit positions reach hard-decisions due to set partitioning. This is referred to as 'soft decision failure'. This failure definitely doesn't occur when the number of partitions in a stage is M. This is avoided in RSMAP by the use of backward recursion and computing the APP's from the branch transitions. Also note that for a given J_{n-k}, the number of bit positions amongst $\log_2 M$, corresponding to the symbol x_{n-k}, where failure happens cannot exceed $\log_2 M - \log_2 J_{n-k}$, since the surviving symbols should at least differ in $\log_2 J_{n-k}$ bit positions.

For the bit positions in stage $(k+1)$ where soft decision failure happens, eqn. (5.16) is used to compute the bit level APP's, where the corresponding symbol level APP's are obtained using

$$\mathrm{P}\left(x_{n-k}=x|\mathbf{z}\right)=\sum_{i=0}^{K_k-1}\left\{\alpha_k\left(r_i\right)\mathrm{P}\left(x_{n-k}=x\right)\cdot\mathrm{P}\left(z_{n-k}|s_k=r_i,x_{n-k}=x\right)\right\} \quad (5.17)$$

We proceed from stage 1 to n, applying soft-decision failure correction wherever necessary. This algorithm differs from the RSMAP detection algorithm in the computa-

tion of the final symbol APP's also. We first determine $\hat{x}_{1,j},\ j = 0, 1, \cdots, K_n - 1$ using the procedure outlined in Subsection 5.5.1. The survivor state history for all nodes in stage n is obtained by substituting k with n in from eqn. (5.13). The APP's are determined using

$$\mathrm{P}\left(x_{n-k} = x|\mathbf{z}\right) = \sum_{\forall j:\mathbf{p}_{n,j}[n-k]=x} \alpha_n\left(r_j\right),\ k = 0, 1, \cdots, n-1 \tag{5.18}$$

Note that waiting till the last stage to compute APP's results in even x_n getting some diversity advantage from the later stages. The bit level APP's are obtained by substituting APP's computed from eqn. (5.18) in eqn. (5.16). Note that at bit positions where the failure happened, bit-level APP's computed using information only up to that stage are used, instead of 0 or 1 probabilities obtained by substituting eqn. (5.18) in eqn. (5.16). This is suboptimal since, those bit positions are not enjoying the diversity advantage of the later stages, unlike other bits. This affects the performance of this algorithm for certain set-partitions, an observation validated by simulations.

5.6.2 Max-Log Approximation Algorithm

The famous max-log approximation [26] can be applied to these algorithms to reduce the complexity even further. For the sake of brevity we are skipping the exact equations obtained under this approximation. We prove from our simulation results that applying max-log approximation on RSMAP algorithm does not affect its performance greatly. On the other hand, this approximation reduces the complexity significantly. The exact reduction in complexity due to this approximation on RSMAP algorithm is given in the following section. This approximation could also be combined with forward RSMAP algorithm developed in Subsection 5.6.1.

5.7 Complexity Analysis

In this section, we compute and compare the implementation complexity for all these algorithms. Complexity associated with QR decomposition and the computation of bit

level APP's from symbol level APP's is ignored, since it is same for all the algorithms described. Complexity is measured in terms of the exact number of exponential operations, branch metric computations, comparisons, additions and multiplications required. Complexity involved in slicing operations is not included. A single computation involving eqn. (5.4) and the negative of exponent in the exponential in eqn. (5.14) refers to one branch metric computation for hard and soft algorithms respectively. For RSSD algorithm (Subsection 5.3.3), at stage k, we need to compute K_k branch metric computations using equation (5.4), after employing slicing amongst all intra-subset symbols to determine the survivor symbol. Thus, this algorithms requires a total of $\sum_{k=1}^{n} K_k$ branch metric computations. Also, we observe that $\sum_{k=1}^{n} K_k$ additions are required for this algorithm. Finally, we need to implement $(K_n - 1)$ comparisons and also trace back to determine the transmitted data.

The complexity of implementing the soft-detection algorithms is considerably higher than RSSD. We only estimate the number of computations required to find the symbol level APP's. All the algorithms compute bit level APP's using eqn. (5.16). For RSMAP in Subsection 5.5.1, from each node at every stage, we need to compute M branch metrics and one exponential operation for every branch metric computation. Hence the number of exponential and branch metric computations necessary is $M \sum_{k=0}^{n-1} K_k$. Implementing eqns. (5.11), (5.12) and (5.15) require addition operations. The number of these additions needed are $(2M-1)\sum_{k=0}^{n-1} K_k - nM$. Since all the symbols are assumed to be equiprobable *a priori*, the multiplications involving a priori probabilities and constants in equations (5.12), (5.14) are not included. A normalization of the final symbol APP's to make their sum 1 is sufficient. The number of multiplication operations needed are $2M \sum_{k=0}^{n-1} K_k + 2 \sum_{k=1}^{n} K_k$.

Forward recursion is same in both RSMAP and forward RSMAP algorithm. Hence the number of exponential operations and branch metric computations required are same as that of RSMAP algorithm, $M \sum_{k=0}^{n-1} K_k$. Similarly the number of additions required to compute the transition probabilities is $M \sum_{k=0}^{n-1} K_k - \sum_{k=1}^{n} K_k$. The number of addition operations required in computation of APP's using eqn. (5.18) is upper-bounded by $Mn\,(K_n - 1)$. Similarly the number of multiplication operations required

in computing the forward metrics are $\sum_{k=1}^{n} K_k$. Determining the survivor symbol requires $(M-1)\sum_{k=0}^{n-1} K_k - K_n + 1$ comparisons. Whenever soft-decision failure happens in a stage, irrespective of the number of bit positions where failure happens, the symbol APP's are computed using eqn. (5.17). For failure in stage k, computing APP's using eqn. (5.17) requires $M\left(K_{k-1}-1\right)$ addition and MK_{k-1} multiplication operations. Hence even assuming that failure happens in all stages from stage $n-u+1$ to n, where $u \in [1, n-1] \ni J_k < M, \forall k \leq u$ and $J_k = M, \forall k > u$, the number of addition and multiplication operations required are $M\sum_{k=n-u}^{n-1} K_k - Mn$ and $M\sum_{k=n-u}^{n-1} K_k$ respectively. Hence the number of addition operations is bounded by

$$M\sum_{k=0}^{n-1} K_k - \sum_{k=1}^{n} K_k + Mn\left(K_n - 1\right) + M\sum_{k=n-u}^{n-1} K_k - Mu$$

and the multiplication operations are bounded by $\sum_{k=1}^{n} K_k + M\sum_{k=n-u}^{n-1} K_k$. Note that the number of computations to obtain bit level APP's from symbol level APP's using eqn. (5.16) is same in both RSMAP and forward RSMAP algorithms. In forward RSMAP algorithm, for bit positions where soft decision failure occurred the APP's obtained from eqn. (5.17) and at the remaining positions, the APP's from eqn. (5.18) are used to compute bit level APP's from eqn. (5.16).

From the above two paragraphs we can observe that RSMAP and forward RSMAP require same number of branch metric and exponential operations. They differ greatly in the number of addition and multiplication operations necessary. The table below summarizes this difference:

	in Forward RSMAP, when compared with RSMAP
Additional Additions	$Mn(K_n - 1) - M\sum_{k=0}^{n-u-1} K_k + 1 - M(n-u)$
Additional Multiplications	$-\left[M\sum_{k=0}^{n-u-1} K_k + M\sum_{k=0}^{n-1} K_k + \sum_{k=1}^{n} K_k\right]$

Table 5.1: Difference in complexity between RSMAP and forward RSMAP algorithms

Hence RSMAP requires fewer additions that forward RSMAP, whereas forward RSMAP requires fewer multiplication operations than RSMAP algorithm. From hard-

ware implementation point of view, multiplications are more difficult than additions. Also forward RSMAP requires a small number of comparison operations. Hence forward RSMAP has lower complexity than RSMAP algorithm.

Let us now examine the memory storage requirements for both these algorithms. RSMAP algorithm requires lot of memory since we need to store forward, backward metrics of all states and transition probabilities of all branches. Forward RSMAP has lower memory requirements. We need to store only a vector where we update the bit-level APP's for positions where failure happens and a vector of length K_n (if memory cannot be allocated dynamically) to store the forward metrics of all states in a stage. Temporary variables can be used to store information about transition probabilities. Therefore forward RSMAP is attractive for systems with very high order modulation or for systems with large antenna array, where implementing backward recursion is not feasible due to memory constraints.

The complexity of RSMAP algorithm when max-log approximation is applied is significantly lower. All the computations are effectively done in log domain. Product of exponentials in RSMAP becomes sum of their exponents in max RSMAP algorithm. In max RSMAP algorithm, a winner is selected from amongst the parallel transitions using slicing. Branch metric computed using that winner is the transition probability for that branch. Forward/backward metrics are obtained by adding forward/backward metric of a node to the transition probability metric of each branch originating/terminating from that node. Finally the symbol APP's in log domain are computed by adding forward, backward metrics and transition probabilities corresponding to that symbol. The negative exponents of these APP's in log domain are normalized to obtain the symbol level APP's. This process requires only Mn exponential operations, much lower when compared with RSMAP algorithm and is also independent of partition scheme. This algorithm also doesn't require any multiplication operations, except during normalization of probabilities. The number of branch metric computations and additions required are comparable in these algorithms. Memory requirements of max RSMAP are very similar to RSMAP algorithm. Hence max-log approximation reduces the complexity significantly.

5.8 Simulation Results

A rate $R = \frac{1}{2}$ convolution code with constraint length 3 and generator polynomial $(1 + D^2 \quad 1 + D + D^2)$ [19] as an encoder and a block interleaver with 100 rows is used in the following simulations. We simulated the performance of various detection algorithms with both encoder and interleaver for 4×4 and 4×5 MIMO systems in block fading channels for various partition schemes with QPSK and 8-PSK modulation. The partition schemes are restricted to be of form $[M, J_3, J_2, J_1]$ from our analysis in section 5.4. Specifically, the performance of RSMAP for various partition schemes is compared against the optimal MAP performance for QPSK and we also discuss the performance of these algorithms with $[8\,1\,1\,1]$ and $[8\,4\,2\,1]$ partition schemes for 8-PSK.

Fig. 5.2 compares the performance of RSMAP algorithm with the optimal performance achieved by MAP decoding algorithm for 4×4 MIMO with QPSK modulation. We can clearly observe that, with a suitable partition, in this case $[4\,4\,2\,1]$, RSMAP approaches the performance of MAP. There is a loss of only about 0.1 dB. Similarly, we can observe that RSMAP with $[4\,4\,2\,1]$ partition in case of 4×5 QPSK modulated MIMO system, looses only about 0.1 dB when compared with the MAP performance. Hence RSMAP algorithm with a carefully chosen partition scheme is near-optimal. The choice of partition depends on the complexity-performance trade-off point of view.

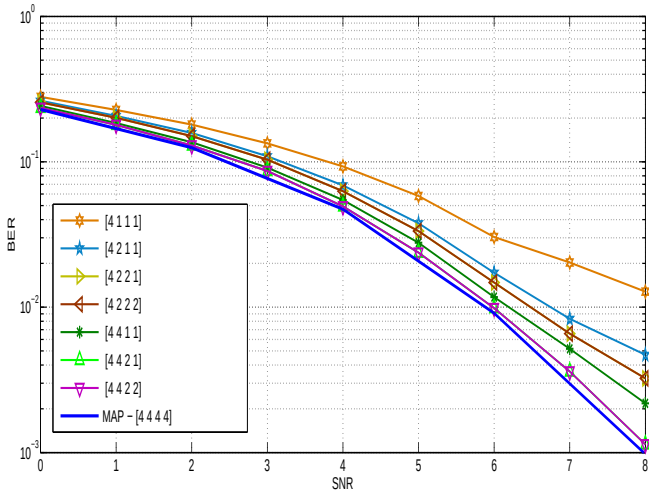

Figure 5.2: Performance of RSMAP for different partition schemes with 4 × 4 QPSK modulated MIMO system

Fig. 5.4 compares the different detection algorithms for transmission of 8-PSK mod-

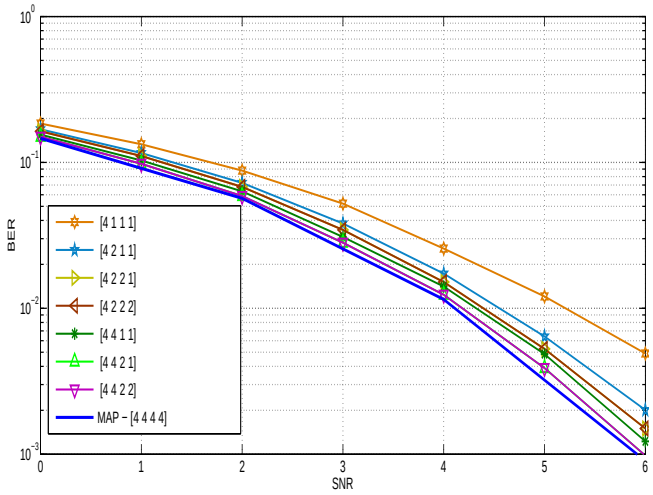

Figure 5.3: Performance of RSMAP for different partition schemes with 4×5 QPSK modulated MIMO system

ulation over a block fading channel via 4×4 MIMO system for $[8\,1\,1\,1]$ partition. We observe that max RSMAP is as good as RSMAP. This is expected and is observed to be true in all cases. Henceforth we only present results for RSMAP algorithm. Also both give about 2 dB gain compared with hard RSSD (RSSD followed by a deinterleaver and a Viterbi convolutional decoder) detector. FRSMAP with soft decision failure refers to the forward RSMAP algorithm with no soft-decision failure correction mechanism. We also observe that when soft decision failure is not corrected, it performs the worst. The fraction of code bits with soft decision failure is very high for this partition. As a result a significant portion of the input to the decoder contains incorrect soft information and hence the performance worsens significantly.

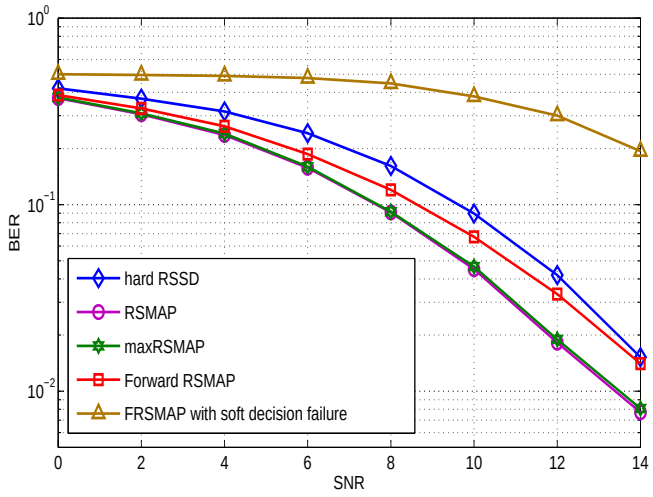

Figure 5.4: Performance of various detection algorithms for coded 4×4 MIMO with 8-PSK for $J = [8\,1\,1\,1]$ partition

Also note that forward RSMAP algorithm approaches the performance of hard RSSD at high SNR for $[8\,1\,1\,1]$ partition scheme. To understand this effect, we must realize that there are two factors at play here, one is the soft processing gain, which is about 2 dB in this case and the effect of diversity at high SNR. For the partition $J = [8\,1\,1\,1]$, soft decision failure in stages 2, 3 and 4 results in the corresponding bit positions enjoying diversity gain of order 2, 3 and 4 respectively. But in case of RSSD, all bit positions enjoy the diversity order of 4, since we use energy from every stage in making a final hard decision. This difference in diversity orders between the two algorithms is indicated by the different slopes for the BER curves of these two algorithms. As a result at high enough SNR, hard RSSD outperforms forward RSMAP, which can be observed around 14 dB. It is important to note that even for $J = [8\,1\,1\,1]$ partition where soft decision failure happens very frequently, forward RSMAP is better than hard RSSD in low to medium SNR region in which we are interested in practice.

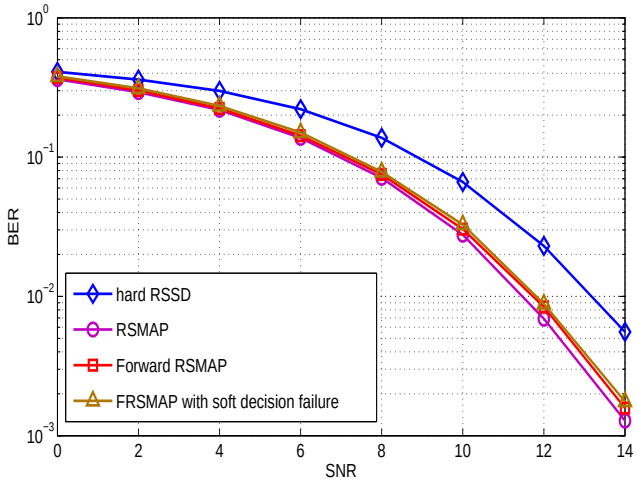

Figure 5.5: Performance of various detection algorithms for coded 4×4 MIMO with 8-PSK for $J = [8\,4\,2\,1]$ partition

For $J = [8\,4\,2\,1]$ partition, we observe from Fig. 5.5 that FRSMAP with soft decision failure and forward RSMAP give nearly same performance. This is because soft-decision failure happens with very small probability for this partition and hence the effect of not correcting the soft-decision failure is not significant. Since soft-decision failure is rare for this partition scheme, most bits enjoy the full diversity benefit under both forward RSMAP and RSSD. This can be observed by noting that the curves corresponding to these two algorithms have nearly same slope. As a result the forward

RSMAP outperforms hard RSSD due to soft processing gain (about 2 dB). We also observe that for this partition where soft decision failure happens very rarely, removing backward recursion only results in a loss of performance by about 0.3 dB. This is very analogous to what happens in the case of equalization of frequency selective channels.

For this partition, where the performance of forward RSMAP and RSMAP are comparable, forward RSMAP soft detection algorithm is a better choice than RSMAP algorithm due to its lower complexity and memory requirements. Hence for partitions where soft decision failure happens less frequently forward RSMAP is the optimal choice.

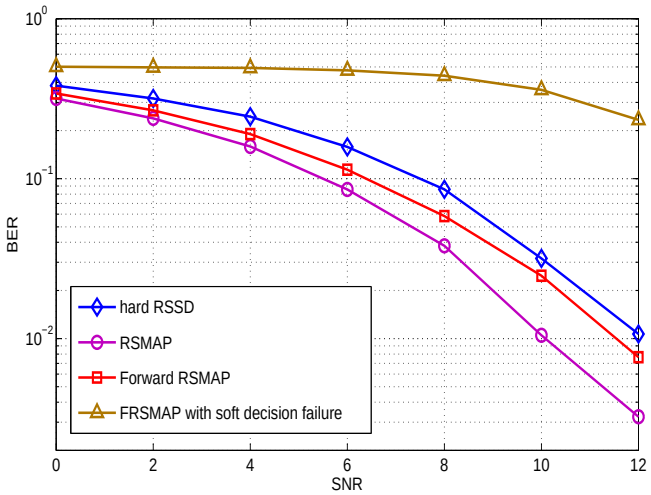

Figure 5.6: Performance of various detection algorithms for coded 4×5 MIMO with 8-PSK for $J = [8\,1\,1\,1]$ partition

Figs. 5.6 and 5.7 compare the performance of the same set of algorithms for 4×5 MIMO system employing 8-PSK modulation. BER in this system is about an order of magnitude lower than in case of 4×4 MIMO systems, since the system is limited now by the first stage with a better diversity order of 2. This extra diversity order decreases the error rate, and the explanations given in previous paragraphs are still valid.

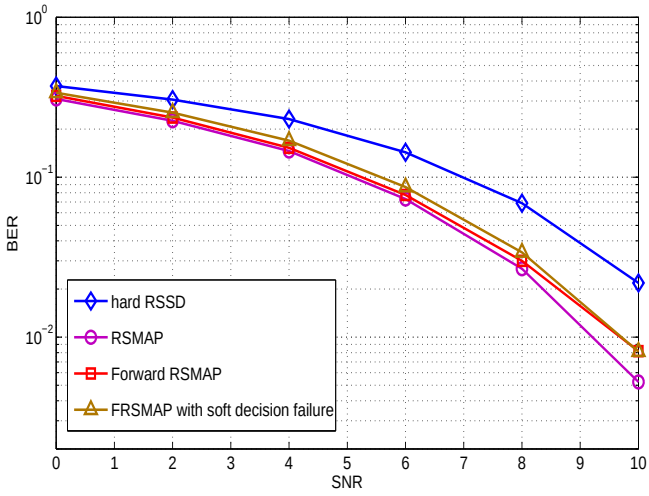

Figure 5.7: Performance of various detection algorithms for coded 4×5 MIMO with 8-PSK for $J = [8\,4\,2\,1]$ partition

CHAPTER 6

Conclusions

A general framework for an iterative receiver is developed and presented. This is applicable for any inner and outer code decoder that takes in soft information and outputs soft information. We then focused on developing low-complexity decoding algorithms for frequency selective fading (ISI) channels and MIMO systems. These algorithms were found to be near MAP algorithm in performance. Also the iterative receiver developed for frequency-selective fading channels, along with SAIC prefilter was used to suppress a single co-channel interferer. For MIMO systems, we developed forward-only soft-detection algorithm that requires considerably lower memory and lower complexity than the RSMAP algorithm.

Some possible directions for future work are

- Develop a forward-only low-complexity equalization algorithm for HOM systems.
- Evaluate the performance of iterative SAIC receiver for the general case of a EDGE user with a dominant GMSK interferer.
- Develop a low-complexity max forward RSMAP detection algorithm for MIMO in flat-fading channels.
- Extend the detection algorithms developed for MIMO in flat-fading to MIMO in frequency-selective channels.

APPENDIX A

Channel Estimation

Here, we observe that both user's signals in principle propagate through the same channel. If the received symbols corresponding to the time-aligned training sequences of both users are collected in a vector $\mathbf{y}$, this vector can be expressed as

$$\mathbf{y} = \cos(\alpha)\ \mathbf{A}_1\ \mathbf{h} + \sin(\alpha)\ \mathbf{A}_2\mathbf{h} + \mathbf{B}_1\ \mathbf{g} + \mathbf{w} \tag{A.1}$$

where $\mathbf{A}_1$, $\mathbf{A}_2$ and $\mathbf{B}_1$ represent $(\ N - L\ )$ x $(\ L + 1\ )$ Toeplitz convolution matrices corresponding to the training sequences of user 1, user 2 and the interferer respectively with the training sequence length N, and $\mathbf{h} = [h[0]\, h[1] \ldots h[L]]^T$, and $\mathbf{g} = [g[0]\, g[1] \ldots g[L]]^T$. For simplicity, the factor j in (eqn. 4.6) has been absorbed in $\mathbf{A}_2$.

A.1 Joint ML estimation

Equation (A.1) is rewritten as

$$\mathbf{y} = l\, \mathbf{S}_1\, \mathbf{\Theta} + \sqrt{1 - l^2}\, \mathbf{S}_2\, \mathbf{\Theta} + \mathbf{S}_3\, \mathbf{\Theta} + \mathbf{w} \tag{A.2}$$

where $l = \cos(\alpha)$, $\mathbf{\Theta} = [\mathbf{h} \quad \mathbf{g}]^T$, $\mathbf{S}_1 = \begin{bmatrix} \mathbf{A}_1 & 0 \end{bmatrix}$ and $\mathbf{S}_2 = \begin{bmatrix} \mathbf{A}_2 & 0 \end{bmatrix}$ and $\mathbf{S}_3 = \begin{bmatrix} 0 & \mathbf{B} \end{bmatrix}$ The joint Maximum-Likelihood (ML) estimates for $\mathbf{\Theta}$ and α result from minimizing the L_2-norm of the error vector

$$\mathbf{e} = \mathbf{y} - \hat{l}\, \mathbf{S}_1\, \hat{\mathbf{\Theta}} - \sqrt{1 - \hat{l}^2}\, \mathbf{S}_2\, \hat{\mathbf{\Theta}} - \mathbf{S}_3\, \hat{\mathbf{\Theta}},$$

where $\hat{l}$, $\hat{\mathbf{\Theta}}$ denote the estimated quantities. Differentiating $\mathbf{e}^H\ \mathbf{e}$ w.r.t. l and $\mathbf{\Theta}$ and setting the derivative to zero results in the following two conditions for ML estimates

of $\mathbf{\Theta}$ and l.

$$2\,l\,a = b - c\left(\frac{1-2l^2}{\sqrt{1-l^2}}\right) - d\left(\frac{l}{\sqrt{1-l^2}}\right) - f, \tag{A.3}$$

$$\mathbf{\Theta} = \left(\mathbf{P}^H\,\mathbf{P}\right)^{-1}\,\mathbf{P}^H\,\mathbf{y}. \tag{A.4}$$

where

$$\begin{aligned}
a &= \mathbf{\Theta}^H\left(\mathbf{S_1}^H\mathbf{S_1} - \mathbf{S_2}^H\mathbf{S_2}\right)\mathbf{\Theta} \\
b &= \left(\mathbf{y}^H\mathbf{S_1}\mathbf{\Theta} + \mathbf{\Theta}^H\mathbf{S_1}^H\mathbf{y}\right), \\
d &= \left(\mathbf{y}^H\mathbf{S_2}\mathbf{\Theta} + \mathbf{\Theta}^H\mathbf{S_2}^H\mathbf{y} - \mathbf{\Theta}^H\left(\mathbf{S_2}^H S_3 + S_3^H\mathbf{S_2}\right)\mathbf{\Theta}\right) \\
c &= \mathbf{\Theta}^H\left(\mathbf{S_1}^H\mathbf{S_2} + \mathbf{S_2}^H\mathbf{S_1}\right)\mathbf{\Theta}, \\
f &= \mathbf{\Theta}^H\left(\mathbf{S_1}^H S_3 + S_3^H\mathbf{S_1}\right)\mathbf{\Theta}, \\
\mathbf{P} &= l\,\mathbf{S_1} + \sqrt{1-l^2}\mathbf{S_2} + S_3
\end{aligned}$$

Equations (A.3) and (A.4) may be viewed as ML estimate of l given $\mathbf{\Theta}$, and ML estimate of $\mathbf{\Theta}$ given l respectively. However, it does not seem possible to obtain a closed form solution to $l, \mathbf{\Theta}$ from the coupled equations, one can solve them iteratively from any initial choice of l.

A.2 Blind estimation

The received signal during training sequence is written as

$$\mathbf{y} = l\;\mathbf{A}_1\;\mathbf{h} + \sqrt{1-l^2}\;\mathbf{A}_2\mathbf{h} + \;\mathbf{B}_1\mathbf{g} + \mathbf{w}. \tag{A.5}$$

When the knowledge of l is available,

$$\mathbf{y} = \left[\;l\mathbf{A}_1 + \sqrt{1-l^2}\mathbf{A}_2\;\;\mathbf{B}\right]\;\mathbf{\Theta} + w = \mathbf{S}\,\mathbf{\Theta} + \mathbf{w}. \tag{A.6}$$

Let $\phi_{\mathbf{\Theta\Theta}} = E\left[\mathbf{\Theta\Theta}^H\right] \cong I_{2(L+1)}$ be the second order statistics of the channel. The pdf of $\mathbf{y}$ given l is given by

$$f_{\mathbf{y}/l}(y) = \frac{1}{\pi^M} \times \frac{1}{det(\phi_{\mathbf{yy}/l})} \times e^{\left\{-\mathbf{y}^H \phi_{\mathbf{yy}/l}^{-1} \mathbf{y}\right\}} \tag{A.7}$$

where $\phi_{\mathbf{yy}_{/l}} = \mathbf{S}\,\phi_{\mathbf{\Theta\Theta}}\,\mathbf{S}^H + \eta\, I_M$. The ML estimate of l can be obtained by maximizing the $\ln(f_{\mathbf{y}/l}(y))$

$$\begin{aligned} \therefore \hat{l} &= \arg\max_{l} \{-\mathbf{y}^H \phi_{\mathbf{yy}_{/l}}^{-1} \mathbf{y} - \ln(det(\phi_{\mathbf{yy}_{/l}}))\} \\ \Longrightarrow \hat{l} &= \arg\min_{l} \{\mathbf{y}^H \phi_{\mathbf{yy}_{/l}}^{-1} \mathbf{y} + \ln(det(\phi_{\mathbf{yy}_{/l}}))\} \\ \hat{\mathbf{\Theta}} &= \left(\mathbf{S}^H\mathbf{S}\right)^{-1} \mathbf{S}^H\mathbf{y} \end{aligned}$$

Here one-dimensional search is required which is less complex than iteratively solving the two equations and the performance is almost same. When implementing a single-dimensional search, MS can use the knowledge of α in previous frame and can get α of the current frame with out much computation. So, essentially the blind technique involves a one-dimensional search in the first burst of communication and it can be updated to design low complexity receivers.

APPENDIX B

Single Antenna Interference Cancellation Algorithms

We describe the implementation of SAIC algorithm for the case of VAMOS in GMSK. We use the channel estimation techniques presented in appendix A to get estimates ofα and θ, the received symbols are obtained from equation (4.6) as follows.

$$y\left[n\right] = \sum_{k=0}^{L} h(k)\ \left[\cos(\alpha)\, a_1(n-k) + j\ \sin(\alpha)\, a_2(n-k)\right] + \sum_{k=0}^{L} g(k)\, b_1(n-k) + w(n)$$

$$\implies \mathbf{y}_n = \sum_{k=0}^{L} \mathbf{H}_k \mathbf{a}_{n-k} + \underbrace{\sum_{k=0}^{L} \mathbf{G}_k \mathbf{b}_{n-k} + \mathbf{w}_n}_{\text{impairement vector } \mathbf{q}_n}, \tag{B.1}$$

where

$$\mathbf{H}_k = \begin{bmatrix} h_I(k) & -h_Q(k) \\ h_Q(k) & h_I(k) \end{bmatrix},\ \mathbf{G}_k = \begin{bmatrix} g_I(k) \\ g_Q(k) \end{bmatrix}$$

$$\mathbf{a}_k = \begin{bmatrix} \cos(\alpha)\, a_1(k) \\ \sin(\alpha)\, a_2(k) \end{bmatrix}, \mathbf{w}_n = \begin{bmatrix} w_I(n) \\ w_Q(n) \end{bmatrix}, \mathbf{b}_k = b_1(k).$$

As shown in figure (4.2), the vector impairment signal can be suppressed using a 2-D MMSE DFE-feed forward filter . We denote the $\mathbf{F}_n$ as matrix 2-D prefilter that is used. This matrix prefilter is designed to shorten the length of channel as well to suppress the interference (should maximise the SINR at the output of prefilter). MMSE based prefilter can be used to solve the probelm. For example, [1],[12] explores the channel-shortening prefilters for MIMO channels. We use prefilter designed in [1], but it gives a coloured impairment at the output of filter. On passing the 2×1 received signal

vector $\mathbf{y}_n$ through the 2-D WL MMSE filter($\mathbf{F}$), the resultant output can be written as

$$\mathbf{V}_n = \sum_{k=0}^{N_b} \mathbf{B}_k \, \mathbf{a}_{n-k} + \mathbf{e}_n. \tag{B.2}$$

where $\mathbf{e}_n$ denotes 2×1 impairment vector and $\mathbf{B}_n$ is the resultant matrix channel obtained by 2-D filtering.We use ITC constraint in [1] to get monic, causal, minimum-phase resultant matrix channel $\mathbf{B}_n$.

REFERENCES

[1] N. Al-Dhahir. FIR channel-shortening equalizers for MIMO ISI channels. *Communications, IEEE Transactions on*, 49(2):213 –218, Feb 2001.

[2] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate (corresp.). *Information Theory, IEEE Transactions on*, 20(2):284 – 287, Mar 1974.

[3] L.G. Barbero, T. Ratnarajah, and C. Cowan. A low-complexity soft-MIMO detector based on the fixed-complexity sphere decoder. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 2669 –2672, April 2008.

[4] L.G. Barbero and J.S. Thompson. Fixing the Complexity of the Sphere Decoder for MIMO Detection. *Wireless Communications, IEEE Transactions on*, 7(6):2131 –2142, June 2008.

[5] Won-Joon Choi, R. Negi, and J.M. Cioffi. Combined ML and DFE decoding for the V-BLAST system. In *Communications, 2000 IEEE International Conference on*, volume 3, pages 1243 –1248 vol.3, 2000.

[6] A.P. Clark and M. Clayden. Pseudobinary Viterbi detector. *Communications, Radar and Signal Processing, IEE Proceedings F*, 131(2):208 –218, April 1984.

[7] A. Duel-Hallen and C. Heegard. Delayed decision-feedback sequence estimation. *Communications, IEEE Transactions on*, 37(5):428 –436, May 1989.

[8] M.V. Eyuboglu and S.U.H. Qureshi. Reduced-State Sequence Estimation with set partitioning and decision feedback. *Communications, IEEE Transactions on*, 36(1):13 –20, Jan 1988.

[9] G.D. Forney, Jr. Maximum-Likelihood sequence estimation of digital sequences in the presence of intersymbol interference. *Information Theory, IEEE Transactions on*, 18(3):363 – 378, May 1972.

[10] G.D. Forney, Jr. The Viterbi Algorithm. *Proceedings of the IEEE*, 61(3):268 – 278, March 1973.

[11] G. Foschini. A reduced state variant of maximum likelihood sequence detection attaining optimum performance for high signal-to-noise ratios. *Information Theory, IEEE Transactions on*, 23(5):605 – 609, Sep 1977.

[12] A. Hafeez, R. Ramesh, and D. Hui. Maximum snr prefiltering for mimo systems. In *Signal Processing Advances in Wireless Communications, 2005 IEEE 6th Workshop on*, pages 186 – 190, June 2005.

[13] B. Hassibi. An efficient square-root algorithm for BLAST. In *Acoustics, Speech, and Signal Processing, 2000. ICASSP '00. Proceedings. 2000 IEEE International Conference on*, volume 2, pages II737 –II740 vol.2, 2000.

[14] K.S. Karthik, Kiran Kuchi, and Bhaskar Ramamurthi. A note on reduced set sequence detection for low complexity, near-ML MIMO detection. *submitted to Communications Letters, IEEE*, 2011.

[15] R. Koetter, A.C. Singer, and M. Tuchler. Turbo equalization. *Signal Processing Magazine, IEEE*, 21(1):67 – 80, Jan. 2004.

[16] Kuei-Chiang Lai and Li-Wei Lin. Low-complexity adaptive tree search algorithm for MIMO detection. *Wireless Communications, IEEE Transactions on*, 8(7):3716 –3726, July 2009.

[17] D. Le Ruyet, T. Bertozzi, and B. Ozbek. Breadth first algorithms for APP detectors over MIMO channels. In *Communications, 2004 IEEE International Conference on*, volume 2, pages 926 – 930 Vol.2, June 2004.

[18] Won Lee and F. Hill. A Maximum-Likelihood Sequence Estimator with Decision-Feedback Equalization. *Communications, IEEE Transactions on*, 25(9):971 – 979, Sep 1977.

[19] S. Lin and D.J. Costello. *Error Control Coding: Fundamentals and Applications*. Pearson Education India, 1983.

[20] S. Loyka and F. Gagnon. Performance analysis of the V-BLAST algorithm: an analytical approach. *Wireless Communications, IEEE Transactions on*, 3(4):1326 – 1337, July 2004.

[21] M. Magarini and A. Spalvieri. Coset detection in MIMO systems based on spatial multiplexing. *Communications Letters, IEEE*, 10(5):390 – 392, May 2006.

[22] R. Meyer, W.H. Gerstacker, F. Obernosterer, M.A. Ruder, and R. Schober. Efficient receivers for gsm muros downlink transmission. In *Personal, Indoor and Mobile Radio Communications, 2009 IEEE 20th International Symposium on*, pages 2399 –2403, Sept. 2009.

[23] R. Meyer, W.H. Gerstacker, R. Schober, and J.B. Huber. A single antenna interference cancellation algorithm for increased GSM capacity. *Wireless Communications, IEEE Transactions on*, 5(7):1616 –1621, July 2006.

[24] J.C. Olivier and W. Kleynhans. Single antenna interference cancellation for synchronised GSM networks using a widely linear receiver. *Communications, IET*, 1(1):131 –136, February 2007.

[25] S. Qureshi and E. Newhall. Adaptive receiver for data transmission over time-dispersive channels. *Information Theory, IEEE Transactions on*, 19(4):448 – 457, Jul 1973.

[26] P. Robertson, E. Villebrun, and P. Hoeher. A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain. In *Communications, 1995 IEEE International Conference on*, volume 2, pages 1009 –1013 vol.2, Jun 1995.

[27] Antonio M. Tulino and Sergio Verdú. Random matrix theory and wireless communications. *Commun. Inf. Theory*, 1:1–182, June 2004.

[28] G. Ungerboeck. Channel coding with multilevel/phase signals. *Information Theory, IEEE Transactions on*, 28(1):55 – 67, Jan 1982.

[29] A.J. Viterbi. An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes. *Selected Areas in Communications, IEEE Journal on*, 16(2):260 –264, Feb 1998.

[30] E. Viterbo and J. Boutros. A universal lattice code decoder for fading channels. *Information Theory, IEEE Transactions on*, 45(5):1639 –1642, Jul 1999.

[31] P.W. Wolniansky, G.J. Foschini, G.D. Golden, and R.A. Valenzuela. V-BLAST: an architecture for realizing very high data rates over the rich-scattering wireless channel. In *Signals, Systems, and Electronics, 1998. ISSSE 98. 1998 URSI International Symposium on*, pages 295 –300, Sep-Oct 1998.

[32] Xu Zhu and R.D. Murch. Performance analysis of maximum likelihood detection in a MIMO antenna system. *Communications, IEEE Transactions on*, 50(2):187 –191, Feb 2002.

LIST OF PAPERS BASED ON THESIS

1. Mahesh Gupta, Rakesh Malladi, Kiran Kuchi, R David Koilpillai ‘SAIC Receiver Algorithms for VAMOS Downlink Transmission’ to be submitted to *ISWCS 2011*

2. Rakesh Malladi, Karthik K S, Kiran Kuchi, R David Koilpillai ‘Soft Decision Algorithms for Detection of MIMO Systems Using Set Partitioning’ to be submitted to *IEEE Transactions on Wireless Communications*

3. Rakesh Malladi, Kiran Kuchi, R David Koipillai ‘Forward/Backward Soft Detection Algorithm for MIMO Systems in Flat Fading Channels’ to be submitted to *International Conference on Communications, 2012*