LOW COMPLEXITY TURBO EQUALIZATION FOR CODED INTERSYMBOL INTERFERENCE CHANNELS

FOO SIANG MENG

NATIONAL UNIVERSITY OF SINGAPORE

2007

LOW COMPLEXITY TURBO EQUALIZATION FOR CODED INTERSYMBOL INTERFERENCE CHANNELS

FOO SIANG MENG

(B.Eng. (Hons.), NUS)

A THESIS SUMBITTED

FOR THE DEGREE OF MASTER OF ENGINEERING

DEPARTMENT OF

ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2007

Acknowledgements

I would like to thank Prof. Marc Andre Armand for his invaluable guidance throughout my entire preparation for this thesis. I would also like to show my appreciation to Dr. George Mathew who has painstakingly taught me most of the technical stuff on the rich field of equalization.

I would also like to express my greatest gratitude to Dr. Qin Zhiliang who has taught me on turbo equalization for the past three and a half years.

I also wish to thank all my friends who have shown great support to me over the years, in particular, Darren Chin and family, Jaron Chang and Francis Tang.

Finally yet importantly, I would like to thank Data Storage Institute and Dr. Cai Kui for the great technical and research support rendered for the period of my M.Eng. studies.

Contents

Summary

The application of the turbo principle in mitigating the detrimental effects of intersymbol interference (ISI) channels has gained widespread interest for the past decade due to their remarkable bit error rate (BER) performances. In this thesis, we consider turbo equalization over coded ISI channels, wherein extrinsic information of transmitted code bits is exchanged iteratively between a soft-input/soft-output (SISO) channel equalizer and an outer decoder. The exact implementation of the SISO channel equalizer is usually based on the Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm, which has a computational complexity growing exponentially with channel memory length. Practical implementation thus requires low complexity alternatives to the BCJR algorithm with comparable BER performances.

In the literature, many low complexity SISO channel equalizers have been proposed. The proposed schemes generally fall into two categories, namely, trellis-based equalization algorithms and filter-based equalization algorithms. In this thesis, we propose a novel approach to reduce the computation complexity of the SISO channel equalizer while maintaining insignificant performance degradation as compared to the BER-optimal BCJR implementation. The proposed method is based on the observation that we may view the SISO channel equalization task as a combinatorial optimization problem with a finite set of binary-constrained solutions. Heuristic search methods which were previously employed to solve large-scale combinatorial optimization problems could thus be adapted for use as an equalization algorithm.

Numerous heuristic search algorithms exist in the literature such as the local search (LS), evolutionary programming (EP), genetic algorithm (GA), greedy algorithm, etc. In this thesis, we focus on studying the application of the LS algorithms to turbo equalization due to its superior performance when a relatively reliable initial solution is available from the outer decoder in the previous iteration. BER performance comparisons and computational complexity analysis reveal that the proposed heuristic-based LS turbo equalizer is a viable alternative to the trellis-based BCJR turbo equalizer, the filter-based minimum-mean-squared-error (MMSE) turbo equalizer and even their low complexity variants.

To understand the reasons behind the superior performance of the proposed heuristic-based LS turbo equalizer, analysis is carried out through the use of the EXtrinsic Information Transfer (EXIT) chart. Through EXIT chart analyses, numerous interesting insights on the heuristic-based LS turbo equalizer are unearthed. In particular, we observe indirectly through the EXIT chart that our proposed heuristic-based LS turbo equalizer is more robust against the detrimental effects of imperfect channel knowledge as compared to both the trellis-based BCJR turbo equalizer and the filter-based MMSE turbo equalizer. Consequently, simulation results obtained show that the proposed heuristic-based LS turbo equalizer has the best BER performance as compared to the other two turbo equalizer approaches. This important finding suggests that, in practice, our proposed heuristic-based LS turbo equalizer could be a serious contender to the other existing techniques where perfect channel knowledge is usually unavailable to the receiver.

List of Tables

List of Figures

List of Abbreviations

AWGN	additive white Gaussian noise
ISI	intersymbol interference
SISO	soft-input/soft-output
LLR	log-likelihood ratios
APP	*a posteriori* probability
EXIT	EXtrinsic Information Transfer
SNR	signal-to-noise ratio
BER	bit-error rate
RSC	recursive systematic convolutional
BCJR	Bahl-Cocke-Jelinek-Raviv
FIR	finite impulse response
RBF	radial basis function
SOVA	soft-output Viterbi algorithm
FER	frame-error rate
ZF	zero forcing
MMSE	minimum mean-square error
FC-BCJR	full complexity BCJR
SW-BCJR	sliding-window BCJR
LS	local search

MAP	maximum *a posteriori*
PDF	probability density function
CIR	channel impulse response
BPSK	binary-phase-shift-keying
WMF	whitened matched filter
DE	density evolution

Chapter 1

Introduction

In this chapter, we first give a brief introduction to the concept of turbo equalization, after which a survey of some of the existing equalization algorithms is provided in Section 1.2. Due to the iterative nature of the turbo equalizer where repeated equalization and decoding have to be carried out on the same set of transmitted data bits for improved performance, it would be important to grasp an understanding of the complexity concerns for a turbo receiver. A discussion on the complexity concerns will be carried out in Section 1.3. Thereafter, some low complexity equalization algorithms are briefly mentioned. In Section 1.5, the motivation and a summary of the work presented in this thesis are highlighted. Finally, the organization of this thesis is given.

1.1 Introduction to Turbo Equalization

The discovery of turbo codes by Berrou *et at*. [1] in 1993 with iterative decoding has ignited significant research interest due to their remarkable near-capacity performance over memoryless additive white Gaussian noise (AWGN) channels. Soon after the introduction of turbo codes and its corresponding receiver called the turbo decoder, it was recognized that iterative decoding could be incorporated as a general methodology for advanced receiver design. The term “turbo processing” was subsequently coined in [2] to describe the general strategy for joint decoding and detection.

In 1995, Douillard *et al.* proposed the concept of "turbo equalization", which extends from the application of turbo processing in joint equalization and channel decoding [3]. Specifically, Douillard *et al.* presented an iterative receiver called the turbo equalizer that is capable of mitigating the effects of intersymbol interference (ISI) due to multi-path effects on convolutionally-coded Gaussian channels, provided that the channel impulse response is known to the receiver.

In essence, the implementation of an iterative receiver (also known as turbo receiver) involves the use of detection/decoding modules that employ soft-input/soft-output (SISO) algorithms [4]-[8]. Soft decision values, usually expressed in the form of log-likelihood ratios (LLR), are computed by one of the modules and thereafter passed to the other. The estimates of the data bits transmitted are then refined by sharing information between the two SISO modules in an iterative fashion. More specifically, the extrinsic output of one SISO module can be used as *a priori* information by the next SISO module.

The reason for the passing of soft information from one module to the other is to ensure no information is lost between the SISO modules. Beside this, the passing of only extrinsic information from one SISO module to the next is also an important requirement for any receiver employing turbo processing scheme. This is to prevent "positive feedback" problems which may destabilize the information passed. Essentially, extrinsic information refers to new information that is derived by a particular SISO module at a particular stage of iteration. From a general intuitive perspective, the extrinsic information from a particular SISO module could be obtained by taking the *a posteriori* probability (APP) LLR generated by the SISO module after each stage of detection/decoding and subtracting all the inputs that are used in computing this APP LLR.

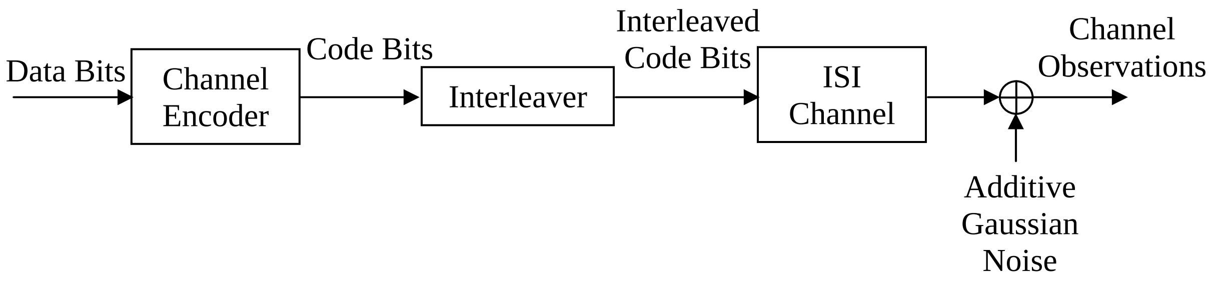

Figure 1.1: Transmitter section of a data transmission model

In this thesis, the transmitter section of the data transmission system considered is shown in Figure 1.1 where the channel encoder and the ISI channel are separated by an interleaver. The function of the pseudo-random interleaver, as the name implies, is to rearrange or scramble a block of code bits from the channel encoder in a pseudo-random manner.

In Figure 1.2 on the next page, the structure of the original turbo equalizer proposed by Douillard *et al.* [3] is presented. Here, the generation of extrinsic information to be used as *a priori* information for the next SISO module is explicitly shown by the inclusion of adders. In the feedforward path, the equalizer and channel decoder are separated by a de-interleaver which performs the reverse operation of the interleaver, *i.e.*, to undo the scrambling done by the interleaver. The distinct feature of a turbo equalizer is the presence of a feedback path from the decoder to the equalizer to allow iterative exchanges of refined estimates. The interleaver used in the feedback path is identical to that utilized in the transmitter end.

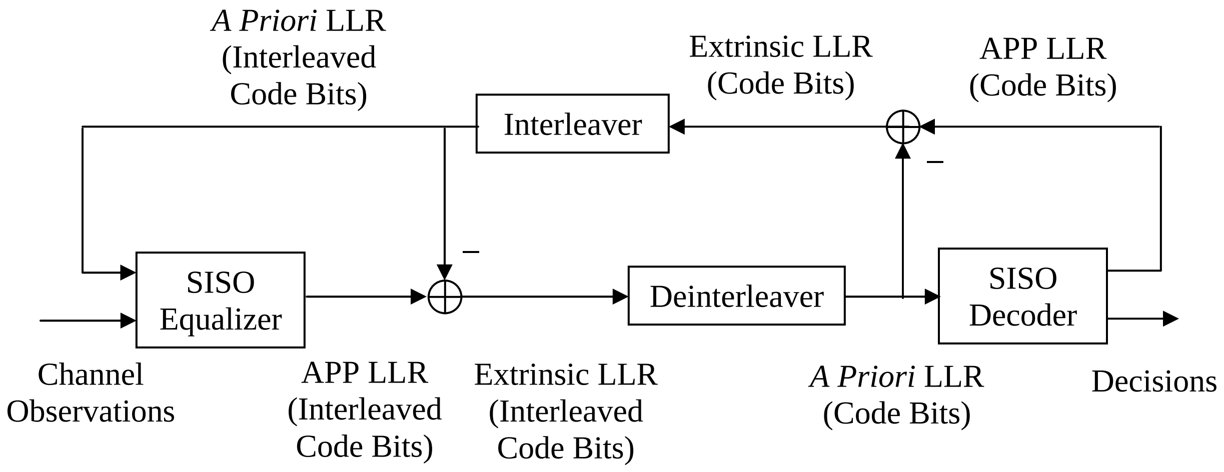
A Priori LLR (Interleaved Code Bits)
Interleaver
Extrinsic LLR (Code Bits)
APP LLR (Code Bits)
–
–
SISO Equalizer
Deinterleaver
SISO Decoder
Channel Observations
APP LLR (Interleaved Code Bits)
Extrinsic LLR (Interleaved Code Bits)
A Priori LLR (Code Bits)
Decisions

Figure1.2: A general receiver model performing turbo equalization

The presence of the interleaver/deinterleaver is crucial in decorrelating the error events introduced by the equalizer between neighboring bits. These error "bursts" are hard to deal with, in particular, by a channel decoder which employs convolutional codes. Besides decorrelating the error events, the presence of the interleaver and de-interleaver at the receiver end also serve to provide independence, at least locally and for several iterations, between neighboring LLR estimates. This independent assumption is a critical property that is utilized in the modeling of the LLR estimates, which subsequently allows an open-loop simulation of the respective SISO modules in the turbo receiver to be carried out. Such open-loop simulations on the SISO modules form the basis for the generation of EXtrinsic Information Transfer (EXIT) chart [9][10], which is an important tool for gaining insights into the operation of a receiver employing turbo processing.

The iterative exchanges of refined estimates are to be carried out until a prescribed number of iterations are reached. This prescribed number of iterations could be obtained through the use of the EXIT chart and it should be varied accordingly for different signal-to-noise ratio (SNR) values for optimum bit-error rate (BER) performance. In general, there is an inverse relationship between the number of iterations for optimum BER performance and the SNR of the received signal. In other words, a decrease in the SNR of the received signal would lead to an increase in the number of iterations required for optimum BER performance and vice versa. It is to be noted that this generalization is only true provided that the SNR is above a certain value called the decoding threshold. The EXIT chart will be formally presented in Chapter 4 where the above relationship will be discussed in detail.

In this thesis, the investigation into the performance of a turbo equalizer employing different equalization algorithms is carried out. No studies are carried out with respect to an optimum construction of the interleaver; pseudo-random interleaving is assumed throughout the whole thesis. The channel code used for error correction is a recursive systematic convolutional (RSC) code and its corresponding decoder employs the BER optimum Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [11] which operates on the code constraints specified indirectly by the code trellis.

Various equalization algorithms suited for used in a turbo equalization scheme are available in the literature. In the next section, we provide a review of existing equalization techniques with an emphasis on some of the more well-known algorithms. These well-known equalization algorithms will also form the basis for comparison, in terms of BER performance, etc., when we present our proposed equalization algorithm based on heuristic search methods in Chapter 3.

1.2 Introduction to Existing Equalization Algorithms

The derivation of differing equalization algorithms are primarily motivated by the numerous ways of visualizing the ISI channel or the channel equalization problem.

From a signal processing perspective, an ISI channel essentially behaves as a band-limited linear filter that introduces bandwidth restriction on the transmitted bits. This bandwidth restriction introduces superpositioning of the transmitted bits' pulses when the bandwidth of the transmitted bits is larger than the bandwidth of the channel. Typically, a finite impulse response (FIR) filter with real/complex coefficients is used to represent an ISI channel.

From another school of thought in the realm of coding theorists, an ISI channel could be visualized as a rate-1 non-recursive convolutional encoder with additive and multiplicative operations defined over the field of real numbers. Since a convolutional encoder is simply a finite state machine, a trellis diagram could be used to describe the behavior of the ISI channel.

With these two representations of an ISI channel, two well-known classes of equalization techniques evolved naturally. One equalization methodology compensates for ISI using a filter-based algorithm [5][12] while the other aims to reduce the detrimental effects of ISI using a trellis-based algorithm [11][13].

In recent years, some researchers [14][15] have came up with a new method in tackling the channel equalization task by viewing it as a received phasor classification problem where the phasors may become linearly non-separable due to channel disturbances. This concept extends from the field of artificial intelligence where they utilize the well-known radial basis function (RBF) used in neural networks to design a nonlinear reduced-complexity Jacobian RBF-assisted equalizer. This methodology came about from the realization that the RBF network has an equivalent structure to the so-called optimal Bayesian equalization solution and can provide the conditional density function of the transmitter bits as soft-output for use by the channel decoder.

In this thesis, as a form of background review, we narrow down our scope to cover the more well-known and established algorithms, namely, the trellis-based and filter-based equalization algorithms. For a better understanding of the RBF-assisted equalization technique, we refer the interested reader to [14][15].

1.2.1 Trellis-Based Equalization Algorithm

The task of equalization can be formulated in terms of a first-order finite-state Markov random process observed through additive noise. Any finite-state Markov process can be represented by a trellis diagram. With this representation, the problem of estimating the APP of the state sequence of a Markov source observed through noise has two well-established trellis-based solutions, namely the BCJR algorithm [11] and the soft-output Viterbi algorithm (SOVA) [13]. Once the state sequence is estimated, it is a trivial task to determine the transmitted code bits associated with it.

Essentially, both the trellis-based algorithm exploits two fundamental properties of the underlying Markov process. The first property is that Markov process is first order. The second property is that the conditional probability of a particular observation through white noise, i.e., the received channel observation at a particular time instance, given the entire state sequence is equal to the conditional probability of the observation given only the state transition that is related to that time instance. These two properties along with the rules of conditional probability and the assumption that the channel noise is independent, allow a recursive and computationally efficient manner for computing the extrinsic LLR values for each bits.

The BCJR algorithm was formally presented in 1974 as an alternative to the Viterbi algorithm, introduced in 1967, for the decoding of convolutional codes. It is noted that the SOVA is an extension of the classical Viterbi algorithm for providing the reliability information of bit estimates.

Basically, the BCJR algorithm is an optimum algorithm in minimizing the BER whereas the SOVA is an optimum algorithm in minimizing the frame error rate (FER). Hence, the SOVA is generally used for sequence estimation whereas the BCJR algorithm is used for symbol-by-symbol estimation. Besides differing in their optimality criterion, another key difference between these two algorithms is that the states estimated by the SOVA must form a connected path through the trellis, whereas that estimated by the BCJR algorithm need not be connected.

From the numerous results available in the literature, it is well-known that the BCJR algorithm has a better BER performance when compared to the SOVA. Hence, only the BCJR algorithm will be presented as a representative of the class of trellis-based equalization algorithm for comparisons to our proposed heuristic-based equalization algorithm to be made.

1.2.2 Filter-Based Equalization Algorithms

The trellis-based equalization algorithms are optimal either in terms of minimizing BER or FER. However, they have large computational complexities and hence are unfavorable for use in an actual system implementation, especially in a turbo equalization setup.

As such, researchers have been exploring sub-optimal equalization methods that typically consist of linear processing of the received signal through the use of linear filters. The parameters of these filters can be selected using a variety of optimization criteria, such as the zero forcing (ZF) or the minimum mean squared error (MMSE) criteria [5].

The ZF design approach essentially, as the name implies, forces all the unwanted ISI to zero. By viewing the frequency response of the filter designed using the ZF criteria, it is observed that the equalizer frequency response is basically an inversion of that of the channel frequency response. Hence the concatenation of the ISI channel and the ZF equalizer results in an overall flat frequency response as seen from the output of the ZF equalizer which translates to zero ISI in the time domain. As additive noise is present at the input of the equalizer, this inversed frequency response will lead to undue noise enhancement if the ISI channel frequency response has near-zero spectral content in some frequency regions. As such, the ZF design approach is not widely used.

A more general approach which overcomes the noise enhancement problem in ZF equalizers is through the use of the MMSE design criteria. In designing the filter, the MMSE approach takes into account both the ISI as well as the additive noise perturbing the received signal. A compromise is stricken by the MMSE approach which avoids infinite noise enhancement at the expense of some residual ISI at the output of the equalizer. A well-known FIR filter which employs the MMSE design criteria is the Wiener filter.

Non-linear filters could also be implemented for equalization purposes. Well-known non-linear filters that are used for equalization includes decision feedback equalizers [16] and the Kalman filter [12]. We refer the interested reader to [12][16] for a more in-depth understanding of their implementation.

In this thesis, we showcase the MMSE equalization algorithm for comparisons to our proposed heuristic-based equalization algorithm to be made. This is because, in our point of view, the MMSE equalization algorithm is a relatively simple yet effective technique in mitigating the effects of ISI, among all other filter-based equalizer design approaches.

1.3 Complexity Concerns for a Turbo Receiver

The remarkable performance of turbo equalization scheme comes at the cost of large decoding delay and high complexity, especially so when equalization and decoding are performed using the BER optimum trellis-based BCJR algorithm.

The BCJR algorithm essentially operates as two Viterbi algorithms with one running in the forward direction and the other running in the backward direction. The forward Viterbi algorithm is used to compute forward metrics whereas the backward Viterbi algorithm is used to compute backward metrics. The final LLR output is then obtained by appropriately combining these two metrics. The details of the BCJR algorithm will be described in Chapter 2. Due to the requirements for the computation of the backward metrics, it is necessary for an entire sequence to be received before the equalization or decoding operations could commence. As such, large decoding delay and huge memory requirements for storing the various metrics are to be expected.

Moreover, as the BCJR algorithm operates on the code/channel trellis whose number of trellis states has an exponential relationship to the code/channel memory length, the BCJR thus has an exponential complexity when computing the required metrics for each particular bit.

In addition, to obtain good performance, the equalization and decoding tasks have to be iterated a few times on each set of data bits, thereby further exacerbating the latency and complexity problems described above.

1.4 Low Complexity Equalization Algorithms

In recent years, after researchers had fully appreciated the potential of turbo processing for reliable communications, numerous efforts have been made in the complexity reduction of such iterative algorithms [17]-[24]. One such focus is the simplification of the equalization task. In the literature, many low complexity SISO channel equalizers have been proposed within the two main categories of equalizers.

Low complexity variants of the trellis-based equalization algorithm include the M-BCJR algorithm [18][19][21], the T-BCJR algorithm [21] and the trellis-splicing algorithm [20]. The general approach adopted in these algorithms is based on simplifying the channel trellis to reduce the computational requirements of the original full complexity BCJR (FC-BCJR) algorithm. The M-BCJR algorithm is derived to ease computational load by reducing the number of channel states at each trellis stage while the T-BCJR algorithm achieves low complexity by decreasing the number of paths searched in the trellis. The trellis-splicing algorithm operates at low latency simply by shortening the channel trellis based on early-decoding of reliable bits. In [22], a sliding-window BCJR algorithm (SW-BCJR) is proposed which allows the BCJR algorithm to operate on a fixed memory span and thereafter output the “smoothed” probability distributions [22] following a given delay. This alleviates the need to receive an entire sequence, thereby reducing decoding delay.

In the filter-based equalizer category, it should be mentioned that this category is already one of the low complexity alternatives to the trellis-based equalizer. As an illustration, in the Master’s thesis by Michael Tühler [24], he proposes a time-varying MMSE equalizer with quadratic computational complexity (in the equalizer filter length) by using a recursive algorithm to efficiently compute the MMSE equalizer filter coefficients. A significant saving in computational complexity is observed when compared with the FC-BCJR equalizer which has exponential computational complexity (in the ISI channel memory length). Then in [5], a further reduction in the complexity of the MMSE equalizer is attained by implementing two time-invariant filters whose coefficients are similarly obtained using the MMSE design criterion, albeit at different initial condition assumptions. This hybrid approach is shown to attain the performance of the time-varying MMSE equalizer by performing the iterative equalization tasks through optimally switching between the two filters. Basically, an optimization criterion is set in this hybrid approach to determine which filter is to be used at each iterations. Neglecting the computation of the optimization criterion, a linear complexity in terms of the equalizer and/or channel memory length is achieved by this hybrid approach.

1.5 Motivation and Summary of Present Work

Motivated by the need for low complexity alternatives while maintaining comparable BER performance to the FC-BCJR algorithm, we propose a novel approach to the channel equalization problem based on heuristic-search methods.

The work in this thesis consists of two parts. First, we propose a heuristic-based equalization approach that utilizes local search (LS) algorithms. LS algorithms are the most widely used heuristic methods [25]-[36] for solving large-scale combinatorial optimization problems and, in general, all other heuristic search methods can be considered as an extension of LS.

In the second part, we utilize the EXIT chart [9] to analyze the LS-based turbo equalizer that we proposed by monitoring the evolution of the extrinsic information as the iterations proceed. This analysis provides enlightening insights into the operation of the LS-based equalizers in a turbo equalization setup and even suggests the feasibility of our proposed equalizer in actual system implementation, where perfect channel knowledge is usually unavailable at the receiver end.

1.5.1 Heuristic–Based Equalization Algorithm Utilizing Local Search

The proposed heuristic search method is based on the observation that we may view the SISO channel equalization task as a combinatorial optimization problem with a finite set of binary-constrained solutions.

Unlike most optimization problems, the channel equalization problem in a turbo equalization setup features a relatively good initial estimate (for example, delivered by other equalizer or fed back from the channel decoder) as a starting point for optimization to begin. In particular, the proposed scheme can be decoupled into several successive stages.

In the first stage, we obtain tentative hard estimates of the transmitted code bits as an initial solution for the optimization problem based on the *a priori* LLR fed back from the channel decoder in the previous iteration. In the second stage, we employ heuristic search methods to optimize an objective function based on the maximum *a posteriori* (MAP) metric and derive a candidate list that consists of all potential solutions. Finally, we produce the *a posteriori* LLR of the code bits by restricting the LLR calculation to the derived candidate list, which can then be used as soft-input for the channel decoder after subtracting the *a priori* LLR from it. In this thesis, we utilize the heuristics based on LS algorithms. We nevertheless point out that other heuristic search methods, such as simulated annealing, evolutionary programming, genetic algorithm, greedy algorithm, etc., could also be employed.

From the simulation results obtained, we show that the proposed receiver utilizing the LS-based algorithm has a negligible BER performance loss when compared to the FC-BCJR equalizer, especially so at moderate-to-high SNR, while significantly reducing the overall computational complexity to a square order magnitude in terms of the ISI channel memory length. Depending on the type of LS algorithm (i.e. 1-Opt LS or *k*-Opt LS) used and the severity of the ISI introduced, the proposed LS-based equalizer is also shown to outperform the equalizer designed using the MMSE criterion. Specifically, for severe ISI condition, the proposed *k*-Opt LS outperforms the time-varying MMSE equalizer with approximately 0.3 dB improvements at a BER of 10^{-3}.

1.5.2 EXIT Chart Analysis of Local Search-Based Turbo Equalizers

The performance of a receiver design is conventionally quantified using the BER metric which could be obtained either via time-consuming simulations or through derivation to arrive at an analytical expression. For a receiver in a turbo setup, such analytical derivation is usually difficult due to the presence of the interleaver. Moreover, in the context of understanding the performance of a turbo receiver or to be specific, a turbo equalizer, the BER metric is usually of limited capability to reflect the iterative nature of the equalization and decoding processes. Essentially, the BER metric could only be viewed as a summary of the performance of a turbo equalizer design.

The EXIT chart, a semi-analytical tool originally pioneered by Stephan ten Brink [9] for the analysis of turbo codes, is a powerful technique to analyze the behavior of a turbo receiver. In brief, the EXIT chart basically tracks the evolution of the extrinsic LLR's probability density function (PDF) indirectly by measuring an information theoretic quantity called mutual information. The EXIT chart is derived by carrying out open-loop simulations on the respective SISO modules to obtain the respective transfer characteristics (also known as transfer function) of both the equalizer and the channel decoder. The transfer characteristics of both the equalizer and channel decoder are then plotted on a single figure to obtain the EXIT chart. From the EXIT chart, the equalization and decoding steps can be visualized explicitly by following a staircase-like trace that is bounded by the two transfer characteristics. This trace is known as the predicted trajectory.

In this thesis, we utilize EXIT chart to carry out an analysis of the proposed LS-based turbo equalizer. As the EXIT chart assumes a Gaussian distribution when modeling the *a priori* input to the respective SISO modules, we found that this assumption does not really hold for a LS-based turbo equalizer. As such, we devise a general technique to obtain the EXIT chart for our LS-based turbo equalizer. The original EXIT chart is then compared to our proposed EXIT chart. Through asymptotic simulations carried out (i.e., very large frame size of typically at least 10^5 data bits per frame), it is found that our proposed EXIT chart is able to predict the performance of the LS-based turbo equalizer more accurately than the original EXIT chart in terms of the number of iterations required for convergence and the associated BER at each iteration. For example, at an SNR of 4.5 dB, the proposed EXIT chart reflects accurately the attainment of fixed point with 15 iterations whereas the original EXIT chart deviate significantly from the actual case by being overly-optimistic with 7 iterations fewer. The decoding threshold obtained from the proposed EXIT chart also agrees with that obtained using density evolution [37]-[39]. This further indicates the accuracy of our proposed EXIT chart in its asymptotic predictions of the LS-based turbo equalizer.

1.6 Organization of Thesis

The rest of the thesis is organized as follows. In Chapter 2, we give a description of some of the equalization algorithms commonly used in a turbo equalizer. These equalization techniques would form the basis for comparison when we present our proposed LS-based equalizer. The focus of this chapter is the trellis-based BCJR algorithm and the filter-based MMSE algorithm. Certain key properties of the BCJR algorithm will be highlighted since this algorithm is also used for the decoding of the RSC code employed in this thesis. The corresponding low complexity variants from these two categories of equalization algorithms will also be briefly mentioned.

In Chapter 3, we present the details of the heuristic-based turbo equalizers utilizing LS algorithm. We first give an introduction to heuristic search algorithm to enable the readers to appreciate the nature of such technique. Following that, in Section 3.2, we formulate the equalization problem formally before addressing this problem from a heuristic approach utilizing LS. Variants of LS, namely 1-Opt and *k*-Opt algorithms will be described. A low complexity implementation of the LS algorithm is then presented in Subsection 3.2.3. After illustrating the implementation details, we carried out some analysis on the LS-based equalizer by looking at its computational complexity and its decoding threshold. Finally, the simulation results are presented together with a discussion on these results in Section 3.5.

The EXIT chart analysis of the LS-based equalizers is exemplified in Chapter 4. We first give an introduction to familiarize the readers with the EXIT chart after which an exposition on the principles of EXIT chart in provided in Section 4.2. Then, the original EXIT chart is presented in Section 4.3 and is shown to be inaccurate in predicting the asymptotic behavior of our LS-based turbo equalizer. A new EXIT chart is proposed in Section 4.4 for our LS-based turbo equalizer and subsequently verified in Section 4.5 to be more accurate than the original EXIT chart in the various predictions. Then, in Section 4.5.6, we carried out further investigation into the LS-based equalizer by observing some key points on the EXIT chart as the SNR varies and under different conditions. This exploration leads to an interesting discovery on our proposed LS-based turbo equalizers, which hinted its robustness against BER performance degradation in situation where imperfect channel impulse response (CIR) knowledge is given to the receiver. Finally, the thesis is concluded in Chapter 5 with some comments on possible directions for further work.

Chapter 2

Survey of Equalization Algorithms for Turbo Equalizers

In this chapter, a selected review on two well-established categories of equalization algorithms, namely, the trellis-based equalization algorithm and the filter-based equalization algorithm are presented. We begin this chapter with a description of the system model so as to provide a unified framework for the presentation of the various equalization algorithms outlined in this thesis. This section will also familiarize the readers in terms of common symbols and notations used while describing the various algorithms. An exposition on the trellis-based equalization algorithm will be presented in Section 2.2. Here, the full complexity BCJR (FC-BCJR) and some of its low complexity variants will be described. The low complexity variants of the FC-BCJR presented here are the M-BCJR and the SW-BCJR. Following that, in Section 2.3, we touch on the filter-based turbo equalizers implemented based on the MMSE design criterion. Low complexity implementation of the MMSE turbo equalizers based on time-invariant filters will also be described. A comparison of the computational complexity between these two categories of equalization techniques is carried out in Section 2.4, after which selected simulation results are presented. Finally, the chapter concludes with a summary in Section 2.6.

2.1 System Model

The transmitter section in consideration is a serial concatenated system consisting of a single RSC encoder and an ISI channel separated by an interleaver. The schematic of this system model is similar to that depicted in Chapter 1 and is replicated here in Figure 2.1.

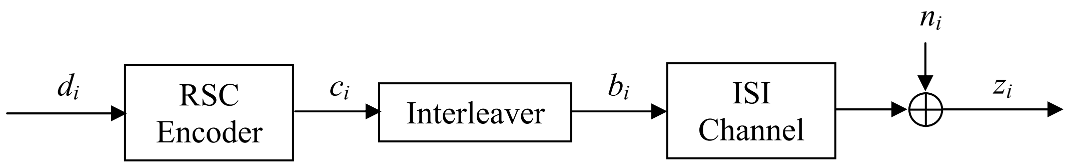

Figure 2.1: Discrete-time equivalent noise-whitened channel model

The binary data bits d_i are first encoded by an RSC encoder and passed to a random interleaver. The interleaved code bit stream b_i is then binary-phase-shift-keying (BPSK) modulated and transmitted over a band-limited ISI channel perturbed by AWGN which is represented as $n(t)$ with single-sided power spectral density level N_o. The received signal can be expressed as

$$r(t) = \sum_{i=-\infty}^{+\infty} b_i g(t - iT_b) + n(t) \tag{2.1}$$

where $g(t) = p(t) \otimes c(t)$. Here, $p(t)$ is the pulse shape filter at the transmitter end, $c(t)$ refers to the continuous-time channel impulse response and $\otimes$ denotes the linear convolution operator. The code bit interval is specified in (2.1) as T_b.

To convert the continuous-time signal $r(t)$ to its equivalent discrete form, we apply a matched filter $g(-t)$ as the receiver front-end together with a sampler operating at a bit rate of $1/T_b$. The corresponding discrete-time signal is thus given by

$$y_i = \sum_{j=-m}^{m} h_j b_{i-j} + w_i \tag{2.2}$$

where $\{h_j : j = -m, -m + 1,\ldots, m\}$ is the discrete-time impulse response of the equivalent channel and w_i denotes additive Gaussian noise. Here, we assume that the ISI span is finite, i.e., $h_j = 0$ for $|j| > m$.

The application of the matched filter $g(-t)$ leads to correlations in the noise sequence at the output of the matched filter, resulting in a colored-noise channel model. It is possible to further process the output of the matched filter by applying a discrete-time noise-whitening filter [40] to attain a corresponding noise-whitened channel given by

$$z_i = \sum_{j=0}^{m} f_j b_{i-j} + n_i \tag{2.3}$$

where $\{f_j : j = 0, 1,\ldots, m\}$ denotes the tap coefficients of the noise-whitened channel model and n_i is the discrete-time AWGN samples. From (2.3), it is observed that the application of the noise-whitening filter leads to a causal description of the discrete-time channel model. The cascade of the matched filter, the sampler and the noise-whitening filter is called the whitened matched filter (WMF).

The relationship between the coefficients of the colored-noise channel model and that of the noise-whitened channel model is given by

$$h_i = \sum_{j=0}^{m-i} f_j f_{i+j} \tag{2.4}$$

Both $\{y_i\}$ and $\{z_i\}$ constitute equivalent sets of sufficient statistics for the estimation of the transmitted data bits b_i. Depending on situations such as the ease of evaluation of performance, etc., one of the set of sufficient statistics or equivalently, one of the types of channel model descriptions, could be preferred instead of the other.

In this chapter, we utilize the noise-whitened channel model as in (2.3) in the exposition of the various equalization algorithms.

2.2 Trellis-Based Turbo Equalizers

In the following subsections, we give a brief review of the FC-BCJR algorithm. The BCJR algorithm [11] is the optimum algorithm for the recovering of a Markov source perturbed by AWGN in terms of BER. We first give an exposition of the BCJR algorithm in the context of equalization over a noisy ISI channel. As the BCJR algorithm could also be used to implement the channel decoder, the variation of the branch metric in the BCJR algorithm to suit the decoding process will be highlighted. From the descriptions of the FC-BCJR algorithm [11], a greater understanding of the turbo principle is obtained. The M-BCJR [21] and SW-BCJR [22] as low complexity alternatives to the FC-BCJR will also be briefly described.

2.2.1 Full Complexity BCJR Algorithm

The FC-BCJR equalizer produces the *a posteriori* information of the transmitted code bits $\{b_i\}$ in the form of LLR as

$$\Lambda_i = \log \frac{P(b_i = +1 \mid \mathbf{z})}{P(b_i = -1 \mid \mathbf{z})} \tag{2.5}$$

$$= \log \frac{\sum_{\{\mathbf{s}_i \to \mathbf{s}_{i+1} : b_i = +1\}} P(\mathbf{s}_i, \mathbf{s}_{i+1}, \mathbf{z})}{\sum_{\{\mathbf{s}_i \to \mathbf{s}_{i+1} : b_i = -1\}} P(\mathbf{s}_i, \mathbf{s}_{i+1}, \mathbf{z})} \tag{2.6}$$

where the channel states at time instant i, $\mathbf{s}_i$, is defined as the m most recent inputs to the channel at time instant i, i.e., $\mathbf{s}_i = (b_{i-1}, b_{i-2}, \ldots, b_{i-m})$. The vector $\mathbf{z}$ refers to an entire frame of observations obtained from the whitened-noise channel, i.e., $\mathbf{z} = (z_0, z_1, \ldots, z_i, \ldots, z_{L-1})$, where L refers to the frame size or equivalently, the trellis length.

The FC-BCJR computes (2.5) by simplifying the calculation of (2.6) through the use of the Markovian properties of the underlying transmission process together with the rules of conditional probabilities to arrive at a computation of the forward recursion α metric and a backward recursion β metric over the 2^m-state trellis of the ISI channel. Assuming a memoryless transmission channel, the joint probability $P(\mathbf{s}_i, \mathbf{s}_{i+1}, \mathbf{z})$ is thus a product of three independent terms written as

$$P(\mathbf{s}_i, \mathbf{s}_{i+1}, \mathbf{z}) = \alpha_i(\mathbf{s}_i) \cdot \gamma(\mathbf{s}_i \rightarrow \mathbf{s}_{i+1}) \cdot \beta_{i+1}(\mathbf{s}_{i+1}) \tag{2.7}$$

where a forward recursion and a backward recursion over the code trellis could be used to compute the α and β metrics respectively as

$$\alpha_i(\mathbf{s}_i) = \sum_{\mathbf{s}_{i-1}} \alpha_{i-1}(\mathbf{s}_{i-1}) \gamma(\mathbf{s}_{i-1} \rightarrow \mathbf{s}_i) \tag{2.8}$$

$$\beta_i(\mathbf{s}_i) = \sum_{\mathbf{s}_{i+1}} \beta_{i+1}(\mathbf{s}_{i+1}) \gamma(\mathbf{s}_i \rightarrow \mathbf{s}_{i+1}) \tag{2.9}$$

The forward and backward recursion are initialized with $\alpha_0(\mathbf{s}_0) = 1$ and $\beta_L(\mathbf{s}_L) = 1$ only for $\mathbf{s}_0$ and $\mathbf{s}_L$ corresponding to the all-zero state, i.e $\mathbf{s} = (-1, \ldots, -1)$ in BPSK style. The rest of the states at this two time instances are set to 0. This initialization is based on the assumption that the trellis starts and ends at the all-zero state. In the event where trellis termination is not possible or unknown, $\beta_L(\mathbf{s}_L) = 1/2^m$ for all the possible states at the trellis end will then be initialized instead.

For complexity and precision reasons, the BCJR algorithm is usually implemented in the logarithmic domain (log-domain) [41]. Two realizations of the BCJR algorithm in the log-domain are the max-log-MAP algorithm and the log-MAP algorithm [41]. Both realizations perform the multiplications in (2.7) to (2.9) as additions except with a difference in how they compute the addition in the log-domain.

The branch metric γ based on the noise-whitened channel model described in (2.3) when expressed in the log-domain is given by

$$\gamma(\mathbf{s}_i \rightarrow \mathbf{s}_{i+1}) = -\frac{1}{2\sigma^2}\left(z_i - \sum_{j=0}^{m} f_j b_{i-j}\right)^2 + \frac{1}{2} b_i L_a(b_i) \tag{2.10}$$

where $L_a(b_i)$ denotes the *a priori* LLR of bit b_i fed back from the decoder and σ^2 is the channel noise variance. The *a priori* LLR $L_a(b_i)$ is the interleaved extrinsic information $L_e(b_i)$ produced by the RSC decoder.

The FC-BCJR algorithm could also be implemented for the colored-noise channel model described by (2.2) with an equivalent BER performance by re-expressing the branch metric as [42] [43]

$$\gamma(\mathbf{s}_i \rightarrow \mathbf{s}_{i+1}) = -\frac{1}{2\sigma^2}\left(2y_i b_i - 2\sum_{j=1}^{m} h_j b_i b_{i-j}\right) + \frac{1}{2} b_i L_a(b_i) \tag{2.11}$$

where the implementation of the noise-whitening filter could be avoided.

The FC-BCJR algorithm described above is exemplified in the context of equalization over a trellis-based rate-1 ISI channel. As the RSC code constraints could be indirectly specified by a code trellis, the FC-BCJR algorithm may also be used for decoding operations. The algorithms for equalization and decoding proceed in a similar manner except with some minor differences highlighted below:

- The equalizer is only required to compute the APP of the channel inputs $\{b_i\}$. On the other hand, the decoder is required to compute not only the APP of code bits $\{c_i\}$, but also the APP of data bits $\{d_i\}$. Hard decisions taken on the APP of the data bits would then be considered as estimates of the transmitted bits.
- As the channel is of rate-1, each of the channel trellis edges corresponds to a single channel output. However, for a code trellis, each edge corresponds to more

than one output. In this thesis, a rate-1/2 RSC code is considered and as such, the code trellis edge has two outputs for every state transition.

- For a turbo equalizer structure depicted in Figure 1.2, the decoder does not have access to the channel output. Therefore, the first term in the branch metric stated in (2.10) or (2.11) reduces to a constant and could be eliminated in the calculation. As such, the branch metric for the RSC decoder reduces to

$$\gamma(\mathbf{s}_i \rightarrow \mathbf{s}_{i+1}) = \frac{1}{2}\sum_{q=0}^{n-1}(2c_{i,q} - 1)L_a(c_{i,q}) \tag{2.12}$$

  where n refers to the number of code bits per data bit. Note in (2.12) that the branch metric for the trellis-based RSC decoder depends solely on the *a priori* information $L_a(c_{i,q})$ provided by the channel equalizer.

The above descriptions highlighted the key differences between the trellis-based equalizer and trellis-based decoder. As a general note, in this thesis, the code trellis of the RSC encoder is terminated by an appropriate selection of the tail bits to flush the content of the encoder to the all-zero state. The channel trellis is also terminated by simply appending m '-1's bits to the BPSK modulated interleaved code bits.

An important observation that can be made from the implementation of the FC-BCJR algorithm for both the equalizer and decoder is that the only "new" information as the iterations proceed enters the equalizer or decoder in the form of *a priori* information at the branch metric described by (2.10) – (2.12). This new *a priori* information is the extrinsic information produced by the previous SISO module. The significant BER improvement of a turbo equalizer comes about solely from an improvement in the reliability of the *a priori* information as the iterations proceed. This exchange of refined estimate of the *a priori* information is the essence of a turbo receiver.

2.2.2 Low Complexity Variants of BCJR Algorithm

A) M-BCJR Algorithm

The M-BCJR algorithm [18][19][21] is a reduced state-trellis technique which aims to reduce the computational complexity of the original FC-BCJR by limiting the number of trellis states kept active to a predetermined constant M where $M < 2^m$. As mentioned before, 2^m is the number of trellis states per stage and m is the channel memory length.

In more details, the forward recursion on α_{i-1} to α_i described in (2.8) is carried out using only the M largest metric values of α_{i-1} while the rest are declared inactive or dead. This same principle is applied in the generation of the backward beta recursion metric. However, to facilitate the appropriate combining of the alpha and beta metrics to form the LLR output, the backward recursion is only executed on the region of the trellis where the forward metrics are still alive. Hence, the exponential computational complexity of the FC-BCJR algorithm is reduced accordingly to the predetermined value M in M-BCJR algorithm

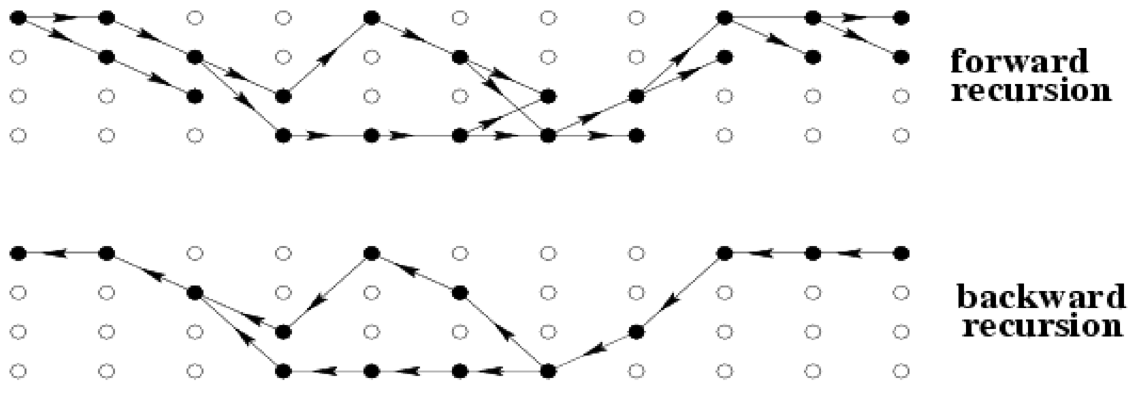

Figure 2.2: Trellis paths of M-BCJR algorithm where $M = 2$

As an illustration on the operation of the M-BCJR algorithm, a trellis section is shown in Figure 2.2. This figure is adapted from the paper by Fragouli *et al.* [18]. Here, the channel memory m is of length 2. The number of trellis states per stage is thus $2^m = 4$. In the trellis section depicting the paths taken for the forward recursion, note that the number of trellis states per stage that is kept alive is $M = 2$. Also shown in Figure 2.1 is the paths taken for the execution of the backward recursion. Visualizing the paths selected by the forward recursion based on the M-BCJR as a tree of active nodes, the backward recursion is thus a subtree of the chosen active nodes. The final APP LLR is then generated using the edges in this common subtree.

B) SW-BCJR Algorithm

The SW-BCJR algorithm is proposed in [22] to reduce the latency associated with the original FC-BCJR algorithm in generating the APP LLR. Instead of the need to receive an entire data frame, the SW-BCJR algorithm relaxes this constraint by operating on a fixed memory span and output the APP LLR after a given delay D, where $D << L$. The channel trellis operated on by the SW-BCJR algorithm remains exactly the same as that of the one operated on by FC-BCJR algorithm. The only difference between the two algorithms lies in the way the backward recursion is initialized.

In the SW-BCJR algorithm, the backward recursion for the generation of APP LLR at the i^{th} time instant is initialized at $(i+D)^{\text{th}}$ time instant instead of the end of the trellis at time instant L. This initialization is carried out using the value of the forward recursion metrics at this $(i+D)^{\text{th}}$ time instant. The computation of the backward recursion is then carried out as usual but based on this new manner of initialization.

With this method of initialization, the intrinsically block oriented FC-BCJR algorithm is modified to allow a continuous decoding/equalization of the received stream. As such, decoding/equalization delays associated with the FC-BCJR algorithm is greatly reduced in the SW-BCJR algorithm.

2.3 Filter-Based Turbo Equalizers

In the following subsections, we give a review on the implementation of the filter-based equalizer designed using the MMSE criterion. The exposition outlined here summarizes the procedures and highlights the key equations in deriving the MMSE equalizer and its corresponding extrinsic information output. For a more detailed presentation, the interested readers could refer to the paper by Michael Tüchler *et al*. [5].

Following that, two low complexity implementations of the MMSE equalizer are briefly described. These two low complexity variants are based on approximate implementations of the original time-varying MMSE equalizer.

2.3.1 MMSE Equalization Algorithm

The MMSE equalizer described here is implemented using a linear FIR filter of length N whose time-varying filter coefficients $c_{i,k}$, $k = -N_1, 1 - N_1, \dots, N_2$, where $N = N_1 + N_2 + 1$, are obtained using the MMSE design criterion.

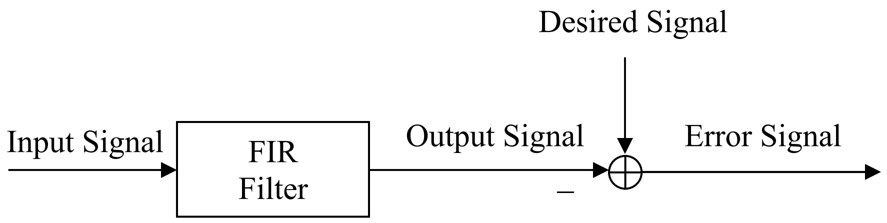

Figure 2.3: Schematic diagram of a filter with an emphasis on its role to reshape the input signal to match the desired signal

In essence, the concept of MMSE filtering or estimation is to select the filter coefficients $c_{i,k}$ such that the error signal as depicted in Figure 2.3 is minimized in the mean square sense. Through this design criterion, the MMSE filter attempts to estimate the part of the desired signal that is correlated with its input signal; any uncorrelated component present in its input signal is unaffected. Generally, the uncorrelated component embedded in the input signal is noise. Perfect estimation of the correlated component in the desired signal is possible only in the event when there is no noise at the input of the MMSE filter. In the context of channel equalization, this perfect estimation due to an absence of input noise results in a ZF equalization approach, thereby translating to zero ISI at the MMSE filter output.

In the general situation where input noise is present, the optimum MMSE equalizer attempts to strike a good compromise between ISI reduction and noise enhancement at the equalizer output. This compromise is determined by the SNR of the signal at the equalizer input. If the input signal is of high SNR, the MMSE equalizer would concentrate on reducing the mis-equalization and hence reducing the detrimental effects of ISI at the equalizer output. However, for low SNR input signal, most of the effort of the MMSE equalizer would be focused on noise reduction instead. Typically, low SNR input signal is encountered in an ISI channel transmission process where spectral nulls are present in some frequency regions of the channel frequency response.

Relating Figure 2.3 to the noise-whitened channel model described in (2.3), the input signal could be visualized as the channel observations z_i and the desired signal as the interleaved code bits b_i. We denote the output signal of the filter as x_i.

To apply the MMSE estimation approach to the turbo equalization scheme, it is necessary to derive the extrinsic information from the equalizer output x_i for use as *a priori* information by the SISO decoder. In Section 2.2, the APP LLR Λ_i of the transmitted code bits b_i is obtained from the BCJR algorithm. Extrinsic information $L_E(b_i)$ is then derived from this APP LLR Λ_i by subtracting the *a priori* LLR $L_a(b_i)$ used in the computation from it. For the MMSE equalizer case, instead of generating the APP LLR, the algorithm could be modified to produce the extrinsic information directly from the statistics of the equalizer output x_i. Specifically, the MMSE equalizer is able to produce the extrinsic information described as

$$L_e(b_i) = \log\frac{P(b_i = +1 \mid x_i)}{P(b_i = -1 \mid x_i)} - \log\frac{P(b_i = +1)}{P(b_i = -1)} \tag{2.13}$$

$$= \log\frac{P(x_i \mid b_i = +1)}{P(x_i \mid b_i = -1)} \tag{2.14}$$

where the second term in (2.13) is the *a priori* LLR $L_a(b_i)$ of the interleaved code bits b_i obtained from the RSC decoder.

As an overview, the implementation of the MMSE equalizer for use in a turbo equalization scheme could be summarized in four steps:

(1) Derive the optimum equalizer filter coefficients $c_{i,k}$ using the MMSE design criterion;

(2) Obtain the equalizer output x_i by filtering the observations z_i through the MMSE-optimum filter derived in (1);

(3) Determine the first order statistics of the optimum equalizer output x_i;

(4) Compute the extrinsic information $L_e(b_i)$ of the transmitted code bits b_i from the first order statistics determined in (3) by assuming a Gaussian distribution on the conditional probability $P(x_i \mid b_i = \pm 1)$.

The four steps described above are to be iterated for the generation of the extrinsic information $L_e(b_i)$ for each time instant i. As such, an exact implementation of the MMSE equalizer requires a re-computation of the optimal time-varying filter coefficients for every time instant i.

The MMSE-optimal filter coefficients is derived based on the first and second order statistics of the underlying jointly wide sense stationary transmission processes, namely, the auto-covariance function of the channel observations vector $\mathbf{z}_i$ and the cross-covariance function between the desired signal b_i and the channel observations vector $\mathbf{z}_i$. Here, vector $\mathbf{z}_i$ is of length N instead of L as described in (2.25) and (2.26). Since these two statistical quantities is inevitably related to the *a priori* LLR $L_a(b_j)$ for $j = (i - m - N_2, \ldots, i, \ldots i + N_1)$ as the iterations proceed, it is thus necessary to set the *a priori* LLR $L_a(b_i)$ to 0 when computing the optimal-MMSE filter coefficients $c_{i,k}$ and also for the generation of the filter output x_i. The *a priori* LLR $L_a(b_j)$ for $j \neq i$ is unaffected and used accordingly when computing this MMSE-optimal filter coefficients for the generation of the extrinsic information $L_e(b_i)$ at this particular ith time instant.

As such, the time-varying MMSE-optimal filter coefficients (in vector form) for the detection of the ith bit is consequently set to

$$\mathbf{c}_i = \left(\sigma^2\mathbf{I}_N + \mathbf{F}\mathbf{V}_i\mathbf{F}^T + (1 - v_i)\mathbf{s}\mathbf{s}^T\right)^{-1}\mathbf{s} \tag{2.15}$$

where σ^2 is the channel noise variance, $\mathbf{I}_N$ is the identity matrix of order N, v_i is the variance of the transmitted bit b_i derived from the *a priori* LLR $L_a(b_i)$ and the superscript T refers to matrix transpose. Here, $\mathbf{V}_i$ is a diagonal matrix defined as

$$\mathbf{V}_i = \mathrm{Diag}\left(v_{i-m-N_2}, v_{i-m-N_2+1}, \cdots, v_{i+N_1}\right) \tag{2.16}$$

and $\mathbf{F}$ is the N x $(N + m)$ channel convolution matrix based on the noise-whitened channel model written as

$$\mathbf{F} = \begin{bmatrix} f_m & f_{m-1} & \cdots & f_o & 0 & \cdots & \cdots & 0 \\ 0 & f_m & f_{m-1} & \cdots & f_0 & 0 & \cdots & 0 \\ \vdots & & \ddots & \ddots & \ddots & & & \vdots \\ \vdots & & & \ddots & \ddots & \ddots & & \vdots \\ 0 & \cdots & \cdots & 0 & f_m & f_{m-1} & \cdots & f_0 \end{bmatrix} \tag{2.17}$$

The vector **s** in (2.15) is utilized in the derivation of the filter coefficients in a way similar to an indicator function to remove the *a priori* information on the *i*th bit and is defined as

$$\mathbf{s} = \mathbf{F}\left[\mathbf{0}_{1\times(N_2+m)},\ 1,\ \mathbf{0}_{1\times N_1}\right]^T \tag{2.18}$$

With the derivation of the MMSE-optimal filter coefficients $\mathbf{c}_i$, the filter output x_i can then be expressed as

$$x_i = \mathbf{c}_i^T\left(\mathbf{z}_i - \mathbf{F}\overline{\mathbf{b}}_i + \overline{b}_i\mathbf{s}\right) \tag{2.19}$$

where $\overline{b}_i$ is the mean of the transmitted code bits b_i derived from the *a priori* LLR $L_a(b_i)$ fed back from the decoder and $\overline{\mathbf{b}}_i$ is the mean vector defined as

$$\overline{\mathbf{b}}_i = [\overline{b}_{i-m-N_2}, \overline{b}_{i-m-N_2+1}, \cdots, \overline{b}_{i+N_1}]^T \tag{2.20}$$

The mean $\overline{b}_i$ and variance v_i of the transmitted code bits b_i is derived from the *a priori* LLR $L_a(b_i)$ using

$$\begin{aligned}\overline{b}_i &= \sum_{b\in\{+1,-1\}} b\cdot P\left(b_i = b\right)\\ &= P\left(b_i = +1\right) - P\left(b_i = -1\right)\\ &= \frac{e^{L_a(b_i)}}{1+e^{L_a(b_i)}} - \frac{1}{1+e^{L_a(b_i)}}\\ &= \tanh\left(\frac{L_a(b_i)}{2}\right)\end{aligned} \tag{2.21}$$

and

$$v_i = \sum_{b\in\{+1,-1\}} |\,b_i - E(b_i)\,|^2 \cdot P\left(b_i = b\right) \tag{2.22}$$

$$= 1 - |\,\overline{b}_i\,|^2 \tag{2.23}$$

From the MMSE-optimal filter output x_i described in (2.19), the 1st moment and 2nd central moment are then derived from x_i to give the mean $\mu_{i,b}$ and variance $\sigma_{i,b}^2$ of x_i as

$$\mu_{i,b} = b \cdot \mathbf{c}_i^T \mathbf{s} \tag{2.24}$$

$$\sigma_{i,b}^2 = \mathbf{c}_i^T \left(\sigma^2 \mathbf{I}_N + \mathbf{F}\mathbf{V}_i\mathbf{F}^T - v_i \mathbf{s}\mathbf{s}^T \right) \mathbf{c}_i \tag{2.25}$$

$$= \mathbf{c}_i^T \mathbf{s} \left(1 - \mathbf{s}^T \mathbf{c}_i \right) \tag{2.26}$$

Assuming a Gaussian distribution on the conditional probabilities $P(x_i \mid b_i = b)$, $b \in \{+1, -1\}$with parameters described in (2.24) and (2.26), the final extrinsic LLR $L_e(b_i)$ described by (2.14) can then be re-written as

$$L_e(b_i) = \frac{2x_i \mu_{i,+1}}{\sigma_{i,+1}^2} \tag{2.27}$$

$$= \frac{2\mathbf{c}_i^T \left(\mathbf{z}_i - \mathbf{F}\bar{\mathbf{b}}_i + \bar{b}_i \mathbf{s} \right)}{1 - \mathbf{s}^T \mathbf{c}_i} \tag{2.28}$$

This thus completes the descriptions for the exact implementation of the MMSE equalizer based on time-varying filter coefficients for use in a turbo equalization scheme.

From (2.15), it is observed that a matrix inversion operation is required for the generation of the MMSE-optimal filter coefficients $\mathbf{c}_i$, which is a costly operation with a cubic order computation complexity (in the matrix order or equivalently in the equalizer filter length N). A recursive algorithm with a square order computation complexity (in the equalizer filter length N) is employed in [24] to efficiently compute the MMSE equalizer filter. This time-recursive update approach is based on the observation that the submatrices of the matrices to be inverted are identical for time instances i and (i+1). As such, based on an initial condition obtained simply by carrying out the conventional matrix inversion for the derivation of the MMSE-optimal filter coefficients for the first time instant, subsequent derivation of the MMSE-optimal filter coefficients for other time instances could be obtained through this recursive approach.

2.3.2 Low Complexity Variants of MMSE Equalization Algorithms

The low complexity variants of the MMSE equalizer described here forth are implemented based on a time-invariant implementation of the exact time-varying MMSE equalizer described in the previous section. Such approximate implementations of the MMSE equalizer are derived based on two different scenarios where the MMSE equalizer has either zero *a priori* information or perfect *a priori* information.

A) Approximate Implementation I – Zero *A Priori* Information Scenario

The MMSE filter described here is designed based on zero *a priori* LLR, i.e., $L_a(b_i) = 0 \ \forall \ i$ which thus results in $\overline{b}_i = 0$ and $v_i = 1 \ \forall \ i$, as determined from (2.21) and (2.23). Substituting these parameters into (2.15), a time-invariant MMSE equalizer with coefficients $\mathbf{c}_{\text{zero}}$ could then be obtained as

$$\mathbf{c}_{\text{zero}} = \left(\sigma^2 \mathbf{I}_N + \mathbf{F}\mathbf{F}^T\right)^{-1} \mathbf{s} \tag{2.29}$$

where subscript "zero" is used to denote the time-invariant MMSE coefficients derived based on zero *a priori* information.

However, to allow this approximate MMSE filter to be used in a turbo equalization scheme, it must still incorporate the *a priori* information from the SISO decoder into the derivation of the required statistics, bearing in mind that the *a priori* information for the i^{th} bit is still required to be set to 0 when computing the extrinsic information for the i^{th} bit. As such, from the MMSE filter coefficients described by (2.29), the filter output x_i is similarly obtained as in (2.19), from which the required statistics are computed using (2.24) and (2.25). Finally, after obtaining the necessary statistics from the filter output x_i, the extrinsic LLR $L_e(b_i)$ is then obtained using (2.27). Since computing $\sigma_{i,+1}^2$ for each time instant i is computationally expensive, a further simplification to approximate $\sigma_{i,+1}^2$ using the time average could be carried out. As such, the parameter $\sigma_{i,+1}^2$ could be replaced with a time-invariant σ_{zero}^2 defined as

$$\sigma_{\text{zero}}^2 \approx \mathbf{c}_{\text{zero}}^T \left(\sigma^2 \mathbf{I}_N + \left(\frac{1}{L} \sum_{i=1}^{L} v_i \right) \left(\mathbf{F}\mathbf{F}^T - \mathbf{s}\mathbf{s}^T \right) \right) \mathbf{c}_{\text{zero}} \tag{2.30}$$

where L is the frame size of the transmitted code bits b_i.

B) Approximate Implementation II – Perfect *A Priori* Information Scenario

The second approximate implementation of the MMSE filter is derived based on the situation where perfect *a priori* information is available to the equalizer. Since $|L_a(b_i)| \to \infty \;\forall\; i$, this thus translates to $\overline{b}_i = b_i$ and $v_i = 0 \;\forall\; i$, based on (2.21) and (2.23). Substituting these parameters into (2.15), the MMSE filter coefficients is thus

$$\mathbf{c}_{\text{perfect}} = \left(\sigma^2 + \mathbf{s}^T \mathbf{s} \right)^{-1} \mathbf{s} \tag{2.31}$$

where subscript "perfect" is used to denote the time-invariant MMSE coefficients derived based on perfect *a priori* information. Note that $\mathbf{s}^T\mathbf{s}$ is the energy E_f of the ISI channel defined as

$$E_f = \sum_{j=0}^{m} | f_j |^2 \tag{2.32}$$

Following the same procedures as for the zero *a priori* information scenario, the final extrinsic LLR is then obtained as

$$L_{e,\text{perfect}}(b_i) = \frac{2E_f \mathbf{s}^T \left(\mathbf{z}_i - \mathbf{F}\overline{\mathbf{b}}_i + \overline{b}_i \mathbf{s} \right)}{E_f \sigma^2 + \mathbf{s}^T \mathbf{F} \mathbf{V}_i \mathbf{F}^T \mathbf{s} - v_i E_f^2} \tag{2.33}$$

where the variance $\sigma_{i,+1}^2$ could similarly by approximated by its time average as

$$\sigma_{\text{perfect}}^2 \approx \left(E_f \sigma^2 + \left(\frac{1}{L} \sum_{i=1}^{L} v_i \right) \left(\mathbf{s}^T \mathbf{F}\mathbf{F}^T \mathbf{s} - E_f^2 \right) \right) \Big/ \left(\sigma^2 + E_f \right)^2 \tag{2.34}$$

2.4 Computational Complexity Comparison

Beside BER performance, the computational complexity of a SISO module is a major consideration in its actual implementation in a turbo receiver. In this thesis, the foremost aim is to derive an alternative low complexity algorithm for the equalization task carried out in a turbo equalizer, while yet trades off with a negligible BER performance degradation when compared with the BER-optimal FC-BCJR equalization algorithm. With this objective in mind, it is thus necessary to evaluate and compare both the computational complexities and the BER performances of the various equalization algorithms described in this chapter to establish a form of benchmark for use in comparing with our proposed heuristic-based LS equalization algorithm that will be presented in the next chapter.

The computational complexity of some of the equalization algorithms [5] are depicted in Table 2.1 on the next page. The focus is on the SW-BCJR algorithm and the exact time-varying implementation of the MMSE algorithm. Such an emphasis on these two algorithms is due to their relatively good tradeoff between computational complexity and BER performance as compared with the BER-optimal FC-BCJR algorithm. This understanding will become clearer when we present the BER performance of the various equalization algorithms in the next section.

The computational complexities of the two approximate implementations of the exact MMSE equalization algorithm described in Section 2.3.2 are also listed in Table 2.1. Although their individual BER performances are not impressive, it is shown in Chapter 4 indirectly through the use of the EXIT chart that they could achieve the same BER performance as the exact MMSE algorithm when utilized together in a hybrid manner. This is done by carrying out the equalization tasks through an optimal switching between the two approximate implementations as the iterations proceed.

From here on, various trellis-based equalization algorithms will be represented by their respective descriptions used in Section 2.2. For the exact time-varying MMSE equalization algorithm described in Section 2.3.1, we would denote it as "MMSE EXACT". The approximate time-invariant implementations of the MMSE equalization algorithm based on zero and perfect *a priori* LLR will be denoted as "MMSE APRX I" and "MMSE APRX II" respectively.

The number of required operations for the SW-BCJR algorithm is arrived at based on its implementation in the logarithmic domain.

For the MMSE case, any overhead due to initialization (one-time operations for all iterations, for example, to compute $\mathbf{c}_{\text{zero}}$ as the starting point for the first iteration), is neglected. The required *a priori* statistics $\overline{b}_i$ and v_i of $L_a(b_i)$ are assumed to be available for all i and the subsequent computation of $L_e(b_i)$ described by (2.24) – (2.28) are not considered. The number of required operations for the MMSE-based approaches follows from the recursive implementations stated in [5][24].

Table 2.1:
Number of required operations for the detection of one transmitted bit per iteration using varying SISO equalization algorithms where
m: channel memory length; D: SW-BCJR decision delay; N: Equalizer filter length.

Equalization Algorithms	Number of real additions	Number of real multiplications
SW-BCJR	$2^m \times (2mD + 7D + 2m + 10)$	4×2^m
MMSE EXACT	$8N^2 + 2m^2 - 10N + 6m + 8$	$16N^2 + 4m^2 + 18m - 4N + 10$
MMSE APRX I	$4N + 4m$	$4N + 8m + 8$
MMSE APRX II	$10m + 8$	$10m + 10$

2.5 Simulation Results and Discussion

In this section, we present the BER performance results of various SISO equalization algorithms described in this chapter by simulating data transmission using the transmitter structure shown in Figure 2.1 and based on the system model described in Section 2.1. A rate-1/2 RSC encoder with generator polynomials in octal form, $(g_1, g_2) = (7, 5)$, where g_1 is the feedback polynomial and g_2 is the feedforward polynomial, is used as the channel code for all simulations. Beside that, the data d_i frame size is set to $2^{12} = 4{,}096$. No deliberate interleaving optimization is carried out; random interleaving is used for all simulations here.

The turbo equalizer structure depicted in Figure 1.2 is utilized as the receiver. The RSC decoder is implemented based on the BER-optimal FC-BCJR algorithm operating in the log-domain, which works on the code trellis specified by the RSC encoder described above. To facilitate turbo equalization, the interleaving pattern used in the transmitter end, the ISI channel impulse response (2.3) and the AWGN channel noise variance σ^2 are known to the receiver. The noise variance σ^2 is determined according to the SNR E_b/N_o (in dB) defined as

$$\begin{aligned} \text{SNR} &= 10 \cdot \log_{10}\left(\frac{E_b}{N_o}\right) \\ &= 10 \cdot \log_{10}\left(\frac{E_b}{2\sigma^2}\right) \\ &= 10 \cdot \log_{10}\left(\frac{E_c}{2R\sigma^2}\right) \end{aligned} \tag{2.35}$$

where E_b is the energy of the data bit d_i, E_c is the energy of the code bit c_i and R is the code-rate which is the ratio of the number of data bits to code bits. For the simulations shown here, we set $E_c = 1$ and $R = 1/2$ to scale the AWGN noise variance σ^2 accordingly to the prescribed SNR value.

All the BER performances curves presented are the simulation results acquired from the 15th iterations, unless otherwise stated. Here, we consider the first time equalization and decoding tasks with zero *a priori* information at the input of the equalizer as the first iteration. In short, all variable parameters, i.e., code-rate, generator polynomials of RSC code, frame size, interleaving pattern, SNR definition, decoder implementation, predetermined number of iterations etc., which may affect the BER performances comparisons, are kept constant; the only difference in the system setup is the type of equalization algorithms utilized.

Beside the general system parameters described above, other specific parameters relating to the respective equalization algorithms are to be detailed for a fair comparison. For the SW-BCJR equalization algorithm, the decision delay D is set to $2m$, which is twice the memory length of the ISI channel. This value is selected to allow a fair comparison when we present the LS equalization algorithm in the next chapter so as to ensure that both equalization algorithms operate on the same length of observation window. For the M-BCJR equalization algorithm, the number of survival states per trellis stage is set to $M = 5$ [21]. For all the MMSE-based implementations, the filter length is set to $N = 15$ where $N_1 = 9$ and $N_2 = 5$, which is identical to that used in [5].

In the simulations, we also test the efficacy of the various SISO equalization algorithms under different ISI conditions modeled by an ISI channel of memory length $m = 4$ with different coefficients as shown in Table 2.2. Using (2.32), the energy of the respective ISI channels in Table 2.2 could be computed and is found to be unity. The characteristics of the respective channels are also stated in the table.

Table 2.2:
Descriptions and characteristics of ISI channels used in the simulations

Channel Description	Noise-Whitened Coefficients	ISI Due To Distortions By		ISI Condition
		Delay	Amplitude	
Channel I	$f=\left\{\sqrt{0.45},\sqrt{0.25},\sqrt{0.15},\sqrt{0.10},\sqrt{0.05}\right\}$	Yes	Yes	Mild
Channel II (Proakis C Channel)	$f=\left\{0.227, 0.460, 0.688, 0.460, 0.227\right\}$	No	Yes	Severe

The simulations are first carried out using a 5-tap ISI channel whose corresponding noise-whitened coefficients are given by $f=\left\{\sqrt{0.45},\sqrt{0.25},\sqrt{0.15},\sqrt{0.10},\sqrt{0.05}\right\}$. This ISI channel is denoted in Table 2.2 as Channel I and is used in [1] and [44] to demonstrate the performance of turbo equalization schemes. This channel introduce mild ISI contributed by both delay and amplitude distortions. The effect of amplitude distortions on the transmitted data could be appreciated by observing the non-flat magnitude response of Channel I depicted in Figure 2.4. Though not shown here, the phase response characteristic of Channel I is a non-linear function of the frequency, which accounts for the delay distortions on the transmitted data.

The BER performances of the various equalization algorithms illustrated in this chapter under the ISI condition described by Channel I are shown in Figure 2.5. For comparison purposes, the performance of FC-BCJR convolutional decoding over an AWGN channel without any ISI (curve labeled as “AWGN”) is also included to serve as a lower bound for coded ISI systems. Once the BER performance of the turbo equalizer reaches this bound, the detrimental effects of ISI on the transmitted data caused by the channel are completely removed by the equalizer. From Figure 2.5, it is observed that the best BER performance is attained by the turbo equalizer employing FC-BCJR equalization algorithm. Though not shown here, the performance of the SW-BCJR algorithm is almost identical to the FC-BCJR algorithm. As such, from here on, we would not distinguish the BER performances between these two algorithms. The exact implementation of the time-varying MMSE equalizer has a performance similar to that of the M-BCJR algorithm. It is also observed that these algorithms attained the ISI-free bound ("AWGN" curve) at a SNR of about 4 dB. The approximate implementations of the MMSE equalizer, however, have serious performance degradations through the simulated SNR range. The MMSE APRX I performs worse than the SW-BCJR equalization algorithm by approximately 4 dB at a BER of 10^{-3}, while the MMSE APRX II does not work at all for this channel.

Next, we consider data transmission through another ISI channel with noise-whitened coefficients $f = \{0.227, 0.460, 0.688, 0.460, 0.227\}$ which is denoted as Channel II in Table 2.2. This channel is also known as Proakis C channel [40] and has been used in [5] to test the performance of the time-varying MMSE equalizer. This channel causes severe ISI contributed solely by amplitude distortions. The severity of the ISI is due to the presence of spectral nulls as observed in the magnitude response of the channel depicted in Figure 2.6.

The BER performances of the various equalization algorithms utilized to combat ISI in Channel II are shown in Figure 2.7. As before, the best performing equalization algorithm is still that of the SW-BCJR algorithm and it manages to converge to the ISI-free bound at a SNR value of 4 dB with a BER of 10^{-3}. The performance of the FC-BCJR algorithm is indistinguishable from that of the SW-BCJR algorithm and hence not shown for clarity. A difference between the BER performances of the FC-BCJR/SW-BCJR algorithms depicted in Figure 2.7 as compared to Figure 2.5 is the noticeable loss at low SNR region for the severe ISI case. Next, it is also observed that the MMSE EXACT performs better than the M-BCJR algorithm in this channel, although both reach the ISI-free bound at the same SNR value of approximately 5.8 dB; a SNR value of 1.8 dB more as compared to the mild ISI case. Similar to Figure 2.5, the approximate implementations of the time-varying MMSE equalizer (MMSE APRX I and MMSE APRX II) have a large performance loss over this channel.

From the BER performance curves depicted in Figure 2.5 and Figure 2.7, we can conclude that trellis-based SW-BCJR algorithm and the filter-based MMSE EXACT algorithm are good representatives of their respective categories of equalization algorithms in terms of the trade-off between BER performances and computational complexities, under general ISI conditions.

Next, we further explore the BER performances of the SW-BCJR algorithm and the MMSE EXACT algorithm as the iterations proceed for a particular SNR value. The relationship between BER performance and the number of iterations at different SNR values for the two channels depicted in Table II are shown in Figure 2.8 and Figure 2.9 respectively. In general, the BER performance is said to converge to a fixed point if further iterations carried out by the turbo equalizer does not improve BER performance anymore. As observed in these figures, the SW-BCJR algorithm is able to converge faster (i.e., required less iterations to attain a constant BER performance) than the MMSE EXACT algorithm. An important observation that can be seen in these figures is that the number of iterations required for convergence and the corresponding BER attained at convergence are heavily dependent on the SNR value. The reasons behind such phenomena could be understood when we present the EXIT chart in Chapter 4.

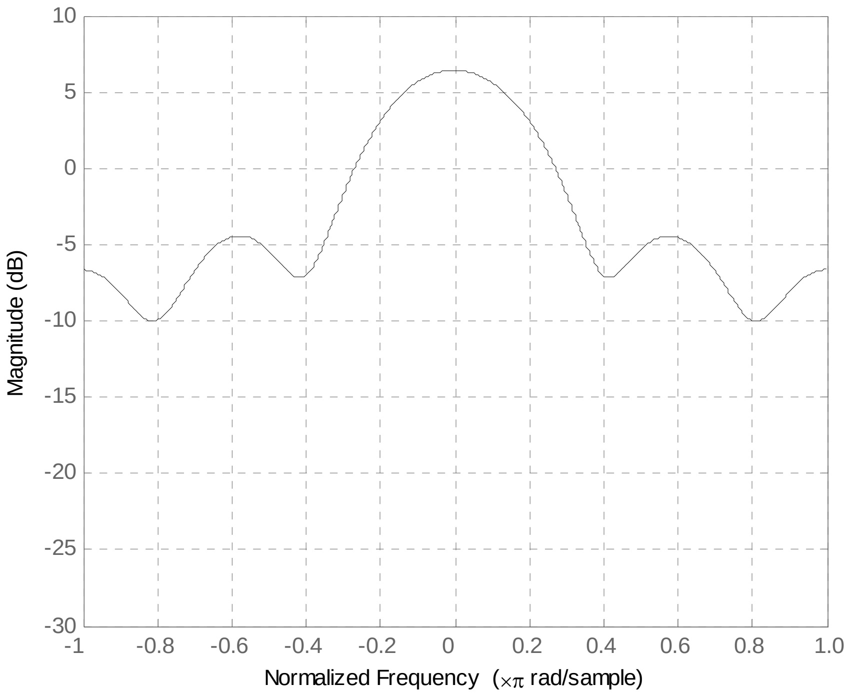

Figure 2.4: Magnitude response of Channel I where
$f = \left\{\sqrt{0.45}, \sqrt{0.25}, \sqrt{0.15}, \sqrt{0.10}, \sqrt{0.05}\right\}$

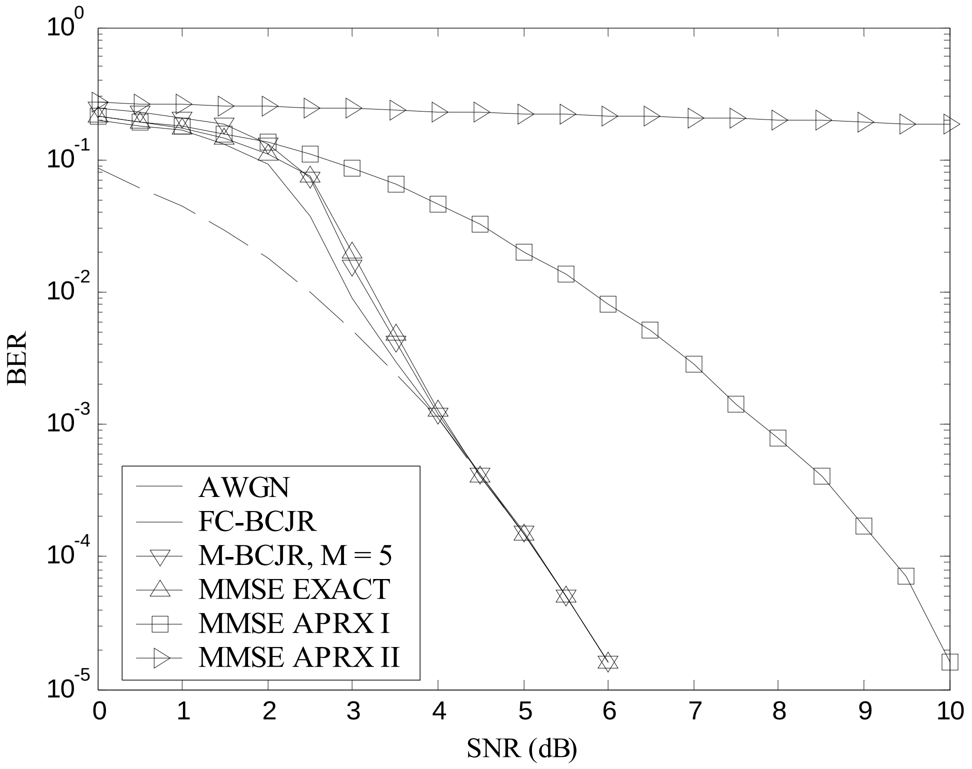

Figure 2.5: Performance of various equalization algorithms for Channel I

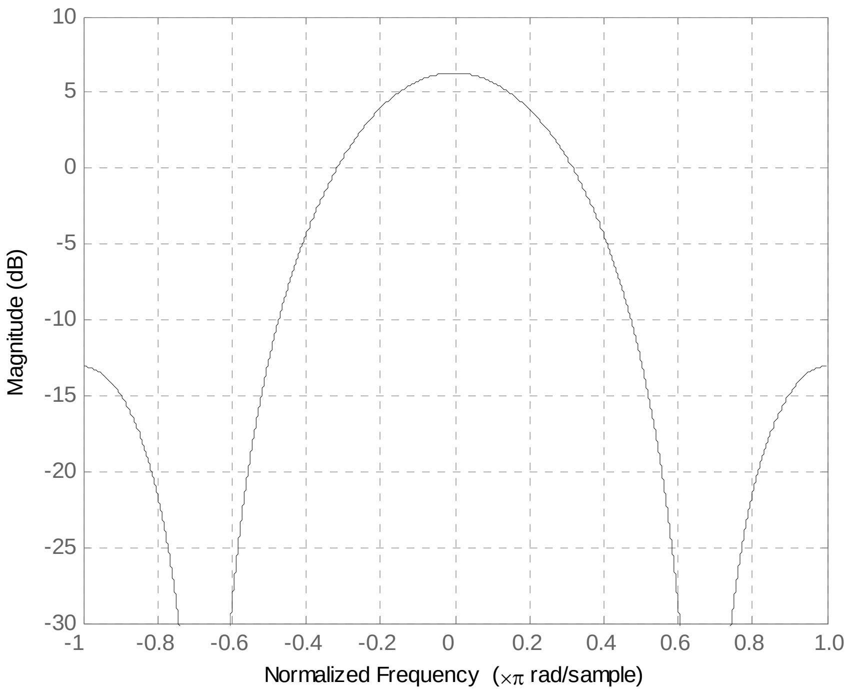

Figure 2.6: Magnitude response of Channel II (Proakis C Channel) where $f = \{0.227, 0.460, 0.688, 0.460, 0.227\}$

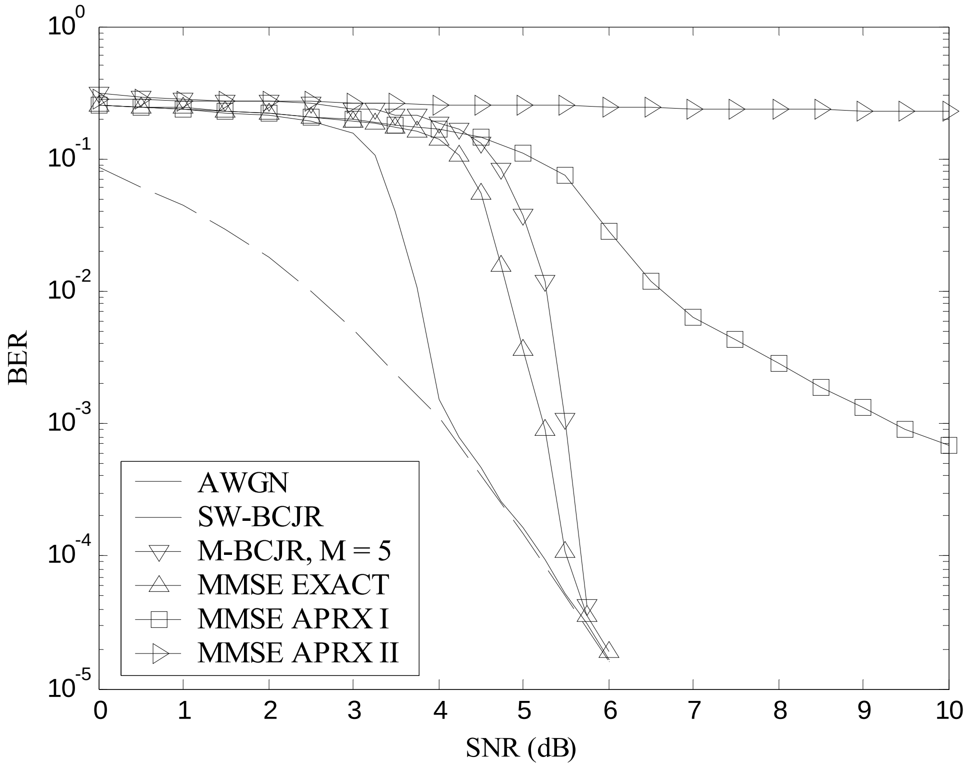

Figure 2.7: Performance of various equalization algorithms for Channel II (Proakis C Channel)

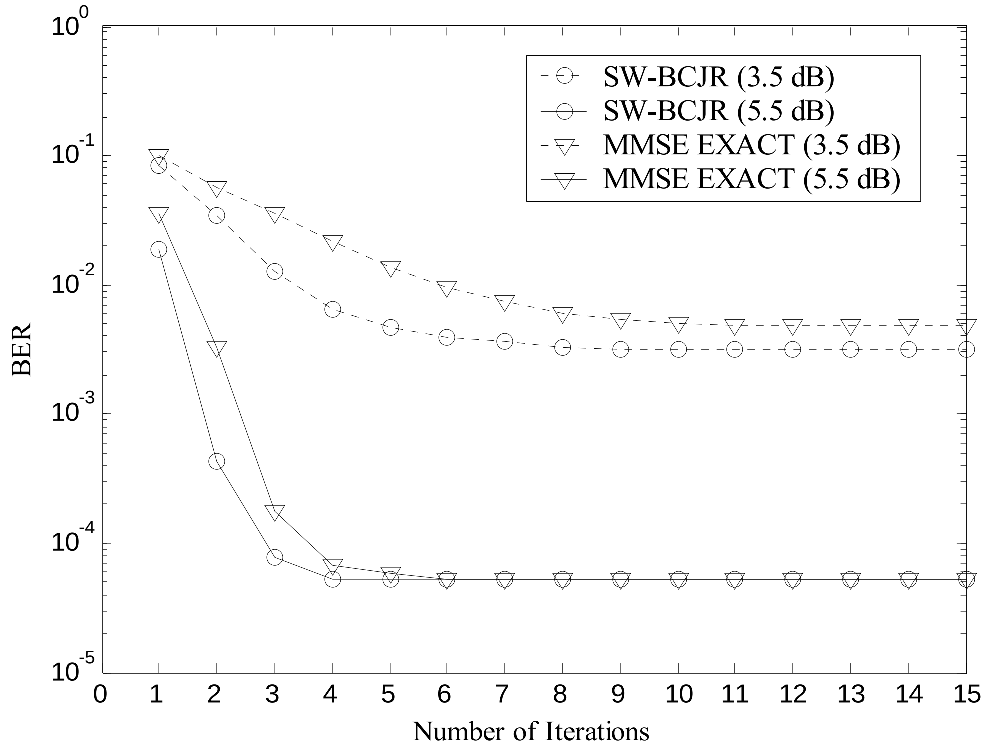

Figure 2.8: Performance of selected equalization algorithms at different iterations for Channel I

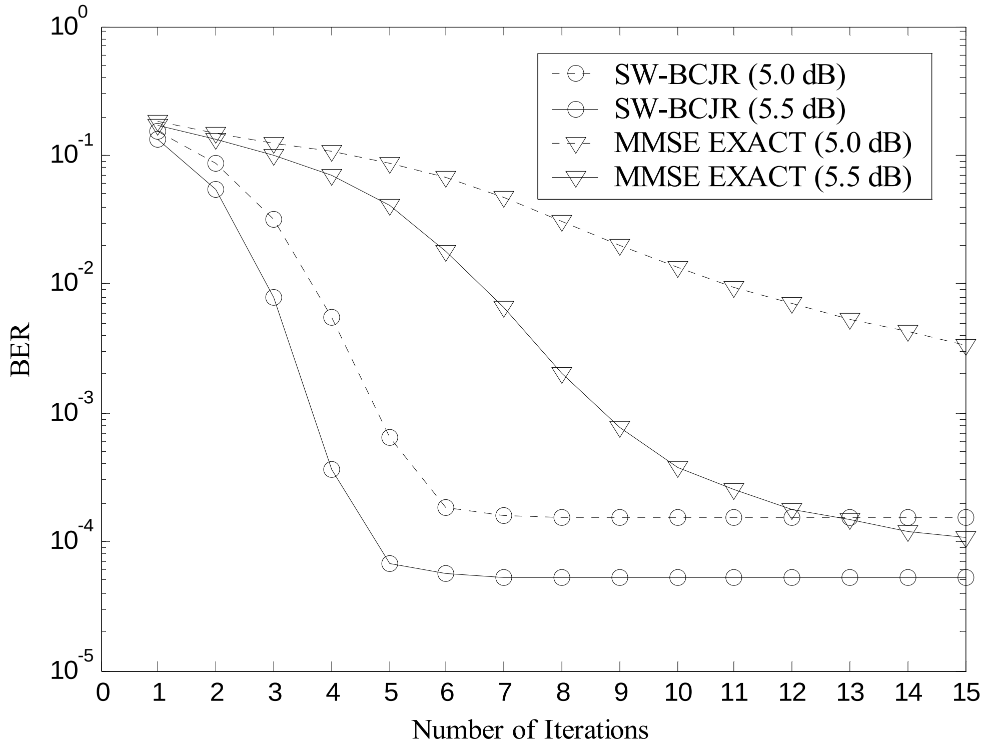

Figure 2.9: Performance of selected equalization algorithms at different iterations for Channel II (Proakis C Channel)

2.6 Chapter Summary

A survey of some of the equalization algorithms for use in a turbo equalization scheme has been carried out in this chapter. Two well-known categories of equalization algorithms from the trellis-based class and filter-based class have been described. The essential implementation details for these two categories to suit the turbo equalization process are also highlighted. In general, these implementation details revolve around manipulating or modifying the respective algorithms so that it could accept soft *a priori* information from the decoder and are able to generate extrinsic information for use by the decoder.

The system model used in this chapter is based on the noise-whitened channel model described in Section 2.1. The relationship between the coefficients of the noise-whitened channel model and that of the colored-noise channel model is also established in this section. This thus lays the groundwork for the presentation of our proposed heuristic-based LS equalization algorithm in the next chapter since it utilizes the colored-noise channel model for an efficient implementation.

BER performance curves for the various equalization algorithms described in this chapter under different ISI conditions have been obtained and presented to illustrate the efficacy of these algorithms to mitigate the detrimental effects of ISI. Beside BER performance, an analytical treatment on the computational complexity for selected algorithms is also presented.

Chapter 3

Heuristic-Based Local Search Turbo Equalizer

In this chapter, we propose a novel class of SISO channel equalizer based on heuristic search methods as applied to a receiver employing turbo equalization. The local search (LS) algorithm, a modern problem-independent heuristic technique, is the focus here. We begin this chapter with a brief introduction into the rich field of heuristic search algorithms which were previously designed to solve large-scale optimization problems. The similarity of such problems to the equalization task is also drawn in this section. Following that, in Section 3.2, the heuristic-based LS turbo equalizer is formally presented. Through this derivation, it could be observed that the turbo equalization setup features a natural adoption of the LS algorithm due to the availability of a good quality initial solution for optimization to commence. Variants of the LS algorithm, namely, 1-Opt and *k*-Opt, will be described in this section, together with their reduced-complexity implementations. Then in Section 3.3, an analysis of the computational complexity of the proposed LS equalizer is provided. Following that, further investigation into the more powerful *k*-Opt LS algorithm is carried out through density evolution to derive its decoding threshold. Simulation results are then presented in Section 3.5 together with a discussion before we conclude the chapter with a summary in Section 3.6.

3.1 Introduction to Heuristic Search Methods

A heuristic [25]-[36] is a search method that is used to chance upon the optimum solution to an optimization problem in a short time. In comparison to exact approaches such as the BER-optimum FC-BCJR algorithm [11] described in Chapter 2 or the FER-optimum SOVA [13], a heuristic might not always find the optimum solution nor provide an assurance to find a solution within a certain range to that of the optimum. Nevertheless, the ability of heuristics to generate a good solution in reasonable time warrants serious consideration in its application to situations whereby the number of candidate (possible) solutions grows exponentially (usually in the solution length) such that simple enumeration schemes are rendered practically infeasible. In fact, both the BCJR algorithm and SOVA which work on the trellis could be considered as elegant enumeration schemes of all the possible solutions with a great reduction in the computational complexity from being exponential in trellis length to being exponential in the channel memory length.

Relating the equalization task to a particular class of optimization problems, i.e., combinatorial optimization problems, a similarity could be seen whereby both manipulate discrete decision variables which are usually binary. As such, based on the two key points mentioned above, we could thus relate the ISI channel equalization task to a nondeterministic polynomial-time (NP)-hard binary-constrained optimization problem, commonly known as a binary quadratic program in the area of combinatorial optimization. Heuristic search methods, typically employed to solve such a problem approximately within a reasonable time, could therefore be applied to the channel equalization task at hand.

To effectively apply heuristics to the equalization problem, a clear notion of solution quality [25] must exist through proper definition of an objective function. With the establishment of such an objective function in relation to the problem statement, any heuristic search method can then be applied and modified to suit the equalization problem. As a consequence, the proposed method outlined in this chapter is actually a general framework in which other heuristics could also be applied in a similar manner.

Among various heuristic search methods, LS algorithm [25][26][28] is the most widely used heuristic for solving large-scale optimization problems due to its relatively high efficiency. Essentially the LS algorithm is a form of improvement heuristics (as opposed to construction heuristics which construct feasible solutions for optimization problems from scratch; a well-known example of a construction heuristic is the greedy algorithm [25][31]) which takes a feasible solution as input and tries to find a better or "improved" solution by stepwise transition in its local neighborhood. This basic idea of a neighborhood search is problem independent. However, in our proposed implementation of the LS-based equalizer, we incorporate some form of problem-specific domain knowledge into the LS algorithm so as to permit a more efficient and effective algorithm for the equalization process to take place.

To enable the LS algorithm to search each neighborhood at each step more efficiently, we utilize the colored-noise channel model in its derivation to allow a more efficient implementation based on the computation of *gains* (of a new solution) and *difference of gains* to evaluate the quality of a solution instead of a direct explicit computation of its objective function. This will be clearer when we present the details in Section 3.2.3. To allow a more effective search to take place, we employ the best improvement approach by selecting the solution with the highest objective value at each optimization steps for use as input solution for the next step to commence, as opposed to the first improving solution found (first improvement approach). The best improvement approach is carried out at the first step of each turbo iteration by initializing the input solution with the hard estimates obtained from the *a priori* LLR fed back from the decoder derived in the previous iteration and the past bit estimates produced by the equalizer in the current iteration. This form of initialization is carried out as it utilizes the most reliable information at hand and thus likely to have a larger objective value, compared with a randomly chosen candidate vector from the solution space.

Ironically, the main disadvantage of the LS algorithm is a direct consequence of its advantageous point mentioned previously. Due to the fact that the LS algorithm is a form of improvement heuristics which serve to improve the quality of the input solution, the LS algorithm is thus susceptible to being trapped in a local optimum [25][28]. In general, all neighborhood search based algorithms are improvement heuristics and hence the performance outcome is highly dependent on the starting solution. This form of search is only guided by local information where no other information is utilized. As such, an inappropriate choice for the starting solution may lead to a local optimum with low objective value and thus large distance away from the optimum solution with respect to the objective function.

In the recent developments of heuristic search methods, hybrid methods that combine two or more search strategies, for example, memetic algorithms [25] which are hybrid of neighborhood search methods and evolutionary algorithms, were proposed to utilize the benefits of evolutionary algorithms to escape the predicament of being trapped in local optima, while preserving high efficiency through the use of neighborhood search methods. Specifically, memetic algorithms are designed to employ variation operators commonly used in evolutionary algorithms to perform "jump" in the search space to escape the attractor region of a local optimum with the aim of reaching the basin of attraction of another local optimum with higher objective value, and hence closer to the optimum solution. In a turbo equalization scheme, the presence of a SISO decoder can be visualized as an effective "operator" to facilitate such a jump since it accepts the interleaved local optimum solution from the LS-based equalizer and further process them using additional new information provided by the code constraints specified indirectly by the code trellis. As such, the turbo equalization setup is a natural format for the utilization of LS algorithm where after being trapped at a local optimum, the decoder may assist the LS-based equalizer in escaping to a better solution region in the next turbo iteration.

An overview of our proposed LS-based turbo equalizer is provided below:

(1) For the first search step (at each turbo iteration), initialize the input solution with the hard decisions derived from the *a priori* LLR fed back from the decoder obtained in the previous iteration and the APP LLR obtained by the equalizer in the current iteration.

(2) Based on the neighborhood of the input solution obtained in (1), find the candidate vector with the largest objective value and update this particular candidate vector as the input solution for the next search step to commence. Save all the candidate vectors that are encountered in each search steps to form a list of candidate vectors. In this thesis, the search process is carried out using variants of LS heuristics, namely 1-Opt LS and *k*-Opt LS.

(3) Compute the bitwise APP LLR based on the candidate list obtained in (2).

3.2 Heuristic-Based Turbo Equalizers

In the following subsections, we present the implementations details of our proposed heuristic-based turbo equalizer utilizing the LS algorithm. Before delving into these details, we first reformulate the equalization problem formally as in Section 2.1 so as to set the stage for the presentation of our heuristic-based LS turbo equalizer.

We begin by rewriting the colored-noise channel model described by (2.2) in matrix form as

$$\mathbf{y}_i = \mathbf{H}\mathbf{b}_i + \mathbf{w}_i \tag{3.1}$$

where the subscript i denotes the observation window for the detection of the ith transmitted bit b_i. In the channel model described by (3.1), the transmitted interleaved code bits vector is of length $(4m + 1)$ defined as $\mathbf{b}_i = [b_{i-2m}, \ldots, b_i, \ldots, b_{i+2m}]^T$ where the superscript T denotes matrix transpose. As before, m denotes the memory length of the equivalent noise-whitened ISI channel. The additive Gaussian noise $\mathbf{w}_i$ and the corresponding received channel observations $\mathbf{y}_i$ are both vectors of length $(2m + 1)$ defined as $\mathbf{w}_i = [w_{i-m}, \ldots, w_i, \ldots, w_{i+m}]^T$ and $\mathbf{y}_i = [y_{i-m}, \ldots, y_i, \ldots, y_{i+m}]^T$ respectively. The $(2m + 1)$ x $(4m + 1)$ channel convolution matrix $\mathbf{H}$ based on the colored-noise channel model (2.2) is written similarly to (2.17) as

$$\mathbf{H}=\begin{bmatrix}
h_{-m} & \cdots & h_0 & \cdots & h_m & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 \\
0 & h_{-m} & \cdots & h_0 & \cdots & h_m & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\
0 & 0 & h_{-m} & \cdots & h_0 & \cdots & h_m & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & \cdots & 0 & \ddots & \cdots & \ddots & & \ddots & \cdots & \ddots & 0 & \cdots & 0 & 0 \\
0 & 0 & \cdots & 0 & 0 & 0 & 0 & h_{-m} & \cdots & h_0 & \cdots & h_m & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & h_{-m} & \cdots & h_0 & \cdots & h_m & 0 \\
0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & h_{-m} & \cdots & h_0 & \cdots & h_m
\end{bmatrix} \tag{3.2}$$

From the presentations described in Chapter 2, it is noted that the trellis-based FC-BCJR algorithm is the BER-optimum equalization algorithm which could be modified to form the SW-BCJR algorithm that allows each bit to be processed in a serial manner by operating on a fixed memory span. From the simulation results, it is observed that the SW-BCJR algorithm suffers negligible performance degradation as compared to the FC-BCJR algorithm despite having a restricted observation window of length $D << L$, where L is the channel trellis length. As such, we adopt a similar sliding-window approach to yield a low-complexity method to approximate the APP LLR calculation of the interleaved code bits b_i stated in (2.5) and (2.6) by considering only the channel observations $\mathbf{y}_i$ of length $(2m + 1)$ instead of the entire received frame of length L as

$$\Lambda_i = \log \frac{P(b_i = +1 | \mathbf{y}_i)}{P(b_i = -1 | \mathbf{y}_i)} \tag{3.3}$$

$$= \log \frac{\sum_{\mathbf{v}_i \in \{-1,+1\}^{4m+1} : b_i = +1} P(\mathbf{v}_i \mid \mathbf{y}_i)}{\sum_{\mathbf{v}_i \in \{-1,+1\}^{4m+1} : b_i = -1} P(\mathbf{v}_i \mid \mathbf{y}_i)} \tag{3.4}$$

where the summations in the numerator and denominator of (3.4) are over all possible 2^{4m} binary vectors $\mathbf{v}_i$ of length $(4m + 1)$ associated with the i^{th} transmitted code bit hypothesis $b_i = +1$ and $b_i = -1$ respectively. Relating the conditional probability $P(\mathbf{v}_i \mid \mathbf{y}_i)$ in (3.4) to the channel model described by (3.1), the APP LLR can thus be expressed as

$$\Lambda_i = \log \frac{\sum_{\mathbf{v}_i \in \{-1,+1\}^{4m+1} : b_i = +1} \exp(\Omega(\mathbf{v}_i))}{\sum_{\mathbf{v}_i \in \{-1,+1\}^{4m+1} : b_i = -1} \exp(\Omega(\mathbf{v}_i))} \tag{3.5}$$

where $\Omega(v_i)$ is a metric for $(4m + 1)$-tuple candidate bit vector $\mathbf{v}_i$ defined by

$$\Omega(\mathbf{v}_i) = \frac{1}{2\sigma^2}\left(2\mathbf{y}_i^T \mathbf{v}_{i,m} - \mathbf{v}_{i,m}^T \mathbf{H} \mathbf{v}_i\right) + \frac{1}{2} \boldsymbol{\lambda}_i^T \mathbf{v}_{i,m} \tag{3.6}$$

Here, $\mathbf{v}_{i,m}$ is a binary vector of length $(2m + 1)$ defined as

$$\mathbf{v}_{i,m} = \left[v_{i-m}, \cdots, v_{i-1}, b_i, v_{i+1}, \cdots, v_{i+m}\right]^T \tag{3.7}$$

and $\boldsymbol{\lambda}_i$ is the *a priori* LLR vector obtained from the interleaved extrinsic output of the SISO decoder defined as

$$\boldsymbol{\lambda}_i = \left[L_a(b_{i-m}), \cdots, L_a(b_i), \cdots, L_a(b_{i+m})\right]^T \tag{3.8}$$

Viewing all the possible candidate vectors $\mathbf{v}_i$ as a binary vector space of dimension-$(4m + 1)$, the vector $\mathbf{v}_{i,m}$ could thus be visualized as a vector in the subspace of $\mathbf{v}_i$ as

$$\mathbf{v}_i = \left[\underbrace{v_{i-2m}, \cdots, v_{i-m-1}}_{m\text{ elements}}, \underbrace{\mathbf{v}_{i,m}}_{\text{vector of length }(2m+1)}, \underbrace{v_{i+m+1}, \cdots, v_{i+2m}}_{m\text{ elements}}\right]^T \tag{3.9}$$

The metric function $\Omega(\mathbf{v}_i)$ defined in (3.6) has a similar form as the branch metric of the FC-BCJR algorithm derived based on the colored-noise channel model stated in (2.11). Expressing the metric function in this manner avoids the occurrence of the squared-term as observed in the branch metric stated in (2.10) derived based on the whitened-noise channel model. The deliberate choice of the colored-noise channel model based on this consideration allows an efficient implementation of the LS algorithm to be carried out. This observation will become apparent when we present the efficient form of LS algorithms in Subsection 3.2.3.

Although the APP LLR Λ_i defined above has its computational complexity reduced by limiting the length of the observation window to $(2m + 1)$ instead of the entire channel frame size L, the summation in (3.5) is still over the entire binary vector space of dimension-$(4m + 1)$. To further reduce this exponential computational complexity (in m), we propose to approximate (3.5) by summing a relatively small number of terms that have large metric values as determined from (3.6) and hence dominate the calculation. In another words, instead of computing the metrics for all possible candidate vectors in the entire vector space of dimension-($4m$ + 1), we approximate the summations in (3.5) as

$$\Lambda_i = \log \frac{\sum_{\mathbf{v}_i \in \boldsymbol{\varepsilon}: b_i = +1} \exp(\Omega(\mathbf{v}_i))}{\sum_{\mathbf{v}_i \in \boldsymbol{\varepsilon}: b_i = -1} \exp(\Omega(\mathbf{v}_i))} \tag{3.10}$$

where $\boldsymbol{\varepsilon}$ is a subset of $\{+1, -1\}^{4m+1}$ comprising of a list of candidate vectors with large metric values. The size of this candidate list $\boldsymbol{\varepsilon}$ is adjustable and as such the complexity of (3.10) is proportional to the cardinality of $\boldsymbol{\varepsilon}$. For example, if the list is selected to be $\boldsymbol{\varepsilon} = \{+1, -1\}^{4m+1}$, then (3.10) is equivalent to (3.5) and the entire vector space is searched. The task now is to select an appropriate list $\boldsymbol{\varepsilon}$ such that this it provides a good approximation to the computation of (3.5) through the use of (3.10) without involving an exhaustive search over the entire vector space in question.

A few important considerations have to be taken into account when generating $\boldsymbol{\varepsilon}$ so as to minimize performance loss and restrict the focus on a targeted space of candidate vectors with large metric values. These considerations are listed below:

(A1) The candidate list $\boldsymbol{\varepsilon}$ used in (3.10) for the computation of the APP LLR of the i^{th} transmitted code b_i must contain at least two vectors $\mathbf{v}'_i$ and $\mathbf{v}''_i$ described as

$$\mathbf{v}'_i = [v'_{i-2m}, \cdots, v'_{i-1}, b'_i, v'_{i+1}, \cdots, v'_{i+2m}] \tag{3.11}$$

$$\mathbf{v}''_i = [v''_{i-2m}, \cdots, v''_{i-1}, b''_i, v''_{i+1}, \cdots, v''_{i+2m}] \tag{3.12}$$

where $b'_i b''_i = -1$. This is to prevent either the numerator or denominator of (3.10) being zero due to an absence of the respective candidate vector in the list, thereby resulting in a negative or positive infinite LLR.

(A2) To minimize performance loss, the candidate list **ε** should contain candidate vectors that correspond to large metric values.

(A3) From the metric function $\Omega(\mathbf{v}_i)$ described by (3.6), it could be observed that the metric value is heavily dependent on $\mathbf{v}_{i,m}$ relative to $\mathbf{v}_i$. Since the size of the search space is directly related to the dimension of the candidate vector, we could thus restrict the size of the search space by finding large metric value vectors of $\mathbf{v}_{i,m}$ instead of $\mathbf{v}_i$. This restriction in the search space thus allows a more focus search to be carried out while at the same time, contributes to further computational complexity reduction when we generate **ε**.

An obvious choice for a particular binary vector $\mathbf{v}_i$ in the list **ε** with large metric value can be obtained by taking a hard decision on the *a priori* LLR fed back from the decoder derived in the previous iteration and using the bit estimates delivered by the equalizer in the current iteration. In another words, we could select this vector to be

$$\mathbf{v}_i = \left[\underbrace{\tilde{b}_{i-2m}, \cdots, \tilde{b}_{i-m-1}}_{m \text{ precursor bits}}, \underbrace{\hat{b}_{i-m}, \cdots, \hat{b}_i, \cdots, \hat{b}_{i+m}}_{\mathbf{v}_{i,m}}, \underbrace{\hat{b}_{i+m+1}, \cdots, \hat{b}_{i+2m}}_{m \text{ postcursor bits}} \right]^T \tag{3.13}$$

where the m precursor bits in (3.13) are obtained based on the APP LLR (of past estimates) derived by the equalizer in the current iteration as

$$\tilde{b}_j = \begin{cases} +1 & \text{where } \Lambda_j > 0 \\ -1 & \text{where } \Lambda_j \leq 0 \end{cases} \quad \text{for } j \in [i-2m, i-m-1] \tag{3.14}$$

and the remaining $(3m + 1)$ elements in $\mathbf{v}_i$ described by (3.13) are derived from the *a priori* LLR fed back from the decoder obtained in the previous iteration as

$$\hat{b}_j = \begin{cases} +1 & \text{where } L_a(b_j) > 0 \\ -1 & \text{where } L_a(b_j) \leq 0 \end{cases} \quad \text{for } j \in [i-m, i+2m] \tag{3.15}$$

The rationale behind this particular choice lies with the fact that this candidate vector described by (3.13) is obtained from the most reliable information at hand and hence likely to be estimated well with a larger metric value compared with a randomly selected vector from the solution space $\{+1, -1\}^{4m+1}$.

Based on the consideration stated in (A3), the m precursor bits and m postcursor bits defined in (3.13) for the generation of the APP LLR of the ith code bit b_i is hence fixed. As such, the task of generating a suitable list $\boldsymbol{\varepsilon}$ comprising of length-($4m$ + 1) candidate vectors $\mathbf{v}_i$ having large metric values thus reduces to a search for a list of appropriate vectors $\mathbf{v}_{i,m}$ of length ($2m$ + 1). In other words, the candidate list $\boldsymbol{\varepsilon}$ contains a list of length-(4m + 1) vectors with variations in its elements only in the positions specified by $\mathbf{v}_{i,m}$. In general, (3.16) depicts the candidate list $\boldsymbol{\varepsilon}$ once populated, where the first vector $\mathbf{v}_i^{(0,0)}$ is identical to that described by (3.13).

$$\boldsymbol{\varepsilon} = \left\{ \begin{array}{ccccccccc} \mathbf{v}_i^{(0,0)} & = & [\tilde{b}_{i-2m}, & \cdots, & \tilde{b}_{i-m-1}, & \mathbf{v}_{i,m}^{(0,0)}, & \hat{b}_{i+m+1}, & \cdots, & \hat{b}_{i+2m}]^T \\ \mathbf{v}_i^{(0,1)} & = & [\tilde{b}_{i-2m}, & \cdots, & \tilde{b}_{i-m-1}, & \mathbf{v}_{i,m}^{(0,1)}, & \hat{b}_{i+m+1}, & \cdots, & \hat{b}_{i+2m}]^T \\ \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\ \mathbf{v}_i^{(a,b)} & = & [\tilde{b}_{i-2m}, & \cdots, & \tilde{b}_{i-m-1}, & \mathbf{v}_{i,m}^{(a,b)}, & \hat{b}_{i+m+1}, & \cdots, & \hat{b}_{i+2m}]^T \\ \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\ \mathbf{v}_i^{(0,2m+1)} & = & [\tilde{b}_{i-2m}, & \cdots, & \tilde{b}_{i-m-1}, & \mathbf{v}_{i,m}^{(0,2m+1)}, & \hat{b}_{i+m+1}, & \cdots, & \hat{b}_{i+2m}]^T \\ \mathbf{v}_i^{(1,0)} & = & [\tilde{b}_{i-2m}, & \cdots, & \tilde{b}_{i-m-1}, & \mathbf{v}_{i,m}^{(1,0)}, & \hat{b}_{i+m+1}, & \cdots, & \hat{b}_{i+2m}]^T \\ \mathbf{v}_i^{(1,1)} & = & [\tilde{b}_{i-2m}, & \cdots, & \tilde{b}_{i-m-1}, & \mathbf{v}_{i,m}^{(1,1)}, & \hat{b}_{i+m+1}, & \cdots, & \hat{b}_{i+2m}]^T \\ \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\ \mathbf{v}_i^{(1,2m+1)} & = & [\tilde{b}_{i-2m}, & \cdots, & \tilde{b}_{i-m-1}, & \mathbf{v}_{i,m}^{(1,2m+1)}, & \hat{b}_{i+m+1}, & \cdots, & \hat{b}_{i+2m}]^T \end{array} \right\} \quad (3.16)$$

Here, the superscript (a,b) is used to denote the different vectors $\mathbf{v}_{i,m}$ encountered in the search process. In the following subsections, we would give a description of the LS algorithm, which is a highly efficient approach to generate such a candidate list.

3.2.1 1-Opt Local Search Algorithm

Essentially, as the name suggests, the LS (Local Search) algorithm [25][28][29] takes a feasible solution as input and tries to search for a better solution by stepwise transition in the vicinity of the input solution. A generalization of the LS algorithm to the channel detection problem yields a fixed-radius sphere search algorithm with the center of the sphere fixed at the input solution. The quality of the solutions generated by the LS algorithm is thus heavily dependent on the quality of the input solution. As such, the vector defined in (3.13) is used as an input solution or specifically, as a starting vector to commence the search.

Since we are required to find a candidate list based on large metric value vectors due to variations only in $\mathbf{v}_{i,m}$, we could thus utilize the metric function described in (3.6) as a form of objective (or cost) function in deciding which possible vectors $\mathbf{v}_{i,m}$ to use when populating the candidate list $\boldsymbol{\varepsilon}$. In another words, at each search step p, the 1-Opt LS algorithm is to produce an updated vector $\mathbf{v}^{(p+1)}$ based on the previous estimate $\mathbf{v}^{(p)}$ by maximizing the objective function $\Omega(\mathbf{v}_i)$ described in (3.6) as

$$\mathbf{v}^{(p+1)} = \arg\left\{ \max_{\mathbf{v} \in N_1\left(\mathbf{v}^{(P)}\right)} \Omega(\mathbf{v}_i) \right\} \tag{3.17}$$

where the subscript (i,m) in $\mathbf{v}^{(p+1)}$ and $\mathbf{v}^{(p)}$ described in (3.17) has been dropped to avoid unnecessary cluttering. As mentioned, the initial solution $\mathbf{v}^{(0)}$ to commence the search is set to (3.13) and $N_1(\mathbf{v}^{(p)})$ is defined as a Hamming sphere with radius-1 that consists of all possible binary vectors that differ from the central vector $\mathbf{v}^{(p)}$ by one element, i.e.,

$$N_1\left(\mathbf{v}^{(p)}\right) = \left\{ \mathbf{v} \in \{+1,-1\}^{2m+1}; \left\| \mathbf{v} - \mathbf{v}^{(p)} \right\|_H \leq 1 \right\} \tag{3.18}$$

where $\|\cdot\|_H$ denotes the Hamming weight of its vector argument. The final candidate list will then be formed by all trial solutions encountered in the LS procedures at every step.

For a better appreciation of the 1-Opt LS algorithm, consider an initial input solution vector of length-4 described by $\mathbf{v}^{(0)} = [+1, +1, +1, +1]^T$. For the first search step, i.e., $p = 1$, the 1-Opt neighborhood of $\mathbf{v}^{(0)}$ thus consists of four vectors $[-1, +1, +1, +1]^T$, $[+1, -1, +1, +1]^T$, $[+1, +1, -1, +1]^T$ and $[+1, +1, +1, -1]^T$.

Assuming that the updated vector $\mathbf{v}^{(1)}$ selected based on (3.17) is the vector $\mathbf{v}^{(1)} = [-1, +1, +1, +1]^T$, then for the second search step, i.e., $p = 2$, the 1-Opt neighborhood of $\mathbf{v}^{(1)}$ thus consists of another 4 vectors $[+1, +1, +1, +1]^T$, $[-1, -1, +1, +1]^T$, $[-1, +1, -1, +1]^T$ and $[-1, +1, +1, -1]^T$. The updated vector $\mathbf{v}^{(1)}$ stated here is specially selected such that the first vector stated in the 1-Opt neighborhood of $\mathbf{v}^{(1)}$ is identical to initial input solution $\mathbf{v}^{(0)}$. As such, this particular candidate list $\boldsymbol{\varepsilon}$ generated by a 1-Opt LS with search step $p = 2$ for an input vector of length-4 consists of 8 distinct vectors (including the initial solution) as stated above.

In general, at the end of a p-step 1-Opt LS optimization carried out with a starting vector of length-$(2m + 1)$, the candidate list $\boldsymbol{\varepsilon}_{\text{1-Opt}, p}$ consists of all trial solutions encountered in the LS procedure from which the APP LLR of bit b_i can be approximated according to (3.10). The cardinality of this candidate list is thus $|\, \boldsymbol{\varepsilon}_{\text{1-Opt}, p} \,| \leq 1 + p(2m + 1)$ and hence the 1-Opt LS equalizer developed here has linear computational complexity (in terms of the noise-whitened channel memory length m).

3.2.2 *k*-Opt Local Search Algorithm

The basic principle behind the 1-Opt LS algorithm described in the previous section could be extended to derive a more powerful LS algorithm called the *k*-Opt LS algorithm [25] where the *k*-Opt neighborhood consists of all possible binary vectors that differ from its center vector by one up to *k* elements. For example, a 2-Opt LS algorithm is realized by flipping one up to two elements to reach a neighboring solution. As such, the size of a *k*-Opt neighborhood centered on a vector $\mathbf{v}^{(q)}$ of length $(2m + 1)$ is

$$\left|N_k\left(\mathbf{v}^{(q)}\right)\right| = \sum_{j=0}^{k}\binom{2m+1}{j} \tag{3.19}$$

where the superscript q is used to denote the search step for *k*-Opt LS algorithm similar to the superscript p that is used to denote the search step for 1-Opt LS algorithm.

From (3.19), it is easy to conceive that the computational complexity to search a complete *k*-Opt neighborhood is very high. To efficiently search a subset of the *k*-Opt neighborhood, a divide and conquer approach based on the principle of Lin-Kernighan algorithm [27] used for solving the well-known traveling salesperson problem (TSP) could be applied. The basic idea behind this approach is to decouple the *k*-Opt LS search operation into successive 1-Opt LS procedures. At each *k*-Opt search step q, a variable number of elements in the current solution, i.e., the vector fixed at center of search sphere, are flipped to arrive at a better neighboring solution. For a $(2m + 1)$-tuple candidate center vector considered in this chapter, a list of $(2m + 1)(m + 1) = (2m^2 + 3m + 1)$ solutions is produced at each *k*-Opt step. Firstly, each bit of the input solution is flipped exactly once so that all the solutions in the list are distinct. The update estimate from this 1-Opt LS algorithm is then used as the input solution for a next 1-Opt LS algorithm. Both 1-Opt LS algorithms utilized here are embedded within one *k*-Opt search step. The solution derived by the first 1-Opt LS algorithm may differ in one up to $(2m + 1)$ elements from the initial solution.

For clarity purposes, the pseudo code of the k-Opt LS algorithm for the ISI channel detection problem is depicted below as Algorithm I.

Algorithm I – (k-Opt LS for Soft-Output ISI Channel Detection)

```
(1) Initialization: Obtain the initial solution of length-$(2m + 1)$ $\mathbf{v}^{(0)}$ as in (3.13) and compute the best objective (i.e., metric) value and the best solution based on this initial solution as $\Omega(\mathbf{v}^{(0)}) \rightarrow \Omega'$ and $\mathbf{v}^{(0)} \rightarrow \mathbf{v}'$;
(2) For search step $q = 1, \ldots, Q$ where $Q$ is the total number of $k$-Opt LS search steps:
        a. Let $\mathbf{v}$ denote the current solution $\mathbf{v}^{(q-1)} \rightarrow \mathbf{v}$. Generate a set $\mathbf{C} = \{1, \ldots, 2m + 1\}$ to record positions on which the elements of $\mathbf{v}$ will be flipped;
        b. Find the best neighboring solution $\mathbf{v}_i$ by flipping elements recorded in $\mathbf{C}$, using $\Omega(\mathbf{v}_i) \geq \Omega(\mathbf{v}_j) \forall j \in \mathbf{C}$, where $\mathbf{v}_i$ ($\mathbf{v}_j$, respectively) differs from $\mathbf{v}$ by only the $i$th ($j$th, respectively) element;
        c. If $\Omega(\mathbf{v}_i) \geq \Omega'$, update $\Omega(\mathbf{v}^{(0)}) \rightarrow \Omega'$ and $\mathbf{v}_i \rightarrow \mathbf{v}'$ accordingly;
        d. Reduce the density of $\mathbf{C}$ by excluding the $i$th position as $\mathbf{C} = \mathbf{C} \setminus \{i\}$. Set $\mathbf{v}_i \rightarrow \mathbf{v}$ and repeat from step b until $\mathbf{C} = \Phi$;
        e. If $\Omega' \geq \Omega(\mathbf{v}^{q-1})$, set $\Omega' \rightarrow \Omega(\mathbf{v}^{(q)})$ and $\mathbf{v}' \rightarrow \mathbf{v}^{(q)}$;
(3) End. Store all trial solutions encountered in the $k$-Opt LS procedures described above in the candidate list $\boldsymbol{\varepsilon}_{k\text{-Opt},\, q}$. Compute APP LLR using (3.10) with this list.
```

The cardinality of $\boldsymbol{\varepsilon}_{k\text{-Opt},\, q}$ is thus given by $|\, \boldsymbol{\varepsilon}_{k\text{-Opt},\, q} \,| \leq 1 + q(2m + 1)(m + 1)$. Hence, the k-Opt LS detector based on the Lin-Kernighan implementation has a square order computational complexity (in terms of the noise-whitened channel memory length m).

From the understanding of the operational procedures carried out by the LS algorithm described previously, it is interesting to compare the construction of the candidate list by the LS algorithm and the list-sphere decoding algorithm as described in [45]. The list-sphere decoder constructs a list of candidate symbol vectors by using a modification of the traditional real (or complex) sphere decoder to search a sphere around the zero-forcing (ZF) symbol detector based on a channel-dependent distance function. By doing so, it has to store all possible values of the real (or complex) lattice set. In contrast, the LS algorithm uses a bit vector obtained from other decoding stages as the center of a binary Hamming sphere, where the center and radius of the sphere is updated and fixed respectively, at each search step. The symbol vectors corresponding to binary vectors described here within the sphere are then used to construct a candidate list. Hence, the LS algorithm can thus be viewed as a binary list-sphere decoder that constructs the candidate list directly from the bit level.

3.2.3 Low Complexity Implementation of Local Search Algorithm

The most computationally intensive part of LS algorithm hinges on the calculation of the metric value or the objective function given by (3.6) for every candidate vector encountered in the LS procedures. A direct evaluation of this objective function for a particular candidate vector would require $O(m^2)$ floating point operations, or to be specific, $4m^2 + 8m + 3$ additions and 2 multiplications.

In LS algorithms, the objective function can always be computed based on a neighboring candidate vector or solution that differ by only one bit from the old solution. As the objective value of the old solution is known, the objective value of the new solution can be obtained by simply focusing on the bit that has changed in value. In other words, the structure of the local neighborhood searched by LS algorithms could be exploited to compute the metric values of all candidate vectors in each 1-Opt vicinity by drawing our attention to the gain g_j associated with the change in the jth bit position.

For simplicity of notations, let $\mathbf{v}' = [v_1, \cdots, v_{j-1}, -v_j, v_{j+1}, \cdots, v_{2m+1}]^T$ denote a neighboring solution that differs from the current solution $\mathbf{v}$ by only the jth element. After some simplifications, the gain g_j associated with flipping the jth element in $\mathbf{v}$ could thus be calculated as

$$g_j = \Omega(\mathbf{v}') - \Omega(\mathbf{v}) \tag{3.20}$$

$$= \frac{1}{2\sigma^2}\left[-4v_j\left(y_{i,j} - \sum_{n=1, n\neq j+m}^{4m+1} H_{j,n}\bar{v}_n\right)\right] - \lambda_j v_j \tag{3.21}$$

where v_j is the jth element in the vector $\mathbf{v}_{i,m}$ and $\bar{v}_n$ is the nth element in the vector $\mathbf{v}_i$ and λ_j is the jth element in the vector $\boldsymbol{\lambda}_i$ as described by (3.7), (3.9) and (3.8) respectively. As such, evaluating a candidate vector can be accomplished in $(2m + 1)$ additions and 2 multiplications. With a 1-Opt neighborhood size of $(2m + 1)$, we would thus require $O(m^2)$ operations, or to be specific, $(2m + 1)(2m + 2) = (4m^2 + 6m + 2)$ additions and $(4m + 2)$ multiplications, to compare the metric values of all the candidate vectors in this neighborhood.

From the simulation results that are to be shown in the penultimate section, it could be observed indirectly through BER performances that several search steps, i.e., $p > 1$ for 1-Opt LS or $q > 1$ for k-Opt LS, are usually required in LS algorithms to successively improve the quality of the solution generated. As such, instead of computing the gains at each search step using (3.21), a more efficient approach is to consider only the difference of gains from the second search step onwards.

In more detail, we consider the computation of the difference of gain g'_l at the second search step due to a flipping of the lth element in $\mathbf{v}'$, where we assume that $\mathbf{v}'$ is selected as the input solution for the second 1-Opt search step. For ease of understanding, denote the input solution vector at the second search step as $\mathbf{v}' = [v'_1, \cdots, v'_{l-1}, v'_l, v'_{l+1}, \cdots, v'_{2m+1}]^T$ which differs from the initial input solution **v** at the first search step by the jth element. Let $\mathbf{v}'' = [v'_1, \cdots, v'_{l-1}, -v'_l, v'_{l+1}, \cdots, v'_{2m+1}]^T$ denotes the candidate vector encountered in the second search step with one bit different from the center $\mathbf{v}'$ of the corresponding sphere. Since we are considering the second search step, it is thus implied that all the gains g_j for $j = \{1, \ldots, 2m+1\}$, associated with flipping the jth element in the initial solution **v**, have already been obtained from (3.21). Hence, the difference of gain g'_l associated with flipping the lth element in $\mathbf{v}'$ with respect to the initial vector **v** can then be obtained efficiently as

$$g'_l = \Omega(\mathbf{v}'') - \Omega(\mathbf{v}') \tag{3.22}$$

$$= \begin{cases} -g_j & l = j \\ g_l + \Delta g_{l,j} & \text{otherwise} \end{cases} \tag{3.23}$$

where

$$\Delta g_{l,j} = -8 v_l v_j H_{l,j+m} \tag{3.24}$$

As such, the evaluation of the objective values of all the (2m + 1) 1-Opt neighboring solutions starting from the second search step can thus be achieved in O(m) operations, or to be specific, (4m + 2) additions only.

To further reduce computational complexity, we could apply the abovementioned concept of gains to the initialization of the objective values for each of the first input solutions derived at each turbo iteration. A direct implementation of this initialization based on (3.6) would require $L(4m^2 + 8m + 3)$ additions and $2L$ multiplications. Since we are detecting one bit at a time, a more efficient method is to first calculate the objective values of an L-tuple sequence formed based on the tentative hard estimates fed back from the outer decoder in the previous iteration. If the refined bit estimate at the ith bit position delivered by the LS-based equalizer at the current iteration turns out to be different from the initial estimate, we can then update the metric value of the L-tuple sequence efficiently using the gain already obtained in the previous LS procedures. The updated metric value could then be used as the initial value for the detection of the $(i + 1)$th bit. In this manner, the number of operations required for initialization is at most $(2mL + 4L)$ additions and 2 multiplications.

The implementation details of the proposed LS-based turbo equalizer have been completed. To further reinforce the understanding of the essential principles in this section, the schematic diagram of our proposed LS-based turbo equalizer is shown in Figure 3.1. The structure shown here is similar to the original turbo equalizer proposed by Douillard *et al*. depicted in Chapter 1, except with the explicit inclusion of an additional feedback path where hard decision is taken on the *a priori* LLR to deliver the tentative hard estimates used as a starting input vector for LS algorithm to commence.

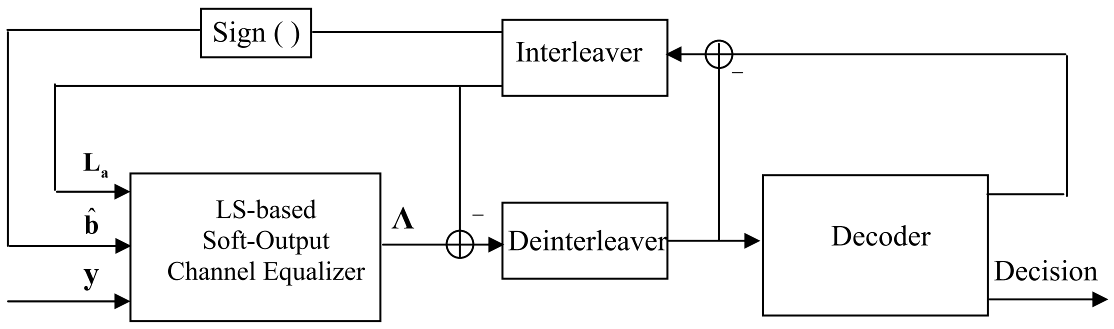

Figure 3.1: Turbo equalizer based on LS algorithms for coded ISI channels

3.3 Computational Complexity Analysis of Local Search Algorithm

The computational complexities of the respective LS algorithms, namely 1-Opt and *k*-Opt, are directly proportional to the cardinality of their respective derived candidate list and the subsequent computation of the candidate vectors' objective values. For a fair comparison with the other equalization algorithms described in Chapter 2, we compute the exact number of required operations based on these two aspects for the detection of each bit. The respective numbers of required operations shown in Table 3.1 below is based on the efficient implementation described in Section 3.2.3. The cost of initialization is also included in the calculation of the computational complexities for the LS algorithms.

Table 3.1:
Number of required operations for the detection of one transmitted bit per iteration using varying SISO equalization algorithms where
m: Channel memory length; D: SW-BCJR decision delay; N: Equalizer filter length;
p: Number of search steps for 1-Opt LS;
q: Number of search steps for k-Opt LS.

Equalization Algorithms	Number of real additions	Number of real multiplications
1-Opt LS	$2m + 4 + (2m + 1)(2m + 2p)$	$4m + 2$
k-Opt LS	$2m + 4 + (2m + 1)(4m + 2)q$	$4m + 2$
SW-BCJR	$2^m \times (2mD + 7D + 2m + 10)$	4×2^m
MMSE EXACT	$8N^2 + 2m^2 - 10N + 6m + 8$	$16N^2 + 4m^2 + 18m - 4N + 10$
MMSE APRX I	$4N + 4m$	$4N + 8m + 8$
MMSE APRX II	$10m + 8$	$10m + 10$

At a quick glance, both the 1-Opt LS and the *k*-Opt LS equalizers have a square order computation complexity in terms of the channel memory length *m*. The SW-BCJR equalizer has an exponential computation complexity in *m* whereas the time-varying MMSE equalizer has a square order computational complexity in terms of both *m* and the filter length, *N*. The approximate MMSE implementations have linear complexity.

3.4 Decoding Threshold of *k*-Opt Equalizers

Density evolution (DE) [37]-[39] is a useful technique to analyze the convergence of iterative algorithms. The details of DE analysis for ISI channels can be found in [5], [38] and [39]. The basic idea is to introduce the assumption that the sequence transmitted over the channel is identically and uniformly distributed (i.u.d). In this section, we briefly describe how to use the DE technique to predict the decoding thresholds for LS-based turbo equalizer.

When applied to iterative equalization and decoding, DE analysis involves the computation of the probability density function (PDF) of the extrinsic information exchanged locally within each of the respective SISO modules and also globally between the different SISO modules. Specifically, the PDF of the extrinsic information f_d at the output of the equalizer can be expressed as a function of the PDF of the extrinsic information f_o at its input, where the subscripts d and o are used to denote extrinsic information delivered by the equalizer and the decoder respectively. Since no analytical expression is known, a Monte Carlo approach [38] is used to obtain the transfer function F_d of the equalizer, i.e.,

$$f_d^{(q)} = F_d\left(f_o^{(q-1)}, E_b/N_o\right) \tag{3.25}$$

where the superscript q denotes the q^{th} turbo iteration between the equalizer and decoder. Here, q could be any integer from one up to Q where Q is the predetermined number of iterations normally initialized in a turbo equalization scheme. Due to the nonlinear nature of ISI channels, the sequence input to the channel is not assumed to be the all-zero vector but is a sufficiently long randomly selected i.u.d sequence and the *a priori* input for the equalizer is generated according to the PDF of the extrinsic LLR (observed through a histogram approach) from the decoder. The PDF f_d is then obtained by observing the histogram of the equalizer output LLR. In a similar vein, we can also obtain the transfer function F_o of the decoder which is given by

$$f_o^{(q)} = F_o\left(f_d^{(q)}\right) \tag{3.26}$$

where an all-zero input sequence can be used here to approximate f_o. From (3.25) and (3.26), we can then represent the iterative receiver as a single-parameter system as

$$f_o^{(q)} = F_o\left(F_d\left(f_o^{(q-1)}, E_b/N_o\right)\right) \tag{3.27}$$

By definition, the decoding threshold is stated as the minimum SNR value E_b/N_o described above in (3.27) in which the decoding errors approach zero when the codeword length L and the total number of iterations Q are increased to infinity [38]. In a mathematical representation, the decoding threshold C_{th} of the iterative system could thus be written as

$$C_{th} = \inf_{E_b/N_o} \left\{ E_b \middle/ N_o : \lim_{L\to\infty} \lim_{q\to\infty} \int_{-\infty}^{0} f_{o,\text{app}}^{(q)}(\xi) d\xi \right\} \tag{3.28}$$

where $f_{o,\text{app}}^{(q)}$ denotes the PDF of the *a posteriori* output from the decoder which could be obtained by

$$f_{o,\text{app}}^{(q)} = f_d^{(q)} \otimes f_o^{(q)} \tag{3.29}$$

As before, the symbol $\otimes$ is used to denote the convolution operator.

Based on the above procedures outlined, the decoding thresholds for the k-Opt LS-based Turbo equalizer as employed in Channel II (Proakis C Channel) described in Chapter 2 for the number of search steps q = 1 and q = 2 are 4.5 dB and 4.2 dB respectively. In the next section, we would show that the decoding thresholds obtained here can be used to predict the waterfall region that is a typical phenomenon in the BER plot of an iteratively decoded system.

3.5 Simulation Results and Discussion

In this section, we present the BER performance results of our proposed heuristic-based LS turbo equalizer. Key results of the 1-Opt LS and *k*-Opt LS turbo equalizer with varying search steps *p* and *q* respectively will be shown. All the general system parameters described in Chapter 2 remains identical here so as to allow a fair comparison with the trellis-based and filter-based equalization algorithms described in the previous chapter. The only difference lies in the receiver structure whereby the LS-based turbo equalizer utilized is shown in Figure 3.1 while all others utilized that depicted in Figure 1.2.

In addition, since it is desirable for the LS-based turbo equalizer to receive a good quality input solution for the search to commence, we utilize the time-invariant MMSE filter derived based on the zero *a priori* LLR scenario (stated in Section 2.3.2 A)) for the first turbo iteration and thereafter switch the equalization task to the LS equalizer for all subsequent iterations. It is to be noted here that this choice is not a necessity; any other equalization algorithms are also possible. Essentially, the intention is to find an equalizer implementation based on the lowest possible computational complexity while providing a relatively good quality initial solution for the LS equalizer to commence its search. In another words, the proposed LS-based equalizer can also be used in the first iteration. However, due to an absence of the *a priori* LLR at the first iteration, a good quality input solution cannot be formed and as such, a randomly chosen input solution from the solution space would be selected to commence the search. Typically, a randomly chosen input solution could be the all-zero vector or the all-one vector. Consequently, this random choice will, in general, have a low objective value and therefore increase the number of iterations required to realize the potential BER improvement. For all simulation results presented here, it will be implicit that the first turbo iteration is carried out by the abovementioned time-invariant MMSE filter.

First, we consider turbo equalization over the mild ISI conditions described by Channel I in Table 2.2 given in the previous chapter. The BER performances over this channel are shown in Figure 3.2. Here, several LS-based equalizers are considered in the simulation studies, namely, the 1-Opt LS equalizer with search steps $p = 1$ and $p = 2$, and the k-Opt LS equalizer with search step $q = 1$. For comparison purposes, the BER performance curves of the various equalization algorithms presented in Chapter 2 are also replicated here. As observed, the 1-Opt LS equalizer has a performance loss of about 0.7 dB compared with the FC-BCJR equalizer at a BER of 10^{-2}. Increasing the value of p to 2 reduces the loss to 0.4 dB. Though not shown here, using more search steps (e.g., $p = 4$) for 1-Opt LS does not translate to much more performance gain. As such, we switch our attention to the more powerful k-Opt LS algorithm with a search step $q = 1$ and discover that it outperforms all other equalization algorithms and is very close in performance to the FC-BCJR equalizer. For the k-Opt LS with $q = 1$, the equalizer searches 46 distinct candidate vectors per bit as opposed to the 1-Opt LS with $p = 1$ and $p = 2$ which only searches 10 and 19 distinct candidate vectors per bit respectively.

Next, we carry out simulation studies over the severe ISI channel denoted as Channel II in Table 2.2 of previous chapter. As mentioned before, this channel is also known as the Proakis C Channel in [40]. All the performance curves for the various equalization algorithms described in Chapter 2 are similarly shown here in Figure 3.3 for ease of comparison. We carry out simulation studies using more powerful LS algorithms, namely the 1-Opt LS with a search step $p = 3$ and the k-Opt LS with a search step q = 1 and q = 2. As observed in Figure 3.3, the 1-Opt LS, despite having a larger search step of p = 3, is not able to effectively mitigate the detrimental effects of ISI introduced by this channel. However, a significant performance gain of about 1.5 dB at a BER of 10^{-3} is attained simply by switching to the k-Opt LS with a search step of q = 1. Noting that the k-Opt LS with q = 1 still has some performance gap to the SW-BCJR performance curve, we increase the search step to q = 2 and discover that a further improvement of approximately 0.8 dB is realized at this BER of 10^{-3}. At this BER, the proposed k-Opt LS with q = 2 outperforms the M-BCJR algorithm with M = 5 and even that of the time-varying MMSE equalizer. The k-Opt LS with q = 2 is also able to attain the ISI-free bound ("AWGN" curve) much earlier than both the M-BCJR and the time-varying MMSE equalizer at an SNR of approximately 5.3 dB.

In Figures 3.4 and 3.5, we present BER performance results of some selected LS-based equalizers as a function of the number of iterations under the two ISI conditions described by Channel I and Channel II respectively. In general, the number of iterations needed to converge to the performance of the SW-BCJR equalizer is a function of the SNR values. Convergence of a turbo equalizer is attained when further iterations does not translate to any additional BER improvements. As observed in Figure 3.4, increasing the search step from p = 1 to p = 2 for the 1-Opt LS leads to a much faster convergence. The switching from a 1-Opt LS to the more powerful k-Opt LS also results in a similar phenomenon. Comparing only the k-Opt LS with that of the time-varying MMSE equalizer in Figures 3.4 and 3.5, it could be observed that in general, under various ISI conditions and SNR values, the k-Opt LS is able to converge much faster than the time-varying MMSE equalizer and in some cases, the convergence for the k-Opt LS even occurs at a much lower BER that that of the time-varying MMSE equalizer. For example, with reference to Figure 3.5, the *k*-Opt LS (with $q = 2$) is able to attain convergence using approximately 12 iterations whereas that of the time-varying MMSE equalizer requires approximately 3 more iterations. Further, though the time-varying MMSE equalizer requires more iterations for convergence, the corresponding BER attained is still slightly worse off than the *k*-Opt LS by half an order of magnitude difference.

In general, a faster rate of convergence for a particular turbo equalizer implementation means fewer numbers of turbo iterations are required to attain a particular BER. This thus translates to a shorter decoding delay and also indirectly lowers the overall computational complexity of the receiver system. To be more specific, if a turbo equalizer employs a low-complexity equalization algorithm but can only attain the desired BER performance after many rounds of iterations, the benefits of its low complexity implementation may thus be significantly reduced. On the other hand, a slightly more computationally intensive algorithm with a comparatively smaller number of iterations needed to attain that desired BER performance may be highly preferred instead. From the rate of convergence and the corresponding BER attained at convergence shown in Figures 3.4 and 3.5 together with their respective computational complexities shown in Table 3.1 in this chapter, we can conclude that the *k*-Opt LS with $q = 1$ and $q = 2$ offer the best of both worlds, in terms of faster convergence with low complexity and excellent BER performances, as compared with all other equalization algorithms described in this thesis. In general, it is also observed that with an increase in the severity of the ISI, an improvement in performance can be attained by switching the search step of the *k*-Opt LS from $q = 1$ to $q = 2$.

Since the *k*-Opt LS equalization algorithm offers an excellent alternative to the trellis-based and filter-based equalization algorithms when employed in a turbo setup, we further illustrate their BER performances curves in Figures 3.6 and 3.7 as the number of iterations proceed for $q = 1$ and $q = 2$ respectively. The respective decoding thresholds (denoted as C_{th}) determined by density evolution are also shown in the figures to mark the start of the waterfall region.

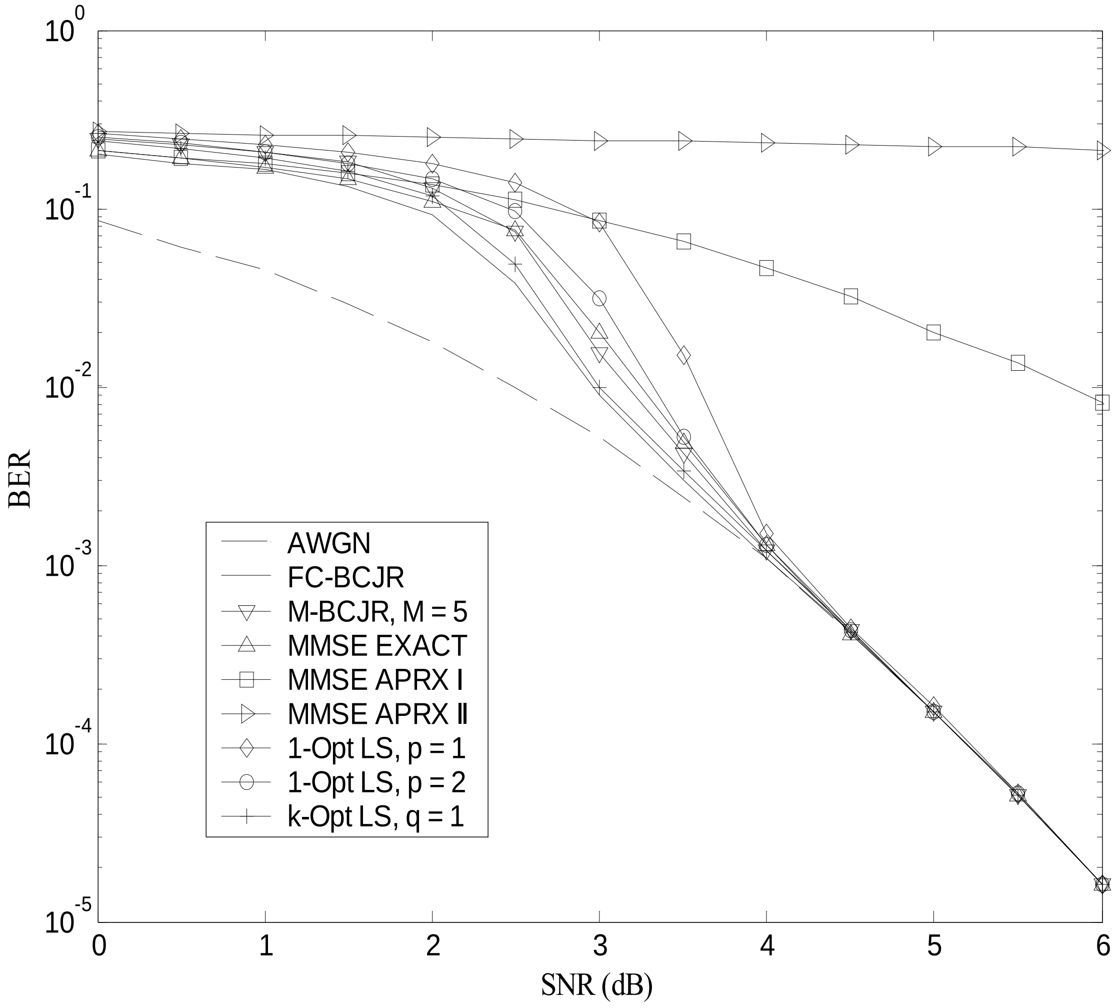

Figure 3.2: Performance comparisons between the proposed heuristic-based LS turbo equalizers and other equalization algorithms for Channel I

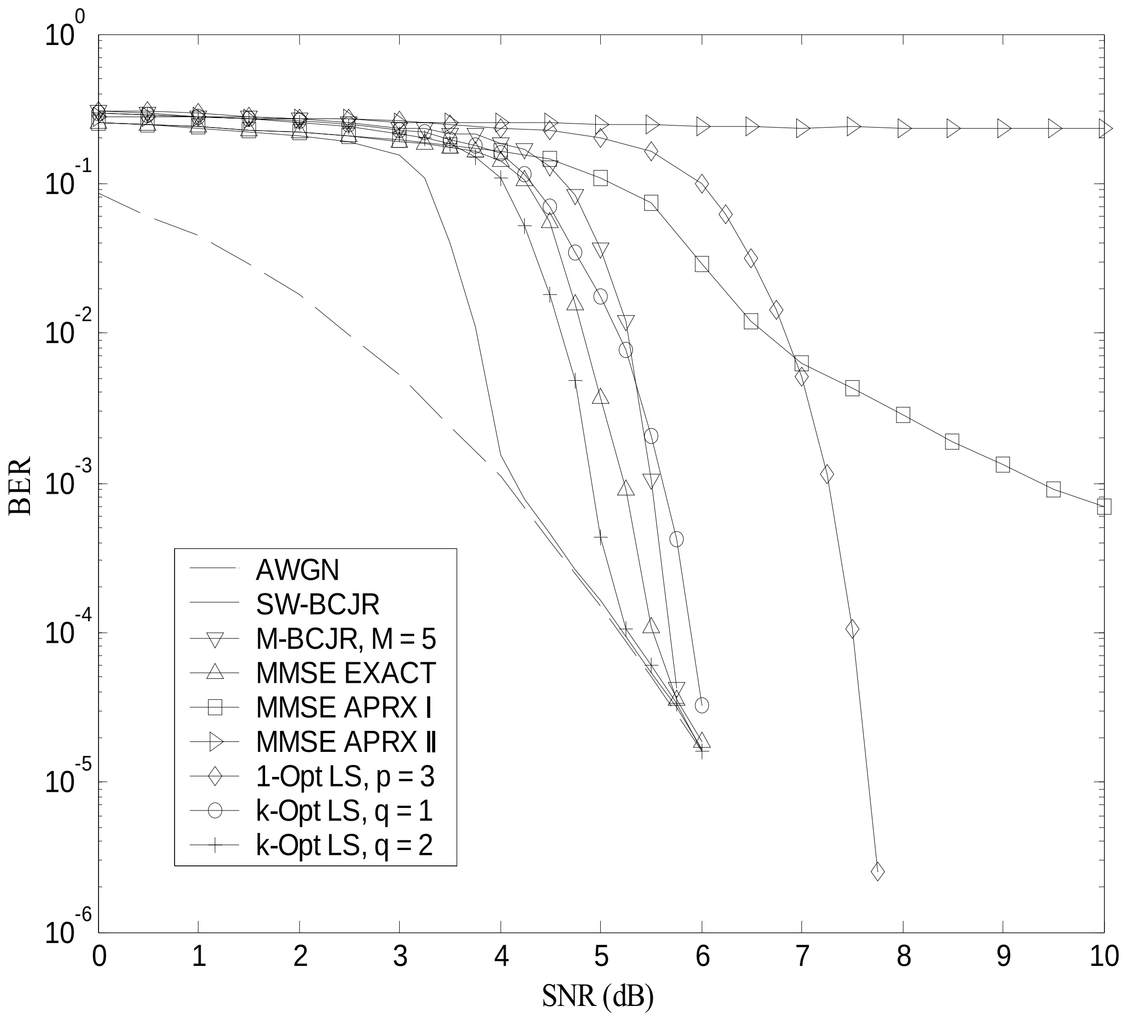

Figure 3.3: Performance comparisons between the proposed heuristic-based LS turbo equalizers and other equalization algorithms for Channel II (Proakis C Channel)

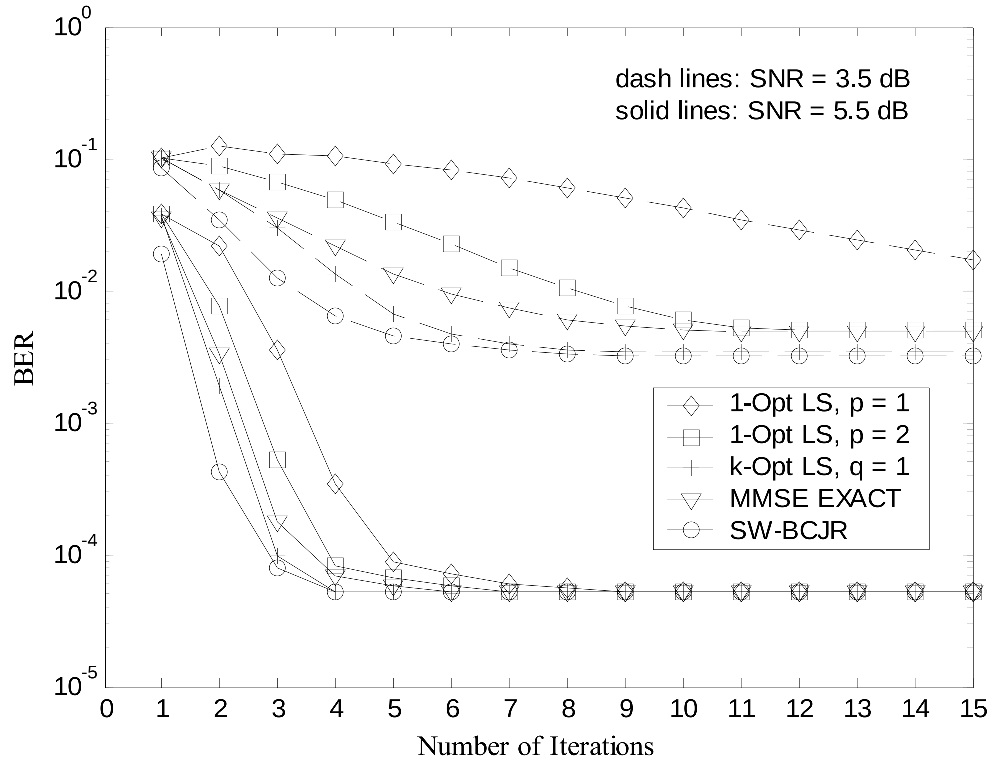

Figure 3.4: Performance of selected equalization algorithms at different iterations for Channel I

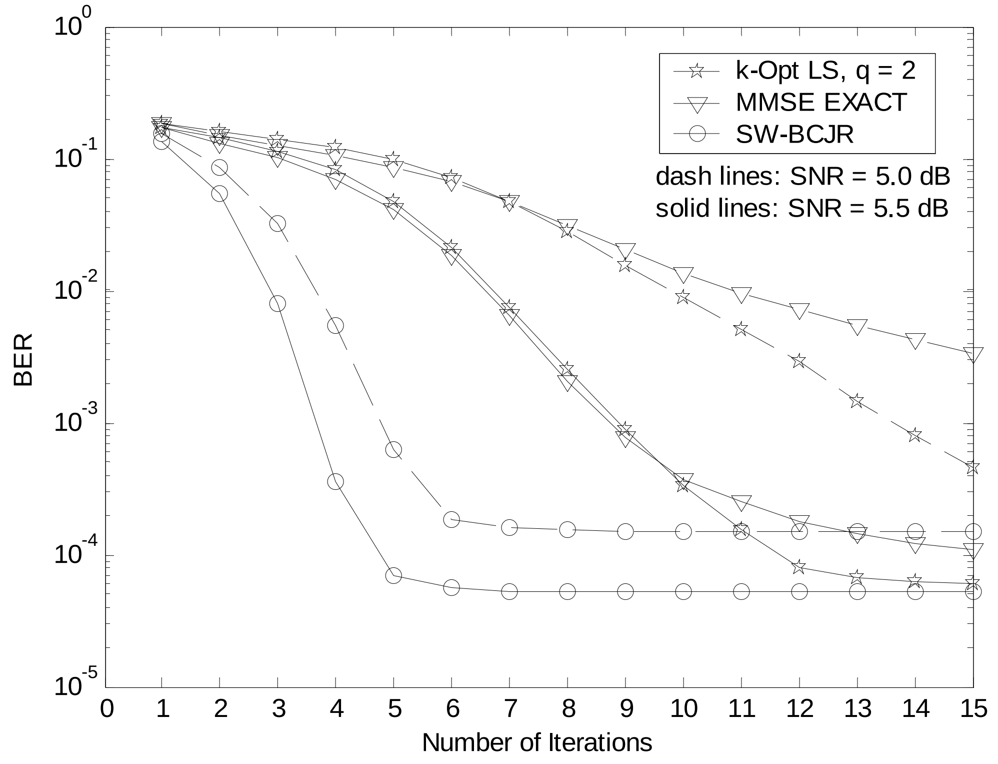

Figure 3.5: Performance of selected equalization algorithms at different iterations for Channel II (Proakis C Channel)

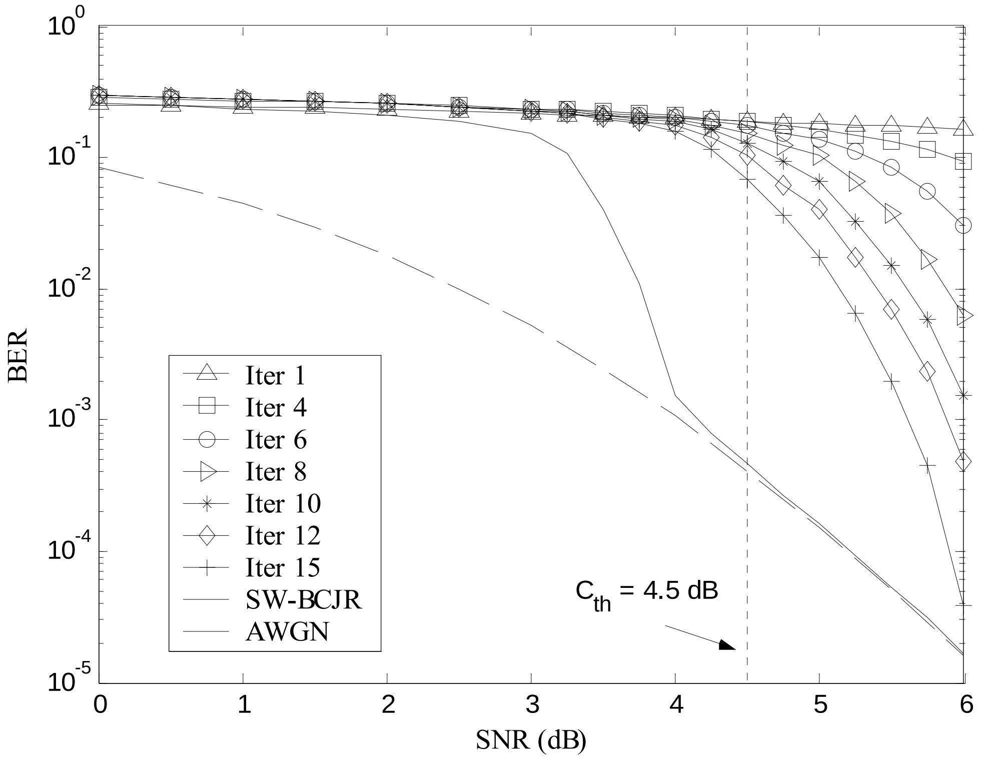

Figure 3.6: Convergence behavior of *k*-Opt LS turbo equalizer with $q = 1$ for Channel II (Proakis C Channel)

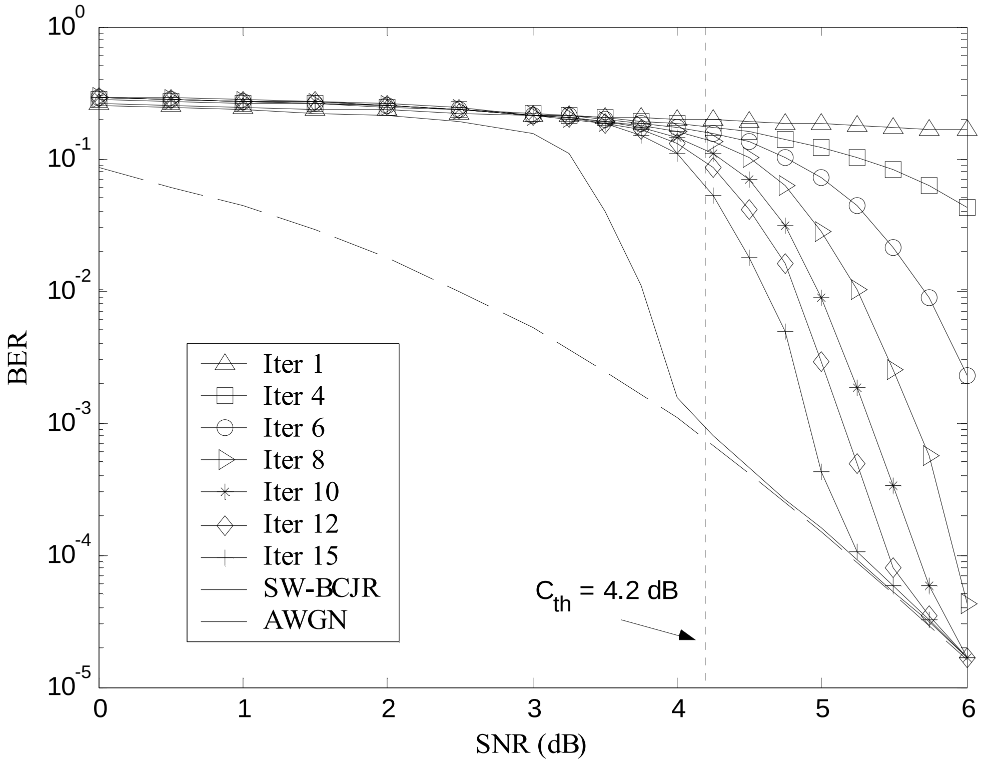

Figure 3.7: Convergence behavior of *k*-Opt LS turbo equalizer with $q = 2$ for Channel II (Proakis C Channel)

3.6 Chapter Summary

The implementation procedures of the proposed heuristic-based LS turbo equalizer have been described in details in this chapter. An introduction to the rich field of heuristics with an emphasis on LS algorithms in relation to the turbo equalization scheme is highlighted as a prelude to the presentation of these implementation details. From the analysis of these algorithms' computational complexities and their corresponding BER simulation results obtained, we show that the proposed LS-based equalizer can provide adjustable performance/complexity tradeoffs with a simple modification in their number of search steps. Generally, the *k*-Opt LS equalization algorithm outperforms that of the 1-Opt LS and BER performance improvements could be expected by increasing the number of search steps. Comparing the *k*-Opt LS equalization algorithm with the trellis-based and filter-based equalization algorithms as employed in a turbo equalization scheme, we also show that the *k*-Opt LS turbo equalizer with a search step of $q = 2$ outperforms the time-varying MMSE equalizer and attains very closely to the performance of the BER-optimal FC-BCJR equalizer under severe ISI conditions, with a much lower computational complexity. As such, the proposed *k*-Opt LS turbo equalizer is a viable alternative to these two well-known classes of equalization algorithms, in terms of both BER performance and computational complexity considerations.

Chapter 4

EXIT Chart Analysis of Heuristic-Based Local Search Turbo Equalizer

The implementation details of the proposed heuristic-based LS turbo equalizer have been presented. Simulation results show that it can attain very close to the BER-optimal FC-BCJR turbo equalizer while yet alleviating the huge computational complexity of the latter tremendously. In this chapter, an investigation into the proposed heuristic-based LS turbo equalizer is carried out through the use of the EXtrinsic Information Transfer (EXIT) chart. The EXIT chart is a useful analytical tool to investigate or predict the asymptotic behaviors of a turbo equalizer in the waterfall region. We begin this chapter with an introduction to the EXIT chart, after which the basis for such an analytical tool is provided in Section 4.2. The original EXIT chart as shown in Section 4.3 is found to be inaccurate in providing useful insights. As such, a new EXIT chart for the *k*-Opt turbo equalizer is proposed in Section 4.4 and subsequently verified in Section 4.5 to be more accurate in its asymptotic predictions than the original EXIT chart. Through the proposed EXIT chart for the *k*-Opt turbo equalizer, comparisons with selected equalization algorithms are then carried out in Section 4.6. Finally, some simulations results are presented after the EXIT chart analysis on the *k*-Opt turbo equalizer reveals its robustness against imperfect channel impulse response knowledge.

4.1 Introduction to EXIT Chart

The EXtrinsic Information Transfer (EXIT) chart [9] was originally proposed by Stephan ten Brink as a semi-analytical tool to investigate the convergence behavior of iteratively decoded parallel concatenated codes (also known as turbo codes). In a similar vein as the application of turbo processing concept to the equalization task, the EXIT chart could also be likewise applied to investigate the convergence behavior of a turbo equalizer [5][46]-[52].

Basically, as the name suggests, the EXIT chart is used to track the iterative transfers of extrinsic information (expressed in the form of LLR's) from one SISO module to the next by monitoring the evolution of the probability density function (PDF) of the extrinsic LLR indirectly through an information-theoretic quantity called mutual information. From the EXIT chart, the improvement on the quality of the extrinsic LLR as the iterations proceed could be visualized explicitly as a staircase-like trace (also known as trajectory) bounded by the transfer functions of the respective constituent SISO modules in the turbo setup.

A direct application of the EXIT chart [5] is to allow an asymptotic analysis (in the sense of very large frame size; typically about 10^5 bits or more) of a particular turbo equalizer implementation in the waterfall region without the need to carry out time-consuming closed-loop simulations on the actual system setup such as that depicted in Figure 1.2. In another words, the staircase-like trajectory bounded by the respective transfer functions of the equalizer and decoder in the turbo setup could be used to predict the average behavior of the turbo equalizer (for very large frame size) in terms of the associated BER at each iteration, the number of iterations required for convergence to a fixed point such that further iterations does not lead to any BER improvements and also its decoding threshold. For clarity, we call the staircase-like trace bounded by the transfer functions depicted in the EXIT chart as the *predicted trajectory* and the trace acquired by closed-loop simulations as the *averaged system trajectory*. The averaged system trajectory is simply obtained by averaging the *snapshot system trajectories* derived from repeated closed-loop simulation trials. Each snapshot system trajectory represents the improvement in the quality of the extrinsic LLR, measured in terms of the mutual information between the output extrinsic LLR derived from the constituent SISO modules in the turbo setup and the respective bits at the transmitter end, as the iterations proceed, for a particular simulated frame carried out at a predetermined SNR value. From a slightly different perspective, the EXIT chart allows the asymptotic averaged system trajectory and its associated parameters of interest at a particular SNR value to be obtained indirectly through the predicted trajectory, without carrying out time-consuming closed-loop simulations on very large frame size.

The transfer functions of each constituent SISO modules, namely, the equalizer and the decoder, are obtained separately by carrying out open-loop simulations independently. Essentially, a transfer function simply depicts the quality of the output extrinsic LLR derived from a particular SISO module, given a specific quality of input *a priori* LLR. For a serially concatenated receiver model such as the turbo equalizer structure depicted in Figures 1.2 and 3.1, an important distinction exists between the transfer function of the equalizer and that of the decoder. Basically, the transfer function of the equalizer depicts the input/output relationship of the LLR quality at a particular SNR value, whereas the transfer function of the decoder has a fixed input/output relationship irregardless of the SNR value. Another aspect to note stems from the simulation setup for the decoder. Since the open-loop simulation carried out to obtain the transfer function for the decoder yields the data estimate as a side product, a relationship between the BER and corresponding value of the mutual information between the output extrinsic LLR and the coded bits can thus be established and used for BER prediction. The simulation setup for generating the transfer function of the equalizer and that of the decoder are shown at the end of this section as Figures 4.1 and 4.2, respectively.

As mentioned earlier on, the quality of the LLR measured by the EXIT chart is quantified through the computation of the mutual information between the respective LLR and its corresponding bits at the transmitter end. Other means of quantifying the quality of the LLR exist in the literature, for example, the measurement of its SNR value [53], the tracking of its means or variances [9], or some form of correlation measure [54] as the iterations proceed.

However, there are several advantages to the use of mutual information as a form of quantifier. Firstly, the use of mutual information as a quantifier is shown in [9] to provide a better agreement between the predicted trajectory obtained by the EXIT chart and the corresponding asymptotic averaged system trajectory. A close agreement between the predicted trajectory and its corresponding asymptotic averaged system trajectory is crucial in allowing an accurate prediction of the various key performance parameters from the EXIT chart. Secondly, since the value of mutual information ranges from zero to one, its use can therefore compactly describes the quality of the extrinsic LLR and hence allows a convenient graphical display of the trajectory, unlike its SNR value counterpart which could have a range up to infinity. Furthermore, the use of mutual information as a quantifier in EXIT chart can also provide an information-theoretic interpretation to the well-known Shannon's noisy channel coding theorem [10][55] through the respective transfer functions' area properties when the inner code for a serially concatenated scheme is of rate-1, such as the data transmission model used throughout the whole of this thesis as depicted in Figure 1.1.

To carry out open-loop simulations of the respective SISO modules, a modeling of the input *a priori* LLR is required. Typically, the sequence of input *a priori* LLR is modeled as consistent uncorrelated Gaussian random variables with its absolute mean half of that of its variance in conjunction with the known transmitted bits. The quality of the input *a priori* LLR quantified through mutual information is varied from zero (which denotes no *a priori* information) to a value of one (which denotes perfect *a priori* information) indirectly through a suitably chosen variance parameter, σ_L^2. Given a particular quality of input *a priori* LLR determined by σ_L^2, the corresponding output extrinsic LLR from the respective SISO modules are then obtained by simulations where the PDF of the output extrinsic LLR are derived through a histogram measurement. From the derived PDF, the mutual information between the output extrinsic LLR and the corresponding bits used in the open-loop simulation is then numerically computed. In short, even though the input *a priori* LLR for each mutual information value is modeled and assumed to be Gaussian with parameters given by $N\left(\sigma_L^2/2, \sigma_L^2\right)$, the mutual information for the output extrinsic LLR is obtained without any reliance on this assumption.

In the next section, a simple derivation is provided to justify the reason behind the Gaussian model and its associated parameters used in the modeling of the input *a priori* LLR. Thereafter, the one-to-one relationship between the values of the variance parameter σ_L^2 and the mutual information is shown.

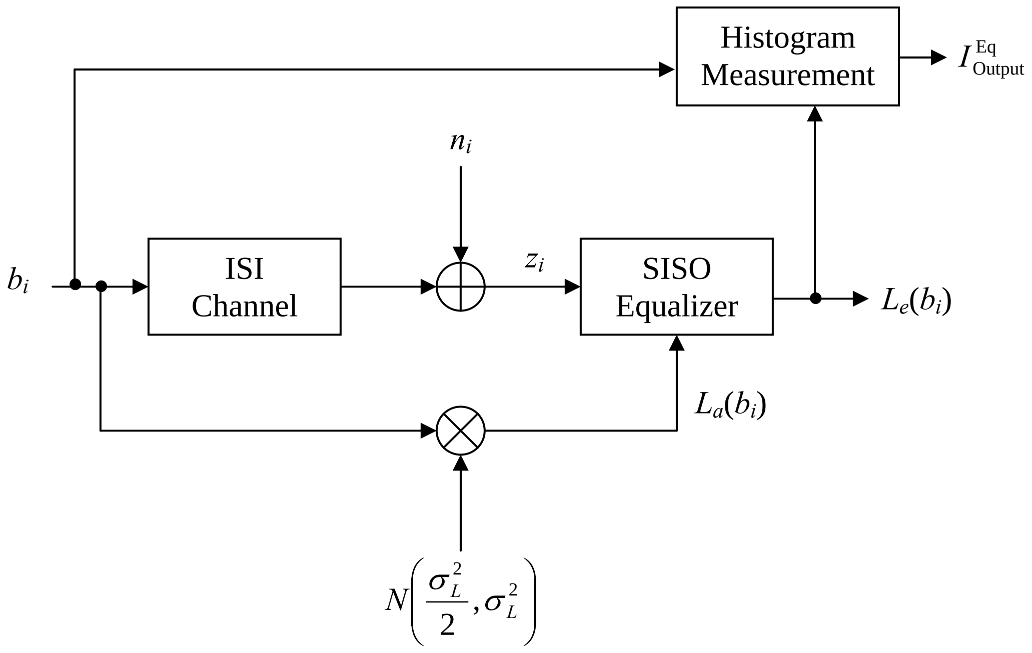

Figure 4.1: Simulation setup for generating the transfer function of the equalizer

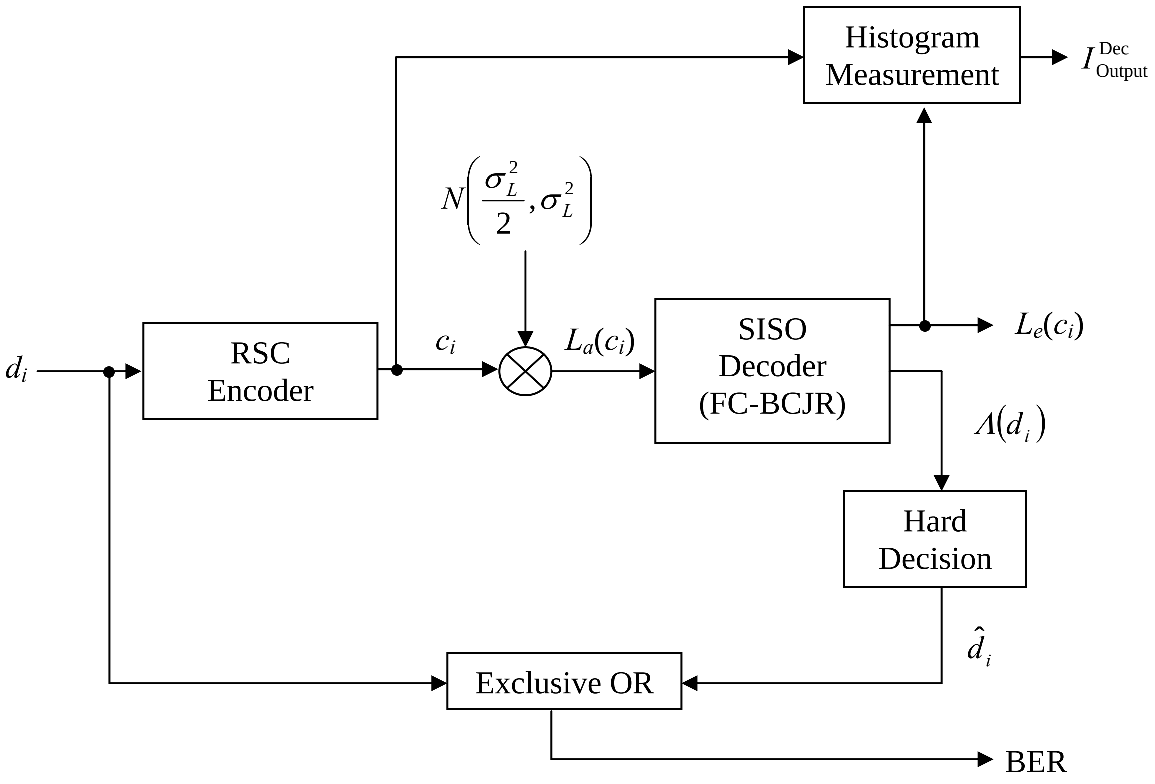

Figure 4.2: Simulation setup for generating the transfer function of the decoder

4.2 Principles of EXIT Chart

A) Input *a priori* LLR model

Consider a simple transmission process when the channel corrupts the transmitted signal by the addition of white Gaussian noise. The corresponding received real-valued discrete-time signal z is thus

$$z = b + n \tag{4.1}$$

where b denotes the transmitted binary bit in BPSK format and n denotes an AWGN sample with mean zero and variance $\sigma_n^2 = N_o/2$ (double-sided noise power spectral density). From (4.1), the corresponding LLR L (in natural log format) could be written as

$$L = \log_e \frac{P(z|b=+1)}{P(z|b=-1)} \tag{4.2}$$

where the conditional PDF in (4.2) is given as

$$P(z|b) = \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left\{ \frac{-(z-b)^2}{2\sigma_n^2} \right\}, \qquad b \in \{+1,-1\} \text{ and } z \in (-\infty, \infty) \tag{4.3}$$

Substituting (4.3) into (4.2) and simplifying, the LLR value can then be expressed as

$$L = \frac{2}{\sigma_n^2} z \tag{4.4}$$

$$= \frac{2}{\sigma_n^2}(b+n) \tag{4.5}$$

From (4.5), we thus could visualized the LLR L as a Gaussian random variable with associated parameters given by

$$\mu_L = \frac{2}{\sigma_n^2} b \qquad \text{and} \qquad \sigma_L^2 = \frac{4}{\sigma_n^2} \tag{4.6}$$

where μ_L and σ_L^2 denotes the mean and variance of the LLR L, respectively.

From (4.6), an important relationship between the absolute value of the mean of the LLR and its corresponding variance is established as

$$\left|\mu_L\right| = \frac{\sigma_L^2}{2} \tag{4.7}$$

The LLR expression used in (4.2) has a form identical to the definition of extrinsic information as observed earlier on in Chapter 2 in the exposition of the MMSE equalizer. Since the extrinsic information derived from one SISO module is used as *a priori* information for the next SISO module in a turbo setup, the above derivation leading to the relationship in (4.7) thus forms the basis for the modeling of the *a priori* LLR at the input of each SISO module. In particular, two important properties required for the modeling of the input *a priori* LLR could be deduced from the derivation, namely:

(1) The input *a priori* LLR of each transmitted bit fed into each SISO module is assumed to be statistically independent.

(2) Each *a priori* LLR is modeled as a Gaussian random variable with its absolute mean and variance related as in (4.7). In addition, the mean of a particular *a priori* LLR takes the sign of its corresponding bit that it represents.

Interestingly, the important relationship in (4.7) can also be derived using the consistency condition [37] for the distribution of the LLR L using

$$P\left(l \middle| X = x\right) = P\left(-l \middle| X = x\right)\exp(xl) \tag{4.8}$$

where the conditional probability of the random variable L given $X = x$ stated in (4.8) is of a Gaussian distribution given by

$$P\left(l \middle| X = x\right) = \frac{1}{\sqrt{2\pi}\sigma_L}\exp\left\{\frac{-\left(l - x\mu_L\right)^2}{2\sigma_L^2}\right\}, \qquad l \in (-\infty, \infty) \tag{4.9}$$

B) A measure of information content – Mutual Information

Mutual information is a measure of the amount of information that one random variable contains about another [55]. In simpler term, it indicates how much one knows about a random variable indirectly through the observation of another. Mutual information is defined between two random variables, say *X* and *Y*, as

$$I(X;Y) = \int\int P(x,y)\log_2 \frac{P(x|y)}{P(x)}\,dx\,dy \tag{4.10}$$

where the range of mutual information (used in the context of the EXIT chart) is

$$0 \leq I(X;Y) \leq 1 \tag{4.11}$$

For $I(X;Y)=0$, it implies no knowledge about the random variable *X* could be derived from observing random variable *Y*. For $I(X;Y)=1$, it indicates that we have a perfect knowledge of random variable *X* simply by observing random variable *Y* alone.

Relating the concept of mutual information to measure the improvement in the reliability of the *a priori* LLR and extrinsic LLR as the iterations proceed for a turbo receiver with respect to the transmitted bits, we could rewrite (4.10) as

$$I(L;B) = \frac{1}{2}\sum_{b=\{+1,-1\}}\int_{-\infty}^{+\infty} P(l|B=b)\log_2 \frac{2P(l|B=b)}{P(l|B=+1)+P(l|B=-1)}\,dl \tag{4.12}$$

where random variable *L* denotes either the *a priori* LLR or the *extrinsic* LLR observed at the receiver end, and random variable *B* refers to the transmitted binary bit associated with the observed LLR *L*. In (4.12), it is assumed that the binary bit, transmitted in BPSK format, is equiprobable. In the context where *L* is the *a priori* LLR, for $I(L;B)=0$, it represents zero *a priori* information and for $I(L;B)=1$, it indicates perfect *a priori* information is available to the respective SISO modules in the turbo setup [5][9][10].

Substituting the statistical model of the *a priori* LLR L derived previously in Section 4.2 A) into (4.12), the input mutual information could then be rewritten as

$$I_{\text{Input}} = 1 - \int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi}\sigma_L} \exp\left\{ \frac{-\left(l - \sigma_L^2/2\right)^2}{2\sigma_L^2} \right\} \log_2\{1 + \exp(-l)\}\, dl \qquad (4.13)$$

where I_{Input} is used to denote the mutual information between the input *a priori* LLR fed to the respective SISO modules in the turbo setup and the associated transmitted bit. From (4.13), a one-to-one relationship exists between σ_L^2 and I_{Input} which is shown in Figure 4.3. For easy reference, the variations of the variance parameter σ_L^2 in relation to the values of the mutual information I_{Input} are also provided in Table 4.1.

To compute the mutual information between the extrinsic LLR and their corresponding binary bits at the transmitter end using (4.12), a histogram approach is used to gather a long sequence of extrinsic LLR into evenly-spaced bins to estimate the PDF of the extrinsic LLR. Using such an approach in (4.12), the output mutual information can then be rewritten as

$$I_{\text{output}} = \frac{1}{2} \sum_{j=1}^{N} \left(p_j \log_2 \frac{2p_j}{p_j + n_j} + n_j \log_2 \frac{2n_j}{p_j + n_j} \right) \qquad (4.14)$$

where N denotes the number of histogram bins used to collect the extrinsic LLR and

$$p_j = \frac{\text{number of extrinsic LLR (due to } b = +1\text{) collected in the } j^{th} \text{ bin}}{\text{total number of extrinsic LLR due to } b = +1} \qquad (4.15)$$

$$n_j = \frac{\text{number of extrinsic LLR (due to } b = -1\text{) collected in the } j^{th} \text{ bin}}{\text{total number of extrinsic LLR due to } b = -1} \qquad (4.16)$$

Note that in the event where the numerator goes to zero or both the numerator and denominator go to zero in (4.14), we use the following convention $0\log(0) = 0$ and $0\log(0/0) = 0$ which could be arrived at from continuity [55].

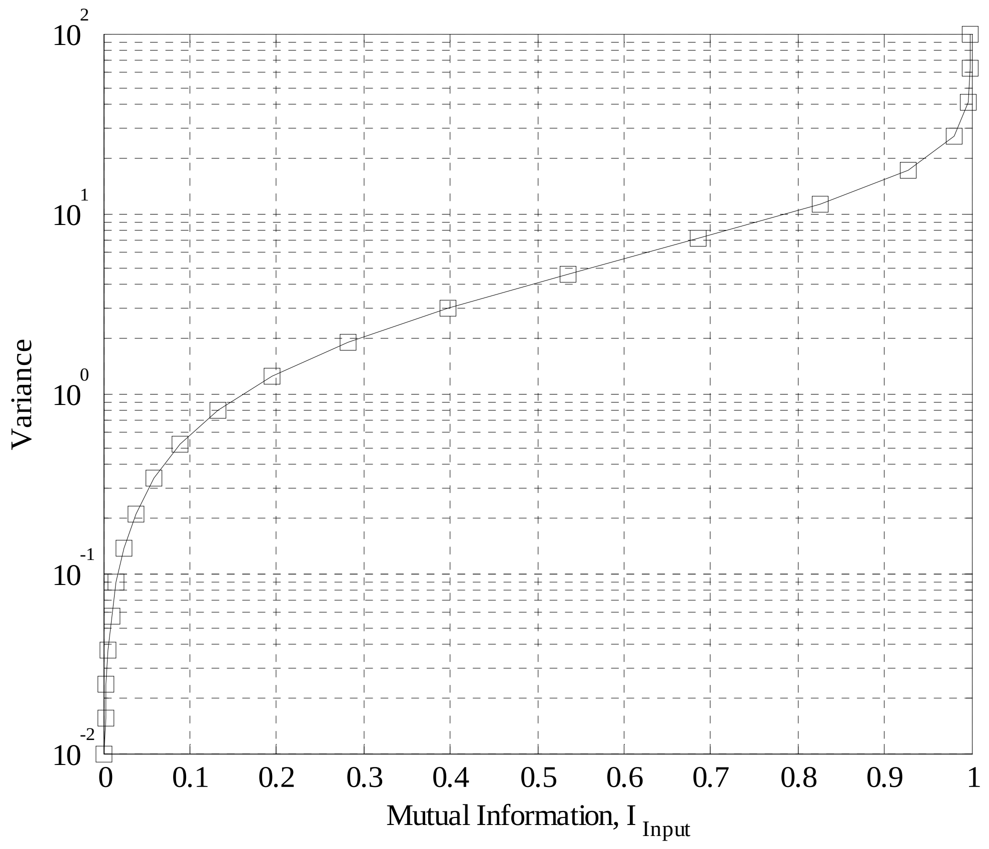

Figure 4.3: One-to-one mapping between variance and input mutual information

Table 4.1: Variations of input mutual information and its corresponding variance

Mutual Information, I_{Input}	Variance, σ_L^2
0.00	0.0100
0.05	0.2871
0.10	0.5951
0.15	0.9257
0.20	1.2816
0.25	1.6658
0.30	2.0819
0.35	2.5344
0.40	3.0291
0.45	3.5732
0.50	4.1761

Mutual Information, I_{Input}	Variance, σ_L^2
0.55	4.8500
0.60	5.6119
0.65	6.4854
0.70	7.5049
0.75	8.7255
0.80	10.2370
0.85	12.2114
0.90	15.0350
0.95	19.9500
1.00	100.0000

4.3 EXIT Chart Analysis on *k*-Opt Turbo Equalizer

The respective transfer functions of the *k*-Opt equalizer and RSC decoder are generated from the simulation setup shown in Figure 4.1 and 4.2, respectively using the Gaussian model for the input *a priori* LLR described in Section 4.2 A) in conjunction with the simulation parameters stated in Table 4.1. The frame size for generating the transfer functions is set to 10^4 averaged over 100 trials. The EXIT chart of the *k*-Opt turbo equalizer is then subsequently obtained by plotting the transfer functions of both the equalizer and decoder in a single figure, with the ordinate of the decoder's transfer function flipped with its abscissa. To facilitate analysis and discussion, we restrict the EXIT chart analysis to the turbo equalizer employing *k*-Opt LS equalizer with search step q = 2 over Channel II (Proakis C Channel) and decoder for a rate-1/2 RSC encoder with generator polynomials in octal form given by (g_1, g_2) = (7, 5). As such from here on, we would simply refer to the abovementioned as the *k*-Opt equalizer and decoder, respectively. The resultant EXIT charts at a SNR of 5.0 dB are shown in Figures 4.4 and 4.5. Note that the SNR description of the EXIT refers to the SNR value of the AWGN samples n_i as in Figure 4.1 for the generation of the equalizer's transfer function.

As mentioned in Section 4.1, a close agreement between the predicted trajectory and the averaged system trajectory is important in allowing an accurate asymptotic analysis and prediction of the key performance indicators of the turbo equalizer through the use of the EXIT chart. From here on, unless otherwise stated, the averaged system trajectories shown in the EXIT charts are obtained through simulations of the actual system setup at the same SNR value as the equalizer's transfer function using a data frame size of 10^5 averaged over 100 trials or equivalently 100 snapshot system trajectories.

In Figure 4.4, the averaged system trajectory obtained using the *k*-Opt equalizer at the first iteration with input solution the all-one vector is shown together with the predicted trajectory obtained from the EXIT chart. From Figure 4.4, it is observed that the predicted trajectory and the averaged system trajectory match well with each other only at the first iteration, but subsequent iterations between the two deviate significantly. A particular iteration can be easily seen from the trajectory as a vertical upward movement depicting the equalization process and a horizontal rightward movement depicting the decoding process.

In Figure 4.5, the averaged system trajectory is obtained through the use of the MMSE APRX I equalizer in the first iteration to provide a better quality input solution (as compared to the all-one vector). Thereafter in subsequent iterations, the equalization task reverts back to the proposed *k*-Opt equalizer. In such a system implementation, the overall turbo equalizer is considered a hybrid one [5]. As such, to obtain the predicted trajectory from the EXIT chart, it is required for both the transfer functions of the MMSE APRX I equalizer and that of the *k*-Opt equalizer to be plotted together. To avoid cluttering of the figures, we only show the transfer function of MMSE APRX I equalizer at zero mutual information since it is only used in the first iteration. The predicted trajectory is subsequently obtained as shown by the dash lines in Figure 4.5. Comparing the predicted trajectory and the averaged system trajectory, it is observed that both trajectories matches well for the first one and a half iterations and subsequently deviate. Here, we refer to the first half iteration as the equalization process.

To understand the reasons behind the deviation between the predicted trajectory and the averaged system trajectory, an investigation into the PDF of the *a priori* LLR and extrinsic LLR is carried out though simulation of the actual system. The investigation into the PDF of the LLR involves two aspects, namely, the progression of the variance parameters over the range of mutual information and the validity of the Gaussian assumption for the modeling of the *a priori* LLR. From the histogram plots obtained from the decoder's extrinsic LLR, it is observed that the mean of the LLR does not increase as much as that depicted indirectly by the simulation parameters stated in Table 4.1.

To quantitatively validate the Gaussian model, we use empirical skewness [56][57] and kurtosis [56]-[58] to determine the Gaussianity of the LLR measured at the output of both the equalizer and decoder, since many PDF have a signature relationship between their skewness and kurtosis. Skewness *S* and kurtosis *K* are the third and fourth moments of a PDF defined as

$$S = \frac{\sum_{i=1}^{M} (l_i - \eta)^3}{(M-1)\sigma^3} \tag{4.17}$$

$$K = \frac{\sum_{i=1}^{M} (l_i - \eta)^4}{(M-1)\sigma^4} \tag{4.18}$$

where M denotes the frame size of extrinsic LLR, l_i denotes the value of the i^{th} LLR. Here, η and σ denote the empirical mean and empirical standard deviation of the extrinsic LLR, respectively. For a Gaussian PDF, its skewness and kurtosis is 0 and 3, respectively. The variation of skewness and kurtosis of the PDF of the extrinsic LLR at the output of the equalizer and decoder as the iterations proceed are obtained from simulation using a data frame size of 32768 (averaged over 100 trials) and are shown in Figures 4.6 and 4.7, respectively. From these figures, it could easily be observed that the Gaussian PDF is not an accurate model for the input *a priori* LLR.

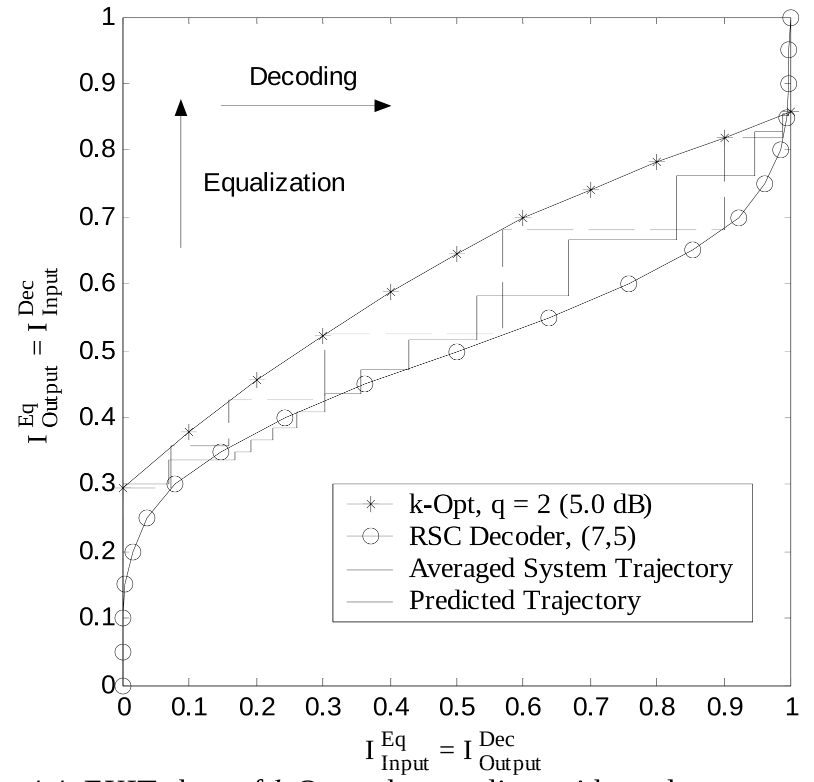

Figure 4.4: EXIT chart of *k*-Opt turbo equalizer with random start at 5.0 dB for Channel II (Proakis C channel)

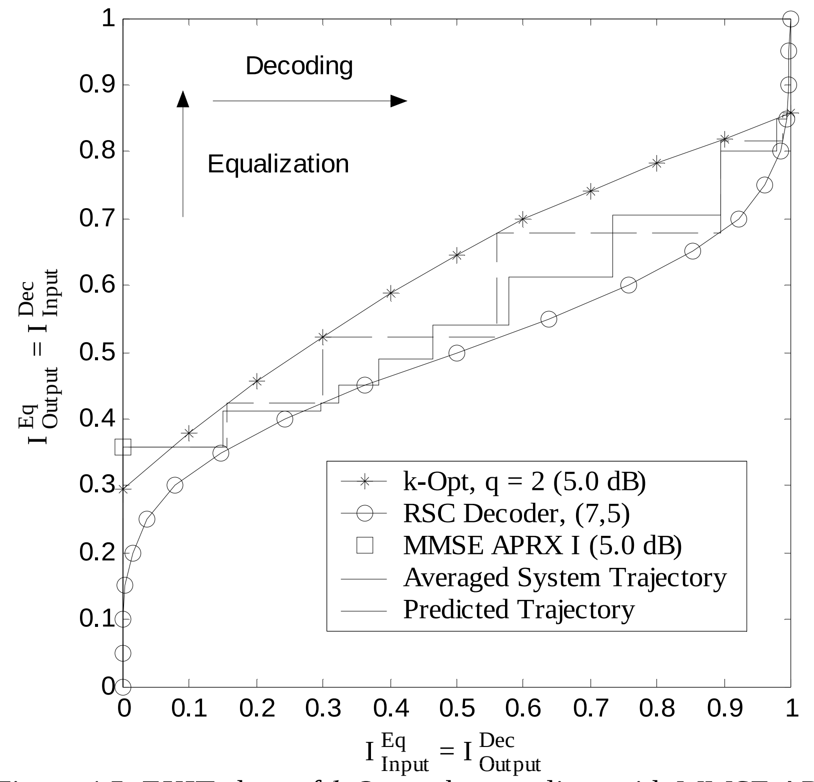

Figure 4.5: EXIT chart of *k*-Opt turbo equalizer with MMSE APRX I as first iteration at 5.0 dB for Channel II (Proakis C channel)

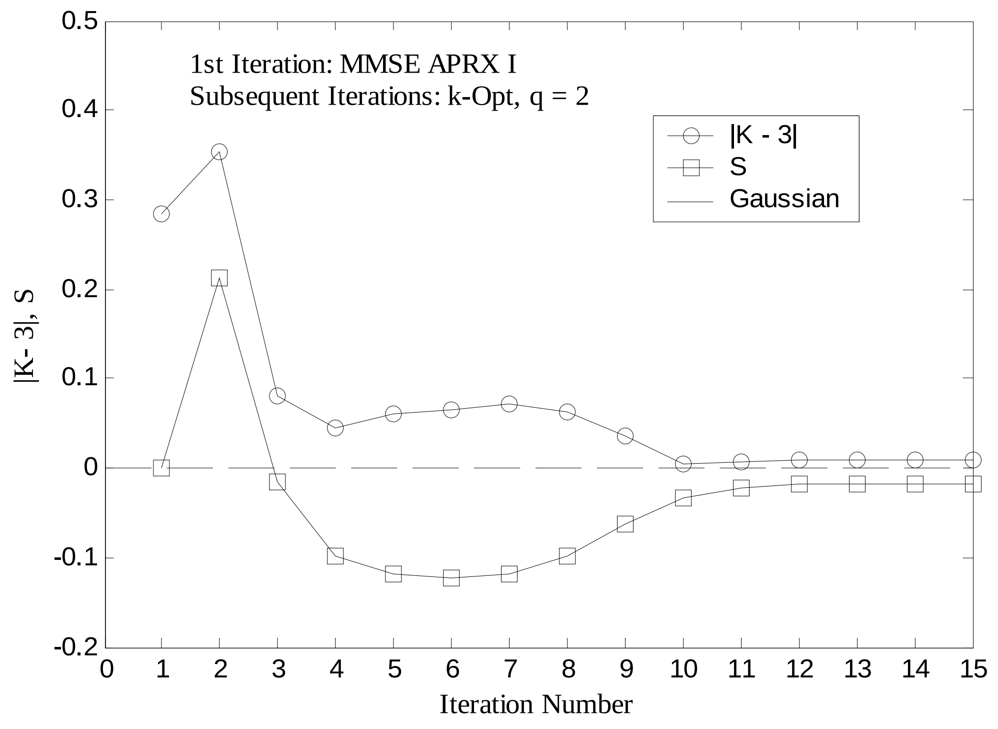

Figure 4.6: Gaussianity of extrinsic LLR measured at the output of equalizer at 5.0 dB

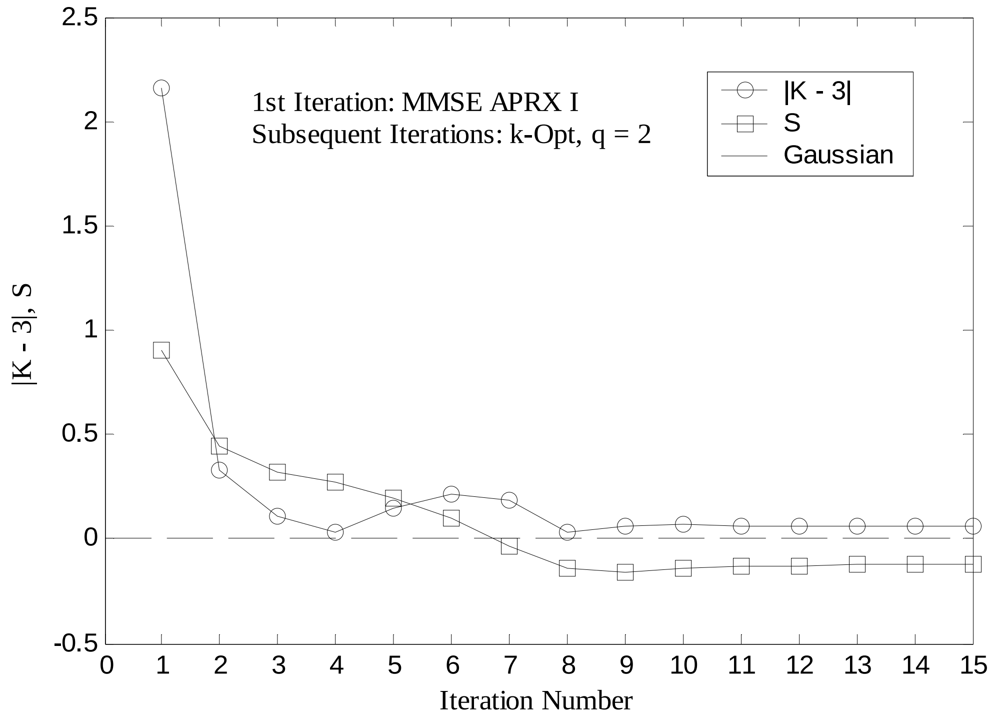

Figure 4.7: Gaussianity of extrinsic LLR measured at the output of RSC decoder at 5.0 dB

4.4 Proposed EXIT Chart of *k*-Opt Turbo Equalizer

From the previous section, the EXIT chart obtained from a consistent Gaussian assumption with the parameters stated in Table 4.1 is found to be inaccurate in providing an asymptotic prediction of the averaged system trajectory of the *k*-Opt turbo equalizer.

As such, in this section, we propose a general approach extended from the principles of the EXIT chart to obtain the respective transfer functions of the *k*-Opt equalizer and its corresponding RSC decoder such that the predicted trajectory acquired from the proposed EXIT chart can better match the averaged system trajectory.

The general idea is based on a "trajectory-fitting" approach together with the assumption that the exact model for the PDF of the *a priori* LLR is not important; the critical aspect simply lies in finding a convenient and suitable description for the input *a priori* LLR at certain values of mutual information that can constantly give an accurate representation of the transfer functions for the equalizer and decoder obtained through the observation of the system trajectories.

An outline of the proposed trajectory-fitting approach is as follows:

(1) At a suitably chosen SNR value that is slightly above the decoding threshold, plot a few snapshot system trajectories (obtained using a frame size of at least 10^5 data bits) on the original EXIT chart obtained as in Section 4.2. If the decoding threshold is unknown, simply plot the snapshot system trajectories at any SNR value where the trajectories can extend as much as possible over the entire range of $I_{\text{Input}}^{\text{Eq}}$.

(2) Approximate the equalizer's transfer function at certain critical points based on the area marked out by the snapshot system trajectories by observing how these points relate to the original transfer function of the equalizer. Get the desirable parameters

from the original transfer function of the equalizer and set it for use in the modeling of the *a priori* LLR for these points using a Gaussian assumption as before.

(3) For the decoder's transfer function, simply choose either a chi-square distribution or a Student's *t* distribution to model the input *a priori* LLR. The extra parameter that is available from such a PDF is the degree(s) of freedom which can be suitably selected to adjust the transfer function of the decoder at various values of input mutual information to fit the system trajectories. Once these parameters are obtained for a decoder matched to a RSC encoder with particular generator polynomials, it can be freely extended to derive the transfer functions of other decoders matched to other generator polynomials since the decoder's transfer function is a fixed function that is independent of SNR values.

4.4.1 Transfer Function of *k*-Opt Equalizer

The original transfer function of the *k*-Opt equalizer is plotted along with five snapshot system trajectories at a SNR of 5.0 dB as shown in Figure 4.8. These snapshot trajectories are obtained through simulations with the *k*-Opt equalizer given an all-one input solution in the first iteration. From the area marked out by the snapshot system trajectories in Figure 4.8, it is observed that we could approximate the transfer function of the *k*-Opt equalizer at three critical points, namely, at input mutual information of values 0, 0.5 and 1.0 accordingly. Two straight lines are then used to connect these critical points to depict the transfer function of the *k*-Opt equalizer at this particular SNR value.

At zero input mutual information, it is observed that the corresponding output mutual information at this particular point is heavily reliant on the quality of the input solution in which the *k*-Opt local search commence. In another words, from the first straight dash line connecting the points corresponding to input mutual information of 0 and 0.5 as drawn in Figure 4.8, it is found that the particular output mutual information at this point of zero input mutual information could be approximated as

$$\mathrm{I}_{\mathrm{Output}}^{\mathrm{Eq}} \approx 0.25 \times \left(\mathrm{I}_{\mathrm{Output}}^{\mathrm{Eq}}\right)_{\mathrm{IN1}} + 0.75 \times \left(\mathrm{I}_{\mathrm{Output}}^{\mathrm{Eq}}\right)_{\mathrm{IN2}} \tag{4.19}$$

where $\left(\mathrm{I}_{\mathrm{Output}}^{\mathrm{Eq}}\right)_{\mathrm{IN1}}$ and $\left(\mathrm{I}_{\mathrm{Output}}^{\mathrm{Eq}}\right)_{\mathrm{IN2}}$ denote the output mutual information of the *k*-Opt equalizer obtained by providing the *k*-Opt equalizer with zero input LLR with the all-one input solution, and a worst case of all-wrong input solution to commence its search respectively.

At input mutual information of 0.5, the critical point relates back to the original transfer function at input mutual information of 0.358 as shown in Figure 4.8. At an input mutual information of 0.358, the corresponding variance parameter as obtained from Figure 4.3 is found to be 2.611. As such, to obtain the transfer function of the *k*-Opt equalizer at input mutual information of 0.5, we replace the variance parameter of 4.1761 as shown in Table 4.1 with 2.611 while yet maintaining the Gaussian model with its mean half of its variance as before.

At input mutual information of 1.0, the output mutual information of the second straight dash line is found to coincide with that obtained from the original transfer function of the *k*-Opt equalizer. As such, the output mutual information for input mutual information of 1.0 is obtained similarly as before.

A further note here is that, except for zero mutual information, the input solution to commence the search is obtained directly by taking hard decision based on the *a priori* LLR generated as described above.

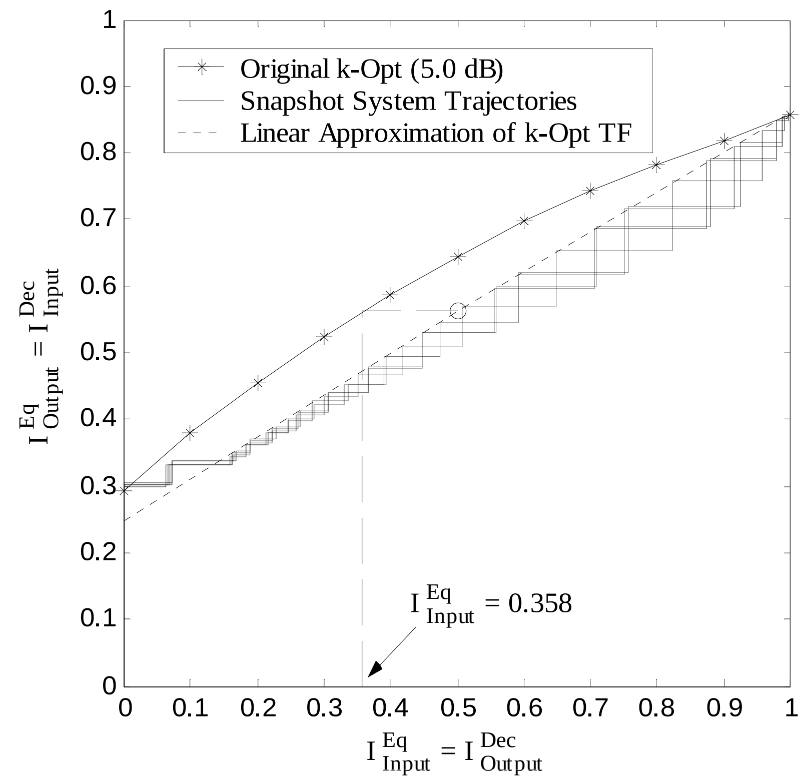

Figure 4.8: Linear approximation for the transfer function of the proposed k-Opt equalizer

4.4.2 Transfer Function of RSC Decoder

To obtain the transfer function of the RSC decoder, we model the input *a priori* LLR with the corresponding PDF depicted in Table 4.2 together with the same variance parameters as stated in Table 4.1. The mean of the input *a priori* LLR at the various values of mutual information is also half of the corresponding variance parameters.

The use of either a chi-square distribution or a Student's *t* distribution is to allow a third parameter (i.e., degrees of freedom) for adjusting the decoder's transfer function to fit the trajectories. It is to be noted that the PDF selected may not be identical to the actual PDF of the input *a priori* LLR. Nevertheless, once selected and fixed, these parameters may be used to obtain the transfer function of the decoder corresponding to other generator polynomials.

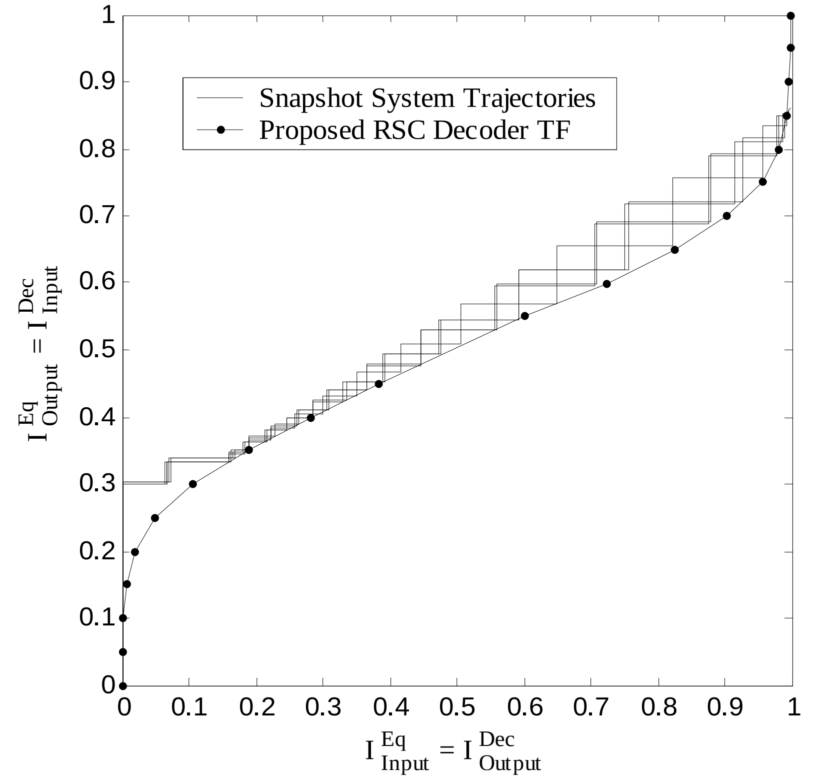

Figure 4.9: A trajectory-fitting approximation for the transfer function of the RSC decoder when used together with a *k*-Opt equalizer

Table 4.2: PDF used to model decoder's input *a priori* LLR

Mutual Information, I_{Input}	Chi-Square Distribution (Degrees of Freedom)
0.00	1
0.05	1
0.10	2
0.15	5
0.20	6
0.25	6
0.30	6
0.35	6
0.40	8
0.45	18

Mutual Information, I_{Input}	Student's *t* Distribution (Degrees of Freedom)
0.55	50
0.60	50
0.65	50
0.70	50
0.75	100
0.80	100
0.85	100
0.90	100
0.95	100
1.00	100

4.5 Verification of Proposed *k*-Opt EXIT Chart

In this section, we verify the accuracy of the proposed EXIT chart as described in Section 4.4 and compare some important results with the asymptotic behavior of the *k*-Opt turbo equalizer obtained by simulating a very large frame size of 10^5 averaged over at least 100 frame trials [5][9]. For the BER comparisons, the asymptotic and the finite frame size simulation are carried out until at least 100 error frames are gathered.

From Figure 4.8, it is observed that the linear approximations of the *k*-Opt equalizer's transfer function is only accurate for the second iteration onwards. As such, to allow an accurate prediction through the proposed EXIT chart, we carried out simulations of the actual system setup for 2 (respectively, 3) iterations for predicting the performance of the *k*-Opt turbo equalizer utilizing an all-one input solution in the first iteration (respectively, for predicting the performance of the hybrid *k*-Opt turbo equalizer with MMSE APRX I in the first iteration) using a finite frame size of 4096 to mark out the asymptotic trajectories for the first 2 (respectively, 3) iterations on the EXIT chart. All the asymptotic predictions from the respective third and fourth iterations onwards are then obtained from the proposed EXIT chart.

The rationale behind this approach is motivated by the fact that statistical independence between the local neighborhoods of the *a priori* LLR at the input of the respective SISO modules in the turbo setup will at least be maintained for the first three iterations for a finite frame size of 4096.

To sum up, the asymptotic behavior of the *k*-Opt turbo equalizer is predicted using the proposed EXIT chart as described in Section 4.4, together with simulations of finite frame size of 4096 for the first 2 or 3 iterations.

4.5.1 Predicted Trajectory vs Averaged System Trajectory

The proposed transfer functions of the *k*-Opt equalizer and its corresponding RSC decoder as shown in Figure 4.10 and 4.11 are obtained as described in Section 4.4 at a SNR of 5.0 dB over Channel II (Proakis C Channel).

In Figure 4.10, we use the proposed EXIT chart to predict the asymptotic averaged system trajectory of a *k*-Opt turbo equalizer (the first iteration utilized *k*-Opt equalizer given an all-one vector as input). Here, the predicted trajectory for the first 2 iterations is obtained using a finite frame size simulation of 4096 while the rest of the iterations are obtained from the proposed EXIT chart. As observed, the predicted trajectory and the asymptotic averaged system trajectory match very well.

For Figure 4.11, we use the proposed EXIT chart to predict the asymptotic averaged system trajectory of a hybrid *k*-Opt turbo equalizer (with the first iteration using the MMSE APRX I equalizer). For this case, the predicted trajectory for the first 3 iterations is obtained using a finite frame size simulation of 4096 while the rest of the iteration are obtained from the proposed EXIT chart. As observed, the predicted trajectory and the asymptotic average system trajectory match very well.

To further verify the accuracy of the *k*-Opt equalizer's transfer function, we have to look at its accuracy in predicting the averaged system trajectory over different SNR values. It is found that the respective predicted trajectories obtained from the proposed EXIT chart with finite frame simulations for the first 2 or 3 iterations and their corresponding asymptotic average system trajectories match very well with each other for SNR values of 4.5, 5.0, 5.5 and 6.0 dB. We will show a summary of the accuracy of our proposed EXIT chart at these SNR values by looking at the number of iterations required to attain convergence to a fixed point in Subsection 4.5.3.

Besides this, the procedure outlined in Section 4.4.2 is used to obtain the decoder's transfer function for an RSC encoder having generator polynomials (g_1, g_2) = (62, 56) with total memory 4 as used in [5]. Though not shown here, the corresponding decoder transfer function obtained using the proposed method is found to match the asymptotic averaged system trajectory very well.

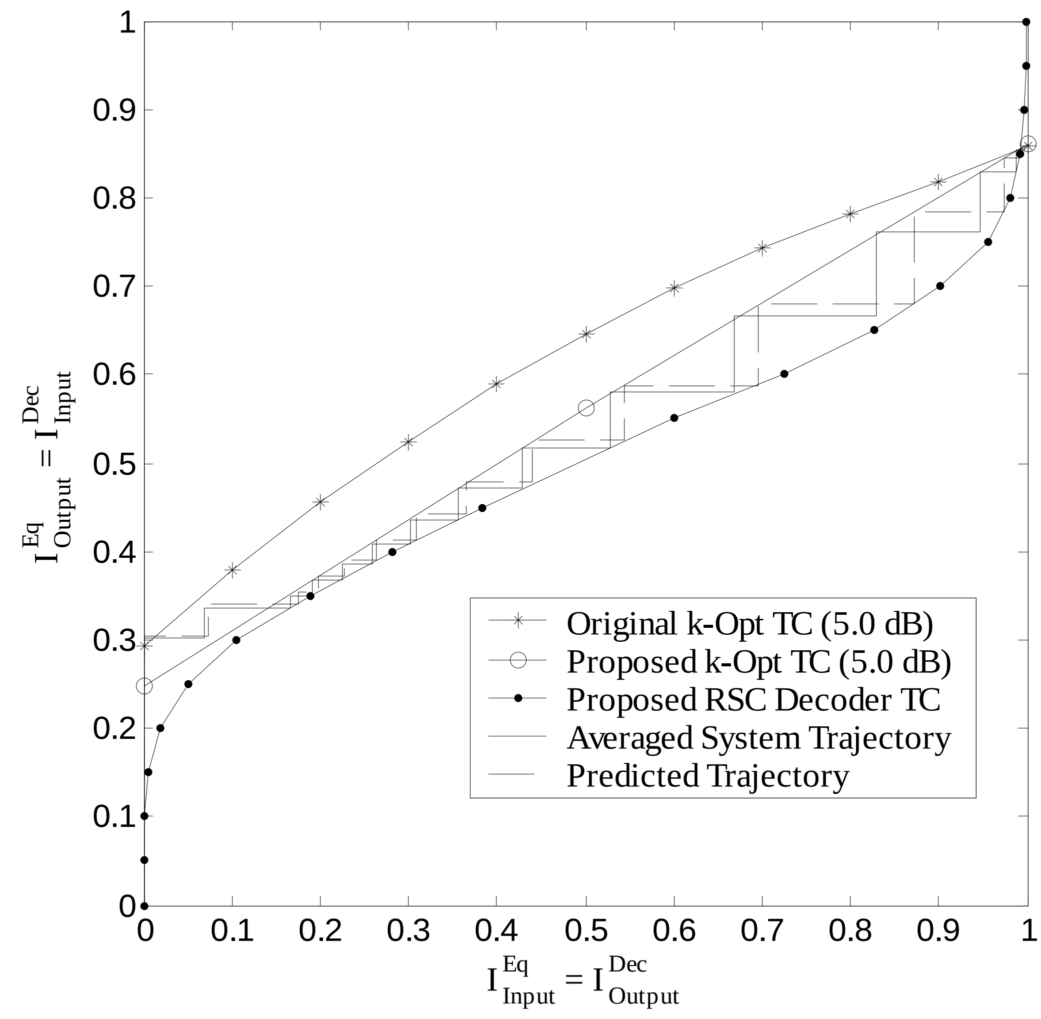

Figure 4.10: Proposed EXIT chart for *k*-Opt turbo equalizer at 5.0 dB with *k*-Opt equalizer given an all-one vector as input solution in the 1st iteration (Note that the first two iterations of the predicted trajectory are obtained by simulation with a finite frame size of 4096 data bits)

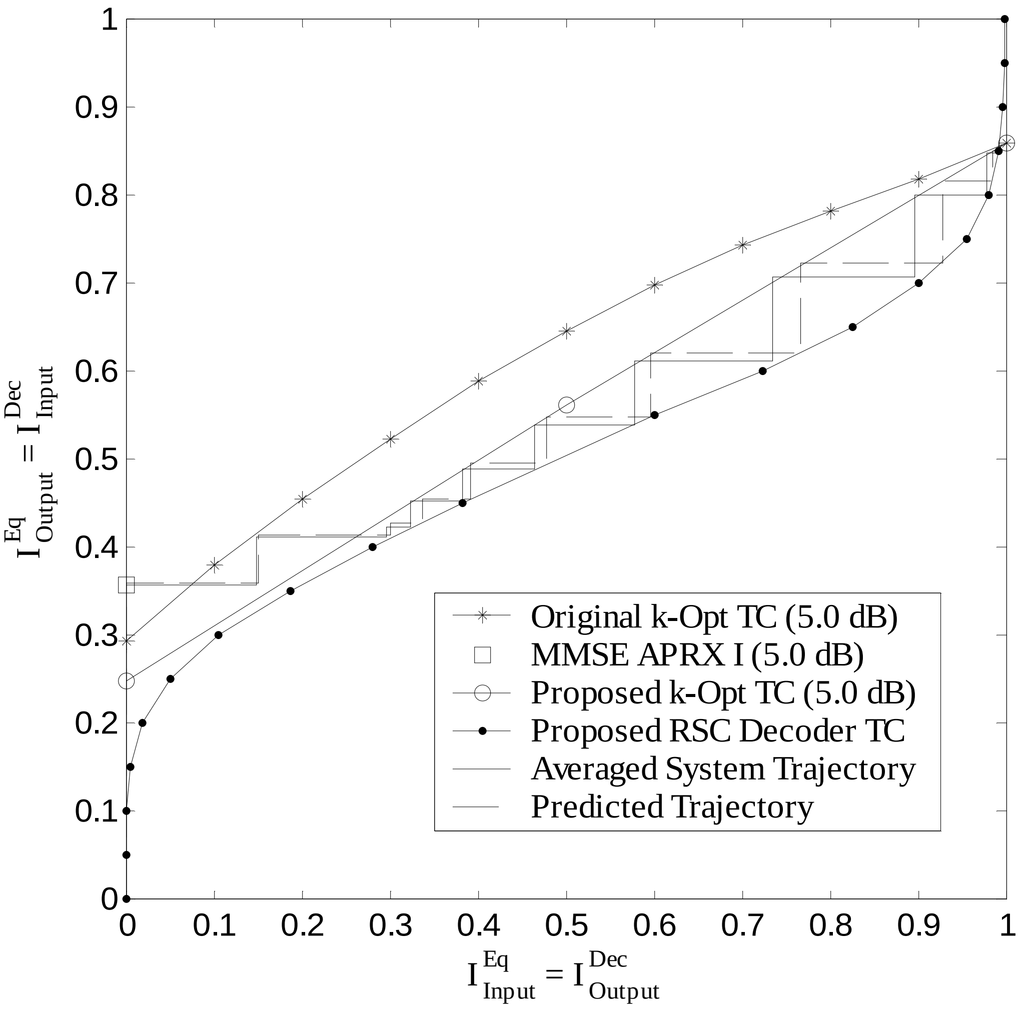

Figure 4.11: Proposed EXIT chart for *k*-Opt turbo equalizer at 5.0 dB with MMSE APRX I in the 1st iteration
(Note that the first three iterations of the predicted trajectory are obtained by simulation with a finite frame size of 4096 data bits)

4.5.2 Comparison of BER Prediction

The corresponding BER associated with the variation of the output mutual information of the decoder as obtained from the open-loop simulation depicted in Figure 4.2 together with the simulation parameters as stated in Table 4.1 and 4.2 are shown in Figure 4.12. Though not shown here, the BER curve obtained as a consequence of generating the original decoder's transfer function as in [5] is exactly the same as that depicted in Figure 4.12.

In Figures 4.13 and 4.14, the asymptotic BER prediction for the first 2 or 3 iterations are obtained from a finite frame size of 4096. Subsequent asymptotic BER prediction for the remaining iterations are obtained by plotting the predicted trajectory using the EXIT chart and taking note of the values of I_{Output}^{Dec} at each iteration. The corresponding BER at these iterations are then obtained using Figure 4.12.

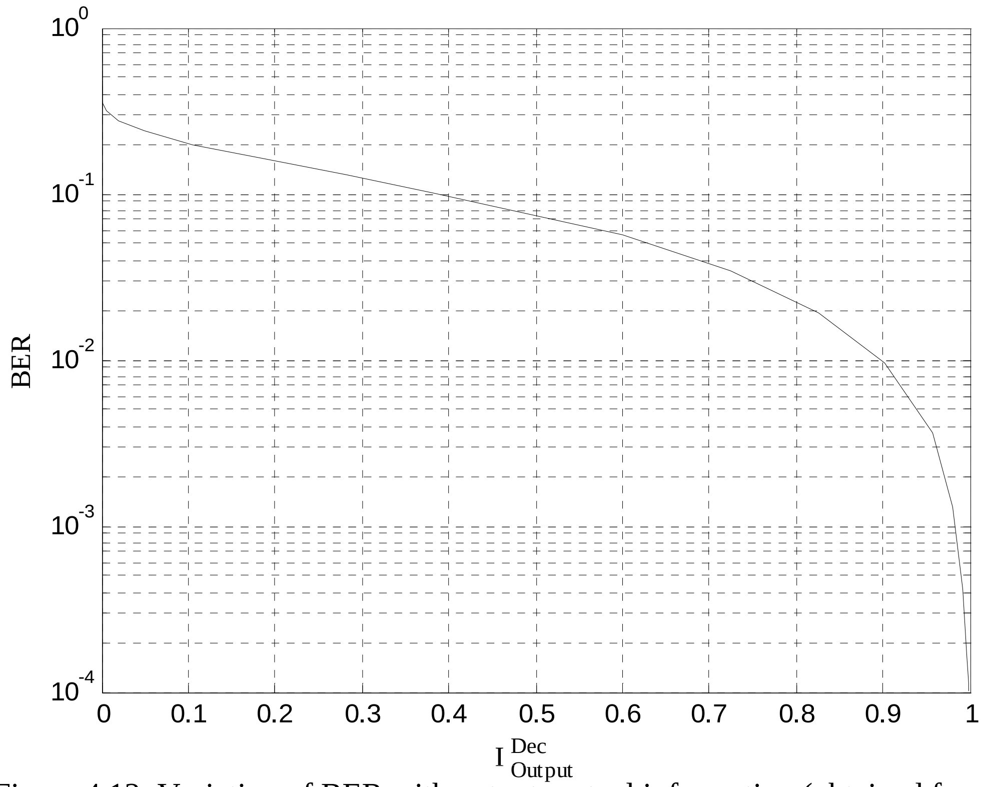

Figure 4.12: Variation of BER with output mutual information (obtained from open loop simulations of decoder's transfer function)

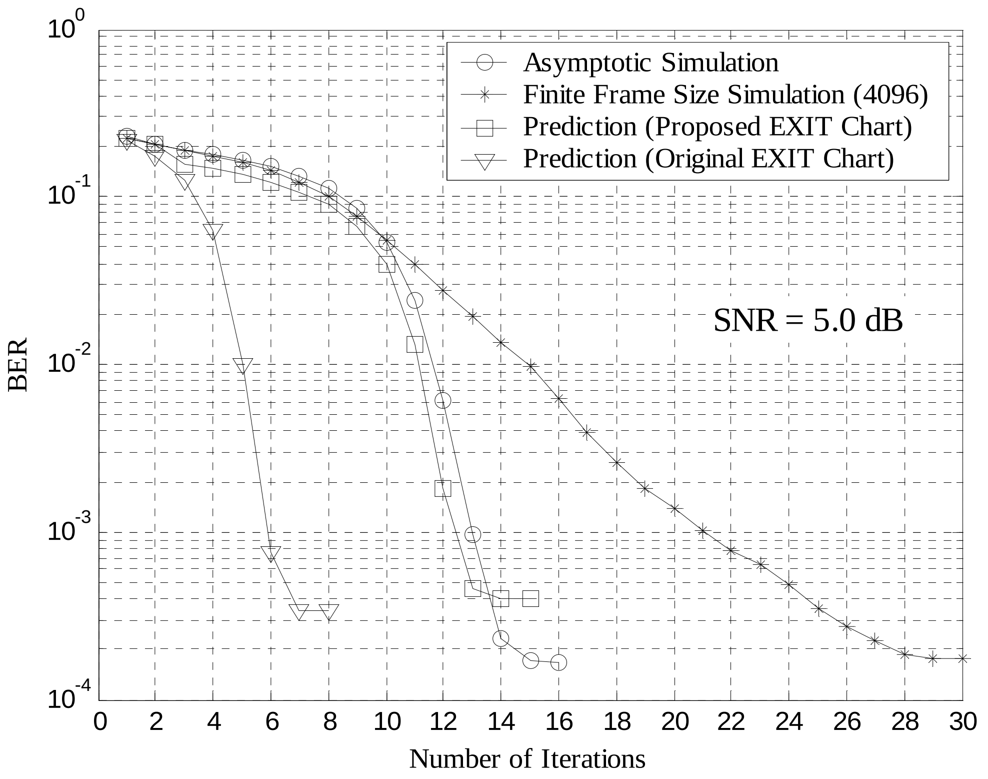

Figure 4.13: Comparison of BER between predictions by proposed and original EXIT charts with that obtained by asymptotic and finite frame size simulations (*k*-Opt equalizer in 1st iteration)

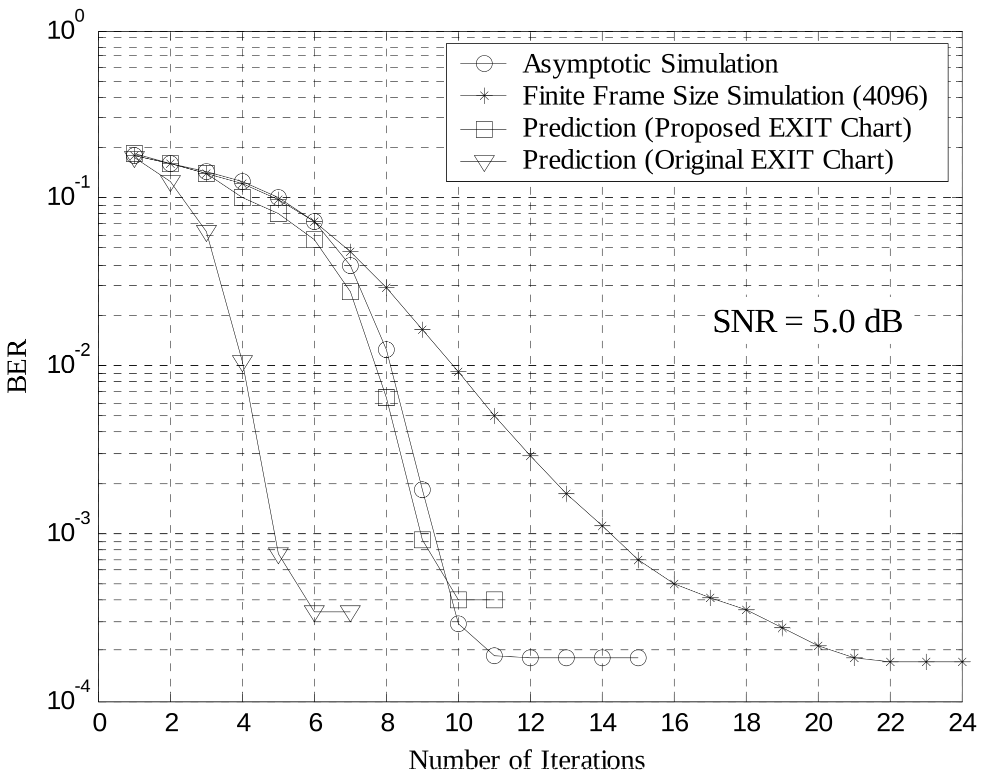

Figure 4:14: Comparison of BER between predictions by proposed and original EXIT charts with that obtained by asymptotic and finite frame size simulations (MMSE APRX I in 1st iteration)

From Figures 4.13 and 4.14, it is observed that proposed EXIT chart is more accurate than the original EXIT chart in predicting the asymptotic BER of the *k*-Opt turbo equalizers. However, it is found that the EXIT chart BER prediction obtained from both the original and proposed EXIT chart is not able to predict the asymptotic BER values of less than approximately $5*10^{-4}$ accurately. This is primarily due to the sensitivity of the BER plot depicted in Figure 4.12 for values of output mutual information approximately greater than 0.990. As such, for a fixed point of the EXIT chart that occurs at output mutual information of greater than 0.990, the BER predictions obtained from the EXIT chart using such a method would not be accurate anymore for the last few iterations that have output mutual information greater than 0.990.

Also shown in Figures 4.13 and 4.14 are the simulated BER values as the iterations proceed for a finite frame size of 4096. It is observed that the predicted BER obtained from the proposed EXIT chart is able to accurately predict the finite frame size BER for about 10 iterations and 7 iterations respectively, at this particular SNR value of 5.0 dB. Furthermore, given sufficient number of iterations, the asymptotic BER can be attained from finite frame size simulations.

4.5.3 Numbers of Iterations to Convergence

We compare the proposed EXIT chart and the original EXIT chart in terms of predicting the asymptotic number of iterations required to reach their respective fixed points on the EXIT chart for the *k*-Opt turbo equalizer here.

The number of iterations predicted from the respective EXIT chart and that obtained from the asymptotic frame size simulations for four SNR values are shown in Table 4.3. It is observed that the original EXIT chart is overly optimistic in predicting the number of iterations required to attain convergence to a fixed point. On the other hand, the proposed EXIT chart is able to predict the number of iterations for asymptotic frame size more accurately.

Table 4.3: Comparisons between original EXIT chart's predictions and proposed EXIT chart's predictions on the number of iterations to convergence to a fixed point with respect to asymptotic frame size simulation for Channel II (Proakis C Channel)

	SNR (dB)			
	4.5	5.0	5.5	6.0
Original EXIT Chart Prediction	8	7	6	5
Proposed EXIT Chart Prediction	15	11	8	7
Asymptotic Frame Size Simulation	15	11	9	7

4.5.4 Decoding Threshold of *k*-Opt Turbo Equalizer

Once an accurate description of the *k*-Opt turbo equalizer's EXIT chart is obtained, it can be used to determine its decoding threshold easily. In general, the decoding threshold of a particular turbo equalizer implementation can be predicted from the EXIT chart by simply determining the SNR value in which the equalizer's transfer function coincide or touches that of the decoder [5][9][10].

For the case of the *k*-Opt turbo equalizer, in addition to finding the SNR value in which the equalizer's transfer function coincides with that of the decoder, it is necessary to carry out simulations at this particular SNR value using a finite frame size of 4096 for 2 iterations to mark out the start of the third iterations in the proposed EXIT chart. This is due to the inability of the proposed EXIT chart in predicting the first 2 iterations of the *k*-Opt turbo equalizer. The decoding threshold of the *k*-Opt turbo equalizer as found from the EXIT chart is 4.2 dB as shown in Figure 4.15 and it is identical to the decoding threshold as determined in Chapter 3 using density evolution [38]. Also shown in Figure 4.15 are the respective points where the third iterations commence for the case where the first iteration utilizes the *k*-Opt equalizer given an all-one input solution and the case where the first iteration utilizes the MMSE APRX I equalizer.

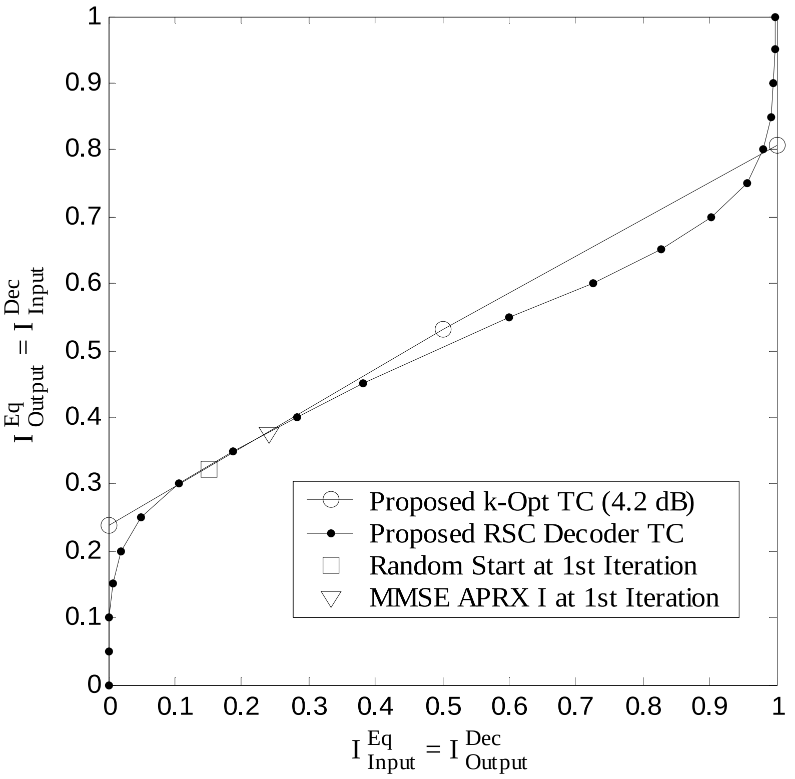

Figure 4:15: Decoding threshold prediction using proposed EXIT chart

The accuracy of the proposed EXIT chart in predicting the asymptotic averaged system trajectory, the expected BER at each number of iterations (up to a BER of approximately $5*10^{-4}$), the number of iterations to convergence to a fixed point and the decoding threshold have been verified to be very precise accordingly.

4.5.5 Effects of Finite Frame Size

The EXIT chart is a semi-analytical tool that is used in understanding the asymptotic behavior (in the sense of very large frame size) of a turbo decoder or turbo equalizer. As such, it would be interesting to find out the extent of deviation from its asymptotic behavior for the case where finite frame size is considered [5].

In Figures 4.16, 4.17 and 4.18, the averaged system trajectories obtained through simulations with at least 100 trials at a SNR of 5.0 dB using finite frame sizes of 512, 4096 and 32768 are shown, respectively. Here, the maximum number of iterations is set to 15 and MMSE APRX I equalizer is used in the first iteration for all cases. As these figures show, greater interleaving gain (associated with a bigger frame size) is manifested in its averaged system trajectory as better quality (measured in terms of mutual information) output extrinsic LLR from the decoder, given a specific quality of input a *priori* LLR. Specifically, the maximum achievable quality of the output extrinsic LLR from the decoder is not exploited fully when smaller frame sizes of 512 and 4096 are utilized. For a frame size of 32768 as shown in Figure 4.18, the averaged system trajectory almost touches the transfer function of the decoder. This indicates that the maximum interleaving gain for the *k*-Opt turbo equalizer is almost exploited fully for a frame size of 32768. Consequently, this also justified the rationale behind the choice of frame size (i.e. 10^5 data bits per frame) that is used for the asymptotic simulations to prove the accuracy of the proposed EXIT chart.

Interestingly, frame size does not affect the quality of the output extrinsic LLR produced by the *k*-Opt equalizer given a specific quality of input a *priori* LLR. This is observed in Figures 4.16 and 4.17 from the averaged system trajectories which touch the proposed transfer function of the *k*-Opt equalizer for a frame size of 512 and 4096 data bits respectively. Beside this, it is also observed that the proposed EXIT chart is able to predict accurately (from the fourth iterations onwards) for at least a few iterations even for finite frame size of 512 and 4096 data bits.

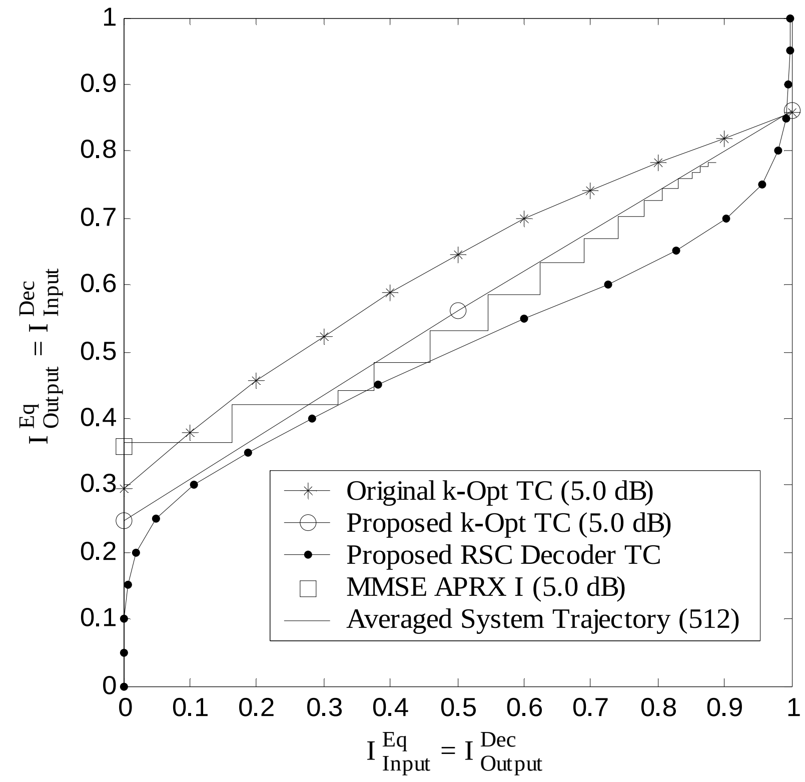

Figure 4:16: Averaged system trajectory of *k*-Opt turbo equalizer (with q = 2) for frame size of length 512 data bits at 5.0 dB

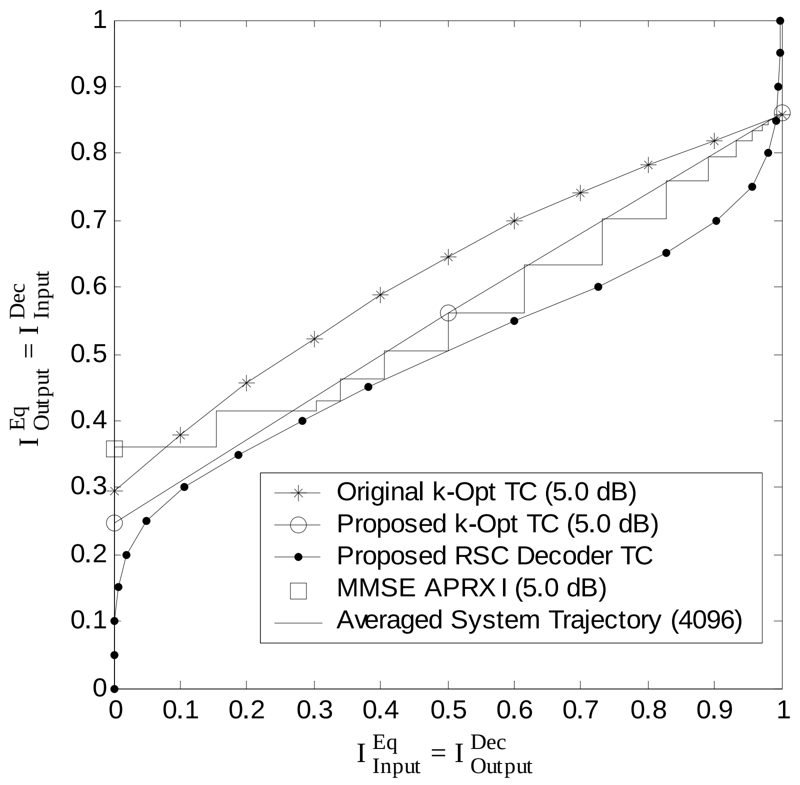

Figure 4:17: Averaged system trajectory of *k*-Opt turbo equalizer (with *q* = 2) for frame size of length 4096 data bits at 5.0 dB

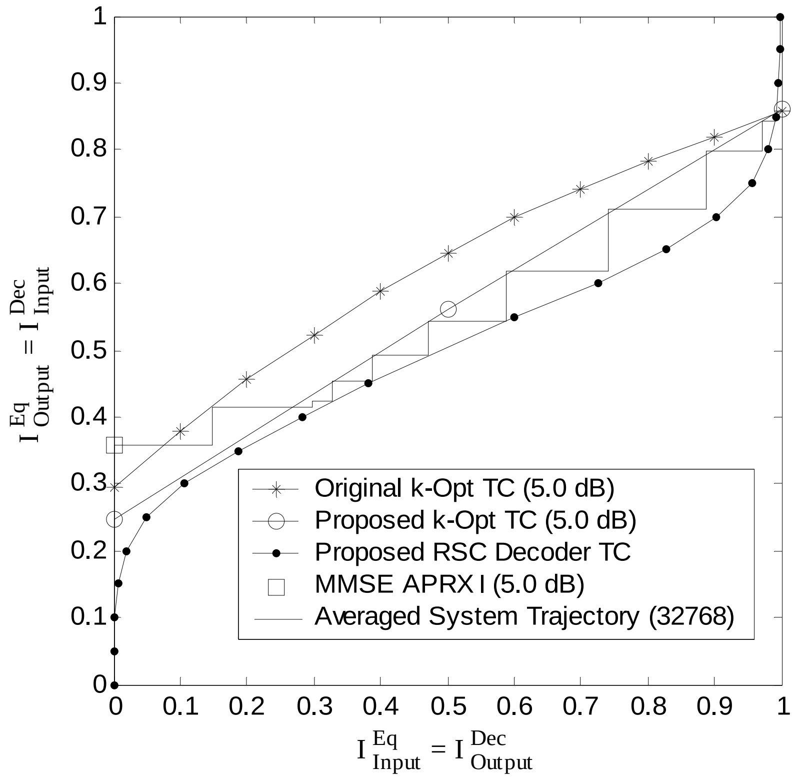

Figure 4:18: Averaged system trajectory of *k*-Opt turbo equalizer (with *q* = 2) for frame size of length 32768 data bits at 5.0 dB

4.5.6 Variation of Critical Points vs SNR (*k*-Opt Equalizer)

To have a greater appreciation of the *k*-Opt equalizer, the variations of its transfer function's critical points at the input mutual information of 0.0, 0.5 and 1.0 over an SNR range from 0 dB to 20 dB are shown in Figure 4.19.

From Figure 4.19, it could be observed that the output mutual information of the *k*-Opt equalizer corresponding to an input mutual information of 0.0 does not increase significantly as the SNR increases. This thus translates to low starting points in the EXIT chart at the first iteration, thereby resulting in a slow convergence to the corresponding fixed points. This impediment could be easily overcome by incorporating another equalizer (such as the MMSE APRX I equalizer) in the first iteration to provide a good quality input solution to jump-start the iterative process, following which the equalization process could then revert back to the *k*-Opt equalizer.

The mid-point of the transfer function (as denoted by "input mutual information = 0.5" in Figure 4.19) for the *k*-Opt equalizer is observed to increase steadily till a value of approximately 0.7 at a SNR of about 12.0 dB. In general, the mid-point of the transfer function for a RSC decoder implemented via the BCJR algorithm is approximately equal to the rate of the RSC code [5][58]. Here, the mid-points of the respective transfer functions are with respect to the abscissa of the EXIT chart as denoted by the axis $\mathrm{I}_{\mathrm{Input}}^{\mathrm{Eq}} = \mathrm{I}_{\mathrm{Output}}^{\mathrm{Dec}}$. In other words, for a rate-1/2 RSC decoder's transfer function at the point where $\mathrm{I}_{\mathrm{Output}}^{\mathrm{Dec}} = 0.50$, its associated ordinate value in the EXIT chart will thus be approximately $\mathrm{I}_{\mathrm{Input}}^{\mathrm{Dec}} = 0.50$. Similarly, for a rate-8/9 RSC decoder's transfer function at the point where $\mathrm{I}_{\mathrm{Output}}^{\mathrm{Dec}} = 0.50$, its associated ordinate value in the EXIT chart will thus be approximately $\mathrm{I}_{\mathrm{Input}}^{\mathrm{Dec}} = 0.89$. As such, it could be easily deduced that the *k*-Opt equalizer cannot work with a code of high rate since the *k*-Opt equalizer's transfer function will lie below that of the corresponding decoder's transfer function.

Next, the variation of the output mutual information for the *k*-Opt equalizer corresponding to an input mutual information of 1.0 is found to be identical to that of the FC-BCJR equalizer (not shown to avoid cluttering of the figure). This indicates that the *k*-Opt equalizer can attain identical performance to the BER-optimal FC-BCJR equalizer implementation, provided sufficient iterations are allowed for the *k*-Opt turbo equalizer to run and the SNR values in consideration are above the decoding threshold of the *k*-Opt turbo equalizer.

Also shown in Figure 4.19 is the variation of the output mutual information of the *k*-Opt equalizer over the SNR range given a perfect input solution and zero *a priori* LLR. This is strictly a hypothetical scenario since it is not possible to obtain a perfect input solution for the local search to commence if the *a priori* LLR given to the *k*-Opt equalizer is zero. Recall that the input solution is obtained simply by taking hard decision on the *a priori* LLR. Nevertheless, this suggests the relative importance and weight given to the input solution as compared to the quality of the *a priori* LLR in the equalization process carried out by the *k*-Opt equalizer. That is, greater importance is placed on the quality of the input solution rather than that of the *a priori* LLR in producing better quality output mutual information.

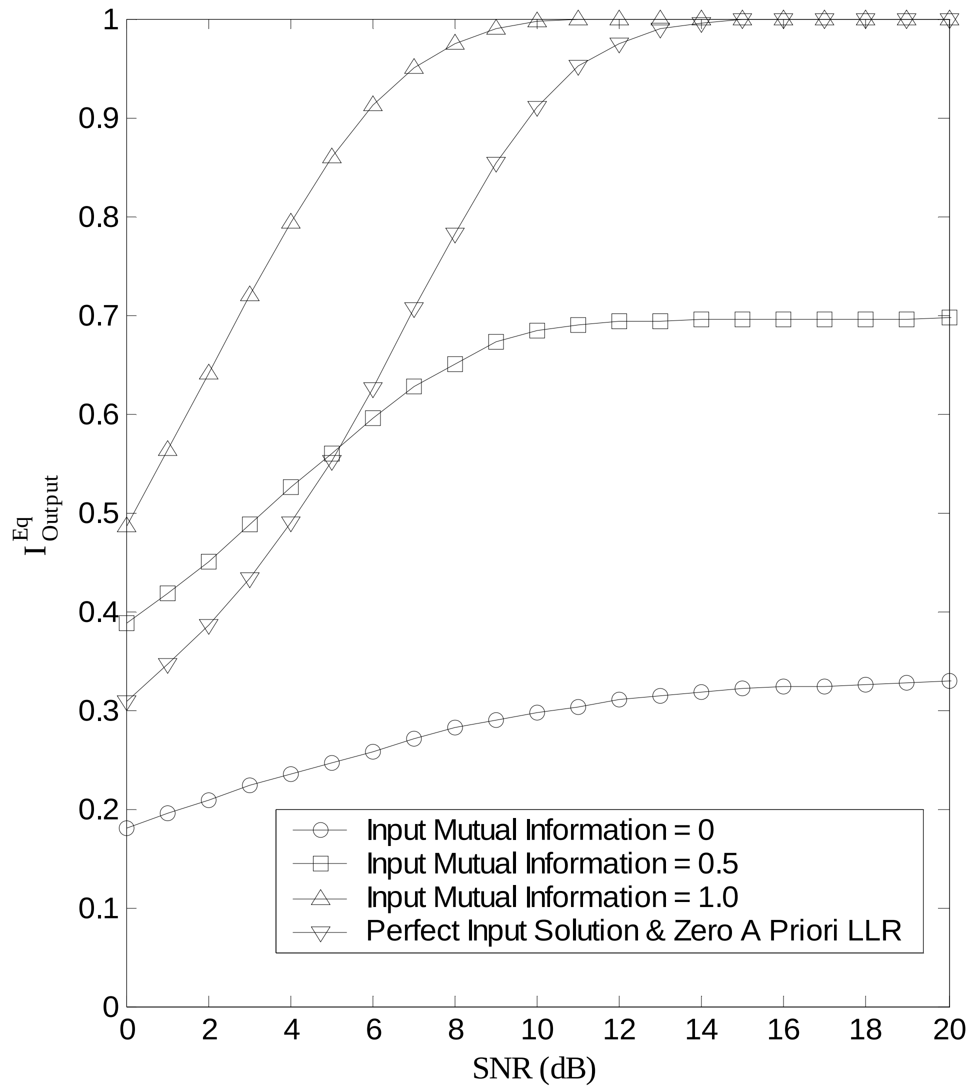

Figure 4.19: Output mutual information of *k*-Opt equalizer vs SNR for different qualities of input *a priori* LLR

4.6 Comparisons with Selected Equalization Algorithms

In this section, we utilize the EXIT chart to compare some asymptotic characteristics of selected equalizers' implementations, namely the FC-BCJR equalizer, the MMSE EXACT equalizer and the Hybrid MMSE equalizer (implemented through the use of either MMSE APRX I or MMSE APRX II equalizers) with that of our proposed *k*-Opt equalizer. The decoder in consideration in this section is the rate-1/2 RSC encoder with generator polynomials (g_1, g_2) = (7, 5) as used throughout the thesis.

Before going further, the various equalizers' transfer functions at a SNR of 5.0 dB and 6.0 dB are shown in Figures 4.20 and 4.21, respectively. Specifically, the transfer functions of the FC-BCJR equalizer, the MMSE EXACT equalizer and the Hybrid MMSE equalizer are obtained using the methods outlined in Sections 4.2 and 4.3. These transfer functions have been shown in [5] to be accurate in predicting the averaged system trajectories. The proposed and original transfer functions for the *k*-Opt equalizer are also included in these figures.

From Figures 4.20 and 4.21, it could be observed that the original transfer functions of the *k*-Opt equalizer are found to exceed that of the FC-BCJR equalizer at an input mutual information of approximately above 0.3 and 0.42 respectively. This is an interesting phenomenon since it shows that given an identical *a priori* LLR input that follows the same description as illustrated in Section 4.2, the *k*-Opt equalizer is able to outperform that of the BER-optimal FC-BCJR equalizer.

Also observed from the two figures is that the transfer functions of MMSE APRX I equalizer and MMSE APRX II equalizer have the same start and end points, respectively, as the MMSE EXACT equalizer. This indicates that the Hybrid MMSE equalizer [5], which is implemented via either MMSE APRX I or MMSE APRX II equalizers, is able to attain the performance of the MMSE EXACT equalizer given sufficient iterations and also under the condition that the SNR values in consideration are above its decoding threshold.

A further point to note here is that despite having different starting points in the transfer functions of the FC-BCJR equalizer, the MMSE EXACT equalizer and the *k*-Opt equalizer (as observed in the proposed transfer functions for the *k*-Opt equalizer), all three equalizer implementations are found to share the same end points for perfect *a priori* LLR as shown in Figures 4.20 and 4.21. This thus implies that all turbo equalizers implemented with the various mentioned equalization algorithms together with an identical decoder (for all cases) are able to attain quite similar final BER performances provided the SNR values in consideration are above their respective decoding thresholds. As such, at a particular SNR value that is well above the maximum decoding thresholds of all the equalizers in comparison, for example at a SNR value of 6.0 dB (refer to Table 4.5 for a list of their respective decoding thresholds), the two vital factors that will discriminate between the various equalizers' implementations will thus be their respective computational complexities and the numbers of iterations that are required to attain convergence.

The issue of computational complexities for the various equalization algorithms has been addressed in Chapter 3. As such, in this section, we would utilize the EXIT chart to compare the various equalizers' implementation in terms of the number of iterations required to attain convergence. For the *k*-Opt equalizer, the number of iterations to attain convergence to a fixed point is obtained using the proposed EXIT chart as outlined in Sections 4.4 and 4.5. The decoding thresholds for various turbo equalizers implemented using different equalization algorithms will also be presented. After which, the variations of the critical points of the transfer functions, at the input mutual information of 0.0, 0.5 and 1.0 specifically, of selected equalization algorithms over a SNR range from 0 dB to 20 dB will be presented. A quick overview of the various equalizers' implementations in terms of their performances over the stipulated SNR range can thus be seen. Besides that, it also allows us to understand the rationale behind using MMSE APRX I equalizer in the first iteration instead of either the FC-BCJR equalizer or the MMSE EXACT equalizer.

Finally, this section ends by investigating the effect of imperfect channel impulse response (CIR) knowledge on the critical points of the various equalizers' transfer functions. This investigation is primarily motivated by the findings in the previous section when we consider the variation of the *k*-Opt equalizer's output mutual information given a perfect input solution and zero *a priori* LLR as the SNR increases. From this hypothetical scenario, it is deduced that the *k*-Opt equalizer places a much heavier reliance on a good quality input solution rather than on a good quality *a priori* LLR. In the context of imperfect CIR knowledge given to the equalizer, the extrinsic LLR derived from the equalizer is expected to suffer degradation in its quality. This degradation will propagate through the decoder and eventually give rise to poor quality *a priori* LLR for the equalizer in the next iteration. As the FC-BCJR equalizer and the MMSE EXACT equalizer depend solely on the quality of *a priori* LLR, such degradation will therefore seriously affect their performances. On the other hand, the degradation in the quality of the *a priori* LLR may not seriously affect the *k*-Opt equalizer's performance due to its greater reliance on the quality of the input solution. Although the input solution is obtained by taking hard decision on the *a priori* LLR, the ensuing degradation in the quality of the input solution is still minimal since it is highly unlikely that the sign of the *a priori* LLR may be flipped due to degradation in its quality. As such, the plot depicting the variation of the *k*-Opt equalizer's output mutual information given a perfect input solution and zero *a priori* LLR as shown in Figure 4.19 can thus be viewed as a lower bound on the end-point of the *k*-Opt equalizer's transfer function in the context where imperfect CIR knowledge is given to the *k*-Opt equalizer.

Bearing this in mind, we go a step further to investigate the impact of imperfect CIR knowledge on the various equalizers' output mutual information at three critical points to observe the extent of degradation in the quality of the extrinsic LLR.

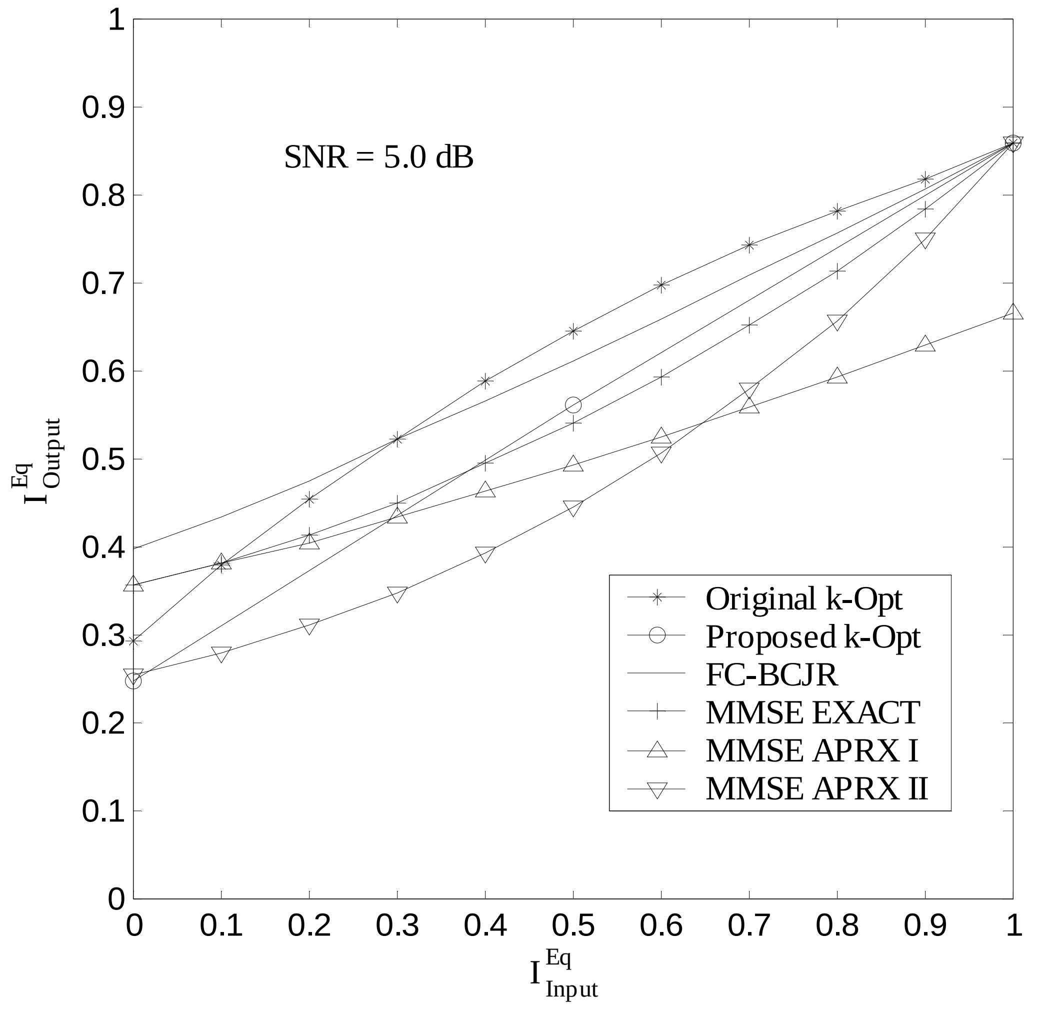

Figure 4.20: Comparison of selected equalizers' transfer functions at a SNR of 5.0 dB

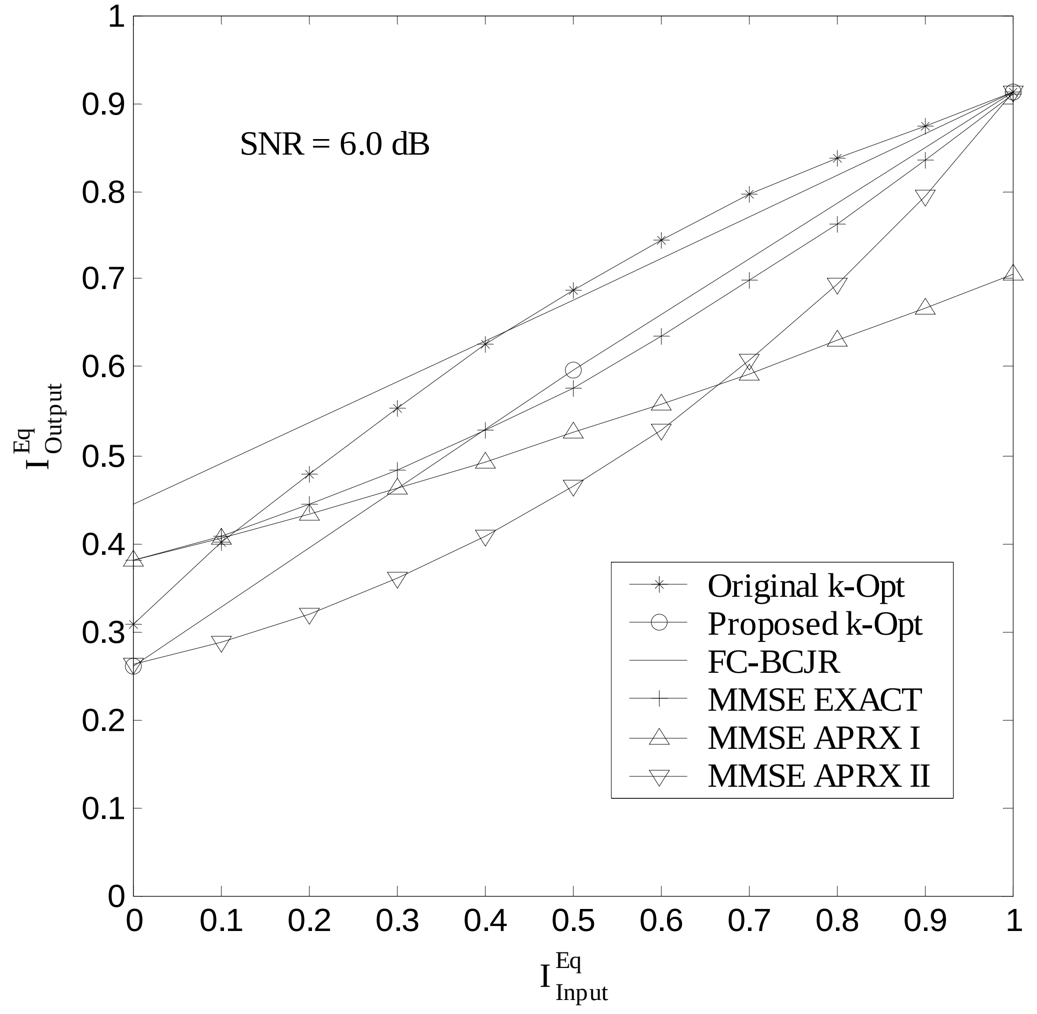

Figure 4.21: Comparison of selected equalizers' transfer functions at a SNR of 6.0 dB

4.6.1 Number of Iterations to Convergence

The number of iterations as obtained from the EXIT chart to attain convergence to a fixed point at a particular SNR value for the various equalizer implementations are shown in Table 4.4. The fixed points for all the equalizer implementations here are identical (except for the case of Hybrid MMSE equalizer at a SNR of 5.0 dB), which indicates that the BER performance of all the equalizers in comparison (upon convergence) are the same. For the case of the *k*-Opt turbo equalizer, the number of iterations reflected in the table include the first iteration that is carried out through the use of the MMSE APRX I equalizer.

As observed in Table 4.4, the use of the FC-BCJR equalizer allows the fastest convergence in terms of the numbers of iterations required. This is followed by the *k*-Opt equalizer and then the MMSE EXACT equalizer. The Hybrid MMSE equalizer does not converge to a good fixed point as compared with the rest of the equalizers at a SNR of 5.0 dB. This is because at this particular SNR, the transfer functions of both the MMSE APRX I and MMSE APRX II are found to lie below that of the decoder's transfer function in the EXIT chart. To be more precise, convergence to a good fixed point is not possible since this particular SNR is below the decoding threshold for the Hybrid MMSE equalizer (see Table 4.5). At an SNR of 6.0 dB which is above the decoding threshold of the Hybrid MMSE equalizer, it is found to be able to converge to the same fixed point as the other equalizers, albeit requiring more iterations as seen in Table 4.4.

From Tables 4.4 and 3.1, it could easily be deduced that there is a tradeoff in terms of the computational complexity of a particular equalizer's implementation and its rate of convergence in terms of the number of iterations.

Table 4.4: Number of iterations to convergence to a fixed point for selected equalizer implementations

SNR \ SISO Equalizer	FC-BCJR	MMSE EXACT	Hybrid MMSE	*k*-Opt LS
5.0 dB	7	12	No Convergence	11
6.0 dB	5	8	14	7

4.6.2 Decoding Thresholds

The decoding thresholds of selected turbo equalizers implemented via the various equalization algorithms mentioned are shown in Table 4.5 below. From the results presented, the FC-BCJR equalizer has the lowest. This attribute of the FC-BCJR equalizer coupled with the fact that it requires the least number of iterations to attain convergence thus explains its superior BER performances in the waterfall region as compared to the other equalizers observed in Figures 3.2 and 3.3. The decoding thresholds of the turbo equalizers implemented using either MMSE EXACT equalizer or *k*-Opt LS equalizer are found to be quite similar. As expected, the Hybrid MMSE equalizer has the highest decoding threshold.

Table 4.5: Decoding thresholds of selected turbo equalizer implementations

SISO Equalizer	Decoding Threshold
FC-BCJR	3.4 dB
MMSE EXACT	4.0 dB
Hybrid MMSE	5.3 dB
k-Opt LS	4.2 dB

4.6.3 Variations of Critical Points vs SNR (Comparisons)

To enable a quick overview of the transfer functions of the various equalizers, the variations in the critical points of their respective transfer functions, as the SNR increases, at an input mutual information of 0.0, 0.5 and 1.0 are shown in Figure 4.22. The parenthesis in the legend of this figure indicates the corresponding values of I^{Eq}_{Input} given to the respective equalizers.

In general, with the exception of the *k*-Opt LS equalizer, the critical points of the various equalizers' transfer functions are found to increase when the SNR increases. As expected, the FC-BCJR equalizer's transfer functions have the best start points (at input mutual information = 0.0) and mid-points (at input mutual information = 0.5) throughout the depicted SNR range. Though not shown in the figure, the start points of the transfer functions for MMSE APRX I equalizer are found to be identical as that of the MMSE EXACT. All the equalizers' transfer functions with the exception of MMSE APRX I equalizer (not shown) are found to have identical end points (at input mutual information = 1.0) as shown in Figure 4.22.

From the BER plots as shown in Figure 3.3, the *k*-Opt LS equalizer (with *q* = 2) is able to attain the AWGN bound at a SNR of 5.0 dB and above. Turning to Figure 4.22, between an SNR of 5.0 and 6.0 dB, the transfer functions' start-points of FC-BCJR equalizer (denoted by —+—) is found to be only slightly better than that of the MMSE EXACT equalizer or equivalently MMSE APRX I equalizer (denoted by —◇—). Since the MMSE APRX I equalizer has the lowest computational complexity compared to MMSE EXACT equalizer and FC-BCJR equalizer, it is thus the best choice to be used in the first iteration for the *k*-Opt turbo equalizer to jump-start the turbo equalization process, which is essential to avoid the low starting points of the *k*-Opt LS equalizer (denoted by —⊖—).

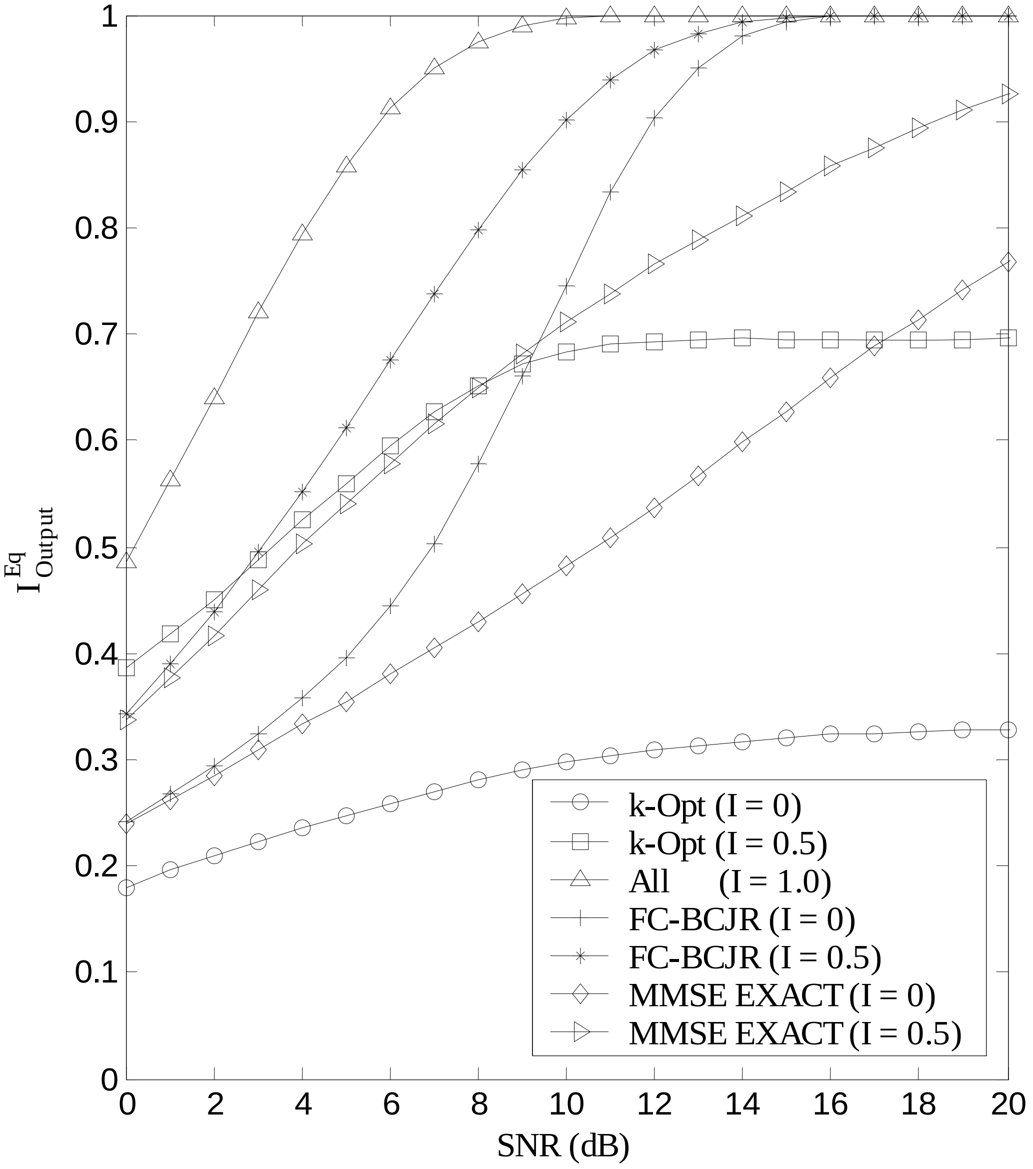

Figure 4.22: Output mutual information of selected equalizers vs SNR for different qualities of input *a priori* LLR

Besides jump-starting the equalization process to a higher value of I_{Output}^{Eq} for the first iteration, the use of the MMSE APRX I equalizer also allows the value of I_{Output}^{Eq} in the second iteration to reach the bound depicted by the original transfer function of the *k*-

Opt equalizer as seen in Figure 4.11 (this phenomenon is also true if FC-BCJR equalizer or the MMSE EXACT equalizer is used in the first iteration). Note that the original transfer function of the *k*-Opt equalizer has a higher $\mathrm{I}_{\mathrm{Output}}^{\mathrm{Eq}}$ values than the transfer function of the MMSE EXACT equalizer for $\mathrm{I}_{\mathrm{Input}}^{\mathrm{Eq}} > 0.1$ at both SNR values of 5.0 dB and 6.0 dB as observed in Figures 4.20 and 4.21. Also, in Figure 4.22, the mid-point of the transfer function for the *k*-Opt LS equalizer (denoted by —⊟—) within an SNR range of 0 to about 10 dB are found to be slightly better than that of the MMSE EXACT equalizer (denoted by —▷—). Furthermore, as observed in Figures 4.20 and 4.21, the *k*-Opt equalizer's transfer functions have a linear relationship connecting the mid- and end-points whereas the MMSE EXACT equalizer's transfer functions have a slight convex relationship (relative to the abscissa of the figures) connecting the mid- and end-points. These three reasons explain the better BER performance of the *k*-Opt turbo equalizer in the waterfall region as compared to the MMSE EXACT turbo equalizer seen in Figure 3.3.

4.6.4 Effects of Imperfect Knowledge of Channel Impulse Response

An investigation into the effects of imperfect channel impulse response (CIR) knowledge on the critical points of the various equalizers' transfer functions is carried out here. All the simulation parameters are kept constant (including the modeling of the *a priori* LLR; possible degradation in the quality of the a *priori* LLR at the input of the equalizers due to imperfect CIR knowledge is not considered here due to the difficulty in the modeling of the *a priori* LLR in such scenario) as that used in generating Figure 4.22 except that the equalizers are only given information on the length of the channel impulse response and not the exact values of the coefficients. With this in mind, together with the constraint of unit energy for the ISI channel for comparison purposes, the equalizers are hence provided with the channel knowledge of $f = \{0.447, 0.447, 0.447, 0.447, 0.447\}$ instead of $f = \{0.227, 0.460, 0.688, 0.460, 0.227\}$ which is the actual channel impulse response used in the transmission.

The effects of imperfect CIR knowledge on the critical points of the transfer functions for the various equalizers are shown in Figure 4.23. Since the *a priori* LLR is modeled as before in the context where perfect CIR knowledge is available to the respective equalizers, the corresponding variations in $\mathrm{I}_{\mathrm{Output}}^{\mathrm{Eq}}$ over the depicted SNR range are thus not indicative of their actual values in such a scenario. In another words, the predicted trajectory obtained from the EXIT charts corresponding to these values of $\mathrm{I}_{\mathrm{Output}}^{\mathrm{Eq}}$ shown in Figure 4.23 at a specific SNR value may not be similar to the averaged system trajectory obtained by simulation of the actual system setup. Nevertheless, Figure 4.23 can still be used to observe the extent of degradation in the quality of the extrinsic LLR between the various equalizers by comparing with Figure 4.22.

In general, comparing Figures 4.22 and 4.23, the values of $\mathrm{I}_{\mathrm{Output}}^{\mathrm{Eq}}$ for the critical points of all the equalizers are lower when imperfect CIR knowledge are given to the equalizers. However, interestingly, the extent of degradation of $\mathrm{I}_{\mathrm{Output}}^{\mathrm{Eq}}$ for the *k*-Opt equalizer at the start and mid-points of its transfer function is found to be less severe compared to that of both the FC-BCJR and MMSE EXACT equalizer.

Recalling that the start point for the *k*-Opt equalizer as seen in Figure 4.22 are the lowest as compared to that of the FC-BCJR and MMSE EXACT equalizer. However, in the context of imperfect CIR knowledge as observed in Figure 4.23, the start points of *k*-

Opt equalizer is found to be better than that of the other two from 0.0 dB up to approximately 12.0 dB. This phenomenon is also manifested in the mid-point of the *k*-Opt LS equalizer's transfer functions depicted in Figure 4.23 to the extent that it is well-above that of the FC-BCJR equalizer and MMSE EXACT equalizer for the entire stipulated SNR range. Surprisingly, the mid-points of both the FC-BCJR equalizer and MMSE EXACT equalizer are observed to deteriorate in the higher SNR regions. The end points of all the equalizers are found to be similar; although that of the FC-BCJR equalizer and the MMSE equalizer are found to be slightly better than that of the *k*-Opt LS equalizer from about 4.0 dB onwards.

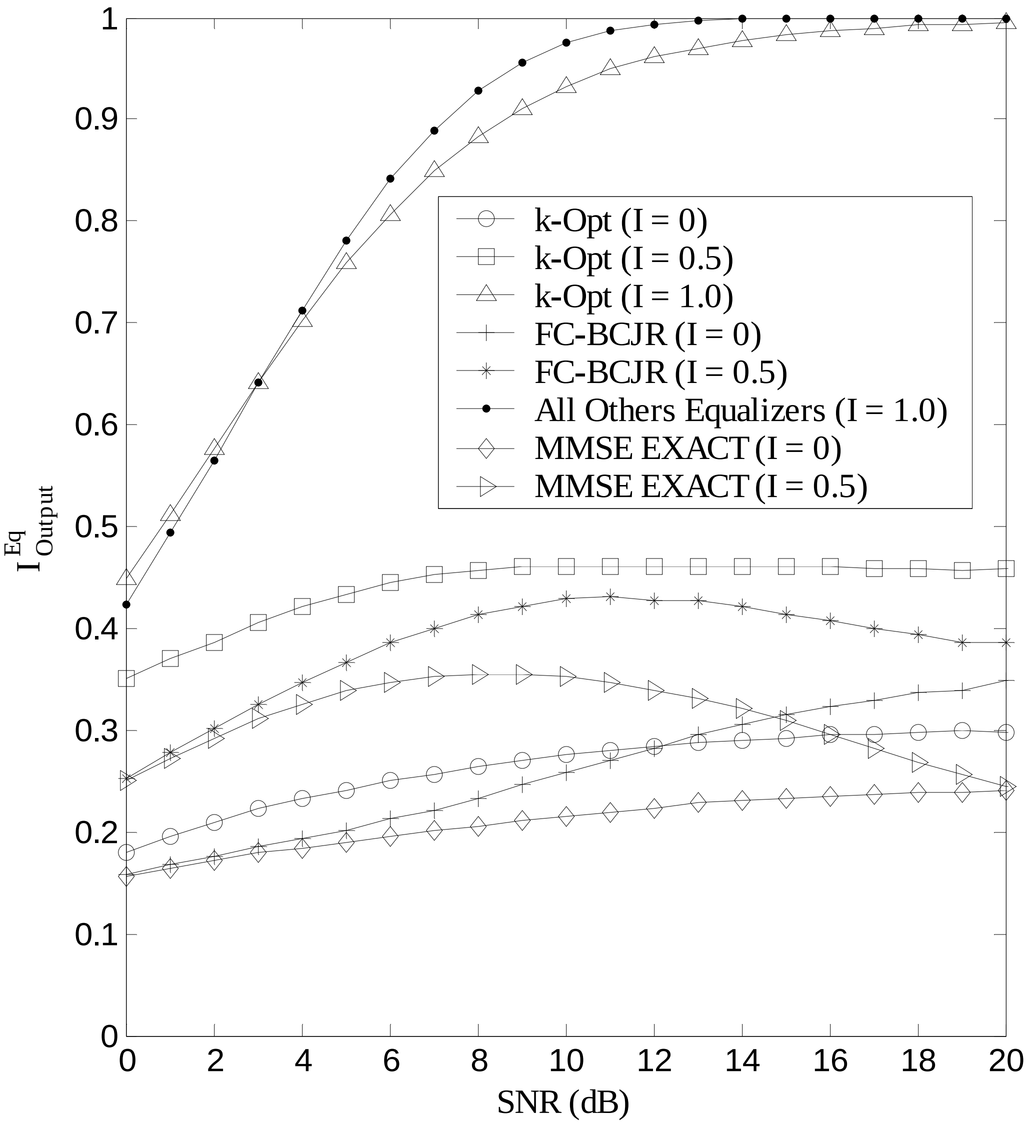

Figure 4:23: Effects of imperfect CIR knowledge on critical points of selected equalizers' transfer functions

4.7 Simulation Results and Discussion

In this section, we present BER performance results for the case where imperfect CIR knowledge is available to the respective equalizers. All the simulation parameters are kept constant as in Chapter 2 and Chapter 3. The only difference is that the equalizer is given the CIR knowledge of $f = \{0.447, 0.447, 0.447, 0.447, 0.447\}$ instead of the actual CIR that is used in the respective simulations.

As a brief summary, the purpose of this investigation is primarily motivated by the following two observations made in Section 4.5, namely:

1) The quality of the extrinsic LLR from the *k*-Opt LS equalizer is found to be heavily dependent on a good quality input solution rather than on a good quality *a priori* LLR. Although the input solution is obtained by taking hard decision on the *a priori* LLR, the ensuing degradation (due to imperfect CIR knowledge) in the quality of the input solution should be minimal since it is highly unlikely that the sign of the *a priori* LLR may be flipped.
2) The extent of degradation in the quality of the extrinsic LLR in the context where imperfect CIR knowledge is provided to the *k*-Opt LS equalizer is less severe as compared to both FC-BCJR equalizer and MMSE EXACT equalizer. Importantly, the quality of the extrinsic LLR for the *k*-Opt LS equalizer in such a scenario is found to be better than that of the BER-optimal FC-BCJR equalizer and the MMSE EXACT equalizer given an *a priori* LLR of input mutual information of 0.5.

With these two observations above, it is possible to reason that the *k*-Opt turbo equalizer could outperform both the FC-BCJR turbo equalizer and the MMSE EXACT turbo equalizer in the situation where imperfect CIR knowledge is available to the equalizer. To prove our hypothesis, we shall present the simulation results to verify our proposition.

We first consider turbo equalization under imperfect CIR knowledge over the mild ISI conditions described by Channel I in Table 2.2. The BER performances of the various equalizers under this scenario are shown in Figure 4.24 (denoted in parenthesis as "Imperfect"). Also shown in this figure are the BER performances of the respective equalizers in the situation where perfect CIR knowledge is available; these BER curves are denoted in parenthesis as "Perfect" in Figure 4.24 and are exactly the same as that depicted previously in Figure 3.2.

From Figure 4.24, the BER performances of the various equalizers under imperfect CIR knowledge are found to be worse compared to the situation where perfect CIR knowledge is available. Where perfect CIR knowledge is available, the best performing equalizer in terms of BER performance is found to be the FC-BCJR equalizer. However, in the context where imperfect CIR knowledge is provided, the best performing equalizer in terms of BER performance turns out to be the *k*-Opt turbo equalizer. In the BER range of around 10^{-5}, the *k*-Opt turbo equalizer is found to be about 1.0 dB away from the AWGN bound. On the other hand, the FC-BCJR turbo equalizer and the MMSE EXACT turbo equalizer are found to be approximately 4.0 dB and 5.6 dB away from the AWGN bound respectively. This is an interesting phenomenon as the *k*-Opt turbo equalizer is able to outperform the BER optimum FC-BCJR turbo equalizer. We note that the BER curve of the MMSE EXACT (under imperfect CIR knowledge) actually worsen off with an increase in SNR within the 7 – 9 dB region. This could be due to error propagation (during the exchange of inaccurate extrinsic information derived from imperfect CIR knowledge) since the corresponding BER with fewer iterations (i.e., less than 15 iterations) within this SNR region is better.

Next, we carry out simulation studies over the severe ISI channel denoted as Channel II in Table 2.2 depicted in Chapter 2. Recall that this channel is also known as the Proakis C Channel. The BER performance of the respective turbo equalizers given perfect (denoted in parenthesis as “Perfect”) and imperfect (denoted in parenthesis as “Imperfect”) CIR knowledge are shown in Figure 4.25. To avoid unnecessary cluttering of the figure, the BER performance for both the FC-BCJR turbo equalizer and MMSE EXACT turbo equalizer are plotted from 6.0 dB onwards.

Similar to Figure 4.24, the BER performance, as observed in Figure 4.25, of the various turbo equalizers under imperfect CIR knowledge are worse than their respective counterparts under perfect CIR knowledge. The *k*-Opt turbo equalizer is also found to be the best performing turbo equalizer in terms of BER performance as compared to the FC-BCJR turbo equalizer and MMSE EXACT turbo equalizer under imperfect CIR knowledge. At a BER of around 10^{-5}, the *k*-Opt turbo equalizer is found to be approximately 5.0 dB away from the AWGN bound. As shown in Figure 4.25, the FC-BCJR turbo equalizer and MMSE EXACT turbo equalizer are not able to attain a BER of 10^{-5} even up to a SNR of 20.0 dB.

We conclude that the *k*-Opt turbo equalizer performs very well where imperfect CIR knowledge is available to the equalizer. This advantageous property of the *k*-Opt turbo equalizer hints its robustness in practical system implementation and also implies that highly precise channel identification process (for a time-invariant channel) may not be necessary when a *k*-Opt turbo equalizer is utilized. For a time-varying channel where training sequences are required to be send through the channel periodically, a shorter training sequence may suffice in ensuring that the *k*-Opt turbo equalizer can still function properly, thereby leading to the advantage of a higher overall code-rate.

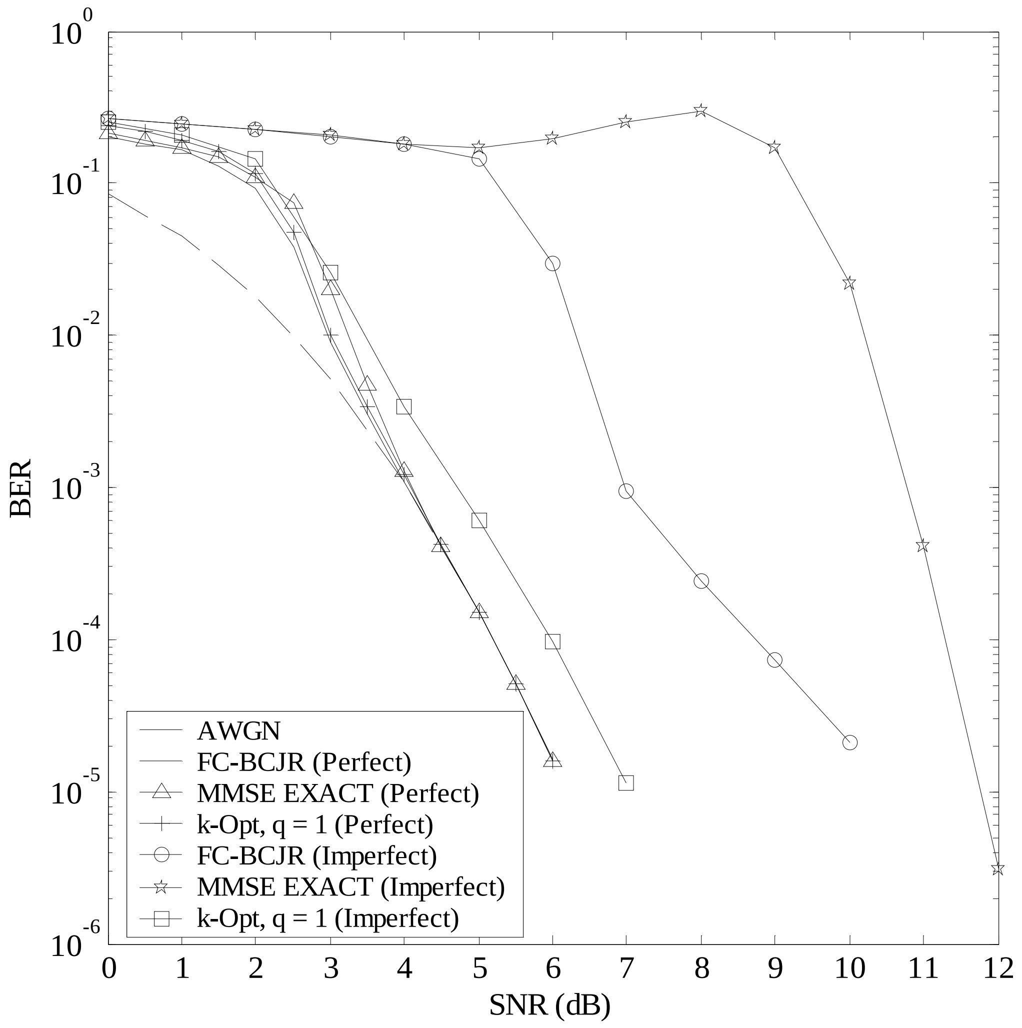

Figure 4.24: Performance comparison between *k*-opt LS turbo equalizer and other equalization algorithms for Channel I under imperfect CIR knowledge of $f = \{0.447, 0.447, 0.447, 0.447, 0.447\}$

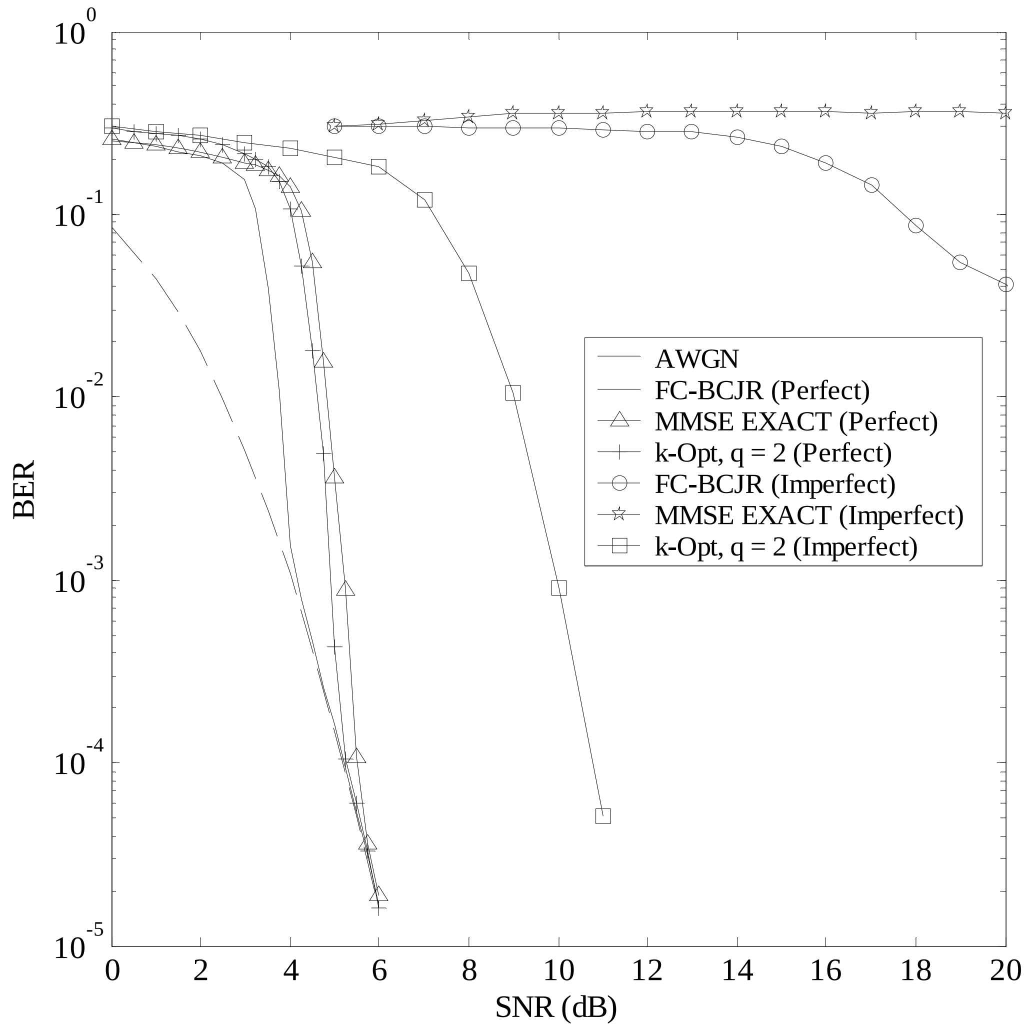

Figure 4.25: Performance comparison between *k*-opt LS turbo equalizer and other equalization algorithms for Channel II (Proakis C Channel) under imperfect CIR knowledge of $f = \{0.447, 0.447, 0.447, 0.447, 0.447\}$

4.8 Chapter Summary

The proposed EXIT chart for the *k*-Opt turbo equalizer has been verified through asymptotic simulations of very large frame size to be more accurate than the original EXIT chart in predicting the BER performances at each iterations and the required number of iterations to convergence to a fixed point. Through the proposed EXIT chart, several important and interesting insights on the *k*-Opt turbo equalizers are discovered. Firstly, it is found that the *k*-Opt equalizer cannot operate in conjunction with a high code-rate RSC code. Secondly, the *k*-Opt equalizer is found to place a heavier reliance on the quality of its input solution rather than on the quality of the *a priori* LLR in producing a good quality extrinsic LLR. Thirdly, the extent of degradation in the quality of the extrinsic LLR in the case where imperfect CIR knowledge is given to the receiver is found to be the lowest for the *k*-Opt turbo equalizer. Motivated by the last two observations obtained through the proposed EXIT chart for the *k*-Opt turbo equalizer, simulation studies on selected turbo equalizers in the case of imperfect CIR knowledge are carried out. From the simulation results and discussion that follows, it is found that the *k*-Opt turbo equalizer may be a serious contender for practical system implementations as compared to the FC-BCJR turbo equalizer and the MMSE EXACT turbo equalizer.

Chapter 5

Conclusions and Further Work

In this thesis, an investigation is carried out on low complexity turbo equalizations over coded intersymbol interference channels, with an emphasis on the aspects of equalization algorithms. In particular, a novel class of low complexity equalization algorithms based on heuristic approach as applied to a turbo equalization scheme has been proposed. Analytical work carried out through the use of the EXIT chart reveals interesting insights on the proposed heuristic-based turbo equalizer that explains its superior performances over their filter-based counterparts. Furthermore, in situations where imperfect channel impulse response knowledge is given to the respective classes of equalization algorithms, we have even discovered that our proposed heuristic-based turbo equalizer surpasses their trellis-based counterparts in terms of BER performances.

The thesis can be divided into three parts. In Part 1, which consists of Chapter 1 and 2, a survey of the existing literature in the area of optimal and sub-optimal (low complexity alternatives) equalization algorithms as applied to a turbo equalization scheme has been carried out. In particular, two well-known classes of equalization algorithms, namely, the trellis-based equalization algorithms and the filter-based equalization algorithms have been studied. From the class of trellis-based equalization algorithms, the BCJR algorithms and its low-complexity variants are focused. Then, from the class of filter-based equalization algorithms, the MMSE EXACT and its low complexity approximate implementations are presented. In Part 2 of the thesis, which comprises of Chapter 3, a novel equalization algorithm as applied to a turbo equalization scheme based on heuristic approach is proposed. Thereafter in Part 3, which consists of Chapter 4, we investigate the proposed heuristic-based LS turbo equalizers through the use of the EXIT chart to draw insights into its asymptotic performances. The details of the work contributed in Part 2 and 3 of the thesis are elaborated further below.

In Part 2 of the thesis, a general framework has been established in the context of turbo equalization such that any heuristic search methods (beside the local search heuristics that is focused in this thesis) can be applied to perform the equalization task. Furthermore, we have shown through simulation results and computational complexity analysis that our proposed local search turbo equalizers utilizing *k*-Opt heuristics is a viable alternative to the trellis-based BCJR turbo equalizers and the filter-based MMSE turbo equalizers (in terms of both BER performances and computational complexity considerations) in situation where perfect channel impulse response knowledge is available to the respective receivers.

In Part 3 of the thesis, a general approach for the application of the EXIT chart has been proposed and is found to be accurate in predicting the asymptotic behaviors of our proposed *k*-Opt turbo equalizer. Through the proposed EXIT chart, a discovery on the robustness of the *k*-Opt turbo equalizer against imperfect channel impulse response knowledge is unearthed. Subsequent simulation results which verified this important findings further reinforce the idea that the proposed *k*-Opt turbo equalizer could well be a serious contender to the BCJR turbo equalizer and the MMSE turbo equalizer in actual system implementation where perfect channel knowledge is usually unavailable to the receiver.

Directions for Further Work

Several important issues arise during the EXIT chart analysis carried out on our proposed *k*-Opt turbo equalizer which would be worth addressing here as possible directions for further work. These issues are highlighted below.

The averaged system trajectory of the *k*-Opt turbo equalizer is not able to attain the prediction given by the original transfer function of the *k*-Opt equalizer. A post-processing operation could be carried out on the extrinsic LLR derived from the *k*-Opt equalizer in a similar manner as that used in [59][60] to allow the averaged system trajectory achieves the bound dictated by the original transfer function. This would thus lead to a faster rate of convergence for the *k*-Opt turbo equalizer and also the possibility of employing a high-rate outer code.

EXIT chart analysis and simulation results show that the *k*-Opt turbo equalizer outperforms the FC-BCJR turbo equalizer and the MMSE EXACT turbo equalizer in situation where imperfect channel impulse response knowledge is given. However, it is still unclear whether it is the use of the heuristic-based equalizer or the use of a mixture of hard and soft *a priori* information that have resulted in such a phenomenon.

Preliminary investigations into the effects of channel knowledge on the BER performances of the respective turbo equalizers reveal that the FC-BCJR turbo equalizer can attain a BER of around 10^{-5} within the stipulated SNR range (refer to Figure 4.25) under Channel II (Proakis C Channel) provided a better channel impulse response knowledge is available. This phenomenon is also observed for the MMSE EXACT turbo equalizer. Viewing the coefficients of a time-invariant channel as a point in a vector space, a simple distance measurement between two points show that the given channel impulse response knowledge of $f = \{0.447, 0.447, 0.447, 0.447, 0.447\}$ is at a distance of approximately 0.3514 and 0.3940 away from that of Channel I and Channel II respectively. Recall that the MMSE EXACT turbo equalizer is able to attain a BER of 10^{-5} for Channel I but not for Channel II when given the channel knowledge of $f = \{0.447, 0.447, 0.447, 0.447, 0.447\}$ for both cases. Consequently, this hinted that there exists a sphere centered around the actual channel impulse response whereby any given channel knowledge that falls outside this sphere will render convergence to a good BER by a particular turbo equalizer's implementation impossible. Bearing this in mind, the radius of such a sphere for each turbo equalizer's implementation could be an additional discriminating criterion, beside BER performances and computational complexity etc., which could be used to differentiate the suitability of the various turbo equalizers' implementations in actual system usage. Conversely, the knowledge of such a parameter could be used to dictate the minimum required accuracy level of channel identification such that a particular turbo equalizer's implementation is expected to perform.

Bibliography

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near Shannon limit error-correcting coding and decoding: Turbo codes,” *Proc. IEEE Intl. Conf. Commun. (ICC)*, Geneva, Switzerland, pp. 1064-1070, May 1993.

[2] J. Lodge and M. Gertsman, “Joint detection and decoding by turbo processing for fading channel communications,” *Proc. Intl. Symp. Turbo Codes and Related Topics, Brest, France*, pp. 88-95, Sept. 1997.

[3] C. Douillard, A. Picart, M. Jezequel, P. Didier, C. Berrou, and A. Glavieux, “Iterative correction of intersymbol interference: Turbo-equalization,” *Eur. Trans. Telecommun.*, vol. 6, pp. 507-511, Sept.-Oct. 1995.

[4] J. Hagenauer, “The turbo principle: Tutorial introduction and state of the art,” *Proc. IEEE Intl. Symp. Inform. Theory (ISIT)*, Brest, France, pp. 1-11, Sept. 1997.

[5] M. Tüchler, R. Koetter, and A. C. Singer, “Turbo equalization: Principles and new results,” *IEEE Trans. Commun.*, vol. 50, no. 5, pp. 754-767, May 2002.

[6] T. V. Souvignier, M. Öberg, P. H. Siegel, R. E. Swanson, and J. K. Wolf, “Turbo decoding for partial response channels,” *IEEE Trans. Commun.*, vol. 48, no. 8, pp. 1297-1308, Aug. 2000.

[7] R. Koetter, A. C. Singer, and M. Tüchler, "Turbo equalization: An iterative equalization and decoding technique for coded data transmission," *IEEE Signal Process. Mag.*, vol. 21, no. 1, pp. 67-80, Jan. 2004.

[8] S. Lin and D. J. Costello, Jr., *Error Control Coding: Fundamentals and Application*, 2nd ed., Prentice-Hall, 2004.

[9] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, no. 10, pp. 1727-1737, Oct. 2001.

[10] J. Hagenauer, "The EXIT chart – Introduction to extrinsic information transfer in iterative processing," EUSIPCO, 2004.

[11] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284-287, Mar. 1974.

[12] X. Li and T. F. Wong, "Turbo equalization with nonlinear Kalman filtering for time-varying frequency-selective fading channels," *IEEE Trans. Wireless Commun.*, vol. 6, no. 2, pp. 691-700, Feb. 2007.

[13] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft decision outputs and its applications," *Proc. IEEE Intl. Conf. Global Telecommun. (GLOBECOM)*, Dallas, Texas, pp. 1680-1688, Nov. 1989.

[14] S. Chen, B. Mulgrew, and P. M. Grant, "A clustering technique for digital communications channel equalization using radial basis function networks," *IEEE Trans. Neural Networks*, vol. 4, no. 4, pp. 570-579, July 1993.

[15] M. S. Yee, B. L. Yeap, and L. Hanzo, "Radial basis function-assisted turbo equalization," *IEEE Trans. Commun.*, vol. 51, no. 4, pp. 664-675, Apr. 2003.

[16] F. Zhao, G. Mathew, and B. Farhang-Boroujeny, “Techniques for minimizing error propagation in decision feedback detectors for recording channels,” *IEEE Trans. Magn.*, vol. 37, no. 1, pp. 592-602, Jan. 2001.

[17] D. Fertonani, A. Barbieri, and G. Colavolpe, “Reduced-complexity BCJR algorithm for turbo equalization,” *IEEE Trans. Commun.*, vol. 55, no. 12, pp. 2279-2287, Dec. 2007.

[18] C. Fragouli, N. Seshadri, and W. Turin, “On the reduced trellis equalization using the M-BCJR algorithm,” AT&T Labs-Research, pp. 1-13, Nov. 1999.

[19] M. Sikora and D. J. Costello, Jr., “A new SISO algorithm with application to turbo equalization,” *Proc. IEEE Intl. Symp. Inform. Theory (ISIT)*, Adelaide, Australia, pp. 2031-2035, Sept. 2005.

[20] B. J. Frey and F. R. Kschischang, “Early detection and trellis splicing: Reduced-complexity iterative decoding,” *IEEE J. Select. Areas Commun. (JSAC)*, vol. 16, no. 2, pp. 153-159, Feb. 1998.

[21] V. Franz and J. B. Anderson, “Concatenated decoding with a reduced-search BCJR algorithm,” *IEEE J. Select. Areas Commun. (JSAC)*, vol. 16, no. 2, pp. 186-195, Feb. 1998.

[22] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, “A soft-input soft-output maximum a posteriori (MAP) module to decode parallel and serial concatenated codes,” TDA Progress Report 42-127, pp. 1-20, Nov. 1996.

[23] C. Hu, K. C. The, Y. L. Guan and Z. Qin, “Performance of iterative multiuser receivers for turbo-coded asynchronous DS-CDMA systems over flat-fading channels,” *IEE Proc. Commun.*, vol. 152, no. 5, pp. 705-712, Oct. 2005.

[24] M. Tüchler, *Iterative Equalization using Priors*, M.Sc Thesis, University of Illinois, 2000.

[25] P. Merz, *Memetic Algorithms for Combinatorial Optimization Problems: Fitness Landscapes and Effective Search Strategies*, Ph.D Thesis, University of Siegen, Dec. 2000.

[26] P. Merz and B. Freisleben, "Greedy and local search heuristics for unconstrained binary quadratic programming," *J. Heuristics*, vol. 8, pp. 197-213, 2002.

[27] S. Lin and B. Kernighan, "An effective heuristic algorithm for the traveling salesman problem," *Operation Research*, vol. 21, pp. 498-516, 1973.

[28] E. H. L. Aarts and J. K. Lenstra, *Local Search in Combinatorial Optimization*, Wiley, 1997.

[29] P. H. Tan and L. K. Rasmussen, "Multiuser detection in CDMA - A comparison of relaxations, exact, and heuristic search methods," *IEEE Trans. Wireless Commun.*, vol. 3, no. 5, pp. 1802-1809, Sept. 2004.

[30] C. Ergün and K. Hacioglu, "Multiuser detection using a genetic algorithm in CDMA communications systems," *IEEE Trans. Commun.*, vol. 48, no. 8, pp. 1374-1383, Aug. 2000.

[31] A. AlRustamani and B. R. Vojcic, "A new approach to greedy multiuser detection," *IEEE Trans. Commun.*, vol. 50, no. 8, pp. 1326-1336, Aug. 2002.

[32] J. Hu and R. S. Blum, "A gradient guided search algorithm for multiuser detection," *IEEE Commun. Lett.*, vol. 4, no. 11, pp. 340-342, Nov. 2000.

[33] M. Naeem, S. I. Shah, and H. Jamal, "Multiuser detection in CDMA fast fading multipath channel using heuristic genetic algorithms," *IEC05 Proc. Conf. 5th Intl. Enformatika*, Prague, Czech Republic, vol. 7, pp. 440-445, Aug. 2005.

[34] Z. Qin, K. Cai, and X. Zou, "A reduced-complexity iterative receiver based on simulated annealing for coded partial-response channels," *IEEE Trans. Magn.*, vol. 43, no. 6, pp. 2265-2267, Jun. 2007.

[35] Z. Qin, K. Cai, and X. Zou, "Turbo multiuser detection based on local search algorithms," *Proc. IEEE Intl. Conf. Commun. (ICC)*, Glasgow, Scotland, pp. 5987-5992, Jun. 2007.

[36] Z. Qin and K. C. Teh, "Reduced-complexity turbo equalization for coded intersymbol interference channels based on local search algorithms," to appear in *IEEE Trans. Veh. Tech.*, 2007.

[37] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 599-618, Feb. 2001.

[38] A. Kavčić, X. Ma, and M. Mitzenmacher, "Binary intersymbol interference channels: Gallager codes, density evolution, and code performance bounds," *IEEE Trans. Inform. Theory*, vol. 49, no. 7, pp. 1636-1652, July 2003.

[39] J. Li, K. R. Narayanan, E. Kurtas, and C. N. Georghiades, "On the performance of high-rate TPC/SPC codes and LDPC codes over partial response channels," *IEEE Trans. Commun.*, vol. 50, no. 5, pp. 723-734, May 2002.

[40] J. G. Proakis, *Digital Communications*, 4th ed., New York: McGraw-Hill, 2001.

[41] M. C. Valenti, *Iterative Detection and Decoding for Wireless Communication*, Ph.D Thesis, Virginia Polytechnic Institute and State University, July 1999.

[42] S. Verdú, *Multiuser Detection*, U.K.: Cambridge Univ. Press, 1998.

[43] G. Colavolpe and A. Barbieri, "On MAP symbol detection for ISI channels using the Ungerboeck observation model," *IEEE Commun. Lett.*, vol. 9, no. 8, pp. 720-722, Aug. 2005.

[44] D. Raphaeli, "Combined turbo equalization and turbo decoding," *IEEE Commun. Lett.,* vol. 2, pp. 107-109, Apr. 1998.

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

[46] K. Li and X. Wang, "EXIT chart analysis of turbo multiuser detection," *IEEE Trans. Wireless Commun.*, vol. 4, no. 1, pp. 300-311, Jan. 2005.

[47] K. R. Narayanan, X. Wang, and G. Yue, "Estimating the PDF of the SIC-MMSE equalizer output and its applications in designing LDPC codes with turbo equalization," *IEEE Trans. Wireless Commun.*, vol. 4, no. 1, pp. 278-287, Jan 2005.

[48] S. J. Lee, A. C. Singer, and N. R. Shanbhag, "Linear turbo equalization analysis via BER transfer and EXIT charts," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 2883-2897, Aug. 2005.

[49] R. V. Tamma, *Iterative Equalization and Decoding using Reduced-State Sequence Estimation Based Soft-Output Algorithms*, M.Sc. Thesis, Texas A&M University, May 2003.

[50] R. D. R. Lopes, *Iterative Estimation, Equalization and Decoding*, Ph.D Thesis, Georgia Institute of Technology, July 2003.

[51] C. M. Kellett and S. R. Weller, "Fixed points of EXIT charts," *Proc. 7th Aust. Commun. Theory Workshop*, Perth, Australia, pp. 137-142, Feb. 2006.

[52] A. O. Yilmaz, "Waterfall region analysis for iterative decoding," *Proc. IEEE 15th Intl. Symp. Personal, Indoor and Mobile Radio Commun. (PIMRC)*, vol. 4, pp. 2576-2580, Sept. 2004.

[53] D. Divsalar, S. Dolinar, and F. Pollara, "Iterative turbo decoder analysis based on density evolution," *IEEE J. Select. Areas Commun. (JSAC)*, vol. 19, no. 5, pp. 891-907, May 2001.

[54] K. R. Narayanan, "Effect of precoding on the convergence of turbo equalization for partial response channels," *IEEE J. Select. Areas Commun. (JSAC)*, vol. 19, no. 4, pp. 686-698, Apr. 2001.

[55] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, New York: Wiley, 1991.

[56] J. K. Tugnait, "Identification and deconvolution of multichannel linear non-Gaussian process using higher order statistics and inverse filter criteria," *IEEE Trans. Signal Process.*, vol. 45, no. 3, pp. 658-672, Mar. 1997.

[57] X. Luo and D. N. Schramm, "Kurtosis, skewness and non-Gaussian cosmological density perturbations," *J. Astrophysical*, vol. 408, pp. 33, 1993.

[58] F. R. Rad and J. Moon, "Turbo equalization utilizing soft decision feedback," *IEEE Trans. Magn.*, vol. 41, no. 10, pp. 2998-3000, Oct. 2005.

[59] G. Lechner and J. Sayir, "Improved sum-min decoding of LDPC codes," *Proc. IEEE Intl. Symp. Inform. Theory Applications (ISITA)*, Parma, Italy, Oct. 2004.

[60] G. Lechner, and J. Sayir, "Improved sum-min decoding of irregular LDPC codes using nonlinear post-processing," *NEWCOM-ACoRN Joint Workshop*, Vienna, Austria, Sept. 2006.