NON-EQUIPROBABLE MULTI-LEVEL CODING FOR THE ADDITIVE WHITE GAUSSIAN NOISE CHANNEL WITH TIKHONOV PHASE ERROR

By

LI NI

A dissertation submitted in partial fulfillment of
the requirements for the degree of

DOCTOR OF PHILOSOPHY

ELECTRICAL ENGINEERING

WASHINGTON STATE UNIVERSITY
School of Electrical Engineering and Computer Science

DECEMBER 2005

To the Faculty of Washington State University:

The members of the Committee appointed to examine the dissertation of LI NI find it satisfactory and recommend that it be accepted.

Chair

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Dr. Benjamin Belzer for his invaluable guidance and consistent support during my five years graduate study in Washington State University. Without his advice and encouragement, this dissertation would not have been possible. His broad knowledge and experience in communication theory have helped me to build a solid foundation in digital/wireless communication area, which will definitely be an asset in my future professional career.

I would like to extend my sincere appreciation to Dr. Thomas Fischer and Dr. Ali Saberi. By sitting in their classes, I learned invaluable knowledge in information theory, data compression, signal processing, stochastic process theory and control theory. This knowledge helped me a lot in my research work.

The work in this dissertation was partially supported by the National Science Foundation under grant CCR-0098357. I want to express my appreciation to NSF. Also, I want to thank Comtech AHA Corporation for the internship, where I earned invaluable industry experience.

I also wish to thank the staff of the Electrical Engineering and Computer Science Department, for their help and support.

Finally, I would like to express my deepest gratitude to my parents and my family, for their love, care, patience and encouragement.

NON-EQUIPROBABLE MULTI-LEVEL CODING FOR THE ADDITIVE WHITE GAUSSIAN NOISE CHANNEL WITH TIKHONOV PHASE ERROR

Abstract

by Li Ni, Ph.D.
Washington State University
December 2005

Chair: Benjamin Belzer

In this dissertation, we present new design techniques for non-equiprobable multi-level codes with iterative decoding, for the AWGN channel with Tikhonov phase error. This channel models the phase estimation error of a phase lock loop (PLL) circuit with pilot-tone carrier recovery. Previously published results have shown that the capacity-achieving PDF for the average-power-constrained AWGN/Tikhonov channel is discrete-amplitude-uniform-phase (DAUP), and that the DAUP-input capacity can be closely approximated by multi-ring PSK, which we use in our codes. A key innovation is serially concatenated binary-input binary output codes with non-equiprobable output bits. These codes consist of an outer non-linear block code and an inner recursive convolutional code. The codewords of the non-linear outer code consist of selected sequences from a convolutional code of known free distance. At 1 bit/symbol, the multi-level codes approach the DAUP-input capacity within 1.1 dB, and outperform comparable equiprobable codes by 1.70 dB, at 0.5 bit/symbol outperform the equiprobable constellation-constrained capacity by 0.33 dB. Although targeted at the AWGN/Tikhonov channel, our MLC architecture is easily adapted to other channels for which the capacity-achieving PDF is DAUP.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In a digital communication system, the propagation delay in the transmitted signal results in a carrier phase offset, which must be estimated at the receiver if the detector is phase-coherent. The phase-lock-loop (PLL) circuit with pilot-tone is a commonly used phase estimation scheme in wireless communication receivers. Fig. 1.1 describes the receiver block diagram with PLL. The pilot tone $c(t)$ can be isolated from $y(t)$ by means of a tone-in-band scheme such as [3]. Due to the loop noise, the PLL outputs the recovered phase with a phase estimation error $\theta_a - \hat{\theta}_a$, so the channel phase cannot be perfectly recovered. From the point of either theory or practice, it is a challenge to investigate the coding problem on such a channel with the phase estimation error. In this dissertation, we design power and bandwidth efficient coded modulation for the partially coherent additive white Gaussian noise (PCAWGN) channel, where the channel noise is assumed to be additive white Gaussian noise (AWGN) and the partial coherence comes from the phase estimation error in PLL circuit. Specifically, the AWGN/Tikhonov channel models the phase estimation

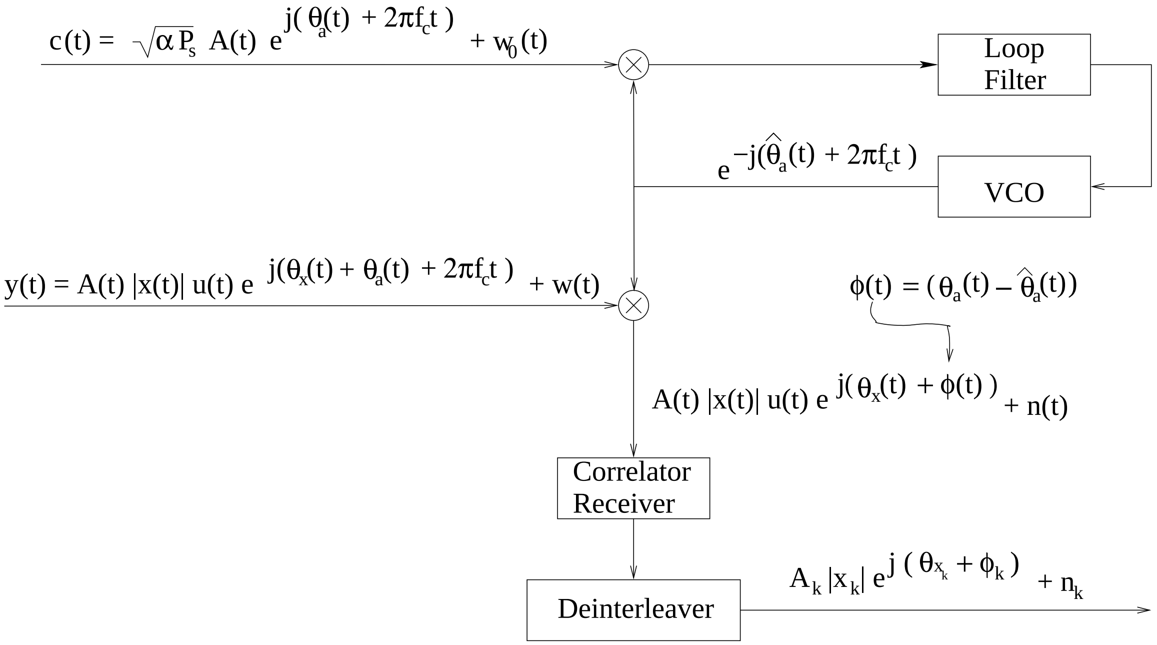

$c(t) = \sqrt{\alpha P_s}\, A(t)\, e^{j(\theta_a(t) + 2\pi f_c t)} + w_0(t)$
Loop Filter
$e^{-j(\hat{\theta}_a(t) + 2\pi f_c t)}$
VCO
$y(t) = A(t)\,|x(t)|\,u(t)\,e^{j(\theta_x(t) + \theta_a(t) + 2\pi f_c t)} + w(t)$
$\phi(t) = (\theta_a(t) - \hat{\theta}_a(t))$
$A(t)\,|x(t)|\,u(t)\,e^{j(\theta_x(t) + \phi(t))} + n(t)$
Correlator Receiver
Deinterleaver
$A_k\,|x_k|\,e^{j(\theta_{x_k} + \phi_k)} + n_k$

Figure 1.1: Block diagram of correlator receiver with pilot-tone PLL

error with the Tikhonov probability distribution function (PDF). The starting point of channel coding is to look into the Shannon channel capacity. Some results of previous work on PCAWGN channel capacity give us guidelines for designing the modulation and channel coding. Since the PCAWGN channel capacity indicates the potential of non-equiprobable signaling, the combination of non-equiprobable multi-ring PSK constellations and multi-level coding allows us to achieve both coding and shaping gain, and is therefore a good choice for the PCAWGN channel.

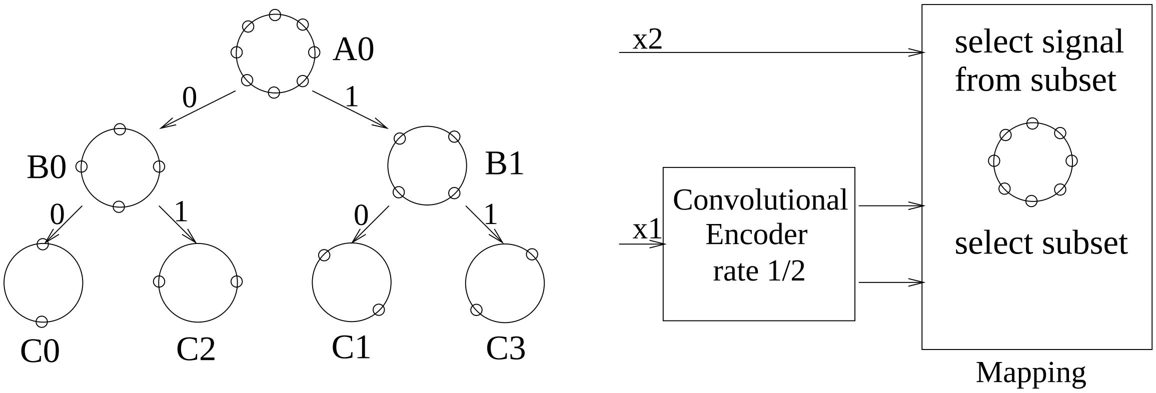


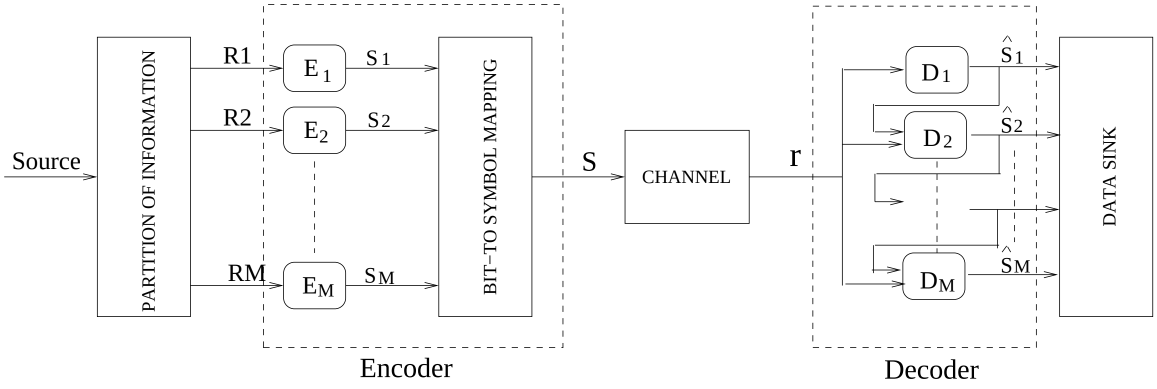
Figure 1.2: 8-PSK Set Partitioning and its TCM encoder

1.1 Background on Coded Modulation and Shaping for the AWGN Channel

The goal of the coded modulation is to jointly optimize coding and modulation in order to approach channel capacity, subject to power and bandwidth constraints. It allows the achievement of significant coding gains without compromising bandwidth efficiency. The most powerful applicable coded modulation schemes were presented in 1976/1977 by Ungerboeck [7], [8] and Imai and Hirakawa [9]. Ungerboeck's approach is named trellis-coded modulation (TCM). It is based on mapping by set partitioning to maximize the minimum intra-subset Euclidean distance, which minimize the symbol error rate. This scheme employs redundant non-binary modulation in combination with a convolutional code, which is a finite-state encoder represented by a trellis structure. This trellis controls the selection of modulation signals to generate coded signal sequences. In Fig. 1.2, a simple 8-PSK set partitioning and its TCM encoder are shown. One uncoded information bit is used to select the signal from a subset C_i, $0 \leq i \leq 3$, and two coded bits from a convolutional encoder are used to select the subset C_i. The decoder uses a soft-decision maximum-likelihood (ML) sequence algorithm, e.g. Viterbi algorithm (VA), with squared Euclidean distance (SED) metric. A simple four-state 8-PSK TCM can achieve 3 dB gain over uncoded 4-PSK modulation on AWGN channel. With more complex TCM, the coding gain can reach 6 dB or more.

In contrast, Imai's idea of multilevel coding (MLC) uses several error-correcting codes in parallel. Fig. 1.3 shows multi-level coding's encoder and decoder structure. The transmitted symbols are constructed by concatenating the bits from the encoders at each level into a binary codeword, which is then mapped to a constellation symbol. Usually, these codes are binary error-correcting codes and have different error-correcting capabilities. The decoder uses multi-stage decoding, i.e., each level's decoding is based on not only the channel output, but also the decoded results from the upper levels. At each level, either hard or soft decision decoding can be used. Soft decision will give the extra performance gain (typically 1-2 dB). This MLC scheme provides flexible transmission rates. Additionally, any code, e.g., block codes, convolutional codes, or concatenated codes, can be used as component codes in a MLC scheme.

The field of "shaping" has grown up out of the traditional field of constellation design. The shape of the signal constellation determines the average signal power. The power reduction due to constellation shaping is called shaping gain. Among all the multidimensional constellations, the spherical signal constellation is optimal and minimizes average



Ei : i^{th} encoder for i^{th} level constellation set partition
Di : i^{th} decoder corresponding to i^{th} encoder

Figure 1.3: Multi-level coding encoder and decoder

signal power, and in the limit as $N \rightarrow \infty$, the shaping gain of the N-sphere over the N-cube approaches $\pi e/6 \approx 1.53$ dB. Forney introduced shaping of multidimensional constellations including cross and generalized cross constellations, lattice-bounded constellations and trellis-bounded constellations in [31], [32] and [30]. Shaping codes based on non-equiprobable signaling are described in [29].

A shaping code is a probabilistic code for use with points in a modulation signal constellation. It saves power by using non-equiprobable signaling, without either reducing transmission rate or increasing bit error probability. A shaping encoder is a nonlinear encoder which can generate output sequences in a multidimensional space so that the coded sequences can induce a desired probability distribution in a low dimensional space which can achieve capacity. For the well-known coherent AWGN channel, the capacity-achieving signal is Gaussian and hence a good shaping code must induce a nonuniform, Gaussian-like distribution over a somewhat expanded constellation. In general, for fixed-rate shaping, a high-dimensional spherical constellation with a uniform distribution of signal points is used, which will induce the desired Gaussian distribution on its lower-dimensional constituent constellation. Most of the shaping gain on the coherent AWGN channel occurs at high rates. Hence large constellations are needed to realize significant shaping gain on the coherent AWGN channel at high transmission rates.

For the coherent AWGN channel, several methods to obtain shaping gain have been found. One is called shaping on regions [29]. In this approach, a low-dimensional signal constellation Ω is partitioned into nested annular sub-constellations $\Omega_0,\cdots,\Omega_{T-1}$ of equal size by scaling a basic region. Signal points in the same sub-constellation Ω_i are used equiprobably, and the shaping code selects sub-constellation Ω_i with probability P_i. To implement the shaping code, a simple table look-up is used to address points in the constellation. Trellis shaping, proposed by Forney [30] is a method of using the "Voronoi region" of a trellis code to shape the constellation. Trellis shaping selects a minimum-weight sequence from an equivalence class of possible transmitted sequences by a search through the trellis diagram of a shaping convolutional code. It is a dynamical programming that induces a truncated Gaussian distribution onto a constellation with a finite number of symbols. Motivated by the scalar-vector quantizer (SVQ), a shaping method called SVQ shaping is described in [33]. A class of constellations called SVQ constellations is defined and SVQ indexing algorithms are specified. The authors of [33] show that the N-sphere shaped constellation optimal for AWGN is a special case of a SVQ constellation, and so the indexing algorithms of the SVQ constellation can be used for N-sphere shaping.

The above-mentioned approaches require sufficiently many signals in the constellation to be selected. The SVQ shaping approach [33] requires that the number of signals to be shaped be larger than the corresponding coding rate. Forney's trellis shaping has the same requirement, while the regional shaping does not have such a requirement.

The shaping code we describe in Chapter IV is related to the shaping on regions approach of [29]. It uses a non-linear block code to select sequences with the desired probability. We don't use the shaping code on higher dimensions since the capacity achieving PDFs for the AWGN channel with Tikhonov phase error are not continuous in low dimension, and therefore are not induced by geometrically simple high-dimensional shape.

Recently, Huber in [25] explains the MLC scheme from the point of information theory and combines shaping and MLC. The techniques of [25] make digital communication close to the Shannon limit possible. In [25], the concept of equivalent channel is introduced, which is based on the well-known chain rule for mutual information, and also practical design rules for MLCs are formulated. The capacity rule, the balanced distance rule and two other rules based on the random coding exponent and cutoff rate are investigated. It is shown that if and only if the individual rates of the component codes are properly chosen, power and bandwidth-efficient digital communication close to the Shannon limits is possible. Also significant shaping gains are achievable in practice if the optimum assignment of code rates to the individual levels and optimum sharing of redundancy between coding and shaping is used. In [25], Forney's trellis shaping algorithm [30] is combined with an

MLC scheme, where the trellis shaping algorithm is used to generate a distribution of signal points approximating the theoretical Maxwell-Boltzmann distribution while preserving the optimum entropy. The codes in [25] achieve performance within 1 dB of the 2D AWGN capacity at 4 bits/symbol/Hz.

1.2 Channel Capacity of AWGN Channel with Tikhonov Phase Error

In [1, 2], the AWGN/Tikhonov channel capacity is studied under an average power constraint, and it is shown that the capacity-achieving input for this channel is discrete-amplitude-uniform-phase (DAUP), i.e., consists of concentric rings around the origin. An alternate proof of these results (without the lower bound computation) appears in [4, 5]. Assuming a finite number of rings, [1] uses a gradient approach to numerically compute the tightest known lower bound on the capacity of the AWGN/Tikhonov channel. The results show that the DAUP-input capacity has more than 0.8dB gain over the Gaussian-input capacity when the transmission rate is no more than 1 bits/symbol/Hz, the pilot-tone power fraction $\alpha = 0.01$, and the loop-bandwidth symbol-interval product $B_LT = 0.01$. This observation suggests the potential of good performing non-equiprobable signaling codes.

In Fig. 1.4, the capacity lower bound with a finite-number-of-rings DAUP input for the channel with $\alpha = 0.01$ and $B_LT = 0.01$ is numerically computed. At low SNR, this bound is significantly higher than the Gaussian-input bound [6]. Four-ring QPSK with the DAUP ring probabilities gives essentially the same results as DAUP. This motivates us to choose

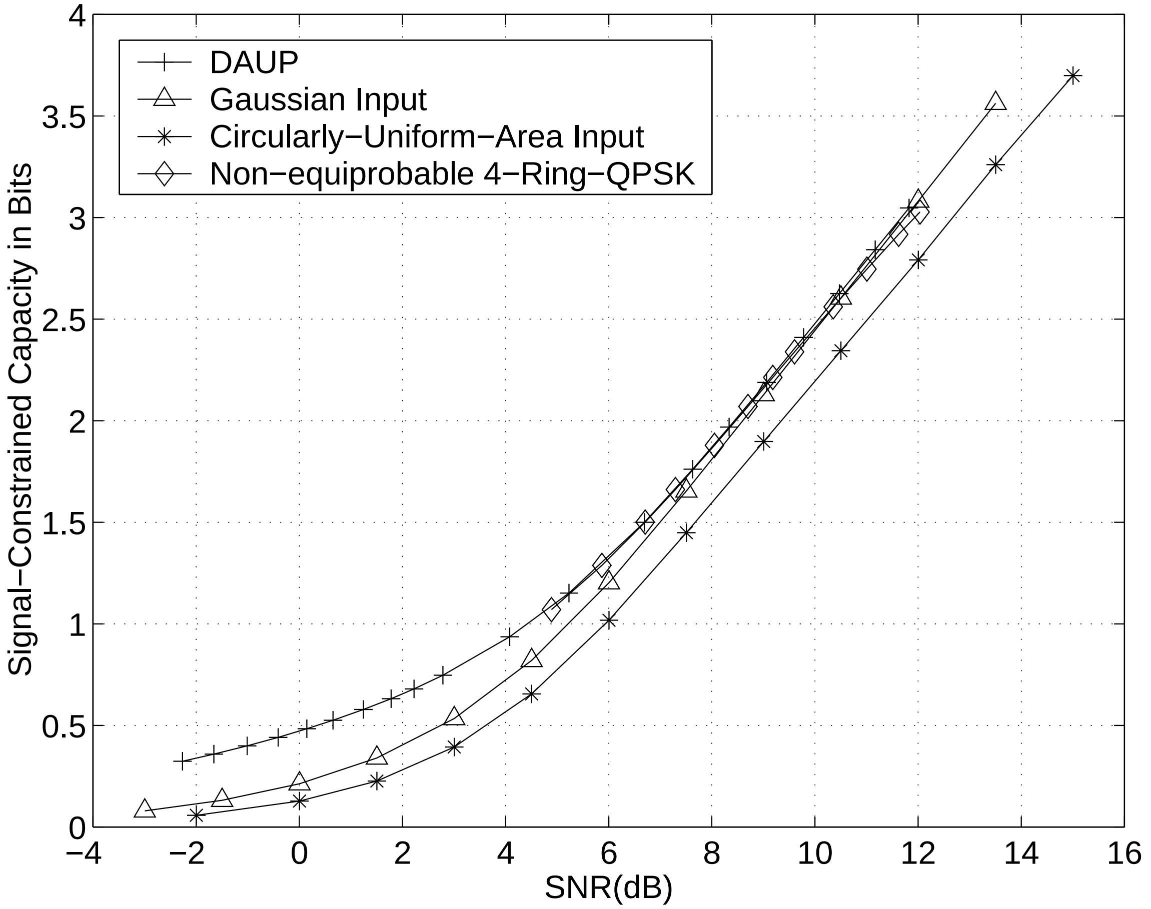


Figure 1.4: Mutual information of DAUP, Gaussian, uniform circular and non-equiprobable 4-ring QPSK inputs, for the AWGN/Tikhonov channel with $B_LT = 0.01$.

proper constellations to approximate the DAUP capacity-achieving inputs. The shaping gain, lower bounded by the gap between the DAUP and circularly-uniform-area in input, is $\geq$ 1.5 dB at 1bit/symbol and $\geq$ 3.5 dB at 0.5 bit/symbol.

The computation of signal-constrained capacities of non-equiprobable and equiprobable signaling for the channel with $\alpha = 0.01$ and $B_LT = 0.01$ motivates us to design codes with non-equiprobable signaling in the following chapters. Fig. 1.5 shows signal-constrained capacities of non-equiprobable and equiprobable signaling for the channel with $\alpha = 0.01$ and $B_L T = 0.01$. At rate 1.0697 bit/symbol/Hz, the four-ring BPSK with the DAUP ring probabilities and ring radii gives the result close to that of DAUP, with only 0.024 dB loss. In contrast, the mutual information of four-ring BPSK with the DAUP ring radii but equiprobable ring probabilities gives about 0.50 dB loss. At rate 0.5251 bit/symbol/Hz, 3-ring MPSK (using BPSK on the two inner rings and QPSK on the outer ring) with the DAUP ring probabilities and ring radii approximates the DAUP capacity with 0.02 dB loss. For the equiprobable case, the loss is about 1.91 dB.

1.3 Gaussian Approximation Technique for PCTCM

1.3.1 Turbo Codes and LDPC Codes

Iterative decoding of product or concatenated codes, using two or more “soft-in/soft-out” decoders with fairly simple component codes in an interleaved scheme, has been termed “turbo” decoding. Using log-likelihood algebra, any decoder can be used which accepts soft inputs, including a *priori* values, and delivers soft outputs. When the decoder input LLRs are subtracted from the soft output LLRs, the result is the extrinsic information LLRs. The extrinsic information is used as a *priori* information for the other decoder(s).

Parallel concatenated turbo coded modulation (PCTCM) encoder architectures can be classified into two main types. The bit-interleaved architecture of [39] uses a separate interleaver for each input bit shared between the two constituent encoders. The symbol-

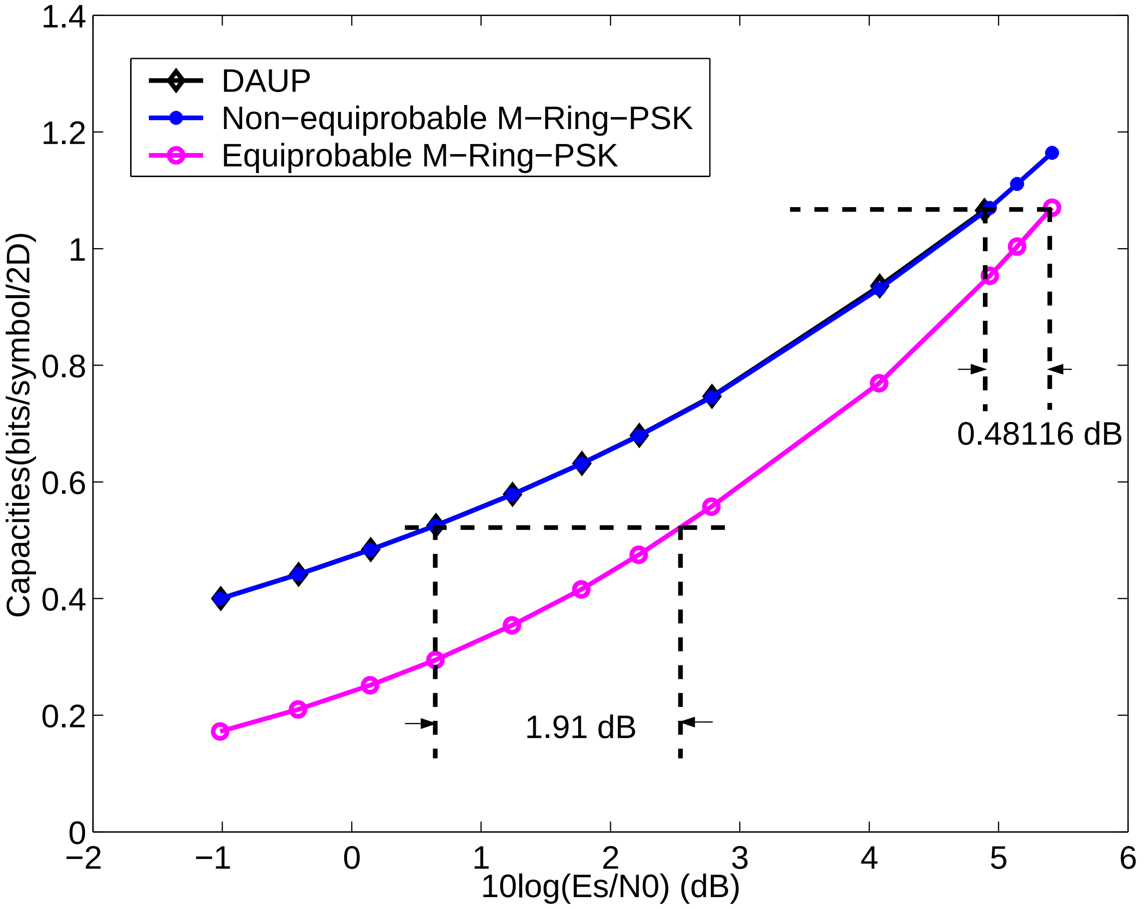


Figure 1.5: Mutual information of DAUP, M-Ring-PSK non-equiprobable and M-Ring-PSK equiprobable inputs, for the AWGN/Tikhonov channel with $B_L T = 0.01$.

interleaved approaches of [40] and [41] employ symbol interleavers, which permute the symbol sequence but preserve the mapping of input bits to symbols. The authors of [41] obtain about a 0.4 dB gain over [39] in the waterfall region of the bit-error-rate (BER) performance curve; however, they point out that the gain in performance at lower SNRs given by symbol-interleaving comes at the cost of a higher error floor. In Chapter III, we employ the the bit-interleaved architecture of [39], and propose a Gaussian approxima-

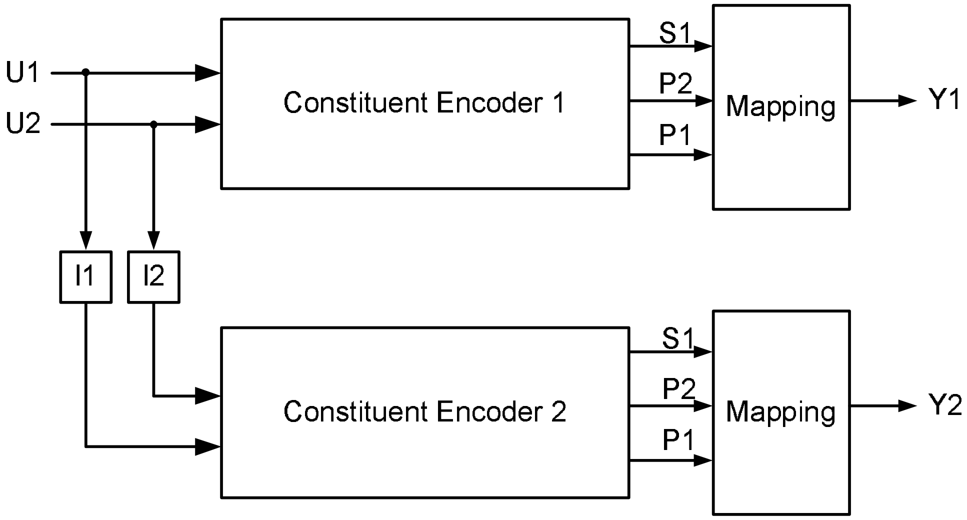


Figure 1.6: Rate 2/6 PCTCM encoder

tion technique to improve performance in the waterfall region. Fig. 1.6 is an example of rate $2/6$ bit-interleaved PCTCM encoder. Two constituent systematic encoders (usually we use systematic recursive convolutional codes) are concatenated in parallel, and separate interleavers I1 and I2 are used to avoid the situation that finite-weight inputs generate finite-weight outputs. The mapping block is a non-linear bit-to-symbol mapping. Iterative decoding follows the "turbo principle". As shown in Fig. 1.7, two MAP decoders are serially concatenated and exchange extrinsic information. The extrinsic information inputs of decoder 2 are interleaved versions of the decoder 1 outputs, and then decoder 2 outputs the deinterleaved extrinsic information to decoder 1. The interleaver and deinterleaver are denoted π and π^{-1}.

Recently, another class of codes exhibiting similar characteristics and performance was rediscovered. This class of codes, called low-density parity-check (LDPC) codes, was

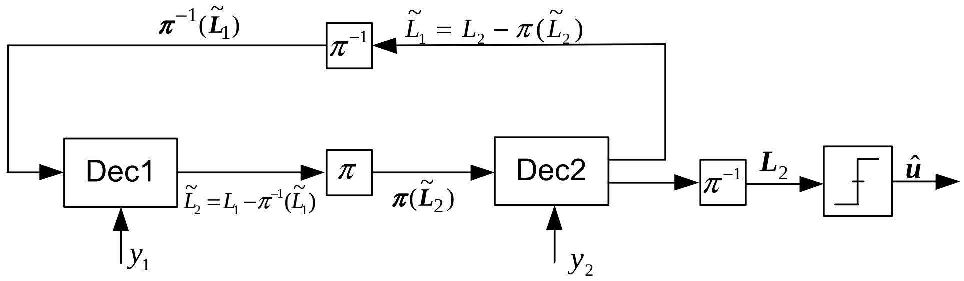


Figure 1.7: Turbo decoder

first introduced by Gallager in 1963 [36]. The decoding of LDPC uses a message-passing (belief-propagation) algorithm, where the message, i.e. extrinsic information, is transfered along the branches connecting the variable and check nodes forward and backward. This decoding scheme is parallelizable and has lower complexity than turbo decoding. Currently, interest has focused on computing the message's density function, and then using the density function to compute the convergence threshold, which indicates the code's performance compared with the Shannon limit. Codes can be optimized to have convergence thresholds within less than 0.1 dB of the Shannon limit, when the block size is large ($\sim 10^6$ bits). In [37], Richardson uses density evolution to design irregular LDPC codes under message-passing decoding. In [38], Chung approximates the message densities as Gaussian random variables and analyzes sum-product decoding of LDPC codes.

1.3.2 Gaussian Approximation for PCTCM

Gaussian approximation can also be used in turbo decoder analysis. In [23], it is demonstrated that the extrinsic information from constituent MAP decoders is well approximated by Gaussian random variables when the channel inputs to the decoders are Gaussian (from an AWGN channel). For the AWGN with Tikhonov phase error, even though the channel inputs are not Gaussian, we have observed experimentally that the extrinsic information can be well approximated as Gaussian [34]. In Chapter III (and [34]), the Gaussian approximation technique is used on bit-interleaved PCTCM to predict the convergence threshold. In the convergence threshold prediction, the extrinsic information SNR input/output relation is computed. The decoder used has a similar structure to the turbo decoder in Fig. 1.7 but without iterative operations. Fig. 1.8 is the decoder used for Gaussian approximation.

There are several problems that arise when applying the above described Gaussian approximation method to PCTCM. First, in PCTCM, two binary inputs are applied and the outputs from the linear encoders are independent. When these outputs are mapped to constellation symbols, the extrinsic information is not independent due to the nonlinearity of the mapping. By tracking the ratio of two decoder's SNRs and maintaining the correct ratio during the convergence threshold prediction, these problems are solved. The Gaussian approximation technique of Chapter III applicable to any channel where bits are correlated due to bit-to-symbol mapping.

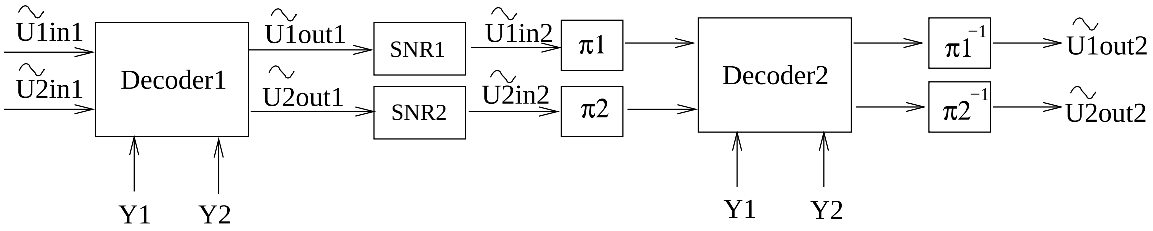


Figure 1.8: Decoder structure for Gaussian approximation

1.4 PEP Metric Computation

In a coded modulation scheme, let $\mathbf{X}_L$ and $\hat{\mathbf{X}}_L$ denote two complex symbol sequences of length L. The pairwise error probability (PEP) $P(\mathbf{X}_L \to \hat{\mathbf{X}}_L)$ is defined as the probability that $\hat{\mathbf{X}}_L$ is chosen by the receiver when $\mathbf{X}_L$ is actually sent, when the receiver chooses only between $\mathbf{X}_L$ and $\hat{\mathbf{X}}_L$. From [20], we know that the union bound on the symbol error probability can be expressed in terms of the PEP:

$$P(e) \leq \sum_{L=1}^{\infty} \sum_{\mathbf{X}_L} \sum_{\mathbf{X}_L \neq \hat{\mathbf{X}}_L} P(\mathbf{X}_L) P(\mathbf{X}_L \to \hat{\mathbf{X}}_L)$$

Appendix A describes the PEP computation for the AWGN/Tikhonov channel, when the squared Euclidean distance (SED) metric is used for symbol sequence comparisons..

The VA decoding simulation shows that the result based on ML metric outperforms the one based on the SED metric by 2 dB for the AWGN channel with moderate Tikhonov phase error, e.g. $B_L T = 0.01$. However due to the lower computational complexity of the SED decoding metric, the PEP based on the SED metric may be used as the code design criterion on such a channel. For the AWGN channel with large phase errors, the authors in [34] show that PEP based on optimal ML metric instead of SED is a good code design criterion which gives the turbo codes with low error floor and low convergence thresholds. The difference between our derivation of PEP and the description in [21] is as follows. Our results are valid for arbitrary constellations and give an exact expression when code length $N = 1$, and a central limit theorem (CLT) approximation for $N > 1$. In [21], the CLT approximation was used to derive the SED-based PEP for M-PSK constellations, with an upper bound given for the case $N = 1$.

1.5 Multi-level Coding for AWGN Channel with Tikhonov Phase Error

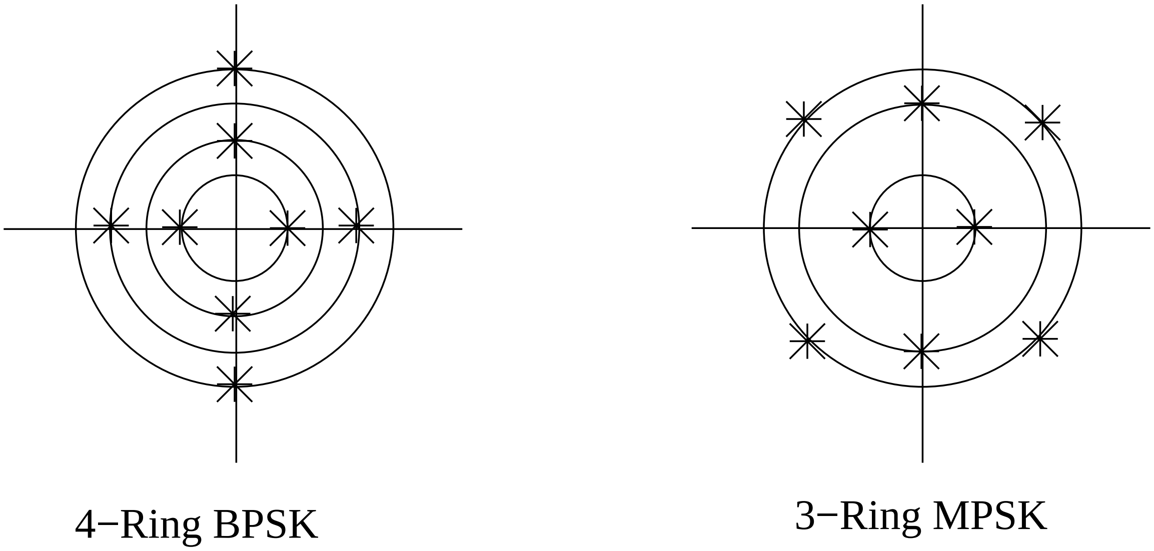


Figure 1.9: Four-ring BPSK and three-ring MPSK constellations

The main contribution of this dissertation is a MLC architecture for multi-ring PSK constellations with unequal ring probabilities. Fig. 1.9 shows examples of four-ring BSPK and three-ring MPSK. On the three-ring MPSK, BPSK is used on the two inner rings and QPSK on the outer ring. The ring radii and probabilities are equal to those of the channel capacity-achieving DAUP distributions. In practice, we use PSK modulations on each ring and the phase is not uniformly distributed. Numerical computations show that this

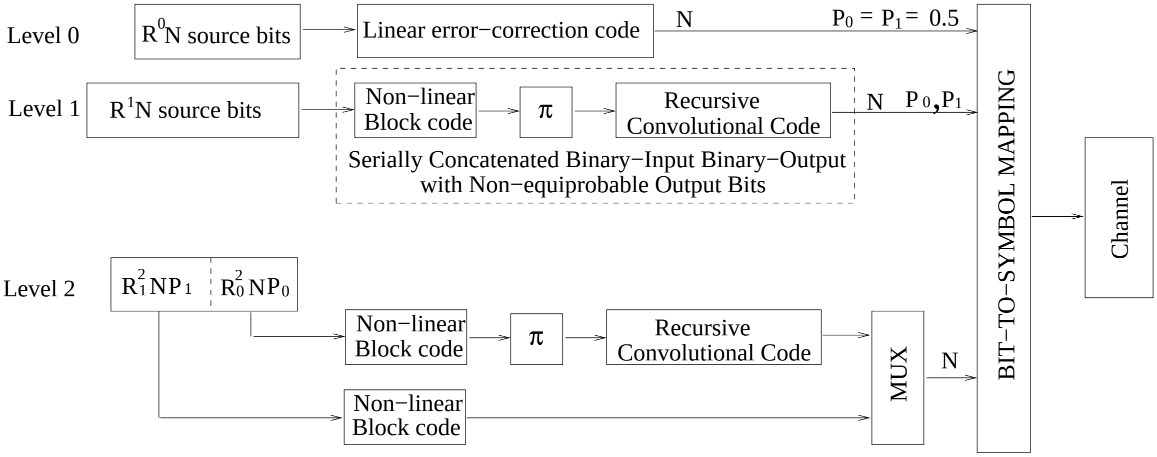


Figure 1.10: MLC encoder with multi-ring PSK constellation

constellation-constrained capacity is a good approximation of DAUP capacity.

In [35], the block non-coherent AWGN channel is considered, where the unknown phase is constant for a block N complex symbols and independent from block to block. The capacity-achieving PDF for this channel is also proved to be DAUP. The scheme called unitary modulation is used, which is different from our multi-ring PSK. The unitary modulation consists of one point located at zero and other points located on the outer ring based on a discrete Fourier transform. The shaping gain is achieved by increasing the frequency of the zero mass point transmission, i.e. augmenting the unitary modulations with the zero sequence. The coding scheme is the standard SCCC with convolutional codes as inner and outer codes.

Fig. 1.10 is our MLC encoder structure. The rate and desired binary bit probability of each level are derived from MLC's capacity design rule. Each decoder level corresponds to a level in the constellation set partitioning. The information blocks are partitioned according to the individual rate. Different codes are used at each level to achieve the desired probabilities. For the level 0, we use the irregular repeat-accumulate (IRA) codes of [26]. In our multi-level coding with multi-ring PSK modulation, a key innovation is serially concatenated binary-input binary-output codes with non-equiprobable output bits, used at level 1 and level 2. These codes consist of an outer non-linear block code and an inner recursive convolutional code. The codewords of the non-linear outer code consist of selected sequences from a convolutional code of known free distance (the “design code”), so that the outer code free distance is at least that of the design code. Both the design code and the inner convolutional code are designed according to the technique in [11, 12]. The encoded bits at each level are grouped column by column and mapped to the symbol on the multi-ring PSK. The decoder of our MLC is multi-stage decoding shown in Fig. 1.11. Multi-stage decoding proceeds from top to bottom; the decoded results of upper levels are used in decoding the lower levels. It is thus necessary to achieve very small bit error rates on the upper level codes to prevent error propagation. This fact motivates the use of high performance turbo decoding schemes at each level.

Although targeted at the AWGN/Tikhonov channel, our MLC architecture is easily adapted to other channels for which the capacity-achieving PDF is DAUP. The list of such channels includes, e.g., the peak-power-constrained AWGN channel [13], the memoryless Rayleigh fading channel [14], the non-coherent AWGN channel (a special case of the AWGN/Tikhonov channel) [15], and the non-coherent Ricean fading channel with average power and 4th moment constraints [16, 17].

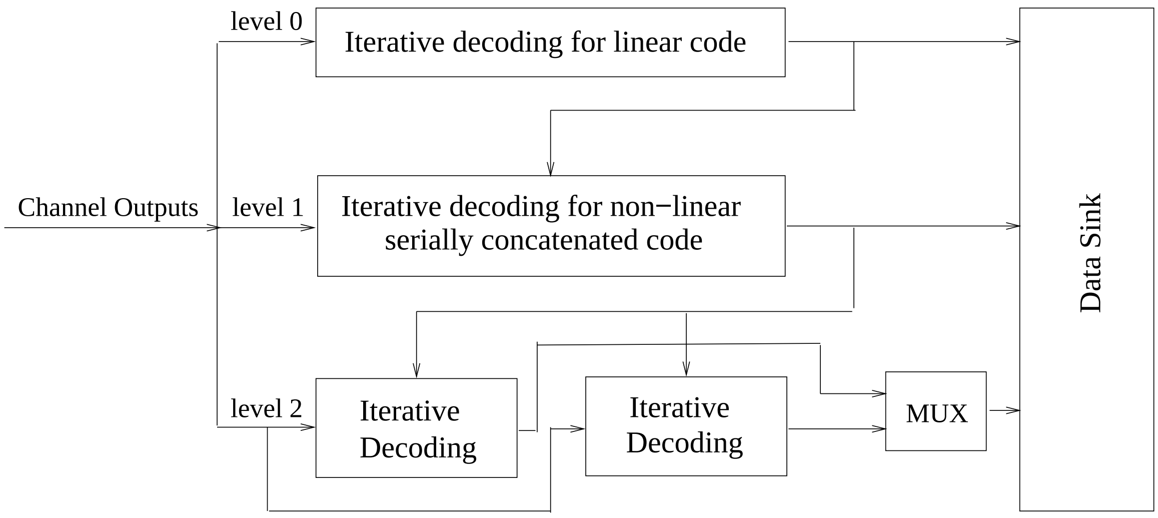

level 0
Iterative decoding for linear code
Channel Outputs
level 1
Iterative decoding for non−linear serially concatenated code
Data Sink
level 2
Iterative Decoding
Iterative Decoding
MUX

Figure 1.11: MLC multi-stage decoder

This dissertation is outlined as follows. A brief summary of the channel model and channel capacity is given in Chapter II. Also, different DAUP-approximating constellations with target radii and probabilities are listed. In Chapter III, the Gaussian approximation for parallel concatenated bit-interleaved turbo coded modulation is introduced, which predicts the convergence threshold and gives a criterion for code search. Chapter IV provides the details of multilevel coding for the AWGN/Tikhonov channel, including code design approach with capacity rule, combination of coding and shaping for multi-ring PSK, non-equiprobable serial concatenated codes, and design examples. The simulation results are given in Chapter V. Chapter VI concludes the dissertation. Appendix A describes the PEP computation using SED metric.

Chapter 2

AWGN with Tikhonov Phase Error Channel Model and Channel Capacity

In this chapter, we introduce the AWGN with Tikhonov phase error channel model, where a PLL is used for carrier phase tracking in a coherent receiver. We approximate DAUP signals with multi-ring PSK (MRPSK) constellations with the DAUP ring radii and probabilities. The constellation-constrained capacity of the MRPSK is computed and compared with the DAUP capacity lower bound.

2.1 Channel Model

In coherent receivers, the phase-locked loop (PLL) is commonly used to track carrier phase. Fig. 1.1 describes the receiver block diagram with PLL. It is assumed that the PLL input $c(t)$ is a pilot-tone with carrier phase $\theta_a(t)$ and constant amplitude $\sqrt{\alpha P_s}A(t)$, where P_s is the average symbol power and α is a positive real number ($0 < \alpha < 1$). It is assumed that the channel amplitude gain $A(t) = 1$. The pilot tone $c(t)$ is used by the PLL to estimate the carrier phase $\hat{\theta}_a$. Due to the loop noise the PLL outputs the recovered carrier phase but with the phase error $\phi = \theta_a - \hat{\theta}_a$, which can be described by a random variable. For a first-order PLL, the phase estimation error ϕ has the Tikhonov probability density function (PDF) [18]. The interleaver-deinterleaver in the PLL circuit makes the phase estimation errors in different symbol intervals independent, and also makes the noise independent. Hence the resulting channel is memoryless.

The discrete-time AWGN/Tikhonov channel model used in this dissertation is:

$$y_i = x_i e^{j\phi_i} + n_i \tag{2.1}$$

where $x_i = x_{i1} + jx_{i2}$ and $y_i = y_{i1} + jy_{i2}$ are the transmitted and received symbols in the ith symbol interval, ϕ_i has Tikhonov PDF

$$p(\phi_i) = \frac{\exp(\rho \cos \phi_i)}{2\pi I_0(\rho)}, \qquad -\pi < \phi_i \leq \pi \tag{2.2}$$

and n_i is a complex zero-mean Gaussian r.v. with independent real and imaginary parts, each having variance $\sigma_n^2 = N_0/2$. In (2.2), $I_0(\cdot)$ is the 0th order modified Bessel function of the first kind, and $\rho = \alpha E_s/(2N_0 B_L T)$, where B_L is the one-sided PLL bandwidth, T is the symbol interval, E_s is the average symbol energy, and α is the power fraction alloted to the carrier pilot reference; in this dissertation, we always use $\alpha = 0.01$. The ϕ_i and n_i are assumed independent from one symbol interval to the next. The model assumes that the carrier pilot tone is not interleaved, so the PLL tracks the time-varying channel phase with an effective observation interval of $1/(B_LT)$ symbols. There is a trade-off in choosing the value of B_LT. Reducing B_LT reduces the effect of noise on the PLL tracking; however, $1/(B_LT)$ should be less than or equal to the coherence time of the channel phase process $\theta_a(t)$, so that the PLL responds quickly enough to track $\theta_a(t)$. Fig. 2.1 is the Tikhonov PDF of the phase estimation error. Different PLL SNRs ρ give different PDF shapes. The smaller ρ, the wider the PDF. When ρ goes to 0, the PDF is uniformly distributed, which means the PLL does not perform well. The larger power fraction α, the larger ρ, hence, the better PLL performance. However, a larger α also decreases the transmission capacity. So, there is a trade-off.

When the phase error ϕ_i is considered to be modulo 2π, the above Tikhonov model accounts for cycle slips in the first-order PLL [18]. For higher-order PLLs, the Tikhonov model is only accurate at loop SNR's high enough to make cycle slips rare. The effect of cycle slips in higher-order PLL's is not considered in this dissertation. The Tikhonov model has been widely used in past publications on phase robust modulation (e.g., [18, 19, 20, 21]). The Tikhonov model also applies to pilot-tone aided systems with open-loop phase tracking, as the Tikhonov PDF closely approximates the phase error PDF of simple one-shot estimators. (See e.g. [22, pp. 266-269].)

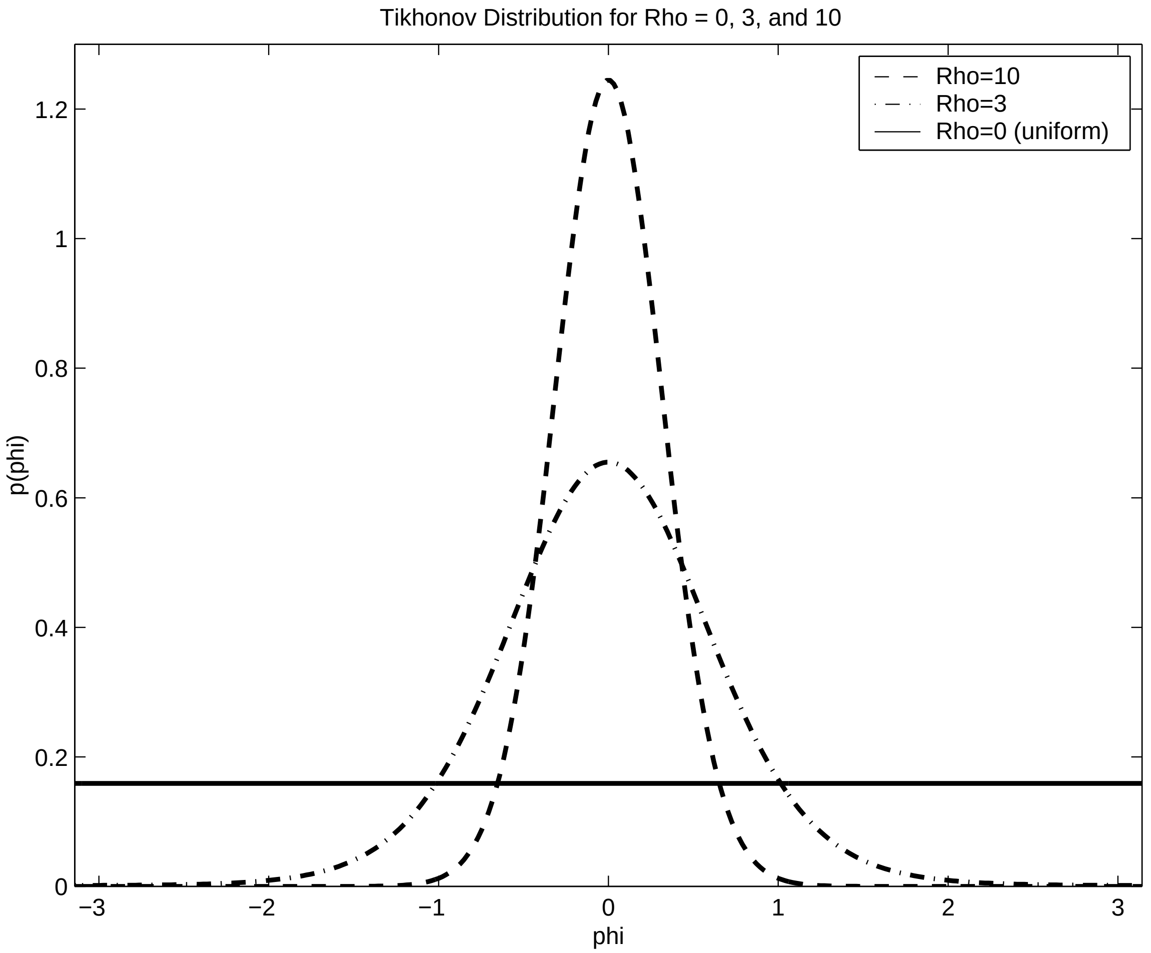


Figure 2.1: Tikhonov PDF for Various Loop SNRs

The conditional channel PDF used in this dissertation is [19]

$$
\begin{aligned}
&p(y_{i1}, y_{i2} | x_{i1}, x_{i2}) = \\
&\qquad \frac{1}{2\pi\sigma_n^2} \exp\left[-\frac{y_{i1}^2 + y_{i2}^2}{2\sigma_n^2}\right] \exp\left[-\frac{x_{i1}^2 + x_{i2}^2}{2\sigma_n^2}\right] \frac{I_0(\nu_{s_i})}{I_0(\rho)}
\end{aligned}
\tag{2.3}
$$

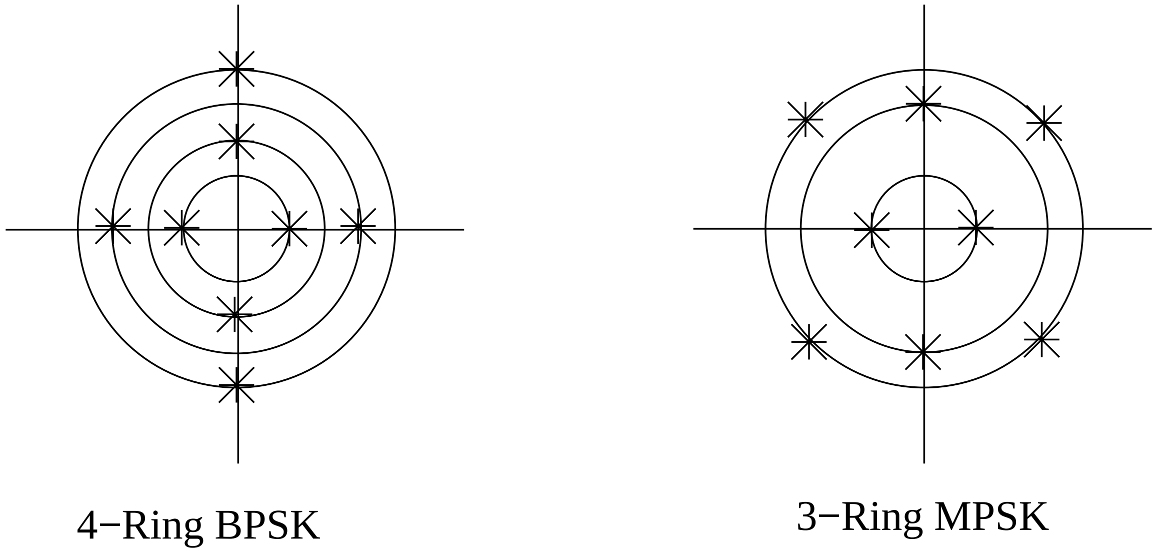


Figure 2.2: Four-ring BPSK and Three-ring MPSK constellations with DAUP ring probabilities and ring radii; the ring radii are not drawn to scale

where

$$\nu_{s_i} = \sqrt{\left[\frac{x_{i1}y_{i1}+x_{i2}y_{i2}}{\sigma_n^2}+\rho\right]^2+\left[\frac{x_{i1}y_{i2}-x_{i2}y_{i1}}{\sigma_n^2}\right]^2}. \tag{2.4}$$

2.2 Channel Capacity and DAUP-Approximated Constellation-Constrained Capacity

We use the four-ring BPSK constellation shown in Fig. 2.2 to approximate the DAUP capacity at rate 1.0697 bit/symbol with $B_LT = 0.01$. As shown in Fig. 1.5, the four-ring BPSK with the DAUP ring probabilities and ring radii has mutual information close to that of DAUP, with only 0.024 dB loss. The DAUP ring probabilities and ring radii are listed in Table 2.1 [1].

Table 2.1: DAUP ring radii and probabilities at rate 1.0697 bit/symbol with $B_L T = 0.01$

ring number	1	2	3	4
radii	0.5	3.2148	5.4537	8.0099
probability	0.60636	0.30856	0.07662	0.00846

Table 2.2: DAUP ring radii and probabilities at rate 0.5251 bit/symbol with $B_L T = 0.01$

ring number	1	2	3
radii	0.1	3.294111	5.814085
probability	0.809123	0.179967	0.010910

At rate 0.5251 bit/symbol, we use 3-ring MPSK in Fig. 2.2 to approximate the DAUP capacity, where we use BPSK on the two inner rings and QPSK on the outer ring. There is only 0.02 dB loss. The DAUP ring probabilities and ring radii are listed in Table 2.2.

Chapter 3

Gaussian Approximation for Parallel Concatenated Bit-Interleaved Turbo Coded Modulation

3.1 Gaussian Assumption for the Extrinsic Information on the AWGN Channel with Tikhonov Phase Error

The extrinsic information from constituent MAP decoders is well approximated by Gaussian random variables when the decoder channel inputs to the decoders are Gaussian [23]. Even though the AWGN/Tikhonov channel variables are not Gaussian, we have observed experimentally that the extrinsic information can be well approximated as Gaussian. After several iterations, the PDF of the extrinsic information will converge to a Gaussian PDF. In Fig. 3.1, the left plot is the histogram of the extrinsic information when $B_LT = 0.01$ after 8 iterations, and its PDF fit with the same mean and variance as the histogram. The Gaussian PDF fits the histogram very well. To verify this fit, we plot the cumulative distribution functions (CDFs) of the extrinsic information and of the Gaussian r.v with the same

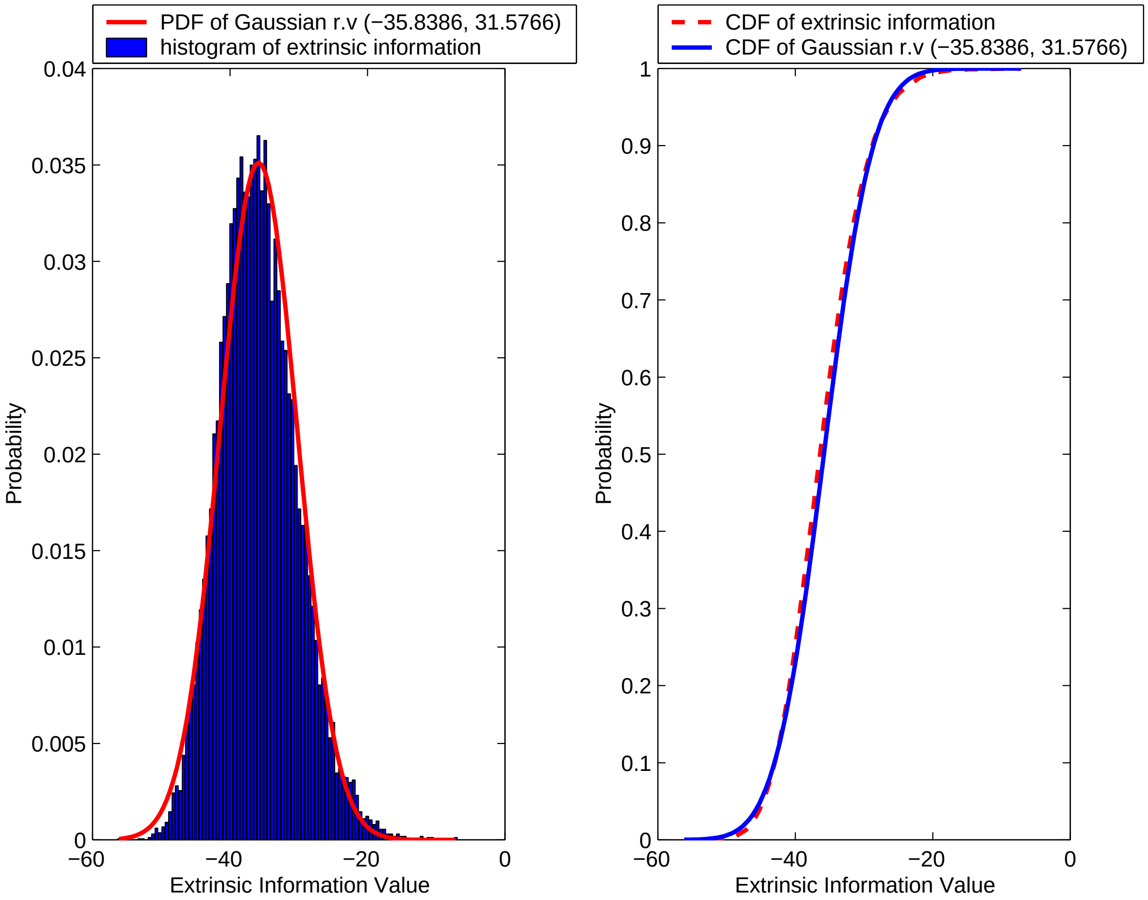


Figure 3.1: Histogram, PDF and CDF of extrinsic information with $B_L T = 0.01$; the numbers in parentheses indicate mean and variance

mean and variance respectively in the right plot in Fig. 3.1. It shows that the CDF of the extrinsic information tracks the Gaussian r.v very well. Fig. 3.2 verifies Gaussian assumption for extrinsic information when $B_L T = 0.1$. It turns out that the Gaussian assumption is reasonable even on the channel with severe phase estimation errors.

Based on these observations, the decoder convergence threshold can be predicted using the Gaussian approximation. The Gaussian approximation for turbo decoding was first described in [23]. The convergence threshold is the minimum E_b/N_0 under which the itera-

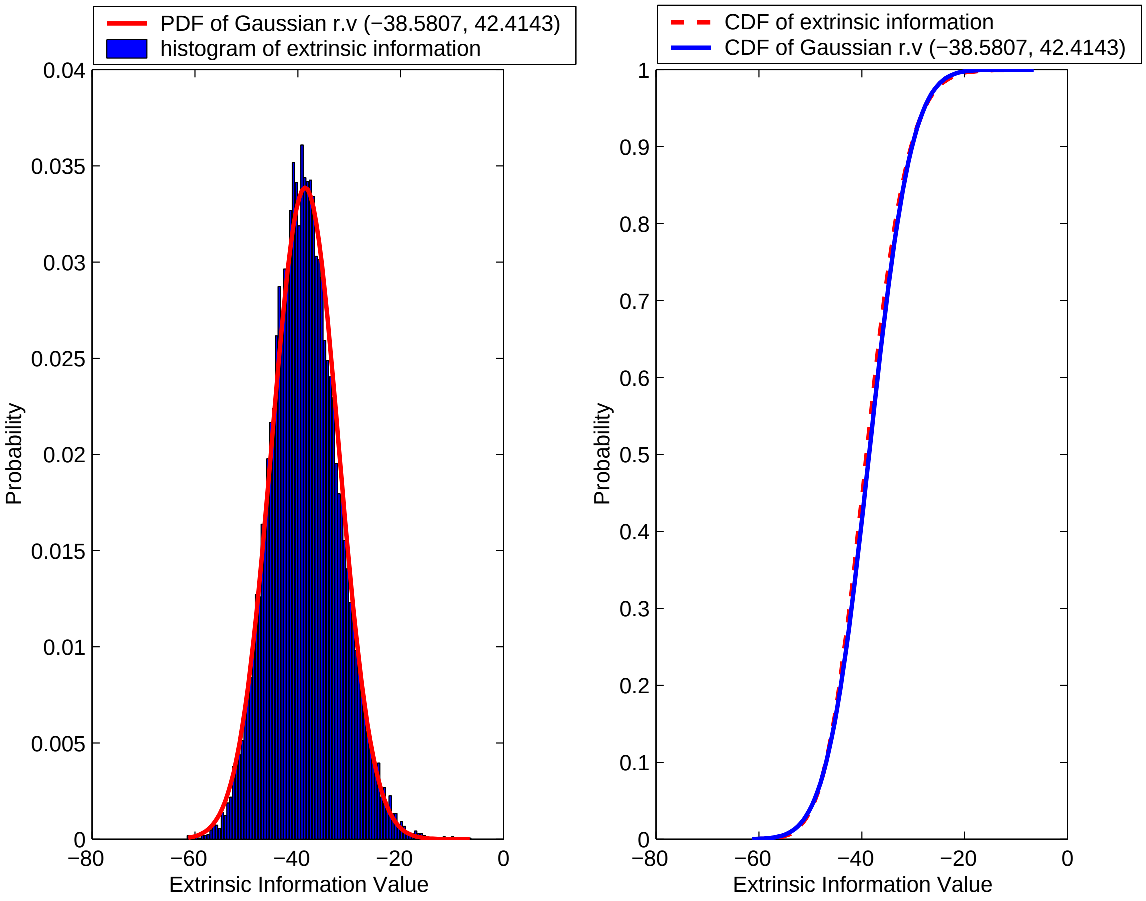


Figure 3.2: Histogram, PDF and CDF extrinsic information with $B_LT = 0.1$

tive decoder will converge with zero error probability. We focus on a single parameter: the extrinsic information SNR which is computed from $P_b = Q(\sqrt{\mathrm{SNR}})$, where P_b is the error probability of the decoder extrinsic information outputs. By feeding independent Gaussian extrinsic information with different SNRs to the decoders, we can get the extrinsic information SNR input/output curve. The convergence threshold is the minimum E_b/N_0 under which there is no intersection between the SNR input/output curve and the line $\mathrm{SNR}_{\mathrm{in}} = \mathrm{SNR}_{\mathrm{out}}$. This method of predicting the convergence threshold is based on the assumption that the intrinsic and extrinsic information are jointly Gaussian and statistically independent. When the turbo decoders are symmetric, the extrinsic information from only one decoder is needed.

3.2 Gaussian Approximation for PCTCM

A problem arises when applying the above Gaussian approximation method to PCTCM. Consider, e.g., a bit-interleaved two-encoder PCTCM in Fig. 3.3 where two binary inputs u_1 and u_2 are applied to each encoder, and the encoder output bits are mapped into constellation symbols. (Although the following discussion assumes two input bits, generalization to more than two input bits is straightforward.) Due to the nonlinear mapping of encoder output bits to constellation symbols, the associated decoder extrinsic information r.v.s $\tilde{u}_1$ and $\tilde{u}_2$ are correlated, so that, at each decoder iteration, a specific ratio exists between their SNRs SNR_1 and SNR_2. We denote SNR_1 as the SNR of $\tilde{u}_1$ and SNR_2 as that of $\tilde{u}_2$. We should maintain the correct ratio $\frac{\mathrm{SNR}_1}{\mathrm{SNR}_2}$ during the convergence threshold prediction. The structure of the decoders is given in Fig. 3.4. Since u_1 is the systematic output of decoder 1, initially $\mathrm{SNR}_1 \gg \mathrm{SNR}_2$ at the output of decoder 1; for decoder 2, $\mathrm{SNR}_2 \gg \mathrm{SNR}_1$ for the same reason. After many iterations, the ratio $\frac{\mathrm{SNR}_1}{\mathrm{SNR}_2} \to 1$. However, assuming that $\frac{\mathrm{SNR}_1}{\mathrm{SNR}_2}$ = 1 at the first iteration leads to an incorrect prediction of the convergence threshold.

In our approach to predicting the convergence threshold, by concatenating the two decoders as in Fig. 3.4, we get the extrinsic information SNR input/output relations for $\tilde{u}_1$ and

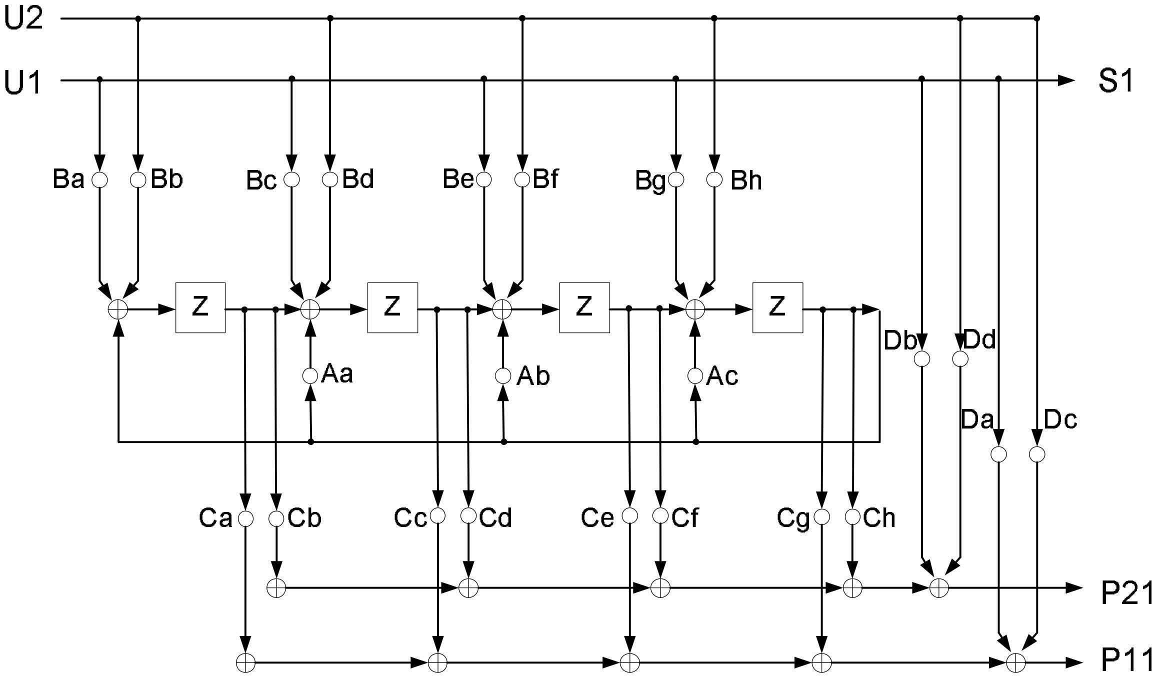


Figure 3.3: Generalized form of constituent encoder

$\tilde{u}_2$ respectively. At each E_b/N_0, the initial $\tilde{u}_{1in1}$ and $\tilde{u}_{2in1}$ are set to zero. After decoder 1, we measure the SNRs of $\tilde{u}_{1out1}$ and $\tilde{u}_{2out1}$ from the decoder 1 extrinsic bit error probabilities P_{1bk} as $\mathrm{SNR}(\tilde{u}_{\mathit{kout1}}) = (Q^{-1}(P_{\mathit{1bk}}))^2$, for $k = 1, 2$. These extrinsic information variables are approximated by independent Gaussian random variables with the correct SNR ratio $\frac{\mathrm{SNR}(\tilde{u}_{\mathit{1out1}})}{\mathrm{SNR}(\tilde{u}_{\mathit{2out1}})}$ and injected into decoder 2. After one iteration, the SNRs of $\tilde{u}_{kout2}$ are computed from decoder 2 extrinsic error probabilities P_{2bk} as $\mathrm{SNR}(\tilde{u}_{\mathit{kout2}}) = (Q^{-1}(P_{\mathit{2bk}}))^2$.

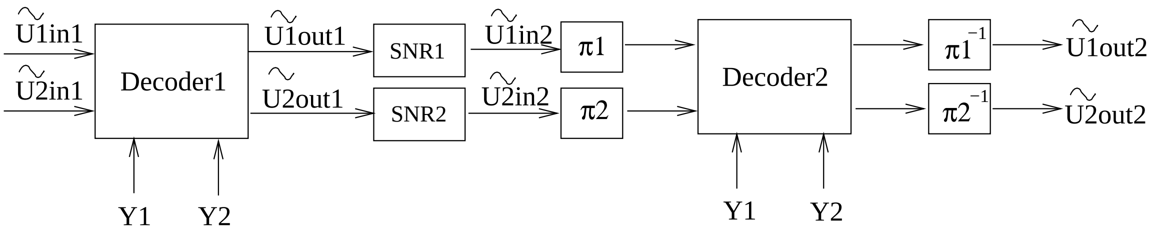


Figure 3.4: Decoder structure for Gaussian approximation

Then we use the ratio $\frac{\mathrm{SNR}(\tilde{u}_{1out2})}{\mathrm{SNR}(\tilde{u}_{2out2})}$ as the ratio $\frac{\mathrm{SNR}(\tilde{u}_{1in1})}{\mathrm{SNR}(\tilde{u}_{2in1})}$ for the next point on the SNR input/output curve. The $\tilde{u}_{1in1}$ and $\tilde{u}_{2in1}$ are independent Gaussian random variables with mean $=$ sign$(u_k) \times 2\mathrm{SNR}(\tilde{u}_{kin})$ and variance $= 4\mathrm{SNR}(\tilde{u}_{kin})$, where $k = 1, 2$. Under such a scheme, the correct ratio between extrinsic information SNRs for $\tilde{u}_1$ and $\tilde{u}_2$ is maintained, so that the correct input/output relation of the extrinsic information is obtained. The convergence threshold is the minimum E_b/N_0 under which neither the SNR input/output curve for $\tilde{u}_1$ nor that for $\tilde{u}_2$ intersects the line $\mathrm{SNR}_{\mathrm{in}}$ = $\mathrm{SNR}_{\mathrm{out}}$.

3.3 Simulation and Prediction Results

The above method to predict the convergence threshold is much faster than an actual turbo decoding simulation. In an actual simulation much time is needed to achieve an accurate estimate of the lowest E_b/N_0 at which P_b falls below 10^{-5}. However, the prediction of convergence threshold using the Gaussian approximation takes only one iteration for each injection of the extrinsic information SNR. At each computation of the output extrinsic information SNR, the intersection condition is checked. If intersection occurs, the computation is stopped and E_b/N_0 is increased. The computation stops at the first E_b/N_0 for which intersection does not occur; this is the predicted convergence threshold.

Table 3.1 compares the ratio $\frac{\mathrm{SNR}_1}{\mathrm{SNR}_2}$ computed by Gaussian approximation and by actual measurement in a turbo decoding simulation.

An example extrinsic information SNR input/output curve is shown in Fig. 3.5. From

Table 3.1: Ratio between the extrinsic information SNR for $\tilde{u}_1$ and $\tilde{u}_2$ (PSK natural mapping with $B_LT = 0.1$)($E_b/N_0 = 17.43$dB)

Iteration	Gaussian Approx.	Actual measurement
1	7.7444	7.9123
2	4.4057	4.4285
3	3.0257	3.0015
4	2.2799	2.2358
5	1.6959	1.6171
6	1.4057	1.3359
7	1.2356	1.2106

Table 3.2: Predicted E_b/N_0 convergence threshold and threshold measured by simulation (QAM radial mapping with $B_LT = 0.1$)

Connection Matrix				prediction	simulation
B	A	C	D	(dB)	(dB)
19	1	33	13	12.14	12.375
38	1	71	1	12.18	12.395
23	4	120	1	12.20	12.405
109	4	150	9	12.24	12.435
219	4	58	9	12.28	12.445

the figure, it is clear that the iterative decoder will converge to $P_b = 0$ for $E_b/N_0 =$ 17.16dB, but for $E_b/N_0 = 17.12$dB the probability of error will be bounded away from zero even if the number of iterations is unbounded. Therefore, the predicted convergence threshold is about 17.16 dB.

The predicted convergence threshold and the actual simulation results are listed in Table 3.2, for the generalized encoder structure of Fig. 3.3. The simulation convergence threshold is the minimum E_b/N_0 at which $P_b \leq 10^{-5}$. The simulation used two 16384 bit S-random interleavers [42], one for each input bit. From the table, it is seen that the pre-

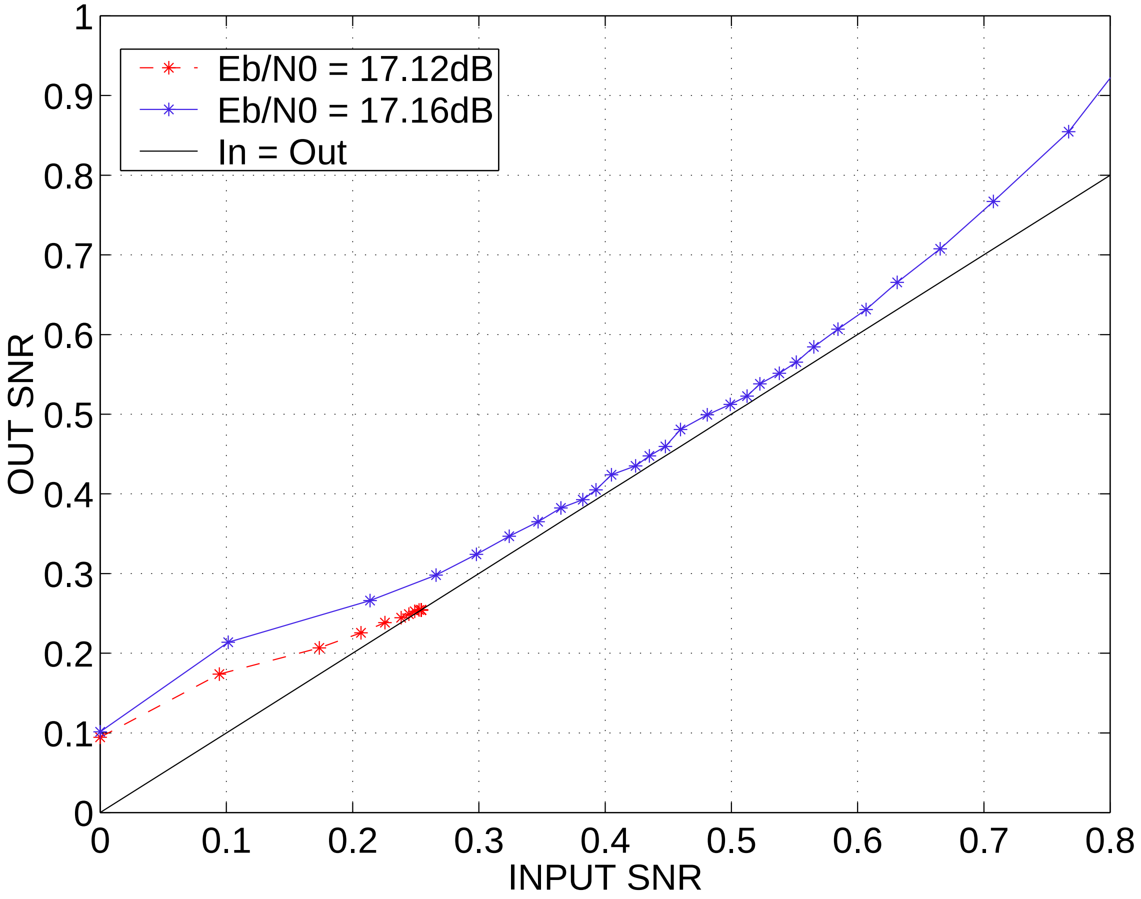


Figure 3.5: Extrinsic information SNR input/output curve (PSK natural mapping with $B_L T = 0.1$)

dicted results track the simulation results reasonably well. The simulated results are about 0.2 dB higher than the predicted threshold, due to use of finite-length interleavers in the simulation. The Gaussian approximation technique described in this chapter is useful for any turbo decoding scheme where the MAP decoders have multiple extrinsic inputs which can be modeled as correlated Gaussian r.v.'s.

Chapter 4

Multi-Level Coding for the AWGN Channel with Tikhonov Phase Error

4.1 Introduction

Since DAUP signals achieve capacity on the AWGN/Tikhonov channel, our approach is to try to approximate this optimal DAUP signal by coding on M-PSK constellations with the DAUP probabilities and radii. Huber showed in [25] that for practical coded modulation schemes, the capacity of the modulation scheme can be achieved by multi-level codes together with multistage decoding if and only if the individual rates of the component codes are properly chosen. Moreover, this multi-level coding technique can be used for arbitrary signaling and labeling of signal points. The combination of coding and shaping is possible. The key point is the well-known chain rule for mutual information. The chain rule provides a model with virtually independent parallel channels for each address bit at the different partitioning levels, called equivalent channels. Among the practical rules for designing and constructing coded modulation schemes, the capacity rule is the most powerful one, and, in theory, makes achieving the Shannon limit possible. And it provides the flexibility of selecting the rate on each equivalent channel. For our non-equiprobable DAUP signals and their approximations by M-PSK constellations, this multi-level technique offers a design framework within which we can search for and find good coded modulation schemes.

4.2 Code Design Approach with Capacity Rule

4.2.1 Set Partitioning

At rate 1.0697 bit/symbol, the partitioning of four-ring BPSK is shown in Fig. 4.1. This partitioning is designed to maximize the squared Euclidean distance (SED) within each subset. Results in [24] show that maximizing SED between coded symbol sequences is a good design strategy for the $B_LT = 0.01$ AWGN/Tikhonov channel. We can also show that this partitioning is consistent with PEP results based on the ML metric. The ML metric between transmitted complex point X and received point Y on the AWGN/Tikhonov channel is:

$$m_{x_i}(x_{i1}, y_{i2}|x_{i1}, x_{i2}) = \frac{|x_i|^2}{2\sigma_n^2} - \log I_0(\nu_{s_i}). \tag{4.1}$$

Equation (4.1) follows by taking the log($\cdot$) of the conditional channel PDF (2.3). For the 4-ring BPSK in Fig. 4.2, we list the PEP based on the ML metric in matrix A. Element $A(i, j)$ is the probability that the j^{th} point is decoded given that the i^{th} point was transmitted, when

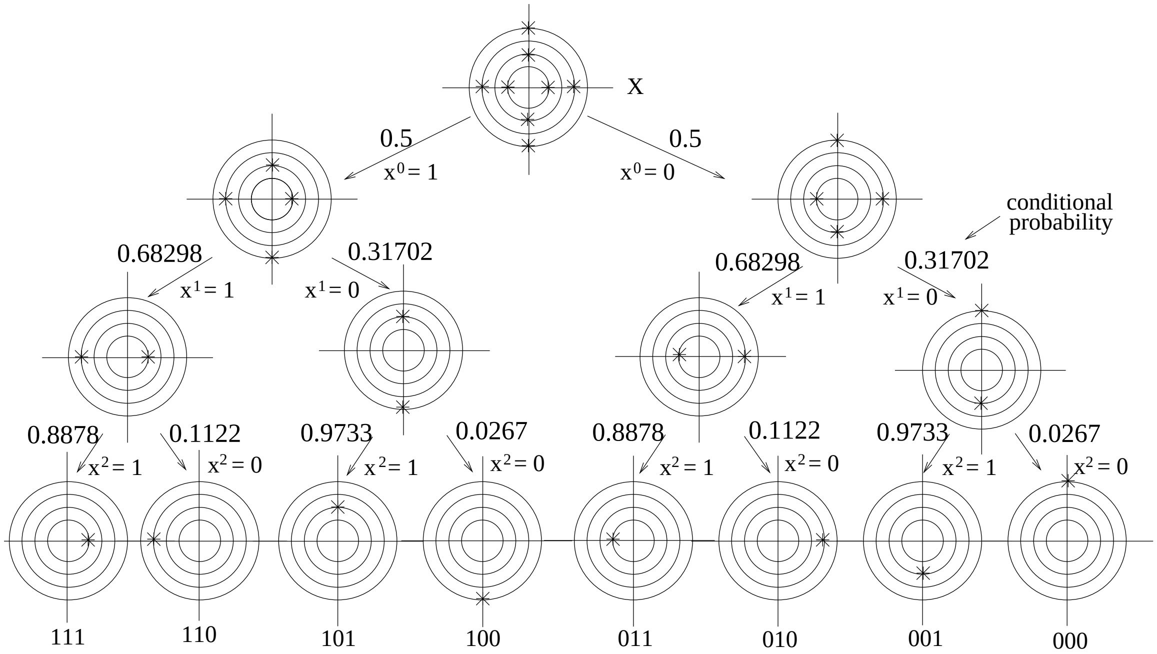


Figure 4.1: Partitioning of 4-Ring BPSK with DAUP ring probabilities for the AWGN/Tikhonov channel with $B_L T = 0.01$

the receiver chooses only between i and j. When $i = j$, PEP is always 1, since the receiver has only one choice. We will exclude the PEP in the diagonal of matrix A. With the result of matrix A, we compare the PEP of two set partitionings listed in Table 4.1 and choose the one with the minimum max PEP within the subsets, which is the one we use in the Fig. 4.1. The max PEP is computed by picking the maximum value from $A(i, j) * p(i)$ for all i and j in the subset, where $p(i)$ is the probability of i^{th} point.

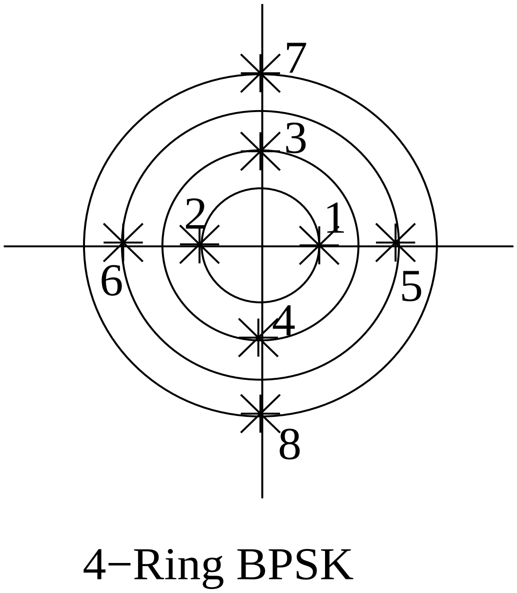


Figure 4.2: 4-Ring BPSK with DAUP ring probabilities for the AWGN/Tikhonov channel with $B_L T = 0.01$

$$\mathbf{A} = \underset{\text{PEP matrix}}{\begin{bmatrix} 1 & 0.28254 & 0.59084 & 0.59084 & 0.00028 & 0.00008 & 0.00000 & 0.00000 \\ 0.28254 & 1 & 0.59084 & 0.59084 & 0.00008 & 0.00028 & 0.00000 & 0.00000 \\ 0.02251 & 0.02251 & 1 & 0.02610 & 0.02442 & 0.02442 & 0.00034 & 0.00006 \\ 0.02251 & 0.02251 & 0.02610 & 1 & 0.02442 & 0.02442 & 0.00006 & 0.00034 \\ 0.00030 & 0.00009 & 0.02454 & 0.02454 & 1 & 0.02249 & 0.01499 & 0.01499 \\ 0.00009 & 0.00030 & 0.02454 & 0.02454 & 0.02249 & 1 & 0.01499 & 0.01499 \\ 0.00000 & 0.00000 & 0.00035 & 0.00006 & 0.01502 & 0.01502 & 1 & 0.02160 \\ 0.00000 & 0.00000 & 0.00006 & 0.00035 & 0.01502 & 0.01502 & 0.02160 & 1 \end{bmatrix}}.$$

At rate 0.5251 bit/symbol, the partitioning of 3-ring MPSK is shown in Fig. 4.3, which maximizes the SED within each subset.

Table 4.1: PEP within subsets comparison on different set partitionings

subsets	max PEP
(1 3 6 8) (2 4 5 7)	0.1791
(1 6) (3 8)	0.000024254
(2 5) (4 7)	0.000024254
(1 3 5 7) (2 4 6 8)	0.1791
(1 5) (3 7)	0.000084890
(2 6) (4 8)	0.000084890

4.2.2 Capacity Computation and Rate Distribution

Using the approach of [25], from the chain rule the mutual information of each equivalent channel i in a l level multi-level code (MLC) can be easily calculated by:

$$\begin{aligned} I(Y;X^i|X^0\cdots X^{i-1}) &= I(Y;X^i\cdots X^{l-1}|X^0\cdots X^{i-1}) \\ &\quad -I(Y;X^{i+1}\cdots X^{l-1}|X^0\cdots X^i). \end{aligned} \tag{4.2}$$

For levels 0, 1 and 2 in Fig.4.1, the mutual informations are:

$$I(Y;X^0) = I(Y;X^0X^1X^2) - I(Y;X^1X^2|X^0), \tag{4.3}$$

$$\begin{aligned} I(Y;X^1|X^0) &= I(Y;X^1X^2|X^0) - 0.68298 I(Y;X^2|X^0,X^1=1) \\ &- 0.31702 I(Y;X2|X^0,X^1=0), \end{aligned} \tag{4.4}$$

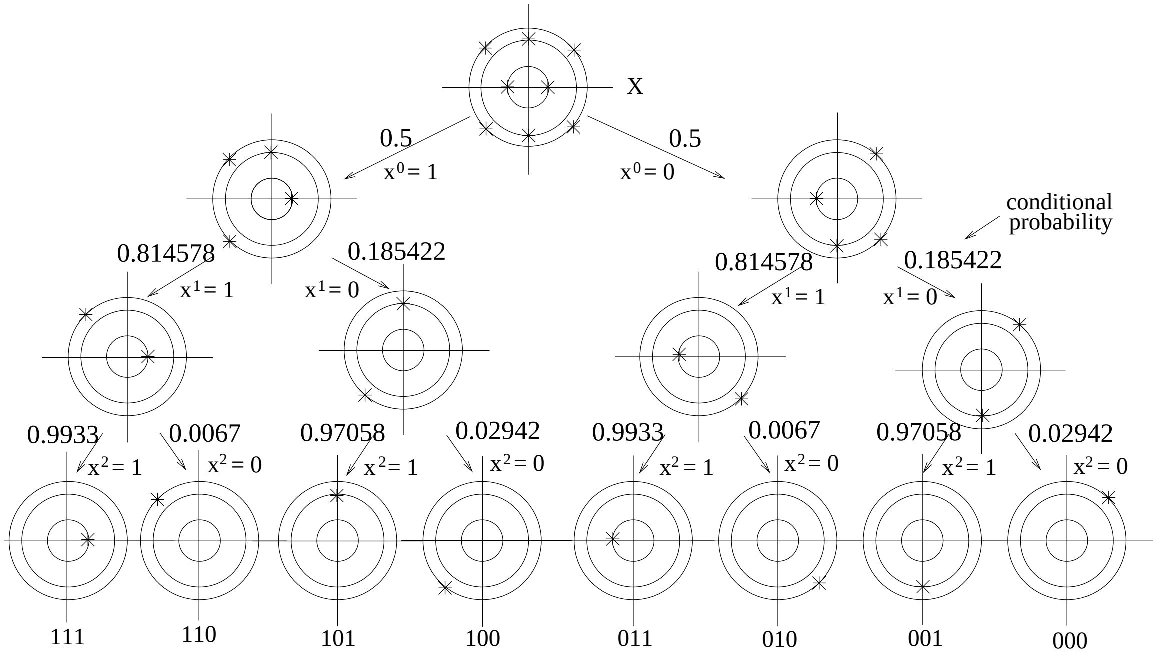


Figure 4.3: Partitioning of 3-Ring MPSK with DAUP ring probabilities for the AWGN/Tikhonov channel with $B_L T = 0.01$

$$I(Y;X^2|X^0X^1) = 0.68298I(Y;X^2|X^0,X^1=1)+0.31702I(Y;X^2|X^0,X^1=0). \quad (4.5)$$

Fig. 4.4 shows the mutual information of the equivalent channels. The rate of each level is designed by the capacity rule, i.e. the rate R^i at the individual coding level i of a multi-level coding scheme should be chosen equal to the capacity C^i of the equivalent channel i. So for 1.0697 bits/symbol, we get:

$$R^0/R^1/R^2 = 0.1613/0.5145/0.3939. \quad (4.6)$$

At 0.5251 bit/symbol, according to the partitioning in Fig. 4.3 and the capacity rule, we

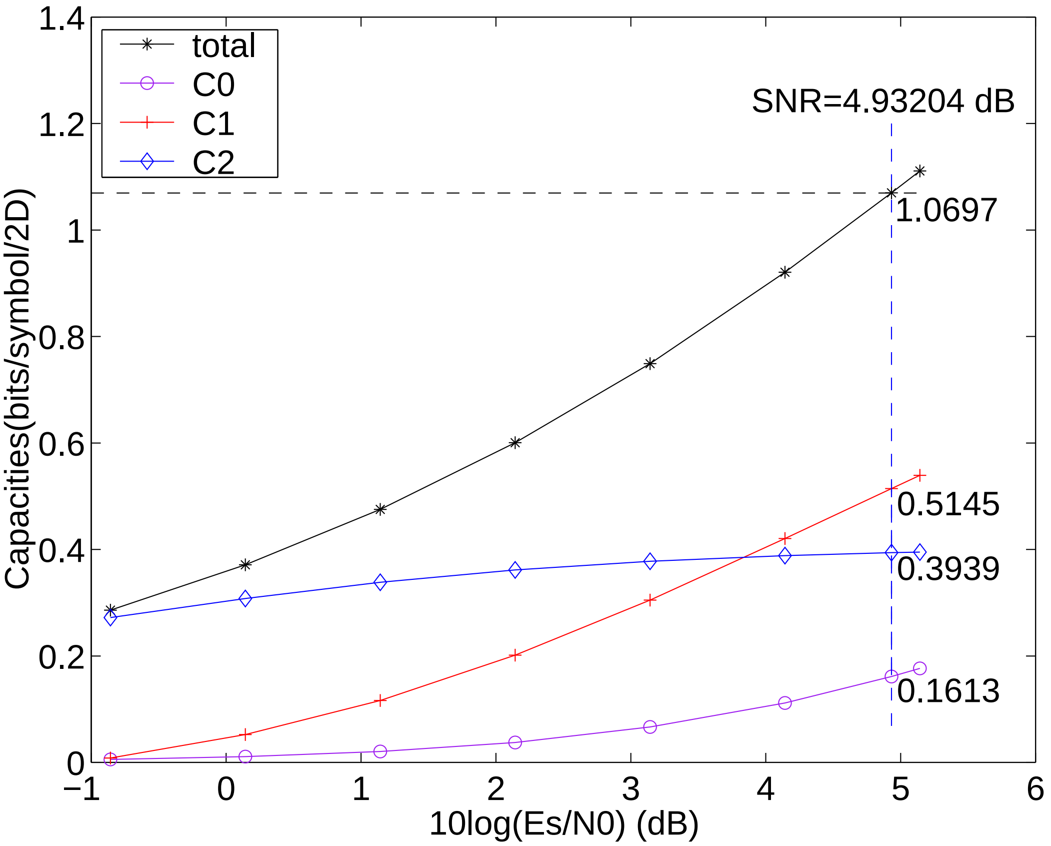


Figure 4.4: Total mutual information C and mutual information C^0,C^1 and C^2 of the equivalent channels, for the AWGN/Tikhonov channel with $B_LT = 0.01$, and the constellation of Fig. 4.1

get the rate at each level as:

$$R^0/R^1/R^2 = 0.0140/0.4580/0.0531. \tag{4.7}$$

Irregular Repeat-Accumulate (IRA) codes [26] and serially concatenated shaping codes are used as component codes in the proposed MLC architecture.

Serially concatenated shaping codes are used on the shaping levels (level 1 and 2).

Each such code consists of a cascade of an outer encoder, an interleaver permuting the outer codewords bits, and an inner encoder whose input words are the permuted outer codeword. The outer is a non-linear code which gives the desired probabilities of "0" and "1" bits. The overall code is designed according to [28]; that is, the outer code is chosen to have largest possible d_{free}, and the inner code is chosen to be a systematic recursive convolutional code with maximum d_{eff}, the minimum weight among all output sequences corresponding to weight-2 input sequences. An iterative decoding algorithm is developed and can offer superior performance.

4.3 Irregular Repeat-Accumulate (IRA) Codes

On the equiprobable level (level 0), we use a linear IRA code [26]. IRA codes are a generalization of the repeat-accumulate codes in [27]. They combine many of the favorable attributes of turbo codes and LDPC codes. Like turbo codes, they can be encoded in linear time. Like LDPC codes, they are amenable to an exact Richardson-Urbanke style analysis. The simulated results show that the performance is slightly superior to turbo codes of comparable complexity, and just as good as the best known irregular LDPC codes. Fig. 4.5 shows a Tanner graph of an IRA code with parameters $(f_1, \cdots, f_J; a)$, where $f_i \geq 0$, $\sum_i f_i = 1$ and a is a positive integer. A Tanner graph is a bipartite graph with two kinds of nodes: variable nodes (open circles) and check nodes (filled circles). For a systematic IRA code, the information nodes corresponding to the information bits, which are transmitted as part of the codeword. The remain codeword bits come from the parity nodes X_1 through X_r. If there are k variable nodes on the left, called information nodes, there are $r = (k \sum_i f_i)/a$ check nodes and r variable nodes on the right, called parity nodes. Each information node is connected to a number of check nodes. The *degree* of a node is the number of edges connected to it. The fraction of information nodes connected to i check nodes is denoted f_i, and a is the check degree, i.e., there are exactly a information nodes connected to each check node. The value of a parity node is determined uniquely by the parity-check equation, i.e., the mod-2 sum of the values of the variable nodes connected to each of the check nodes is zero. This encoding algorithm has the recursive form, so the encoding complexity is linear with the code block size. For the systematic IRA code, the rate is

$$R_{sys} = \frac{a}{a + \sum_i i f_i}. \tag{4.8}$$

IRA codes admit an iterative sum-product message-passing decoding algorithm. All messages are assumed to be in the form of LLRs. The outgoing message from a variable node to a check node represents information about this variable node, and a message from a check node to a variable node represents information about this check node. The outgoing message from a variable node to a check node is the summation of the incoming message from all neighbors except the targeted check node. And the outgoing message from a check node to a variable node follows the tanh rule [26]. The permutation in the encoder is a kind of interleaver; careful design of this permutation will lead to good decoding performance

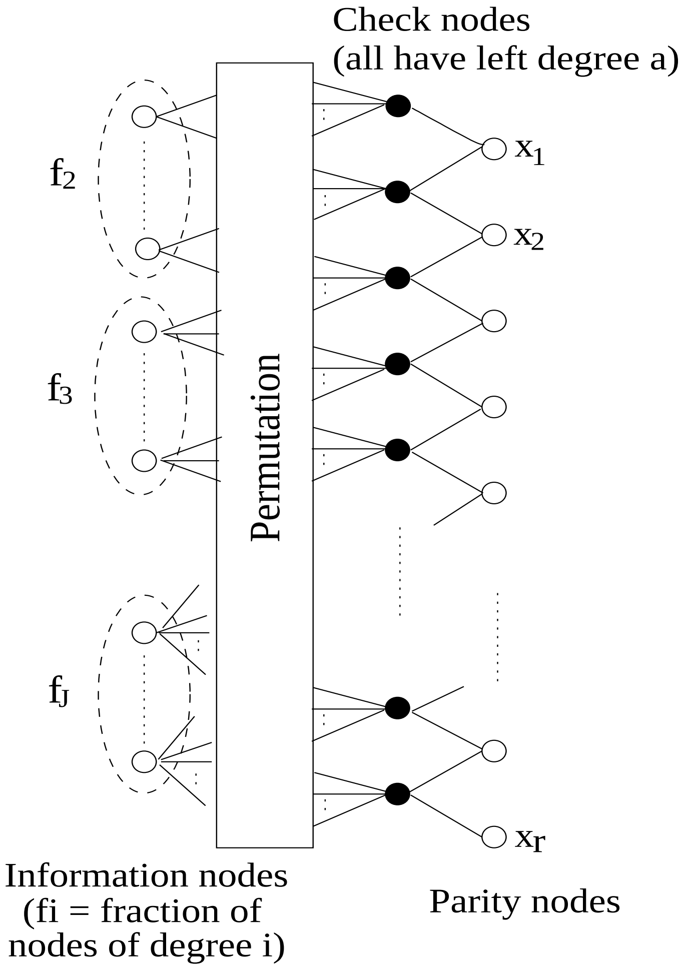


Figure 4.5: Tanner graph for IRA code with parameters $(f_1, \cdots, f_J; a)$

by increasing the girth (the minimum cycle length) of the Tanner graph and improving its connectivity.

In [26], the fixed point analysis based on Gaussian approximation is used to design the degree sequences λ_i of IRA codes with given rates, where λ_i is the fraction of edges between the information and the check nodes that connect to an information node of degree i. With the λ_i, the fraction f_i can be computed by $f_i = (\lambda_i/i)/(\sum_j \lambda_j/j)$. This design problem is converted to a linear programming problem. Given the maximum variable node degree J, the check node degree a and the target rate R, this linear programming will yield good degree sequences such that the IRA codes have noise thresholds close to the Shannon limit. The noise threshold is the maximum noise variance for which the code converges to zero decoding errors when the code block size goes to ∞.

4.4 Non-Equiprobable Serially-Concatenated Code

4.4.1 Encoder and Decoder of Serially Concatenated Shaping Codes

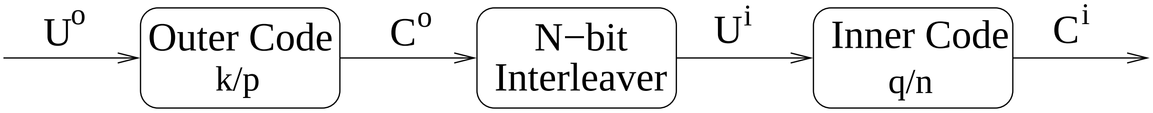


Figure 4.6: Encoder of Serially Concatenated Shaping Code

The structure of the serially concatenated shaping code is shown in Fig. 4.6. It consists of the outer code with rate $R^o = k/p$ which is a non-linear block code, and an inner systematic recursive convolutional code with rate $R^i = q/n$, joined by an interleaver of length N bits. N must be an integer multiple of p and q.

The non-linear block code is encoded using a lookup table. Under a given rate, this lookup table will give the desired probabilities P_0 and P_1 of binary bits. Also a large minimum Hamming distance is desirable. The lookup table consists of selected codeword sequences from a convolutional code. The construction procedure is:

1. Find all remerge paths of length p codeword bits. These remerge paths start and end

at the same state. The quantity p is the length of the lookup table codeword and it must be an integer multiple of the convolutional code's output.

2. Select paths which will give the right probabilities P_0 and P_1 of the binary outputs.

It is guaranteed that the minimum Hamming distance of the lookup table is $\mathrm{d}_{\mathrm{free}}$ of the convolutional code. So we should choose the convolutional code with large $\mathrm{d}_{\mathrm{free}}$. More memories in the convolutional code gives larger $\mathrm{d}_{\mathrm{free}}$, but less paths starting and ending at the same state of length p codeword bits. There is a tradeoff between minimum Hamming distance and number of paths which will give the right probabilities. If the lookup table gives the probability of bit "1" as P_1, then the probability of bit "1" P_1' at the output of the inner code is:

$$P_1' = \frac{P_1 * q + (n - q) * 0.5}{n}. \tag{4.9}$$

where q is the number of systematic bits of the inner convolutional code and n is the number of output bits. Here we assume that the parity bits of the inner code are uniformly distributed. Note that all code designs presented in this dissertation assume $q = n - 1$.

The block SISO (Soft-Input Soft-Output) is used within the iterative decoding algorithm as shown in Fig. 4.7. The notations of U^i and U^o indicate input symbols of the inner and outer code, and C^o is the output symbol of the outer code. The input and output extrinsic informations of SISOs are denoted by “I” and “O”. The SISOs exchange extrinsic information about the inputs or codewords in the form of LLRs (the λs). The interleavers (πs) ensure that the extrinsic information input to a SISO is independent of the SISO's

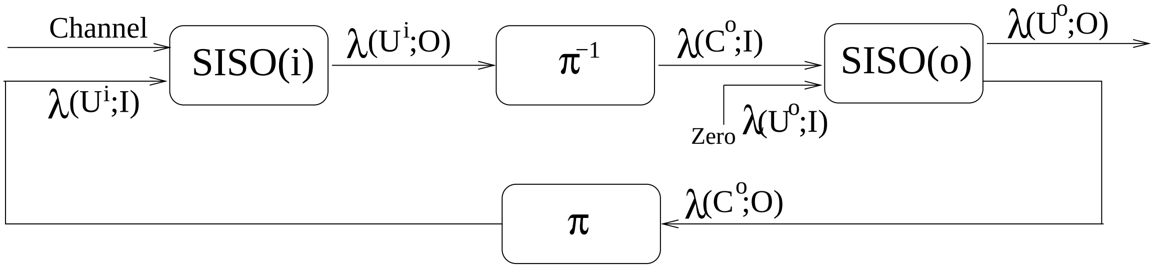


Figure 4.7: Iterative decoder for serially concatenated shaping code uses SISO block for the inner (i) convolutional code and outer (o) non-linear block code

estimate of the input or codeword bits.

4.4.2 SISO of Inner Convolutional Code

For the inner SISO, a MAP algorithm based on the convolutional code trellis is used. It works at the symbol level, i.e. for our (n, q) convolutional code, it operates on information symbols belonging to an alphabet with size 2^q and on code symbols belonging to an alphabet with size 2^n. The log-likelihood ratio (LLR) for a symbol x is defined as [28]:

$$\lambda(x;\cdot) \equiv \log\left[\frac{P(x;\cdot)}{P(x_{ref};\cdot)}\right]. \tag{4.10}$$

When x is a binary symbol, "0" or "1", x_{ref} is assumed to be the "1". When x belongs to an $L-$ary alphabet, x_{ref} is chosen as one of the L symbols (we choose the all "1"s symbol). Here, $P(x;\cdot)$ is given by (2.3) and (2.4). At time k, $k = 1,\cdots,K$, the output extrinsic

LLRs are computed as [28]:

$$\lambda_k(U;O) \;\; = \overset{*}{\max_{e:U(e)=u}} \{\alpha_{k-1}[s^S(e)] + \lambda_k[C(e);I] + \beta_k[s^E(e)]\} + \lambda(U) + h_u \quad (4.11)$$

where $U(e)$ and $C(e)$ are the information and code symbols associated to the edge e and $s^S(e)$ and $s^E(e)$ are the starting and ending states. The prior LLR of symbol U is denoted by $\lambda(U)$, and h_u are normalization constants. Note that, due to the non-linear outer block code, the prior LLRs $\lambda(U)$ seen by the inner code are not 0. The quantities $\alpha_k(\cdot)$ and $\beta_k(\cdot)$ in (4.11) are obtained through the forward and backward recursions, as

$$\alpha_k(s) = \overset{*}{\max_{e:s^E(e)=s}} \{\alpha_{k-1}[s^S(e)] + \lambda_k[U(e);I] + \lambda_k[C(e);I]\} \quad (4.12)$$

$$\beta_k(s) = \overset{*}{\max_{e:s^S(e)=s}} \{\beta_{k+1}[s^S(e)] + \lambda_{k+1}[U(e);I] + \lambda_{k+1}[U(e);I]\}, \quad (4.13)$$

with initial values

$$\alpha_0(s) = \begin{cases} 0 & s = S_0 \\ -\infty & otherwise, \end{cases} \quad \beta_K(S_i) = \begin{cases} 0 & s = S_K \\ -\infty & otherwise. \end{cases}$$

The operator max^* performs the following operation:

$$\overset{*}{\max_{j}}(a_j) \equiv \log \left[\sum_{j=1}^{J} e^{a_j} \right]. \quad (4.14)$$

In order to perform bit interleaving, we need to transform the symbol extrinsic LLRs into bit LLRs before they enter the de-interleaver. The bit LLRs need to be compacted into symbol LLR's before entering the inner SISO. These operations are performed under the assumption that the bits forming a symbol are independent:

$$\begin{aligned}\lambda_i(u;O) &= \max^*_{U:u_i=0}[\lambda(U;O)+\lambda(U;I)] \\ &- \max^*_{U:u_i=1}[\lambda(U;O)+\lambda(U;I)] - \lambda_i(u;I), \qquad (4.15)\end{aligned}$$

$$\lambda(U) = \sum_{i=1}^{q} \lambda_i(u_i). \qquad (4.16)$$

4.4.3 SISO of Outer Non-linear Block Code

For the outer SISO, since the outer block code is non-linear, there is no trellis structure or generating matrix that can be used. The bitwise LLRs are computed based on a lookup table. For simplicity, we denote $(\lambda_1 \cdots \lambda_{k-1}, \lambda_{k+1} \cdots \lambda_p)$ as $\lambda^{'}$; then the probability that the k^{th} outer codeword bit c_k is -1, given the extrinsic information from the inner SISO, is:

$$\begin{aligned}P(c_k=-1|\lambda_1 \cdots \lambda_p) &= \frac{P(c_k=-1,\lambda^{'},\lambda_k)}{P(\lambda^{'},\lambda_k)} = \frac{P(c_k=-1,\lambda_k|\lambda^{'})P(\lambda^{'})}{P(\lambda^{'},\lambda_k)} \\ &= \frac{P(c_k=-1,\lambda_k|\lambda^{'})}{P(\lambda_k|\lambda^{'})} = \frac{P(c_k=-1|\lambda^{'})P(\lambda_k|c_k=-1,\lambda^{'})}{P(\lambda_k|\lambda^{'})} \\ &= \frac{P(c_k=-1|\lambda^{'})}{P(\lambda_k)}\frac{P(c_k=-1|\lambda_k)P(\lambda_k)}{P(c_k=-1)} \\ &= P(c_k=-1|\lambda^{'})\frac{P(c_k=-1|\lambda_k)}{P(c_k=-1)}. \qquad (4.17)\end{aligned}$$

Since the interleaver makes the $\lambda_1 \cdots \lambda_p$ independent, the equality on the third line of (4.17) follows from

$$P(\lambda_k|\lambda^{'}) = P(\lambda_k), \quad \text{and} \tag{4.18}$$

$$P(\lambda_k|c_k = -1, \lambda^{'}) = P(\lambda_k|c_k = -1) = \frac{P(c_k = -1|\lambda_k)P(\lambda_k)}{P(c_k = -1)}. \tag{4.19}$$

Hence, from (4.17),

$$P(c_k = -1|\lambda^{'}) = \frac{P(c_k = -1|\lambda_1 \cdots \lambda_p)P(c_k = -1)}{P(c_k = -1|\lambda_k)}. \tag{4.20}$$

Using the result of (4.17), the extrinsic information of codewords is computed as :

$$\begin{aligned}
\tilde{\lambda}_k(c; O) &= \log \frac{P(c_k = -1|\lambda')}{P(c_k = +1|\lambda')} \\
&= \log \frac{P(c_k = -1|\lambda_1 \cdots \lambda_p)P(c_k = +1|\lambda_k)P(c_k = -1)}{P(c_k = +1|\lambda_1 \cdots \lambda_p)P(c_k = -1|\lambda_k)P(c_k = +1)} \\
&= \log \frac{P(c_k = -1|\lambda_1 \cdots \lambda_p)}{P(c_k = +1|\lambda_1 \cdots \lambda_p)} - \log \frac{P(c_k = -1|\lambda_k)}{P(c_k = +1|\lambda_k)} + \log \frac{P(c_k = -1)}{P(c_k = +1)} \\
&= \log \frac{\sum_{C:c_k=-1} P(C|\lambda_1 \cdots \lambda_p)}{\sum_{C:c_k=+1} P(C|\lambda_1 \cdots \lambda_p)} - \lambda_k + \log \frac{P(c_k = -1)}{P(c_k = +1)},
\end{aligned} \tag{4.21}$$

where p is the length of codewords in the lookup table. Since the interleaver is used,

$\lambda_1 \cdots \lambda_p$ are independent. So

$$\begin{aligned} P(C|\lambda_1 \cdots \lambda_p) &= \prod_{k=1}^{p} P(c_k|\lambda_1 \cdots \lambda_p) \\ P(c_k = -1|\lambda_k) &= \frac{e^{\lambda_k}}{1+e^{\lambda_k}} \\ P(c_k = +1|\lambda_k) &= \frac{1}{1+e^{\lambda_k}}. \end{aligned} \tag{4.22}$$

The LLRs of input bits are computed as:

$$\begin{aligned} \lambda_k(u;O) &= \log \frac{P(u_k = -1|\lambda_1 \cdots \lambda_p)}{P(u_k = +1|\lambda_1 \cdots \lambda_p)} \\ &= \log \frac{\sum_{U:u_k=-1} P(U|\lambda_1 \cdots \lambda_p)}{\sum_{U:u_k=+1} P(U|\lambda_1 \cdots \lambda_p)}, \end{aligned} \tag{4.23}$$

where,

$$P(U|\lambda_1 \cdots \lambda_p) = P(C = \mathrm{Mapping}(U)|\lambda_1 \cdots \lambda_p). \tag{4.24}$$

If $\lambda_k(u;O) \geq 0$, we decode the information bit as 0, otherwise 1.

In order to speedup the decoding, a suboptimal search procedure is implemented as follows:

1. Based on the extrinsic information LLR of codewords from Inner SISO, a codeword is decoded by hard decision.

2. From the whole lookup table, select codewords whose Hamming distance with the decoded codeword is no larger than d. A larger d gives more accurate decoding, but

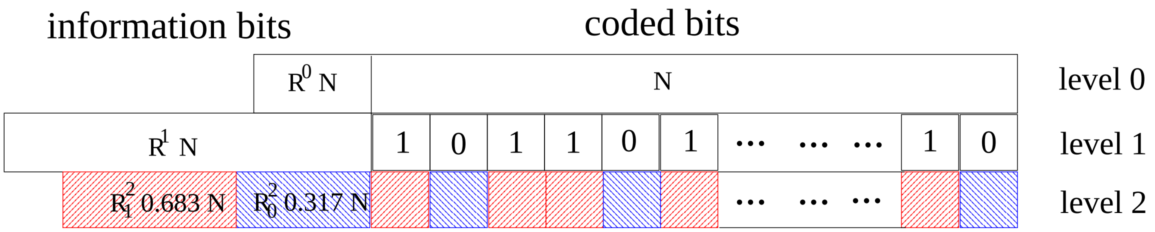


Figure 4.8: Encoders at each level for the rate 1 bit/symbol/Hz MLC

requires longer time. The selected codewords consist of a smaller subtable. In our simulation, we use $d = 6$.

3. Compute the extrinsic information and LLRs as in (4.21) and (4.23) using the subtable.

4.5 Multi-Level Design Examples

In this section, we list the details for code designs of 1 bit/symbol and 0.5 bit/symbol.

4.5.1 Design Example at Rate 1 bit/symbol/Hz

As shown in Fig. 4.1, the partition gives the target multi-level code rate distribution as (4.6), where the rate R^2 code is constructed by two codes with rates 0.5053 and 0.15427 respectively. Fig. 4.8 shows how the encoders work at each level, assuming the ideal rates of (4.6).

- The output block length at each level is N.

- At level 0, a linear code is used and the input block length is $R^0 \times N$ with $R^0 = 0.1613$.

- At level 1, a non-linear shaping code is used with output bit probabilities 0.683 and 0.317, and the input block length is $R^1 \times N$ with $R^1 = 0.5145$. There are $0.683 \times N$ “1” output bits and $0.317 \times N$ “0” output bits. In practice, there may not be exactly $0.683 \times N$ “1” output bits at this level. In this case, a small number of “0” or “1” bits are coded with the wrong code. Because this number is small compared to the large block size, the effect is negligible.

- At level 2, two non-linear shaping codes are used. One corresponds to level 1 “1” bits and has $R_1^2 = 0.5053$ and output bit probabilities 0.8878 and 0.1122. The input block length is $R_1^2 \times 0.683 \times N$ and output block length is $0.683 \times N$. The other code corresponds to level 1 “0” bits and has $R_0^2 = 0.1542$ and output bit probabilities 0.9733 and 0.0267. The input block length is $R_0^2 \times 0.317 \times N$ and output block length is $0.317 \times N$. Overall rate on level 2 is $0.683 \times R_1^2 + 0.317 \times R_0^2 = 0.3939$.

In our design, we use the following MLC rate distribution:

$$R^0/R^1/R^2 = 0.2449/0.5833/0.2415. \tag{4.25}$$

These rates vary from the ideal rates in (4.6), due to tradeoffs involved in achieving the desired ring probabilities.

Fig. 4.9 shows the code design details at each level.

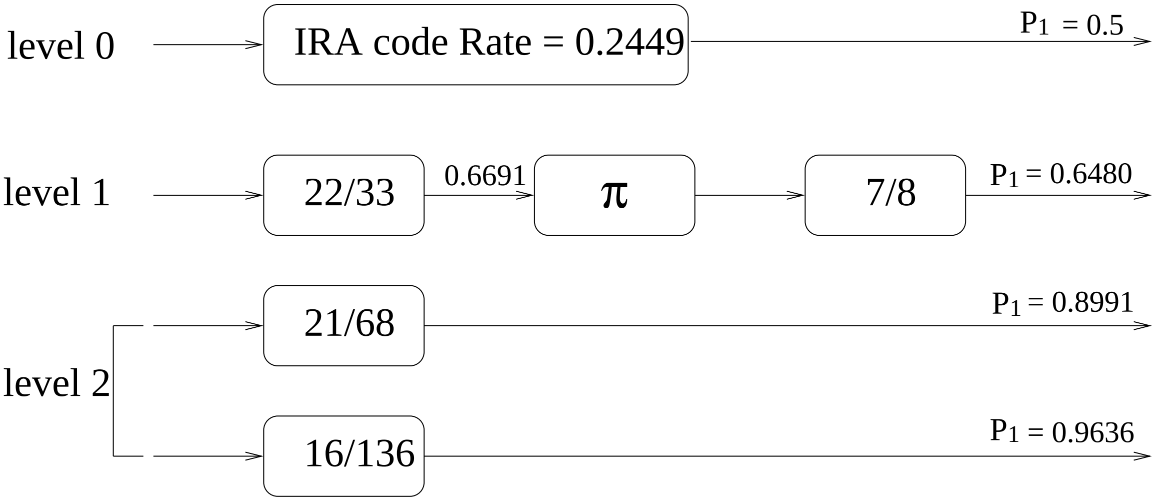


Figure 4.9: Code design example at 1 bit/symbol

- At level 0, a linear IRA code with rate 0.2449 is used. The IRA code is designed with the maximum variable node degree of 34, and fixed check node degree of 3.

- At level 1, the inner code is a rate $\frac{7}{8}$ convolutional code with $\mathrm{d_{free}} = 4$ and 4 memories [11]. The lookup table of the outer code, which is constructed by 3 stages of the $\frac{10}{11}$ convolutional code with $\mathrm{d_{free}} = 3$ and 4 memories [11], gives the probability of $P_1 = 0.6691$; then the probability of “1” at the output of inner code is 0.6480.

- At level 2, there are two different shaping codes. One is constructed by 4 stages of the $\frac{16}{17}$ convolutional code with $\mathrm{d_{free}} = 3$ and 5 memories [11], which gives the probability of $P_1 = 0.8991$. The other is constructed by 8 stages of the same $\frac{16}{17}$ convolutional code, which gives the probability of $P_1 = 0.9636$. Both codes are decoded non-iteratively because of the larger Euclidean distance at level 2.

The final achieved ring probabilities are 0.5826, 0.3392, 0.0654 and 0.0128 from the inner to the outer; the target probabilities are 0.60636, 0.30856, 0.07662 and 0.00846. The average symbol energy of 6.4177 is a little higher than the target of 6.1622.

4.5.2 Design Example at Rate 0.5 bit/symbol/Hz

In our design, we use the following multi-level codes rate distribution:

$$R^0/R^1/R^2 = 0.02202/0.44444/0.05864. \tag{4.26}$$

These rates vary from the ideal rates in (4.7), due to tradeoffs involved in achieving the desired ring probabilities.

Fig. 4.10 shows the code design details at each level.

- At level 0, a linear IRA code with rate 0.02202 is used. The IRA code is designed with the maximum variable node degree of 2800, and fixed check node degree of 2.
- At level 1, the inner code is a rate $\frac{8}{9}$ convolutional code with $\mathrm{d_{free}} = 4$ and 4 memories. The lookup table of the outer code, which is constructed by 3 stages of the $\frac{13}{14}$ convolutional code with $\mathrm{d_{free}} =$ 3 and 4 memories, gives the probability of $P_1 = 0.8028$; then the probability of “1” at the output of inner code is 0.7692.
- At level 2, there are two different codes. One only uses shaping block code which is

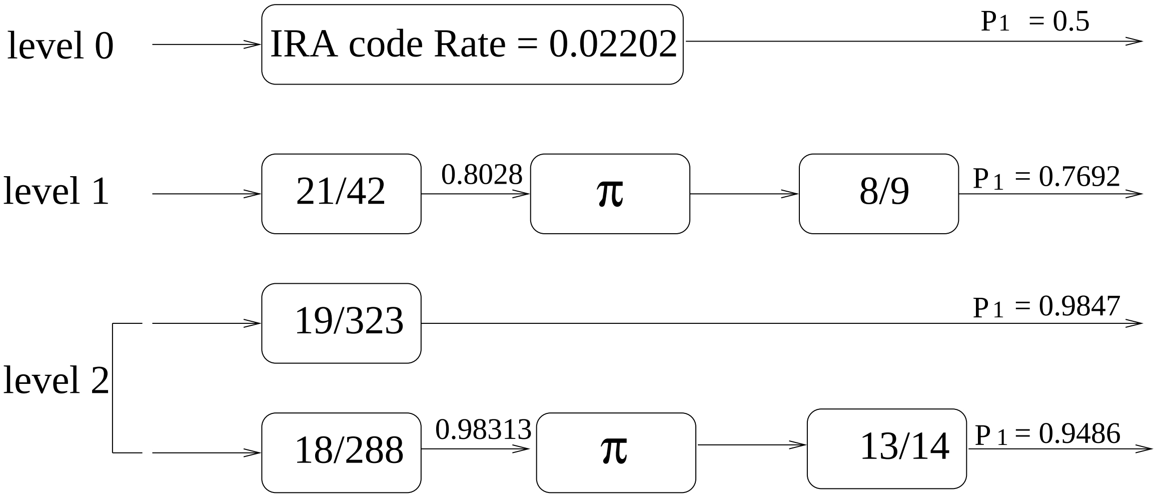


Figure 4.10: Code design example at 0.5 bit/symbol

constructed by 18 stages of the $\frac{15}{16}$ convolutional code with $\mathrm{d}_{\text{free}} = 3$ and 4 memories, with an additional 35 '1's appended to each codeword, and gives the probability of $P_1 = 0.9847$. Iterative decoding is not necessary due to the large SED between signal points at this level. The other is a serially concatenated code. The inner code is a rate $\frac{13}{14}$ convolutional code with $\mathrm{d}_{\text{free}} = 4$ and 4 memories. The look up table of the outer code is constructed by 18 stages of the $\frac{15}{16}$ convolutional code with $\mathrm{d}_{\text{free}} = 3$ and 4 memories, which gives the probability of $P_1 = 0.98313$.

The final achieved ring probabilities are 0.7574, 0.2189 and 0.0236 from the inner to the outer; the target probabilities are 0.809123, 0.179967 and 0.01091. The average symbol energy of 3.1820 is higher than the target of 2.3297.

4.6 Code Search

In 0.5 bit/symbol code design, at level 1, we use a serially concatenated shaping code in Fig. 4.10. To improve this code's performance, there are two solutions. We can either maximize the minimum Hamming distance of the outer block code and the $\mathrm{d_{free}}$ of the inner convolutional code, or decrease the convergence threshold. We found that more memories in the convolutional code gives larger $\mathrm{d_{free}}$, but less paths starting and ending at the same state of length p codeword bits, and so there is a tradeoff between minimum Hamming distance and number of paths which will give the right probabilities. Thus, there are limitations when we seek the first solution.

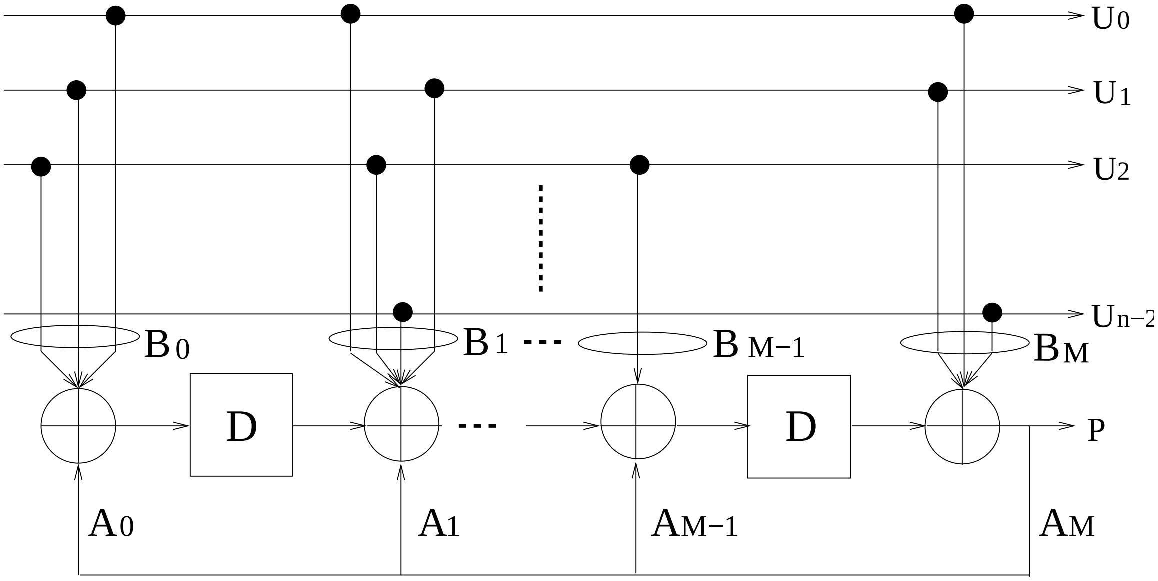


Figure 4.11: An example of $(n, n-1)$ convolutional systematic recursive encoder with M memories

When the Gaussian approximation technique is used to predict a code's convergence threshold, then the convergence threshold can be used as a code search criterion. For level 1's serially concatenated shaping code, since the outer block code is fixed, we can search for inner convolutional codes with good convergence threshold to improve the performance. Fig. 4.11 shows a rate $(n-1)/n$ systematic recursive convolutional encoder with M memories. The connections of the recursive part are denoted by $A_0, A_1 \cdots A_M$, with values 0 or 1, and A_M is always 1. The quantities $B_0, B_1 \cdots B_M$ indicate the input connections to the modulo-2 summers, with values between 0 and 2^{n-1}. Since the outer block code is fixed, its $\mathrm{SNR}_{\mathrm{in}}$/ $\mathrm{SNR}_{\mathrm{out}}$ curve is also fixed as shown in Fig. 4.12. In practice, the curve does not move very much at different E_b/N_0. Given an inner convolutional code, we plot its $\mathrm{SNR}_{\mathrm{in}}$/ $\mathrm{SNR}_{\mathrm{out}}$ curve and check whether there is an intersection with the outer $\mathrm{SNR}_{\mathrm{in}}$/ $\mathrm{SNR}_{\mathrm{out}}$ curve. The convergence threshold is the minimum E_b/N_0 under which there is no intersection between the inner and outer codes' $\mathrm{SNR}_{\mathrm{in}}$/ $\mathrm{SNR}_{\mathrm{out}}$ curves.

All the connections in Fig. 4.11 are taken over GF(2). So for the case $M = 4$ and $n = 9$, all together there will $2^4 \cdot 2^{8 \cdot 5} = 2^{44}$ possible codes. The following constraints are applied to narrow the search space.

1. The recursive connections $A_0, A_1 \cdots A_{M-1}$ can not all be zeros.

2. Each input should have at least one connection with the modulo-2 summer.

After applying these conditions, the total number of codes left in the search space is close to the number $(2^4 - 1) \cdot (2^{40} - 2^{35} \cdot 8)$. This is still a huge search space.

The partial code search results for rate $8/9$ RCCC with 4 memories are listed in Table

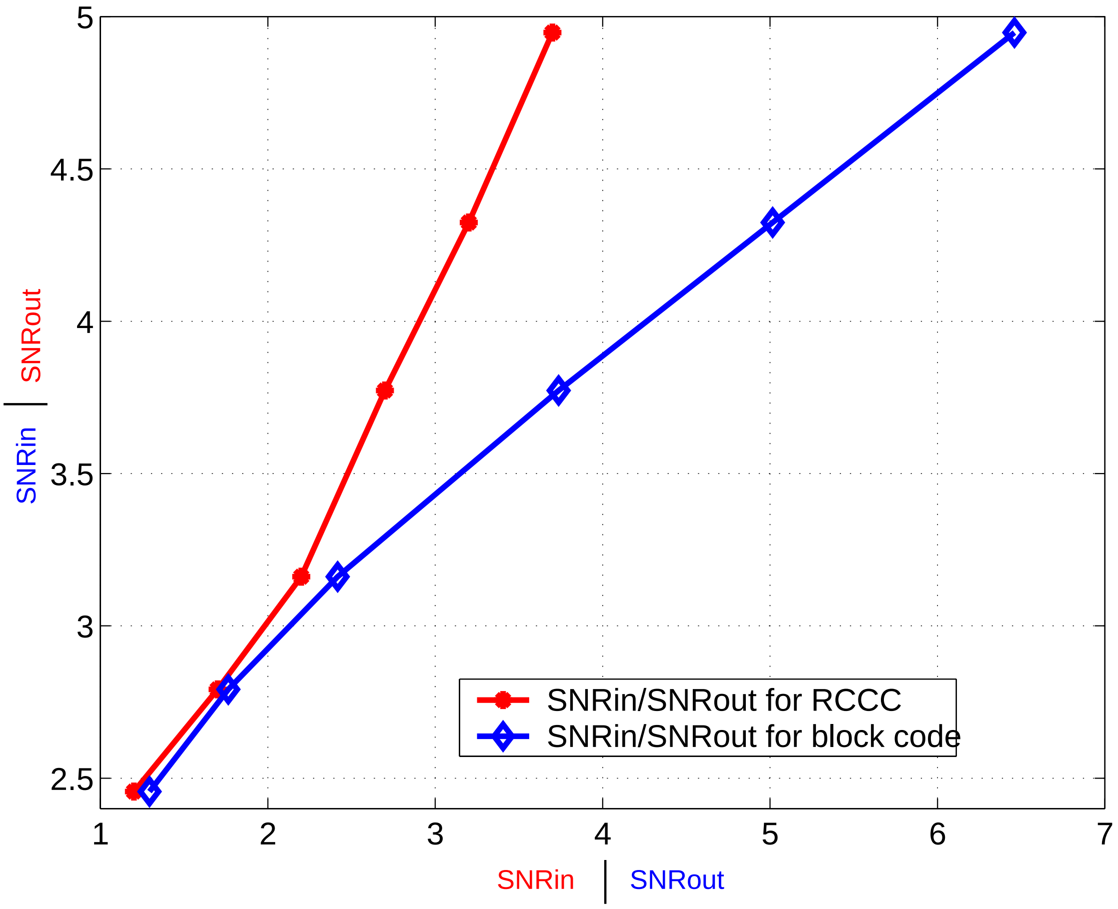


Figure 4.12: $\mathrm{SNR_{in}}/\ \mathrm{SNR_{out}}$ curves of outer and inner codes

4.2. We use the first code in our best simulation results. The last code was used in a previous simulation done before the code search was performed.

4.7 Reduced-Complexity Code Search

To reduce the complexity of code searches for good high-rate $(n, n-1)$ convolutional codes, we use the encoder construction techniques in [12]. Fig. 4.13 is the overall convo-

Table 4.2: Top five codes for rate $8/9$ RCCC code search and their convergence thresholds

Connection Matrix										Convergence Threshold
A_0	A_1	A_2	A_3	A_4	B_0	B_1	B_2	B_3	B_4	(dB)
1	1	1	0	1	255	255	255	255	255	10.10
1	1	0	0	1	255	255	255	255	255	10.10
0	0	0	1	1	255	255	255	255	255	10.10
1	0	0	0	1	0	0	0	0	255	10.20
1	1	0	0	1	239	30	141	91	255	10.20

lutional encoder, which is composed of two constituent encoders: a rate $(n, n-k)$ block encoder associated to the parallel edges, and a low-to-moderate rate $(k, k-1)$ convolutional encoder that defines the dynamical part of the trellis, with additional $n-k$ '0's appended to each codeword. The final output codeword is the modulo-2 summation of two length-n codewords from the block code and the convolutional code. With this construction, the polynomial generator matrix $G(D)$ is given by:

$$G(D) = \begin{bmatrix} G_{BC}(D) \\ X \quad G_{CC}(D) \end{bmatrix},$$

where $G_{BC}(D)$ is the $(n-k) \times n$ generator matrix of the constituent block encoder, X is a $(k-1) \times (n-k)$ all-zero matrix, and $G_{CC}(D)$ identifies the $(k-1) \times k$ generator matrix of the constituent convolutional encoder.

With the encoder structure in Fig. 4.13, we can use the following search procedure.

1. Search all rate $(n, n-k)$ block codes with d_{min} greater or equal to the maximum d_{free}.

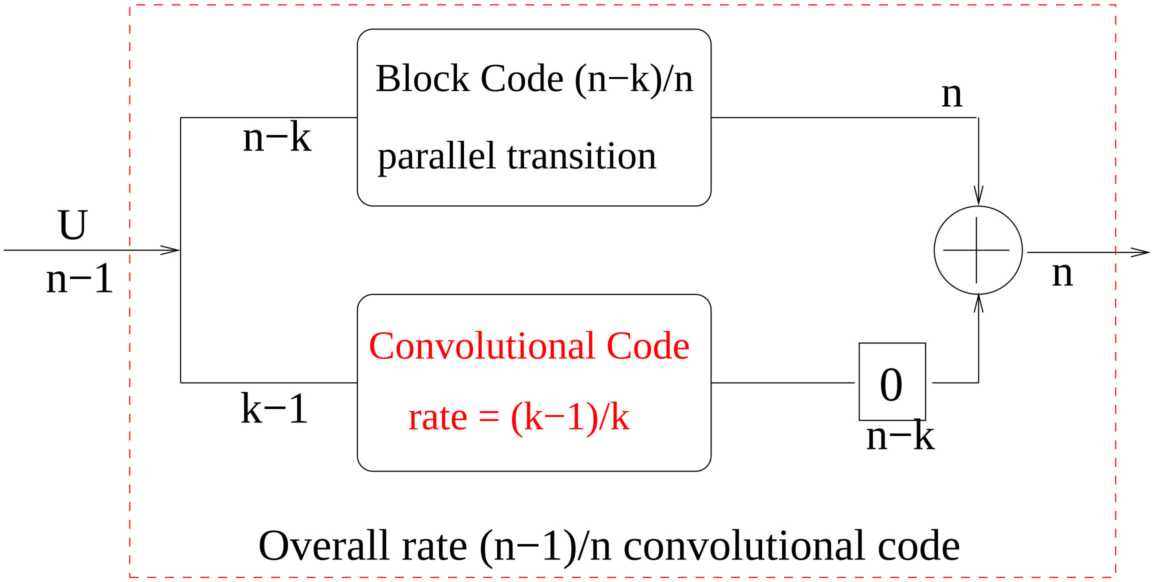


Figure 4.13: The structure of the $(n, n-1)$ convolutional code encoder

2. For each block code, search all rate $(k, k-1)$ RCCCs.

This procedure significantly reduces the search complexity, since it is somewhat split into two simpler searches. In the code search for rate $8/9$ RCCCs with 4 memories, we pick $k = 4$, and the two constituent codes are a rate $5/9$ block code and a rate $3/4$ RCCC. To have $d_{min} \geq 3$, in the 4×5 parity-check matrix of rate $5/9$ block codes, the vectors' weights should be at least 2. There are $C_4^2 + C_4^3 + C_4^4 = 11$ such vectors, and therefore $C_{11}^5 = 462$ parity-check matrices. The overall complexity is 462 systematic block codes times 2^{19} possible RCCCs, which is less than 2^{28}, and represents a factor of 2^{16} of improvement with respect to the direct search. One example of the code search is the following. The generator matrix $G(D)$ is:

$$G(D) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \frac{D^4+D^3+D^2+D+1}{D^4+D^3+D+1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \frac{D^4+D^3+D^2+D}{D^4+D^3+D+1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \frac{D^4+D^3+D^2}{D^4+D^3+D+1} \end{bmatrix}$$

where the upper 5×9 submatrix describes the constituent block encoder and the 3×4 lower right submatrix defines the constituent RCCC encoder. The block code has $d_{min} = 3$. The equivalent systematic encoder $G_{sys}(D)$ is obtained by multiplying $G(D)$ by the inverse matrix of the left 8×8 submatrix of $G(D)$, then

$$G_{sys}(D) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{D^4+D^3+D^2+1}{D^4+D^3+1} \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{D^4+D^3+D}{D^4+D^3+1} \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \frac{D^4+D^3}{D^4+D^3+1} \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \frac{D^4+D^3+D}{D^4+D^3+1} \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & \frac{D^2}{D^4+D^3+1} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \frac{D^4+D^3+D^2+D+1}{D^4+D^3+D+1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \frac{D^4+D^3+D^2+D}{D^4+D^3+D+1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \frac{D^4+D^3+D^2}{D^4+D^3+D+1} \end{bmatrix}$$

In the code search, we use the constraint condition that no numerator and denominator in the right column of G_{sys} cancel out. The codes searched under this condition have lower error floors.

To speed up the code search, we use the Dual-MAP algorithm for high-rate convolutional codes [43] to decode the inner convolutional code. This Dual-MAP algorithm reduces the computational complexity by using reciprocal dual convolutional codes, since the number of codewords to consider is decreased for codes of rate greater than $1/2$. For a rate $(n-1)/n$ RCCC, there are 2^{n-1} possible codewords at each state, while its dual code has rate $1/n$ and only 2 possible codewords at each state. The number of trellis states for the original $(n-1)/n$ RCCC is same as for its dual code. Thus when n is large, the computational complexity is reduced exponentially by using the Dual-MAP algorithm.

The dual convolutional code $C^{\perp}$ of a rate $(n-1)/n$ convolutional code C consists of all code sequences $v^{\perp}(D)$ orthogonal to all code sequences $v(D) \in C$. $C^{\perp}$ is a rate $1/n$ convolutional code generated by any generator matrix $H(D)$ with the property $G(D)H^{T}(D) = 0$. The generator matrix of a reciprocal convolutional code $\tilde{C}$ is obtained by substituting D^{-1} for D in $G(D)$ and by multiplying the jth row, $1 \leq j \leq n-1$, of the resulting matrix with $D^{\delta(j)}$, where $\delta(j)$ is the degree of the jth row of $G(D)$. Then, a sequence $\tilde{v}(D) \in \tilde{C}$ is equal to the time-reversed sequence $v(D^{-1})$.

An example of a rate $2/3$ convolutional code C is described by the polynomial generator matrix

$$G_{sys}(D) = \begin{bmatrix} 1 & 0 & \frac{1}{1+D+D^2} \\ 0 & 1 & \frac{1+D^2}{1+D+D^2} \end{bmatrix}.$$

The rate $1/3$ dual code $C^{\perp}$ is encoded by

$$H(D) = \begin{bmatrix} 1 & 1+D^2 & 1+D+D^2 \end{bmatrix}.$$

Hence, the reciprocal dual code $\tilde{C}^{\perp}$ has the generator matrix

$$\tilde{H}(D) = \begin{bmatrix} D^2 & 1+D^2 & 1+D+D^2 \end{bmatrix}.$$

The bit-wise MAP decoding for high-rate $(n-1)/n$ convolutional codes using its reciprocal dual code is described by the following equations. The LLR of the lth bit is given by

$$L(\hat{v}_l) = L(v_l; y_l) + \log \frac{\sum_{s_1=0}^{2^v-1} \sum_{s_2 \in S_B(s_1)} \alpha_t(s_1) \tilde{\gamma}_t(l, s_1, s_2) \beta_{t+1}(s_2)}{\sum_{s_1=0}^{2^v-1} \sum_{s_2 \in S_B(s_1)} \alpha_t(s_1) (-1)^{b_{l-tn}(s_1,s_2)} \tilde{\gamma}_t(l, s_1, s_2) \beta_{t+1}(s_2)}, \tag{4.27}$$

where

$$\alpha_{t+1}(s) = \sum_{s' \in S_A(s)} \alpha_t(s') \cdot \gamma_t(s', s), \quad 0 \le t < N-1 \tag{4.28}$$

$$\beta_{t-1}(s) = \sum_{s' \in S_B(s)} \beta_t(s') \cdot \gamma_{t-1}(s, s'), \quad 2 \le t \le N \tag{4.29}$$

$$\gamma_t(s_1, s_2) = \prod_{i=0}^{n-1} [\tanh(L(v_{tn+i}; y_{tn+i})/2)]^{b_i(s_1,s_2)}, \tag{4.30}$$

$$\tilde{\gamma}_t(l, s_1, s_2) = \prod_{i=0, i\neq l-tn}^{n-1} [\tanh(L(v_{tn+i}; y_{tn+i})/2)]^{b_i(s_1,s_2)}. \tag{4.31}$$

The notations of $S_A(s)$ and $S_B(s)$ describe the possible transitions between a state s and another state within one trellis stage. To be specific, $S_A(s)$ contains the starting states s_i such that the transitions $s_i \rightarrow s$ exist whereas $S_B(s)$ is the set of destination states s_j within reach from state s (transition $s \rightarrow s_j$). Bit associated with transition $s_1 \rightarrow s_2$ are combined in the $n-$tuple $(b_0(s_1, s_2), \cdots, b_{n-1}(s_1, s_2))$. The time index t indicates the trellis stage, and there are totally N stages.

The reduced complexity code search results for rate $8/9$ RCCC with 4 memories are listed in Table 4.3. All these codes are searched with the block code having the following generator matrix:

$$G(D) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

We note that all five of the top codes in Table 4.3 have slightly smaller (and hence better) convergence threshold than the top codes in the larger code search results shown in Table 4.2.

Table 4.3: Top five codes for rate $8/9$ RCCC reduced complexity code search and their convergence thresholds

Connection Matrix										Convergence Threshold (dB)
A_0	A_1	A_2	A_3	A_4	B_0	B_1	B_2	B_3	B_4	
0	1	1	1	1	61	239	30	85	153	10.06
0	1	0	0	1	75	164	61	135	85	10.06
1	0	1	0	1	35	75	164	75	153	10.08
1	1	0	1	1	153	210	204	239	35	10.08
0	1	1	0	1	241	239	30	204	85	10.08

Chapter 5

Simulation Results

5.1 Simulation Results for $B_L T = 0.01$ at Rate 1 bits/symbol/Hz

The simulation results for 1bit/symbol multi-level coding for AWGN channel with Tikhonov phase error for $B_L T = 0.01$ are shown in Fig. 5.1. The bit error rate curve for the non-equiprobable multi-level codes is about 1.1 dB away from non-equiprobable constellation-constrained capacity at $P_b \approx 10^{-5}$, which outperforms the comparable iteratively decoded equiprobable code by 1.70 dB. The block length used in the non-equiprobable MLC is $N = 246840$. We set the maximum number of iterations on level 0, level 1 and level 2 to be 200, 30 and 1 respectively. (Recall level 2 uses non-iterative decoding for 1 bit/symbol.) The equiprobable code is constructed by multi-level codes using IRA codes as component codes, using the BPSK constellation of Fig. 2.2 with the same ring radii but equiprobable rings. According to the capacity rule of MLC, the rate of each level of the equiprobable

MLC is chosen as:

$$R^0/R^1/R^2 = 0.144577/0.163047/0.762376. \tag{5.1}$$

The average power of this equiprobable MLC is 26.1216. In the simulation, the block length N is set to be 240059 and the maximum number of iterations is 350.

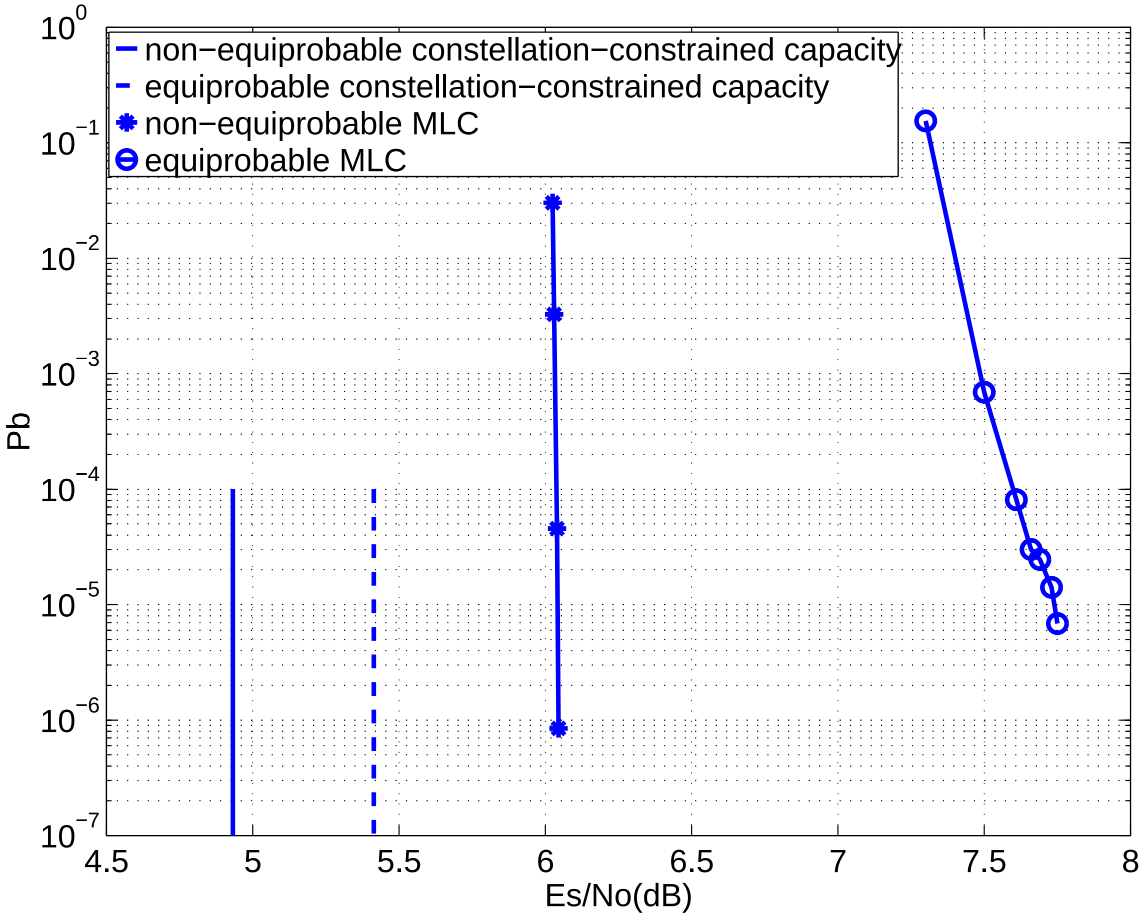


Figure 5.1: Simulation results for 1bit/symbol multi-level coding for AWGN channel with Tikhonov phase error for $B_L T = 0.01$

5.2 Simulation Results for $B_L T = 0.01$ at Rate 0.5 bits/symbol/Hz

The simulation results for 0.5bit/symbol multi-level coding for AWGN channel with Tikhonov phase error for $B_L T = 0.01$ are shown in Fig. 5.2. The bit error rate curve for the non-equiprobable multi-level codes, which is marked with $\diamond$ is about 1.58 dB away from constellation capacity at $P_b \approx 10^{-5}$. It outperforms the equiprobable constellation-constrained capacity by 0.33 dB, and shows that no equiprobable code can beat our codes, since our codes' operation points are below the equiprobable constellation-constrained capacity. The computation of the equiprobable constellation-constrained capacity assumes that three rings are equiprobable with $1/3$ probability on each ring, and the points on each are also equiprobable. The block length used in the non-equiprobable MLC is $N = 158382$. The maximum number of iterations on level 0, level 1 and level 2 is set to be 300, 35 and 1 respectively.

The bit error rate curve with $\diamond$ markers is about 0.2 dB better than the one marked by o. This 0.2 dB gain comes from two parts. One is due to the power savings; by redesigning the non-linear lookup tables on level 1 and level 2, which give the probabilities closer to the desired ones, we get about 0.072 dB power savings. The other is the code search for the RCCC on level 1 with better convergence thresholds. The convergence threshold of the new RCCC used on level 1 is improved by 0.10 dB. At level 0, a low rate IRA code is used. A low rate Turbo-Hadamard code introduced in [44] might be a possible alternative to our low rate IRA codes.

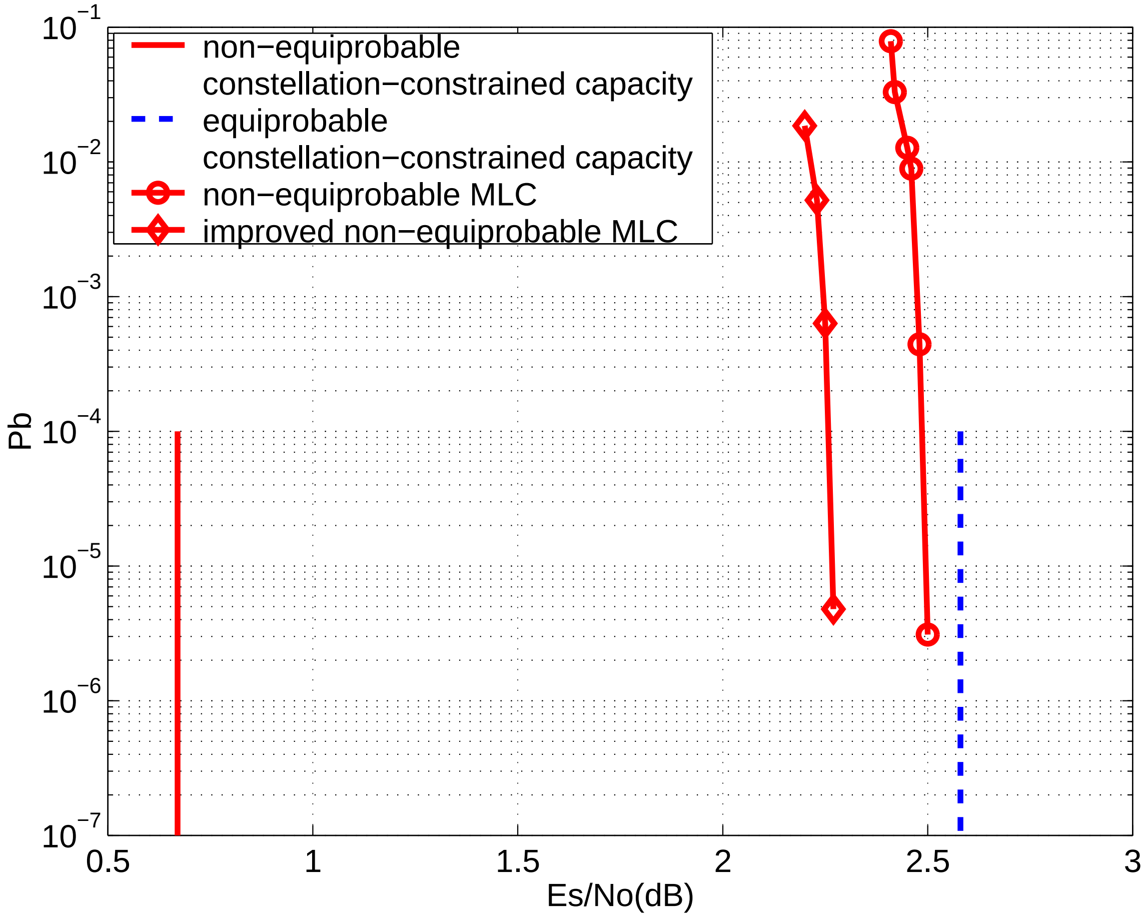


Figure 5.2: Simulation results for 0.5bit/symbol multi-level coding for AWGN channel with Tikhonov phase error for $B_L T = 0.01$

Chapter 6

Conclusion

This dissertation has presented a new multi-level coding architecture for the AWGN channel with Tikhonov phase error. The architecture employs multi-ring PSK constellations with non-equiprobable rings; the ring-radii and probabilities are chosen to closely approximate the capacity-achieving PDF for this channel. To handle the non-equiprobable rings, new serial concatenated binary-input binary-output codes with non-equiprobable output bits have been developed. The performance of the new MLC scheme on the 1 bit/symbol/Hz AWGN/Tikhonov channel with $B_L T = 0.01$ is only 1.1 dB from constellation-constrained capacity, and is significantly better than a comparable equiprobable MLC. At 0.5bit/symbol/Hz, our MLC performance is about 1.58 dB away from constellation capacity, which outperforms the equiprobable constellation-constrained capacity by 0.33 dB. It indicates that no equiprobable code can beat our codes, since our codes' operation points are below the equiprobable constellation-constrained capacity. The non-equiprobable MLC architecture demonstrated in this dissertation has applications not only to AWGN/Tikhonov channel, but also to other important communication channels whose capacity-achieving PDFs are DAUP and can also be approximated by multi-ring PSK.

In this dissertation, another significant work presented is the Gaussian Approximation technique for parallel concatenated turbo-coded modulation which is used to predict the convergence threshold; this technique can be applied to any bit-interleaved Turbo-TCM system.

As for the MLC coding, an important open problem to be solved is to design non-linear non-equiprobable binary codes with algebraic structure and good distance properties, and with low encoding and decoding complexity. This would eliminate the need for lookup tables; we could use much longer and more powerful codes.

Appendix A

PEP Computation Using SED Metric

For the AWGN channel with moderate Tikhonov phase error, e.g. $B_L T = 0.01$, the VA simulation based on the SED metric has a 2 dB loss compared to that of the ML metric. Nonetheless it may be desirable to use the SED decoding metric due to its lower computational complexity, or to ensure backward compatibility with a pre-existing decoder. The PEP based on the SED metric can be considered as the code design criterion on such a channel. In [21], the CLT approximation was used to derive the SED-based PEP for M-PSK constellations, with an upper bound given for the case $N = 1$. The following derivation gives an exact expression when $N = 1$, and a CLT approximation for $N > 1$; both cases are valid for arbitrary constellations.

A.0.1 Exact PEP Computation for Length 1 Sequence

Fig. A.1 is a vector diagram of the transmitted and received complex symbols x and y.

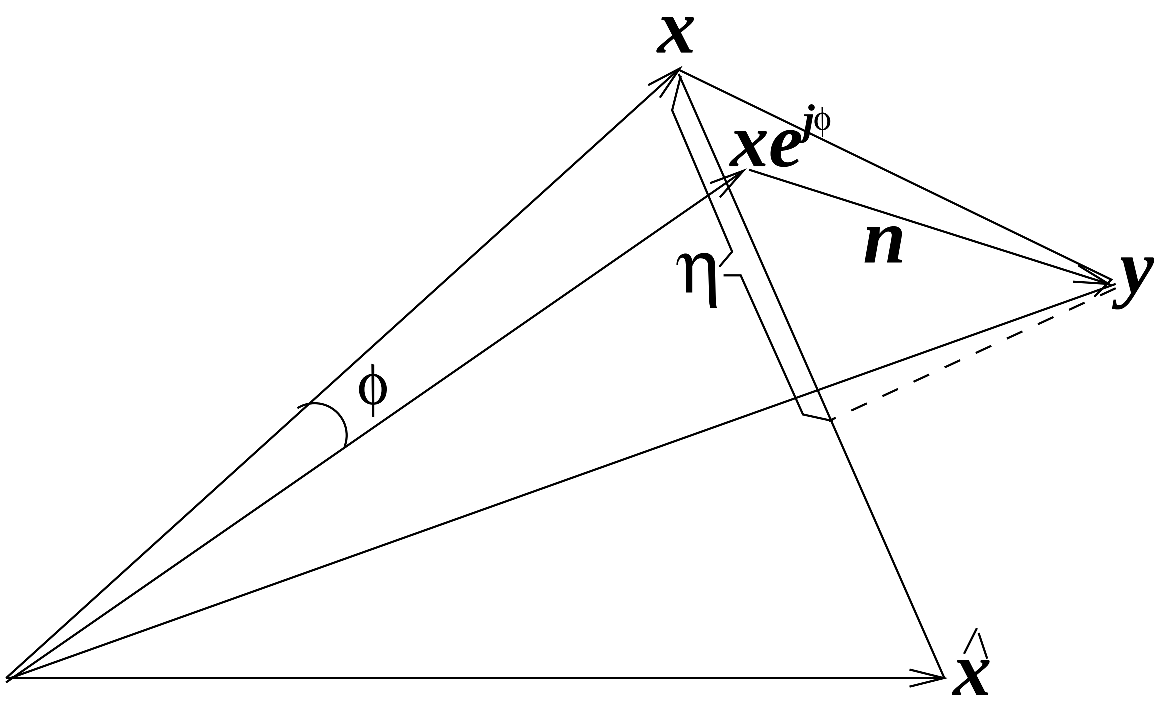


Figure A.1: Vector diagram

Let vector η be

$$\begin{aligned}\eta &= (y-x)\cdot u = (n + xe^{j\phi} - x)\cdot u \\ &= u_1[n_1 + x_1(\cos\phi - 1) - x_2\sin\phi] + u_2[n_2 + x_2(\cos\phi - 1) + x_1\sin\phi],\end{aligned}$$

where $u = u_1 + ju_2$ is the unit vector $u = \frac{\hat{x}-x}{|\hat{x}-x|}$. Conditioned on ϕ, the PEP is

$$p(x \to \hat{x}|\phi) = p(\eta > \frac{|x-\hat{x}|}{2}), \tag{A.1}$$

and η is a Gaussian r.v. with mean μ and variance σ^2, where

$$\mu = E(\eta|\phi) = \frac{\hat{x_1} - x_1}{|\hat{x} - x|}[x_1(\cos\phi - 1) - x_2 \sin\phi] + \frac{\hat{x_2} - x_2}{|\hat{x} - x|}[x_2(\cos\phi - 1) + x_1 \sin\phi], \quad \text{(A.2)}$$

$$\sigma^2 = E[(\eta - \mu)^2] = u_1^2 E[n_1^2] + u_2^2 E[n_2^2] = N_0/2. \quad \text{(A.3)}$$

Then

$$p(x \to \hat{x}|\phi) = \frac{1}{\sqrt{2\pi}\sqrt{\frac{N_0}{2}}} \int_{|\frac{x-\hat{x}}{2}|}^{\infty} \exp\left[-\frac{(\eta - \mu)^2}{N_0}\right] d\eta = Q\left(\frac{|x - \hat{x}|}{\sqrt{2N_0}} - \sqrt{\frac{2}{N_0}}\mu\right), \quad \text{(A.4)}$$

and

$$p(x \to \hat{x}) = \int_{-\pi}^{\pi} Q\left(\frac{|x - \hat{x}|}{\sqrt{2N_0}} - \sqrt{\frac{2}{N_0}}\mu\right) \frac{e^{\rho\cos\phi}}{2\pi I_0(\rho)} d\phi. \quad \text{(A.5)}$$

A.0.2 Exact PEP Computation for Length N Sequences

Let X_N and $\hat{X}_N$ denote two complex symbol sequences of length N. Since the successive ϕ_i and n_i are independent, the PEP is

$$p(X_N \to \hat{X}_N) = \int_{-\pi}^{\pi}\int_{-\pi}^{\pi} \cdots \int_{-\pi}^{\pi} Q\left(\frac{||X_N - \hat{X}_N||}{\sqrt{2N_0}} - \sqrt{\frac{2}{N_0}}\mu\right) \frac{e^{\rho\cos\phi_1}}{2\pi I_0(\rho)} \cdots \frac{e^{\rho\cos\phi_N}}{2\pi I_0(\rho)} d\phi_1 \cdots d\phi_N$$

(A.6)

where

$$\begin{aligned}\mu \;\; &= \;\; \frac{1}{||X_N - \hat{X}_N||}\sum_{i=1}^{N}[(x_{i1}\cos\phi_i - x_{i1} - x_{i2}\sin\phi_i)(\hat{x}_{i1} - x_{i1}) \\ &\quad + (x_{i2}\cos\phi_i - x_{i2} + x_{i1}\sin\phi_i)(\hat{x}_{i2} - x_{i2})]. \qquad \text{(A.7)}\end{aligned}$$

As computation of the N-dimensional integral (A.6) is impractical for large N, we derive a CLT approximation in the next subsection.

A.0.3 Approximate PEP Computation for Length N Sequences Using the Central Limit Theorem

Conditioned on $\phi_1, \phi_2 \cdots \phi_N$, the PEP is

$$p(\eta > d/2|\phi_1, \phi_2 \cdots \phi_N) \qquad \text{(A.8)}$$

where $d = ||X_N - \hat{X}_N||$ and η is the Gaussian r.v. [21]:

$$\begin{aligned}\eta = &\frac{1}{d}\sum_{i=1}^{N}(\hat{x}_{i1} - x_{i1})n_{i1} + \frac{1}{d}\sum_{i=1}^{N}(\hat{x}_{i2} - x_{i2})n_{i2} \\ &+ \frac{1}{d}\sum_{i=1}^{N}(x_{i1}\cos\phi_i - x_{i1} - x_{i2}\sin\phi_i)(\hat{x}_{i1} - x_{i1}) \\ &+ \frac{1}{d}\sum_{i=1}^{N}(x_{i2}\cos\phi_i - x_{i2} + x_{i1}\sin\phi_i)(\hat{x}_{i2} - x_{i2}). \qquad \text{(A.9)}\end{aligned}$$

We note that

$$\frac{d^2}{2} = \frac{1}{2}\sum_{i=1}^{N}\left[(\hat{x}_{i1} - x_{i1})^2 + (\hat{x}_{i2} - x_{i2})^2)\right], \tag{A.10}$$

so that

$$\frac{d}{2} = \frac{1}{2d}\sum_{i=1}^{N}\left[|\hat{x}_i|^2 + |x_i|^2 - 2(\hat{x}_{i1}x_{i1} + \hat{x}_{i2}x_{i2})\right]. \tag{A.11}$$

Rearranging the inequality $\eta > \frac{d}{2}$ by using (A.11) gives:

$$\begin{aligned}&\sum_{i=1}^{N}\frac{\hat{x}_{i1} - x_{i1}}{d}n_{i1} + \sum_{i=1}^{N}\frac{\hat{x}_{i2} - x_{i2}}{d}n_{i2} \\ &> \frac{1}{2d}\{\sum_{i=1}^{N}[|\hat{x}_i|^2 - |x_i|^2] + 2\sum_{i=1}^{N}[(\hat{x}_{i1}x_{i2} - \hat{x}_{i2}x_{i1})\sin\phi_i] \\ &-2\sum_{i=1}^{N}[(\hat{x}_{i1} - x_{i1})x_{i1} + (\hat{x}_{i2} - x_{i2})x_{i2}]\cos\phi_i\}.\end{aligned} \tag{A.12}$$

For simplicity, we rewrite the probability equation as:

$$p(\eta > d/2|\phi_1, \phi_2\cdots\phi_N) = p\left(\eta' > \frac{1}{2d}A\right) = Q\left(\frac{A}{d\sqrt{2N_0}}\right) \tag{A.13}$$

where

$$\eta' = \sum_{i=1}^{N}\frac{\hat{x}_{i1} - x_{i1}}{d}n_{i1} + \sum_{i=1}^{N}\frac{\hat{x}_{i2} - x_{i2}}{d}n_{i2}. \tag{A.14}$$

In (A.13) and (A.14), η' is a Gaussian r.v. with zero mean and variance $\frac{N_0}{2}$ and A is a r.v.

given by:

$$\begin{aligned}
&A = \sum_{i=1}^{N} B_i + 2\sum_{i=1}^{N} C_i \sin\phi_i - 2\sum_{i=1}^{N} D_i \cos\phi_i \\
&B_i = |\hat{x}_i|^2 - |x_i|^2 \\
&C_i = \hat{x}_{i1}x_{i2} - \hat{x}_{i2}x_{i1} \\
&D_i = (\hat{x}_{i1} - x_{i1})x_{i1} + (\hat{x}_{i2} - x_{i2})x_{i2}.
\end{aligned} \tag{A.15}$$

If the error sequence is long enough, the random variable A can be approximated by a Gaussian random variable according to the central limit theorem [21], with mean

$$\begin{aligned}
\mu_A &= \sum_{i=1}^{N} B_i + 2\sum_{i=1}^{N} E[\sin\phi_i]C_i - 2\sum_{i=1}^{N} E[\cos\phi_i]D_i \\
&= \sum_{i=1}^{N} B_i - 2\frac{I_1(\rho)}{I_0(\rho)}\sum_{i=1}^{N} D_i
\end{aligned} \tag{A.16}$$

where $E[\sin\phi_i] = 0$ and $E[\cos\phi_i] = \frac{I_1(\rho)}{I_0(\rho)}$. The variance of A is

$$\begin{aligned}
\sigma_A^2 &= E[(A - \mu_A)^2] \\
&= E\left[\left(2\sum_{i=1}^{N} C_i \sin\phi_i - 2\sum_{i=1}^{N} D_i\cos\phi_i + 2\frac{I_1(\rho)}{I_0(\rho)}\sum_{i=1}^{N} D_i\right)^2\right] \\
&= 4\sum_{i=1}^{N} E[\sin^2\phi_i]C_i^2 + 4\sum_{i=1}^{N} E[\cos^2\phi_i]D_i^2 - 4\frac{I_1^2(\rho)}{I_0^2(\rho)}\sum_{i=1}^{N} D_i^2.
\end{aligned} \tag{A.17}$$

Since A is approximated by a Gaussian r.v. the PEP for length-N sequence is simplified as

$$\begin{aligned} p(X_N \to \hat{X}_N) &= \int_{-\infty}^{\infty} p\left(\eta' > \frac{A}{2d}\right) f(A) dA \\ &= \int_{-\infty}^{\infty} Q\left(\frac{A}{d\sqrt{2N_0}}\right) f(A) dA \end{aligned} \tag{A.18}$$

where $f(A)$ is the Gaussian PDF with μ_A and σ_A^2 given in (A.16) and (A.17).

Bibliography

[1] P. Hou, Capacity and coding/modulation study for partially coherent additive white Gaussian noise channels. PhD thesis, Washington State University, Aug. 2002.

[2] P. Hou, B. J. Belzer and T. R. Fischer, "On the capacity of the partially coherent additive white Gaussian noise channel," in *Proc. 2003 IEEE Int. Symp. on Info. Theory (ISIT 2003)*, Yokohama, Japan, p. 372, June-July 2003.

[3] J. P. McGeehan and A. J. Bateman, "Phase-locked transparent tone-in-band (TTIB): a new spectrum configuration particularly suited to the +transmission of data over SSB mobile radio networks," *IEEE Trans. Commun.*, vol. 32, pp. 81-87, Jan. 1984.

[4] I. C. Abou-Faycal, M.D.Trott and S.Shamai (Shitz), "The capacity of discrete-time memoryless Rayleigh-fading channels," *IEEE Trans. Inform. Theory*, vol. 47, pp. 1290–1301, May 2001.

[5] M. Katz and S. Shamai (Shitz), "One the capacity-achieving distribution of the discrete-time non-coherent and partially-coherent AWGN channels," CCIT report 396, elec. eng., Technion, Israel, Sept. 2002.

[6] P. Hou, B.J. Belzer, and T.R. Fischer, “Shaping gain of the partially coherent additive white Gaussian noise channel,” *IEEE Communications Letters*, vol. 6, pp. 175–177, May 2002.

[7] G. Ungerboeck, and I. Csajka, “On improving data-link performance by increasing channel alphabet and introducing sequence coding,” in *Proc. Int. Symp. Information Theory (ISIT)*, (Ronneby, Sweden, June 1976).

[8] G. Ungerboeck, “Channel coding with multilevel/phase signals.”*IEEE Trans. Inform. Theory*, vol. IT-28, pp. 55–67, Jan 1982.

[9] H. Imai and S. Hirakawa, “A new multilevel coding method using error correction codes,” *IEEE Trans. Inform. Theory*, vol. IT-23, pp. 371–377, May 1977.

[10] U. Wachsmann, R. F. H Fischer, and J. B. Hubber, “Multilevel Codes: Theoretical Concepts and Practical Design Rules,” *IEEE Trans. Inform. Theory*, vol. 45, No. 5, pp. 1361–1391, July 1999.

[11] A. G. Amat, G. Montorsi, and S. Benedetto, “A new approach to the construction of high-rate convolutional codes,” *IEEE Communications Letters*, vol. 5, No. 11, pp. 453–455, November 2001.

[12] A.G. Amat, G, Montorsi, and S. Benedetto, “Design and decoding of optimal high-rate convolutional codes,” *IEEE Trans. Info. Theory*, vol. 50, pp. 867-881, May 2004.

[13] J. G. Smith, “The information capacity of amplitude and variance constrained scalar Gaussian channels,” *Inform. Contr.*, vol. 18, pp. 203-219, 1971.

[14] I. C. Abou-Faycal, M. D. Trott, and S. Shamai (Shitz), “The capacity of discrete-time memoryless Rayleigh-fading channels,” *IEEE Trans. Inform. Theory,* vol. 47, pp. 1290-1301, May 2001.

[15] M. Katz and S. Shamai (Shitz), “On the capacity-achieving distribution of the discrete-time noncoherent and partially coherent AWGN channels,” *IEEE Trans. Inform. Theory,* vol. 50, pp. 2257-2270, Oct. 2004.

[16] M. C. Gursoy, H. V. Poor, and S. Verdú, “On the capacity-achieving distribution of the noncoherent Rician fading channel,” in *Proc. 2003 Canadian Workshop on Info. Theory,* Waterloo, ON, Canada, May 18-21, 2003.

[17] M. C. Gursoy, H. V. Poor, and S. Verdú, “The noncoherent Rician fading channel Part I: Structure of the capacity achieving input,” *IEEE Trans. Wireless Commun.*, to be published.

[18] A. J. Viterbi, *Principles of Coherent Communication*. New York: McGraw-Hill, 1966.

[19] G. J. Foschini, R. D. Gitlin, and S. B. Weinstein, “On the selection of a two-dimensional signal constellation in the presence of phase jitter and Gaussian noise,” *Bell Syst. Tech. Jour.*, vol. 52, pp. 927–965, Jul./Aug. 1973.

[20] E. Biglieri, D. Divsalar, P. J. McLane, and M. K. Simon, *Introduction to Trellis-Coded Modulation with Applications,* New York: Macmillan, 1991.

[21] B. Vucetic and J. Du, “The effects of phase noise on trellis coded modulation over Gaussian and fading channels,” *IEEE Trans. Commun.*, vol. 43, pp. 252–260, Feb./Mar./Apr. 1995.

[22] J. G. Proakis, *Digital Communications,* 4th ed. Boston: McGraw-Hill, 2001.

[23] H. El Gamal and A. R. Hammons, Jr., “Analyzing the turbo decoder using the Gaussian approximation,” *IEEE Trans. Inform. Theory,* vol. 47, pp. 671–686, Feb. 2001.

[24] Y. Zhu and B. J. Belzer, "New Results for the Pairwise Error Probability on the AWGN Channel with Tikhonov Phase, with Applications to Coded Modulation," invited paper in *Proc. 2002 IEEE Symposium on Advances in Wireless Communications (ISWC’02),* pp. 93-94, Victoria BC, Canada, Sept. 2002.

[25] U. Wachsmann, R. F. H Fischer, and J. B. Hubber, “Multilevel Codes: Theoretical Concepts and Practical Design Rules,” *IEEE Trans. Inform. Theory*, vol. 45, No. 5, pp. 1361–1391, July 1999.

[26] H. Jin, A. Khandekarm and R.J. Mceliece, “Irregular Repeat-Accumulate Codes,” *presented at the Second International Conference On Turbo Codes*, Brest, France, pp. 1–8, Sept. 2000.

[27] D. Divsalar, H. Jin, and R.J. Mceliece, “Coding theorems for ’turbo-like’ codes,” *Proc. 36th Allerton Conf. on Communication, Control and Computing*, pp. 201–210, Allerton, Illinois, Sept. 1998.

[28] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, “Serial Concatenation of Interleaved Codes: Performance Analysis, Design, and Iterative Decoding,” *IEEE Trans. Inform. Theory*, vol. 44, NO. 3, pp. 909–926, May 1998.

[29] A.R. Calderbank and L.H. Ozarow, “Nonequiprobable signaling on the Gaussian channel,” *IEEE Trans. Inform. Theory*, vol. 36, No. 7, pp. 726–740, July 1990.

[30] G.D. Forney, “Trellis shaping,” *IEEE Trans. Inform. Theory*, vol. 38, No. 2, pp. 281–300, March 1992.

[31] G.D. Forney, Jr. and L.F. Wei, “Multidimensional constellations – Part I: Introduction, figures of merit, and generalized cross constellations,” *IEEE J. Select. Areas Commun.*, vol. 7, pp. 877–892, Aug. 1989.

[32] G.D. Forney, Jr., “Multidimensional constellations – Part II: Voronoi constellations,” *IEEE J. Select. Areas Commun.*, vol. 7, pp. 941–958, Aug. 1989.

[33] R. Laroia, N. Farvardin, and S.A. Tretter, “On optimal shaping of multidimensional constellations,” *IEEE Trans. Inform. Theory*, vol. 40, No. 4, pp. 1044–1056, July 1994.

[34] Y. Zhu, L. Ni, and B.J. Belzer, “Design of turbo coded modulation for the AWGN channel with Tikhonov phase error,” *IEEE Trans. Commun* , vol. 53, No. 10, pp. 1697–1707, October 2005.

[35] R. Nuriyev and A. Anastasopoulos, “Capacity and coding for the block-independent Noncoherent AWGN channel,” *IEEE Trans. Inform. Theory*, vol. 51, No. 3, pp. 866–883, March 2005.

[36] R.G. Gallager, “Low-Density Parity-Check Codes,” *Cambridge, MA: MIT Press*, 1963.

[37] T.J. Richardson, M.A. Shokrollahi and R.L. Urbanke, “Design of Capacity-Approaching Irregular Low-Density Parity-Check Codes,” *IEEE Trans. Inform. Theory*, vol. 47, No. 2, pp. 619–637, Feb. 2001.

[38] S.-Y. Chung, T.J. Richardson and R.L. Urbanke, “Analysis of Sum-Product Decoding of Low-Density Parity-Check Codes Using a Gaussian Approximation,” *IEEE Trans. Inform. Theory*, vol. 47, No. 2, pp. 657–670, Feb. 2001.

[39] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, “Parallel concatenated trellis coded modulation,” in *Proc. IEEE International Conf. Commun.*, pp. 974–978, June 1996.

[40] P. Robertson and T. Wörz, “Bandwidth-efficient turbo trellis-coded modulation using punctured component codes,” *IEEE Jour. on Sel. Areas in Commun.*, vol. 16, no. 2, pp. 206-218, Feb. 1998.

[41] C. Fragouli and R. D. Wesel, “Turbo-encoder design for symbol-interleaved parallel concatenated trellis-coded modulation,” *IEEE Trans. Commun.*, vol. 49, pp. 425–435, March 2001.

[42] D. Divsalar and F. Pollara, “Turbo codes for PCS applications,” *Proc. IEEE Conf. Communications*, vol. 1, pp. 54–59, June 1995.

[43] S. Riedel, “MAP Decoding of Convolutional Codes Using Reciprocal Dual Codes,” *IEEE Trans. Inform. Theory*, vol. 44, pp. 1176–1186, May 1998.

[44] L. Ping, W.K. Leung and K.Y. Wu, “Low-Rate Turbo-Hadamard Codes,” *IEEE Trans. Inform. Theory*, vol. 49, pp. 3213–3223, December 2003.