2011

Advanced digital and analog error correction codes

Kai Xie
Lehigh University

Advanced digital and analog error correction codes

by

Kai Xie

Presented to the Graduate and Research Committee
of Lehigh University
in Candidacy for the Degree of
Doctor of Philosophy

in

Electrical Engineering
Lehigh University

May 2011

Approved and recommended for acceptance as a dissertation in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

_____

Date

_____

Dissertation Advisor

_____

Accepted Date

Committee Members:

_____

Tiffany Jing Li

_____

Zhiyuan Yan

_____

Shalinee Kishore

_____

Garth Isaak

_____

Erich F. Haratsch

Acknowledgements

I would like to express my profound gratitude to my supervisor Prof. Tiffany Jing Li for her support since the beginning, and for her patience and guidance. I am especially grateful to my committee members, Prof. Zhiyuan Yan, Prof. Shalinee Kishore, Prof. Garth Isaak and Dr. Erich F. Haratsch for their support, suggestion and review of my proposal and dissertation. Without their continuous encouragement and advice, I could never complete this thesis.

I would like to thank all my friends and colleagues, especially Meng Yu, Ruiyuan Hu, Xingkai Bao, Peiyu Tan, Hend Alqamzi, Nattakan Puttarak, Phisan Kaewprapha, Yan Li, Yongmei Dai, Gang Xiong and Min Xiao with whom I have worked together, for their helpful discussions and friendship. I would like to thank my parents, my in-laws, and my wife for their love, understanding and support.

Contents

List of Figures

Abstract

Practical communication channels are inevitably subject to noise uncertainty, interference, and/or other channel impairments. The essential technology to enable a reliable communication over an unreliable physical channel is termed as channel coding or error correction coding(ECC).

The profound concept that underpins channel coding is *distance expansion*. That is, a set of elements in some space having small distances among them are mapped to another set of elements in possibly a different space with larger distances among the elements. Distance expansion in terms of digital error correction has been a common practice, but the principle is by no means limited to the discrete domain. In a broader context, a channel code may be mapping elements in an *analog source space* to elements in an *analog code space*. As long as a similar distance expansion condition is satisfied, the code space is expected to provide an improved level of distortion tolerance than the original source space. For example, one may treat the combination of *quantization*, *digital coding* and *modulation* as a single nonlinear analog code that maps real-valued sources to complex-valued coded symbols.

Such a concept, thereafter referred to as *analog error correction coding* (AECC), *analog channel coding*, or, simply, *analog coding*, presents a generalization to digital error correction coding (DECC). This dissertation investigates several intriguing aspects of DECC and especially of AECC.

The research of DECC focuses on turbo codes and low-density-parity-check (LDPC) codes, two of the best performing codes known to date. In the topic of

turbo codes, this dissertation studies on interleaver design, which plays an important role in the overall performance of turbo codes (at small to medium code lengths) but does not affect the decoding architecture. Before this work, the theoretical foundation of interleaver design and evaluation were rather incomplete, e.g. efficient approaches in measuring "randomness" (one of the most important characteristics for interleavers) were rigorously established. This work proposes two powerful metrics, cycle correlation sum (CCS) and variance of the second order spread spectrum (VSSS), to quantify spread and randomness, two fundamental properties of interleavers, while accounting for the iterative nature of turbo decoding and the weight spectrum of turbo encoding. We evaluate the ensemble of algebraic interleavers, propose design approaches specific to coprime interleavers, a subclass of algebraic interleavers, and provide theoretical insights on selecting parameters. Simulation results show superior performance of the newly designed coprime interleavers to the existing ones.

The second topic analyzes the Gaussian assumption for the stochastic analysis in iterative decoding. Gaussian distribution is widely believed to match the real message density in analyzing iterative decoding, but the justification is largely pragmatic, except for the messages directly coming from Gaussian channels. This work investigates when and how well the Gaussian distribution approximates the real message density and why. We show that the Gaussian assumption is statistically sound when the LLRs extracted from the channel are reasonably reliable to start with, and when the check node degrees of the LDPC code are not very high; but the assumption is much less accurate when one or both conditions are violated. Extensive simulation results are provided to exemplify and verify this discussion.

Besides these topics on digital coding, this dissertation also investigates analog coding, which brings the benefit of avoiding quantization errors for real-world analog sources and hence presents a very promising direction in error-correction coding. As a recent emerging topic, the analysis and understanding of analog coding is far from mature. We categorize analog codes into two classes, summarize the existing analog codes and propose a few new codes.

For linear analog codes, this work initiates some fundamental concepts, defines analytical metrics and theorems, develops the achievable upper and lower bounds, and identifies several classes of linear analog codes that could achieve these bounds. For nonlinear analog codes, we focus on a special type that makes essential use of nonlinear chaotic functions. We develop turbo-like coding structures for chaotic analog codes, and show that they can easily beat the performance bound of linear analog codes. In this, we propose a conjecture that while linear codes are sufficient for digital coding, they are not for analog codes, and nonlinear analog codes, such as those based on chaotic functions, must be used in order to effectively combat error on Gaussian channels.

Chapter 1

Introduction

Essential to reliable communication and storage is the technology of *error correction coding*, which targets correcting errors (deviating from what is true) caused by noise and distortion coming from channels and devices. In reality, not only do communication systems, but people perform error correction every day. For example, we can read from hand-written papers even though everyone's handwriting is different. Our brain will tolerate the deviates between the hand-written alphabet and the standard printed alphabet. When the deviation is within a certain range, our brain will find the most similar and likely letter in the alphabet automatically, which is also the basic operation performed by error correction in wireless communication and storage systems. Similar to the alphabet and letter, a signal space $\mathbb{S}$ must be defined for any system.

Based on the pre-learned knowledge, our brain can effectively find the most likely letter and ignore the deviations. But how to define the "most likelihood" in a system? The concept of "distance" is introduced, and will take different flavors

in variant systems, such as the Hamming distance and the Euclidean distance. During error correction, the system may search the entire signal space $\mathbb{S}$ to find the most likely signal with the smallest distance to the perceived signal. When the deviation is less than a certain threshold, the signal can usually be recovered correctly. The error correction capability thus heavily depend on the distance spectrum, especially the minimum distance among the signals within the space $\mathbb{S}$.

The basic and profound idea behind error correction coding is *distance expansion*, which is also known as *space expansion*. A source signal space $\mathbb{S}$ with small distance among elements will be mapped to a signal space $\mathbb{C}$ with larger distances, termed the *code book*, by adding redundancy. Each element in the code book is termed a *codeword*. Consider, for example, an (n, k, d) binary systematic channel code that encodes source sequences $\mathbf{u} \in \{\mathbf{0}, \mathbf{1}\}^{\mathbf{k}}$ to codewords $\mathbf{c}(\mathbf{u}) \in \{\mathbf{0}, \mathbf{1}\}^{\mathbf{n}}$. A source sequence with neighbors that are only 1 Hamming distance away is now mapped to a codeword whose nearest neighbors are at least d Hamming distance away, thus enabling the detection of up to $(d-1)$ bit errors or correction of up to $\lfloor (d-1)/2 \rfloor$ bit errors. The code rate, defined as $r = k/n$, denotes the payload of the channel code.

Two critical problems running through the studies of error correction coding are:

1. How to construct a code book with a good distance spectrum?
2. Given a received signal, how to effectively find the closest signal in the codebook?

Answers to these two questions remain illusive until the discovery of *turbo codes*

in 1993 and the rediscovery of *low density parity check codes (LDPC)* codes in 1999. They bridged the afore-considered insurmountable theory-practice gap between the practical error correction performance and the Shannon limit [1]. Turbo codes and LDPC codes also revolutionized the coding research with new concepts for successful error correction: a paradigm of building long codes with random construction, and decoding them using soft, iterative decoders with manageable complexity.

However, the random property of the codebook and the nonlinear iterative characteristics of the decoding process are a double-edged sword. On one hand, they enable the remarkable performance of turbo codes and LDPC codes; on the other hand, they also make design and analysis difficult. Many traditional analytical methods become inefficient or inapplicable. Although a theory of an iterative analysis and ensemble analysis is being developed, several questions remain open.

This research is dedicated to the study of advanced error correction technology, in the digital domain as well as in the analog domain. Specific focus will be set on the design, analysis and evaluation of the state-of-the-art and the emerging coding schemes.

The first two chapters cover digital coding, and discuss specific design issues for turbo codes and LDPC codes, the two classes of best-performing codes known to date. The remainder chapter investigates ideas and concepts in analog coding, and explore new ways of error correction.

1.1 Interleaver design for turbo codes

We first study the interleaver design issue for turbo codes. The interleaver, being a critical component of turbo codes, affects both the order of input sequences and the exchange of extrinsic information. It plays two roles in turbo codes: at the encoder side, it makes the constituent encoders work on the same set of information bits but in different orders, which in turn provides a good distance spectrum; at the decoder, it decorreclates the exchanged information, allowing an iterative decoder to approximate the performance of an maximal likelihood (ML) decoder. Therefore, interleaver design has been an interesting research pursuit that spans much of turbo codes's short history.

Intuitively, a good interleaver should process two properties:

1. Spread: two or multiple bits close to each other before interleaving should be separated far apart after interleaving;

2. Randomness: the scrambling rule should not have any apparent or repeatitive patterns.

These basic properties of interleaver have guided the design of good interleavers. However, the theory behind the design criteria is still not complete. Some questions are still open. For instance, how do these two strategies affect and get reflected in the coding performance? How to quantify these two properties, especially the randomness? How to use them to design a good algebraic interleaver? Chapter 2 of this dissertation targets solving at least some of these problems.

Another design challenge for interleaver is how to design interleavers equipped with deterministic formats yet preserve a random characteristic. Generally, large random interleavers deliver good performance, but in practice short deterministic interleavers are preferred over large random ones due to storage and operational concerns. For example, algebraic interleavers are highly desirable, because they can be generated on-the-fly; the system only needs to store a few parameters; and reasonable randomness is exhibited in the interleaving patterns. Coarsely speaking, an algebraic interleaver is an interleaver whose scrambling pattern is completely specified by a well-defined mathematical formula with a few seeding parameters. Additional design difficulty also comes from the decoding perspective, namely, most of the interleaved and concatenated codes use a suboptimal iterative decoding algorithm rather than theoretically optimal maximal likelihood (ML) decoding due to complexity concern.

Taking into consideration of randomness in interleaving pattern, deterministic formats and suboptimal decoding, we propose to first investigate efficient ways to evaluate interleavers in turbo codes. Two simple and powerful metrics, cycle correlation sum (CCS) and variance of the second order spread spectrum (VSSS), are proposed to quantify the spread factor and the randomness factor, and to further measure the relative quality of interleavers and guide the interleaver design. The CCS metric accounts for the iterative nature of the message flow in a turbo decoder and evaluates the impact of interleaver design on the decoder optimization. The VSSS takes into explicit consideration of quantifying the randomness for different interleavers and attempts to build the connection between randomness and performance of an interleaver. These two metrics make it possible to evaluate the performance of an interleaver without lengthy simulation, which, in turn, leads to

good interleaver design guidance. Based on these two design tools, we reevaluate a rich class of algebraic interleavers, the coprime interleavers. Simulation results show that the new coprime interleaver design rules can in general improve the performance while saving the complexity and storage memories.

1.2 Gaussian assumption for LDPC codes

The Chapter 3 of this research evaluates the Gaussian assumption that is used in the iterative decoding of LDPC codes.

Toward a deep theoretic understanding of soft iterative decoding, researchers have conducted active analysis. A soft iterative decoder generally consists of several component soft decoders connected in a parallel, serial or hybrid fashion, passing probabilistic message along the connecting edges between the component decoders. Message-passing algorithm, for which the *a posterior* probability decoding for turbo codes is a specialization, forms the majority of soft iterative decoding mechanisms. Since almost all the message-passing decoders are high-dimensional nonlinear mapping, analysis using conventional methods (such as those based on the codeword space) appears ineffective. On the other hand, stochastic approaches offer a rich source for analyzing the properties of iterative decoding, enabling the modeling of the input and output of a soft decoder as random processes and the tracking of the evolution of their statistic characteristics through iterations. Density evolution (DE), proposed by Richardson *et al* in [2], was one of the pioneering stochastic methods to investigate the convergence behavior for iterative decoding. Density evolution, when applied to code graphs with asymptotically unbounded

girth, can compute thresholds for the performance of LDPC codes and turbo codes with iterative decoding, but tracking the probability density function (pdf) of the messages involves infinite dimensional algebra, and is therefore computationally prohibitive.

To simplify the analysis, researchers started to look into the widely-adapted Gaussian model. Wiberg [3] first demonstrated that the pdf of the extrinsic information (exchanged between component decoders) may be approximated by a Gaussian distribution. This discovery significantly simplified the stochastic analysis, since a Gaussian distribution can be completely characterized by its mean and variance. Following this approximation, [4] succeeded in estimating the thresholds for both regular and irregular LDPC codes. At the same time, [5] showed that the pdf of the extrinsic information in message-passing decoding satisfies and preserves a *symmetry condition*. Realizing that a probabilistic density that is both "symmetric" and Gaussian distributed satisfies $\sigma^2 = 2m$, where m and σ^2 are the mean and the variance of the Gaussian distribution, researchers were able to further simplify the analysis by using a single parameter, either the mean or the variance of the message density, to track down the probabilistic evolution.

As an alternative analysis approach, extrinsic information transfer (EXIT) charts was proposed in [6] to visualize the behavior of an iterative decoder, and especially the evolution of the extrinsic information exchanged between different computational units during the iterative decoding. At its proposition, EXIT charts were considered an effective tool, but one providing not much more knowledge than visualizing the repeated application of the density evolution algorithm with different channel signal-to-noise ratios (SNR) and at different stages of iterative

decoding.

Both EXIT charts and their underlying tool of density evolution make essential use of a prevailing Gaussian assumption, which states that the log-likelihood ratio (LLR) messages exchanged between different component decoders at an arbitrary stage of iterative decoding follow a Gaussian distribution. However, the justification of this assumption is largely pragmatic rather than demonstrated over any rigorous theory. Since this philosophy has shaped the analysis of the iterative decoding for both turbo and LDPC codes, it would be of great importance to provide some statistical analysis on its accuracy. In Chapter 2, we provide a statistical justification for LDPC codes, and [7] provided an analysis on Gaussian assumption for turbo codes.

1.3 Analog error correction codes

While channel coding has been, for much of the decades' long history of modern communications, almost exclusively regarded as a digital-only technology, the principle of space mapping and distance expansion is not intrinsically labeled with "digital only" and needs not be confined in the domain of digital coding.

Analog coding is another possibility and will likely bring huge benefits in certain scenarios. Many the raw signals we obtain from the natural world are analog, such as light and sound. Analog coding allows us to work directly on analog sources without the burden of quantization and filtering. It also avoids the unrecoverable granularity noise caused by quantization.

Analog error correction codes have been considered for solving the peak-to-average-ratio problem in orthogonal frequency division multiplexing (OFDM) schemes, for adding fault-tolerance to massive computation systems [8], for transmitting images and video streams across wireless channels [9], and for joint source-channel coding [10]. However, comparing to the high level of maturity of digital error correction coding in both the theoretical and the practical context, the research of analog coding is still much incomplete [11], [12].

Most existing works on analog coding are isolated and investigate analog codes as straight-forward extensions of digital codes. However, a comprehensive study was not yet shown. This work is the first to systematically structure the analysis and design of analog codes. We develop several new concepts for analyzing and understanding analog codes, including the encoding power gain, average distance/weight ratio and its achievable upper and lower bounds. We also define maximum distance ratio expansible (MDRE) codes, a class of codes similar in spirit to maximum distance separable (MDS) codes in digital coding, and prove that they could achieve those bounds. We also generalize the concept of union bound to analog codes, and show that it is an effective indicator to the performance of analog codes. We also classify analog codes into linear analog codes and non-linear analog codes. Besides, we demonstrate several new codes, and analyze important properties of these codes using our newly-developed concepts and tools.

The first example of nonlinear analog codes was constructed by Chen and Wornell [11] referred to as tent map codes. These codes are based on the chaos theory and exhibit an elegant property of distance expansion similar to its digital counterpart.

Chaos is a universal phenomenon found in a wide spectrum of natural phenomena and nonlinear systems. Prominent features of chaos include nonlinearity, topological mixing and sensitivity to initial conditions. The latter is popularly known as the "butterfly phenomenon" due to a 1972 paper by Lorenz entitled "*Predictability: Does the Flap of a Butterfly's Wings in Brazil Set Off a Tornado in Texas?*" [13]. While the non-periodic, random and fast-diverging evolution of chaotic states are typically viewed as penalty to a system, these same features may be exploited to serve good purposes.

Chen and Wornell explored a natural way of building a chaotic code: the real-valued information, or, the systematic symbol is fed to the chaotic map as the initial state, and a few subsequent states are treated as parity symbols to protect the systematic symbol [11]. In [11], the chaotic map was specified as *tent* map. Several chaotic estimation techniques were developed for decoding the tent map codes, including the maximum-likelihood (ML) based detector [11] [14], the expectation-maximization (EM) algorithm [15], the Bayesian approach [16], and dynamic programming [17].

Chaotic analog coding has a promising potential to be applied in a secure communication system, since variant parameters may be exploited as the secure keys to lead to drastically different encoded sequences. Another major advantage of using chaotic signals in communications is its low-cost implementation. Many chaotic signals, including the popular *tent* map, can be generated by simple electric circuits [18]. Chaotic coding offers additional advantage to analogue sources (such as transmission and recoding of music), since it is free from granularity or source quantization errors.

Although the tent map codes have exhibited interesting properties, our analysis shows that the performance of tent map codes is adversely limited by the unbalanced protection of the sign sequence as well as the short code length.

To avoid unbalanced protection, we propose a more sophisticated chaotic coding strategy that borrows useful ideas from turbo codes. Turbo codes, the renowned class of digital error correction codes that were the first to exhibit performance close to the channel capacity, have enlightened the coding research with several new concepts. One notable feature of turbo codes, for example, is the parallel concatenation of two recursive systematic convolutional (RSC) codes, such that the chance of both component codes producing low-weight codewords is rather small. This ensures that the low-weight codewords or, equivalently, small-distance codeword pairs are scant (the so-called "spectrum thinning" effect). Exploring a similar idea, we propose *chaotic analog turbo* (CAT) codes through the parallel concatenation of two tent maps. Our new codes are to the concatenated turbo codes, as tent map codes are to the individual convolutional codes. The specific concatenation structure will be discussed in detail in Chapter 5. The maximum likelihood decoding algorithm and iterative decoding will be presented. Simulation tests are carried out, and it is shown that the proposed CAT codes noticeably outperform tent map codes of the same code rate. It is also interesting to note that CAT codes can outperform some conventional digital communication schemes, such as BPSK modulation and repetition codes. It should also be pointed out that there is sharp difference between our approach and other notions of chaotic turbo codes in the previous literature [19], [20]. Ours is analog codes, but both [19] and [20] are digital codes.

Next we extend the code length of tent map codes by developing a 2-dimensional chaotic code based on the baker's map. The less-than-desirable performance of the tent map code [11] may be attributed, in part, to the low dimensionality of the underlying chaotic system: the tent map is a 1-dimensional nonlinear function with a scalar input and offers relatively simple relation between the time-evolving states. The CAT code [21] strengthens the inter-state relation by concatenating two tent maps, thus creating a higher level of protection. In Chapter 5, we propose to exploit useful 2-dimensional chaotic systems to construct good chaotic analog codes.

Leveraging rich literature of the chaos theory, we identify the *baker's map*, a 2-dimensional nonlinear function from a unit square to itself, as a desirable candidate. We demonstrate how to apply the baker's map to the tent map to achieve 2-dimensional chaotic coding. Realizing its uneven error protection capability, we further propose a mirrored replication structure to improve the code performance. Unlike the tent map that has many available detection algorithms (such as [15] [16] [17]), the baker's map has hardly any that is suitable for decoding purpose. Hence we also develop a maximum likelihood decoding algorithm. The resultant code, termed the *baker's map code*, successfully strikes a good balance between performance and complexity. Additionally, a comparison between the baker's map code and digital codes (including the convolutional code and the turbo code) reveals a surprisingly good performance achieved by the baker's map code, which is, in some cases, comparable to or better than digital systems.

Chapter 2

Interleaver Design of Turbo Codes

Turbo codes are high performance codes and have found wide utilization in storage and communication systems, including magnetic recording systems, optical communications, digital video broadcasting, space exploring systems and cellular networks. Turbo codes are claimed to achieve near Shannon-limit error correction performance with relatively simple component codes (usually convolutional codes concatenated in a serial or parallel fashion) and large interleavers. Interleavers are essential to the overall performance of turbo codes, since a good interleaver can lead to a lower error floor and earlier decoder convergence, and does not very much affect the structure of the decoder design. In the case of parallel concatenation, the two constituent encoders are working on the same set of information bits but in different bit orders. In other words, when a sequence produces a low-weight output at one constituent encoder, its scrambled counter part will most likely produce a high-weight output on the other. Hence the overall codeword, combined from both outputs, will have a decent weight with a high probability. As for decoding,

interleavers break up error bursts and de-correlate the reliability information exchanged between the two component decoders, such de-correlation warrants the efficiency of the iterative decoding algorithm and narrows the performance gap between iterative decoding and the optimal maximum likelihood decoding.

These two effects of interleavers categorize the design strategies into two groups: the *distance spectrum* criterion and the *effective decoding* criterion. The distance spectrum criterion aims at a large effective free distance $d_{eff,free}$ (for turbo codes) as well as a small multiplicity by mapping the bad pattern which yields a low-weight output at one component code to a good pattern at another component code. In this sense, a general rule is maximizing the minimum spread. However, the spread is not the single most important factor that affects the performance. For example, a row-column interleaver typically has a larger minimum spread than a random interleaver, and may exhibit a better performance at short lengths of no more than a few hundred bits. However, as the length increases to a few thousand bits, its performance may drop noticeably below that of the random interleaver. The reason is that the repetition character of a row-column interleaver increases the multiplicity of its small free distance. Hence, although its minimum free distance may be larger than that of the random interleaver, it may still not perform well, especially at large code lengths. Therefore, *randomness* and *spread* are both critical to the performance of interleavers.

Several metrics have been created to evaluate the spread factor. For example, the *minimum spread criterion*: An interleaver is said to have a minimum spread of S_p if any two bits within a distance of S_p are mapped to two positions that are at leat S_p apart. Crozier relaxed the definition of spread by noting the sum of the

distances between two bit positions before and after interleaving [22], namely, the spread of a bit pair i and j is given by $S_{i,j} = |i - j| + |\pi(i) - \pi(j)|$. However, the minimum spread criterion does not show the whole picture of spread. Suppose we have a good interleaver $\pi(i)$ with length N. Without loss of generality, assume $\pi(0) = 0$. Then we can create a new interleaver $\pi'(i)$ with length $N+1$, by letting

$$\begin{cases} \pi'(i+1) &= \pi(i) \qquad i \neq 0, \\ \pi'(0) &= 0; \end{cases} \tag{2.1}$$

Since the minimum spread of the new interleaver $\pi'(i)$ is 1, it may follow that the minimum spread factor criterion that this interleaver is among the worst. However, as it inherits the majority of the scrambling pattern from the good interleaver, it will perform decently for the most of the time. This simple example illustrates how insufficient the minimum spread criterion is in characterizing the spread factor of an interleaver.

Randomness is another critical factor that affects the performance. For example, a row-column interleaver typically has a larger minimum spread than a random interleaver, and may exhibit a better performance at short lengths of a few hundred bits. However, its performance may drop noticeably below that of the random interleaver as the length increases to a few thousand bits. The connection between the randomness and the performance is not well understood. We do not even have an effective way to quantify the randomness. Since aiming at various targets in different situations, the existing random testing methods are not suitable for use in the interleaver design.

This chapter investigate efficient ways to evaluate interleavers in turbo codes. We proposed two powerful metrics: cycle correlation sum (CCS) quantifies the spread factor, and variance of the second order spread spectrum (VSSS) quantifies the randomness factor. The CCS metric accounts for the iterative nature of the message flow in a turbo decoder and evaluates the impact of interleaver design on the decoder optimization. The VSSS takes into explicit consideration of quantifying the randomness for different interleavers and attempts to build the connection between randomness and performance of interleavers. These two metrics give a comprehensive evaluation to the performance of interleaver in turbo codes, also they make it possible to predict the performance without lengthy simulation, hence are further utilized to guide the interleaver design. The relations behind the generation parameters and the design metrics (CCS and VSSS) are analyzed and proved, and simulation results demonstrated that the new design rules for coprime interleaver improve the performance.

2.1 Typical Interleavers

A length-N interleaver is a single-input single-output device that provides a one-to-one mapping of an alphabet set $A \equiv \{0, 1, ..., N-1\}$ to itself. Let π and π^{-1} denote interleaving and its reverse operation (known as de-interleaving). We say position i is interleaved to position j if

$$\pi(i) = j, \qquad i, j \in A \tag{2.2}$$

$$\text{or} \qquad \pi^{-1}(j) = i, \qquad i, j \in A \tag{2.3}$$

A matrix interleaver, or a row-column interleaver with parameters $M(p \times q, N)$, formats the N input data bits in a matrix of p rows and q columns. The data are written in along the rows and read out along the columns.

A S-random interleaver [23] donated by $S(s = w, N)$, is a randomly generated interleaver which guarantees the minimum spread is at least w, where N is the interleaver length.

Random interleavers, and especially S-random interleavers, generally perform better than row-column interleavers, but the need to store the entire scrambling pattern makes their application costly, especially in systems that have limited storage, but require the use of an exceptionally long code or the support of a few different code lengths. Algebraic interleavers, on the other hand, can be generated on-the-fly using well-defined algebraic formula with only a few seeding parameters. For example, the coprime interleavers are generated by only two parameters. Below we review a few useful classes of algebraic interleavers and coprime interleavers.

A coprime interleaver donated by $C(a, b, N)$, is a structured interleaver whose interleaving pattern is defined recursively as [24]:

$$\begin{cases} \pi(0) = 0; \\ \pi(i) = \mathrm{mod}\big(a\pi(i-1)+b\,,\, N\big), \quad i=1, 2, ...N-1, \end{cases} \tag{2.4}$$

where N is the interleaver length, $\pi(i)$ is the new position to which indice i should be scrambled, and $\mathrm{mod}(x, N)$ denotes the modulo N arithmetic. The seeding parameters a and b need to satisfy the following set of rules to ensure one-to-one mapping:

1. $0 < a < N$, $0 \leq b < N$, and b be relatively prime to N;

2. $(a-1)$ be a multiple of c, for every prime c dividing N;

3. $(a-1)$ be a multiple of 4, if N is a multiple of 4.

Since the value of the starting point $\pi(0)$ has little impact on the interleaving performance, we have set it to 0 in (2.4) for convenience.

The recursion in (2.4) imposes a constraint for sequential implementation which may cause a long delay. An alternative form expresses $\pi(i)$ as a direct function of its indice i and hence allows for parallel implementation:

1) $a \neq 1$:

$$\begin{aligned} \pi(i) &= \mathrm{mod}(b \sum_{j=0}^{i-1} a^j, N) \\ &= \mathrm{mod}(\frac{(1-a^i)b}{(1-a)}, N), \quad i = 0, 1, ...N-1. \end{aligned} \tag{2.5}$$

2) $a = 1$:

$$\begin{aligned} \pi(i) &= \begin{cases} 0, & i = 0, \\ \mathrm{mod}(\pi(i-1)+b, N), & i=1, ...N-1 \end{cases} \\ &= \mathrm{mod}(\, b\, i\, , N), \qquad\qquad i=0, 1, ...N-1. \end{aligned} \tag{2.6}$$

A coprime interleaver $C(1, b, N)$ as defined in (2.6), is also termed linear coprime interleaver and denoted as $LC(b, N)$.

A golden linear coprime interleaver [25] $G(N)$ is an $LC(b, N)$ interleaver, whose b is chosen to be the closest integer to the golden section of N, i.e. b is closest to $\lfloor \frac{(\sqrt{5}-1)\times N}{2} + 0.5 \rfloor$ and relatively prime to N .

Two classes of algebraic interleavers are particularly worth mentioning. The *Welch-Costas interleavers* make essential use of the Costas array, offer performances comparable to random interleavers, and allow for efficient implementations [26]. One drawback, however, is the high complexity in the design procedure, since searching for a primitive element in the Galois field $GF(N)$ can be nontrivial especially for large N. Further, for many practical interleaver lengths of $N = 2^m$, the Welch-Costas interleavers do not exist. Another notable class of algebraic interleavers are the *Takeshita-Costello interleavers* [27], which have been proven to possess several desirable properties as random interleavers. However, since its interleaving pattern can not be derived directly from the input indices, an intermediate sequence of length N has to be computed and stored, thus diminishing the storage advantage of a typical algebraic interleaver.

A Welch-Costas interleaver is generated according to the following rule [26]:

$$\pi(i) = mod((a_1^i), N) - 1, \quad i = 0, 1, ...N - 1, \tag{2.7}$$

where $N+1$ is a prime number and a_1 is a primitive element in $GF(N)$. Note that the constraint on N being a prime number minus 1 excludes the possibility for many interleaver lengths. for example, there does not exist Welch-Costas interleavers at length $N = 32$, 64, 128, 512, 1024, 2048, 4096.

The generating rule of the Takeshita-Costello interleavers is [27]:

$$C_i = mod((a_2 \times (i-1) \times i/2), N), \tag{2.8}$$

$$\pi(C_i) = C_{i+1}, \tag{2.9}$$

where the interleaver length N should be 2^m (m is an integer), and the parameter a_2 should be an odd number smaller than N. As mentioned before, the intermediate sequence $\{C_i\}$ needs to be generated and stored before performing interleaving or de-interleaving.

Since an interleaver that performs well for one turbo code (with specific constituent convolutional codes) in general also performs well for a class of turbo codes with the same constraint length, we thus concentrate the search on one sample turbo code, but the search results generalize to the entire class.

2.2 Metric 1: Cycle Correlation Sum (CCS)

According to the definition, all the coprime interleavers having a length $N = 2^k$ for some integer k can be generated by a pair of parameters a and b, where $a = 4 \times c + 1, 0 \leq c < N/4$, and b is an odd integer. Our first tool is the CCS metric, which regards the correlation between the extrinsic input and output sequences of a BCJR decoder as the indication of the interleaver quality [28].

From the coding theory, the performance of an iterative decoder will approximate that of the optimal decoder when the code graph is free of cycles or when the outbound message from any computing unit does not circulate back. That

latter condition translates to minimal correlation between the outbound message and the subsequent inbound message. In the case of turbo decoders, completion of any round of message exchange between the two component decoders inevitably introduces such undesirable message correlation. To see this, consider bits i and j in the first component code which are interleaved to bits $\pi(i)$ and $\pi(j)$ in the second component code. Since i and j are part of a convolutional codeword, they are inherently correlated. Hence, the reliability information carried by i is transferred to the output extrinsic information of j (through the BCJR decoding), which in turn becomes the input extrinsic information for bit $\pi(j)$. After the BCJR decoding of the second decoder, this reliability information for $\pi(j)$, originated from bit i, gets relayed to bit $\pi(i)$ and, after deinterleaving, is passed back to bit i. Hence, an important measure for the goodness of an interleaver is its ability to minimize the average amount of such correlated message carried from one decoder iteration to the next, where average is performed over all the bits in the sequence.

To quantify the above measure, [28] proposes to evaluate the correlation between the input and output extrinsic information of the BCJR decoding using the standard *correlation coefficients*. It is shown that correlation coefficients are a function of the Hamming distance between two bits and can be approximated by an exponential function. Specifically, [28] formulates the correlation between bits i and j as $e^{-c|i-j|}$, where c is a parameter. Likewise, the correlation between bits $\pi(i)$ and $\pi(j)$ follows $e^{-c|\pi(i)-\pi(j)|}$, and the correlations induced by cycle $i \to j \to \pi(j) \to \pi(i) \to i$ becomes $e^{-c(|j-i|+|\pi(i)-\pi(j)|)}$. Averaging over all such cycles

gives rise to the metric of *cycle correlation sum* [28]:

$$CCS = \sum_{i,j \in A} e^{-c(|j-i|+|\pi(j)-\pi(i)|)} \tag{2.10}$$

where $A \equiv (0, 1, 2, ..., N-1)$, and N is the interleaver length. The parameter c is a constant that is dependent on the component convolutional code, or loosely, the memory size of the component convolutional codes [28]. A lower value of CCS implies less undesirable message correlation introduced in each decoding iteration, a higher efficiency in the iterative turbo decoder, and therefore a better performance achieved by the code. For a more detailed discussion including the computation of CCS, please refer to [28].

To demonstrate the accuracy of CCS, Figure 2.1 compares the CCS predictions and their corresponding performances for linear coprime interleavers ($a = 1$) at length $N = 100$ and 128 bits. For all the possible values of b, the simulated bit error rate (BER) matches remarkably well with the CCS prediction, with a complete and accurate identification of all the worst choices of b (what we should definitely avoid) and a quite accurate identification of the best choices of b (what we wish to attain).

2.3 Evaluating Algebraic Interleavers by CCS

We classify coprime interleavers by their performances as indicated by the CCS metric, and subsequently formulate the rules for good parameters that will lead to performance on par with or better than random interleavers. To complete

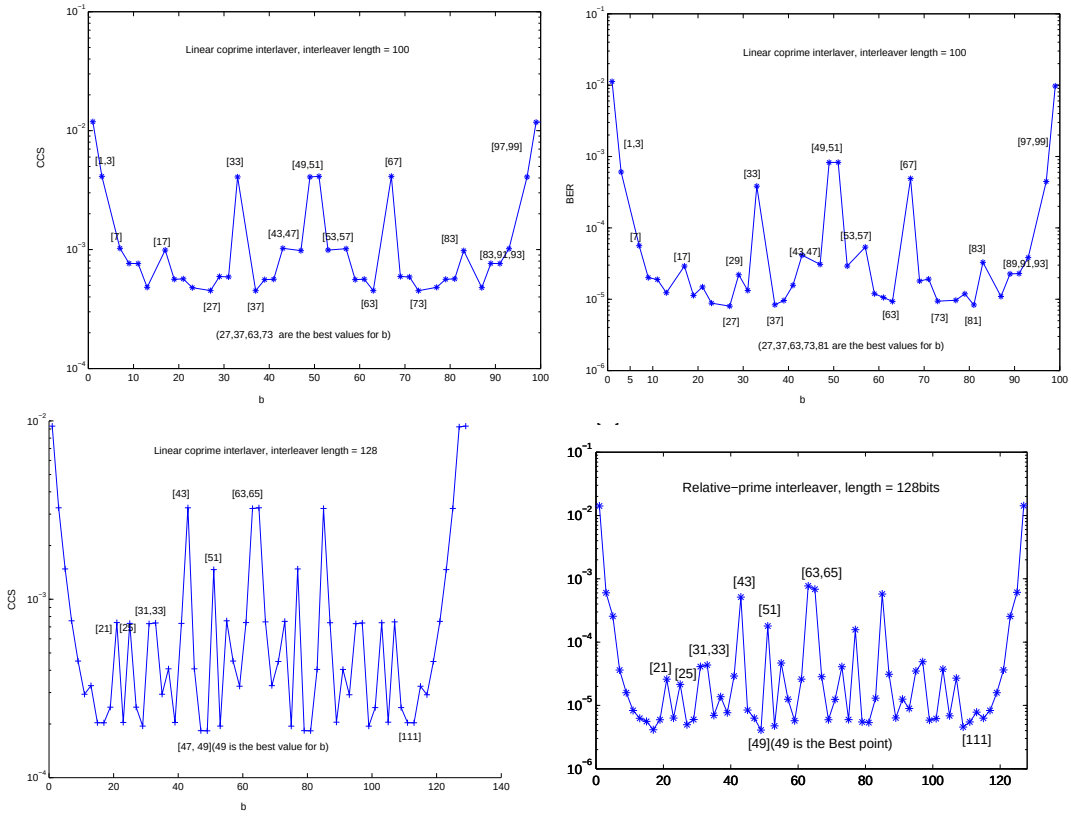


Figure 2.1: Comparison between CCS predictions and simulations results on a turbo code with component code $[1, 5/7]$. Top row: CCS prediction and simulated BER of a length 100 linear coprime interleaver; Bottom row: CCS prediction and simulated BER of a length 128 linear coprime interleaver. Evaluating SNR=3.0dB.

the CCS evaluation, we further compare coprime interleavers with Welch-Costas interleavers, Takeshita-Costello interleavers, random interleavers and S-random interleavers through graph representation and computer simulations.

2.3.1 Analysis and Classification of Algebraic Interleavers

Figure 2.2 evaluates the performance of a host of interleavers with length $N = 128$ bits, including coprime interleavers (and the Golden prime interleaver), the

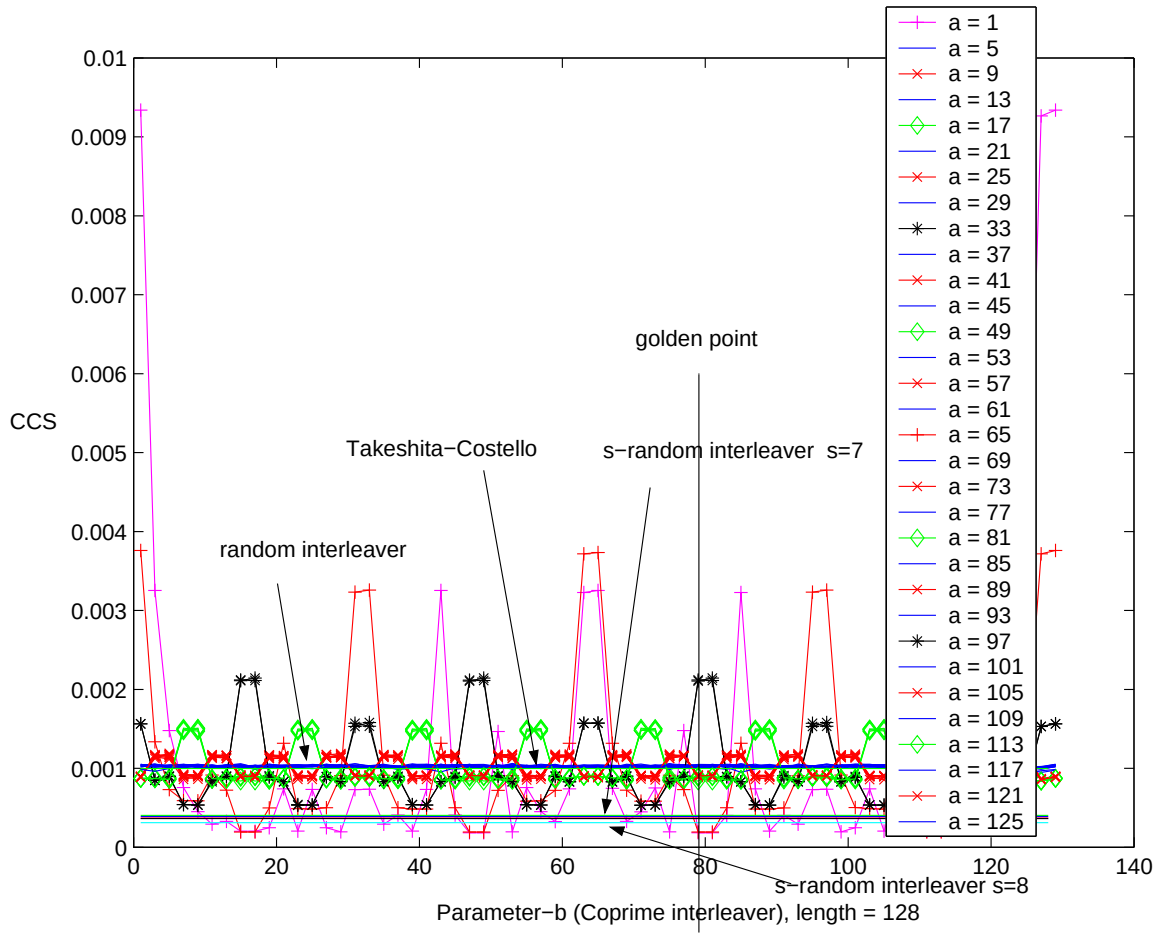


Figure 2.2: The CCS values of coprime interleavers, random interleavers, S-random interleavers and the Takeshita-Costello interleavers. $N = 128$.

Takeshita-Costello interleaver, several random interleavers and S-random interleavers. (Length-128 Welch-Costas interleaver does not exist). The y-axis represents the CCS value. The x-axis represents the value of b for coprime interleavers and the value of a_2 for the Takeshita-Costello interleavers. We tested all the subclasses of coprime interleavers with $a = 4n+1$, $0 \leq n < 32$ and all odd values of b. Different values of a are marked with different line types.

Let us start with S-random interleavers whose performances are delineated by the set of straight horizontal lines located at CCS=0.00035 to 0.0004. From the

plot, most of these straight lines are hugging around CCS=0.0004 and form one thick line. They correspond to the five S-interleavers we found with spread factor $s=7$. The thin line slightly below them at CCS=0.00035 is an S-interleaver with $s=8$. Since the spread factor is upper bounded by $\sqrt{2N}$ for a length N S-random interleaver, these interleavers we tested are about the best S-random interleavers of length 128.

Next, look at the bundle of blue horizontal lines at around CCS=0.001 in Figure 2.2. They correspond to the five random interleavers we tested (generated randomly), the set of Takeshita-Costello interleavers generated using (2.8) and (2.9) with different values of a_2, and several subclasses of coprime interleavers. First, the performances of the Takeshita-Costello interleavers are not sensitive to the parameter a_2 (denoted by the x-axis) and fall right in the random interleaver region according to CCS. This confirms the claim that they are structured interleavers but behave like random interleavers [27]. Similar results of the Welch-Costas interleavers (i.e. perform similar to random interleavers and insensitive to a_1) are obtained for interleaver length of 100 bits, but the plot is omitted due to the space limitation. Third, the subclasses of coprime interleavers that fall in this performance category have $a=5, 13, 21, ...125$. Unlike other subclasses, the performances of these coprime interleavers are consistently close to that of random interleavers regardless of the value of b. It is remarkable to note that this observation is not unique to length $N = 128$. In general, it appears that for any given length N, *there exists subclasses of coprime interleavers which perform unanimously close to random interleavers.* These subclasses, thereafter referred to as *regular coprime interleavers*, are determined by a single parameter a (provided that b is coprime with N). From extensive tests, when $N = 2^m$, the subclasses having $a = 8k-3$

where $k = 1, 2, ..., N/8$ form regular prime interleavers.

In addition, we observe that coprime interleavers can be classified in several categories in accordance to their ensemble CCS values. For the case of $N = 128$ shown in Figure 2.2, regular coprime interleavers clearly form one category. The subclasses with $a = 9, 25, ..., 8k+1, ..., 121$ (marked with red cross) form a second category, whose CCS values are either slight above or slight below that of random interleavers depending on b. Then there is the category with $a = 17, 49, 81, 113$ (marked with green diamonds), whose performances vary more noticeably with the different choices of b. Finally, the subclasses of $a = 1$ and 65 (marked with red plus signs) see the largest performance variation with respect to b. These subclasses consist of a hybrid of "extreme" interleavers, i.e. the worst coprime interleavers that lag far behind the others and the best coprime interleavers that can outperform random and S-random interleavers. We are unable to formulate a rule for the desirable choices of b, but the Golden prime interleaver with $a = 1$ and $b = 0.618 \times N = 79$ is certainly one good example.

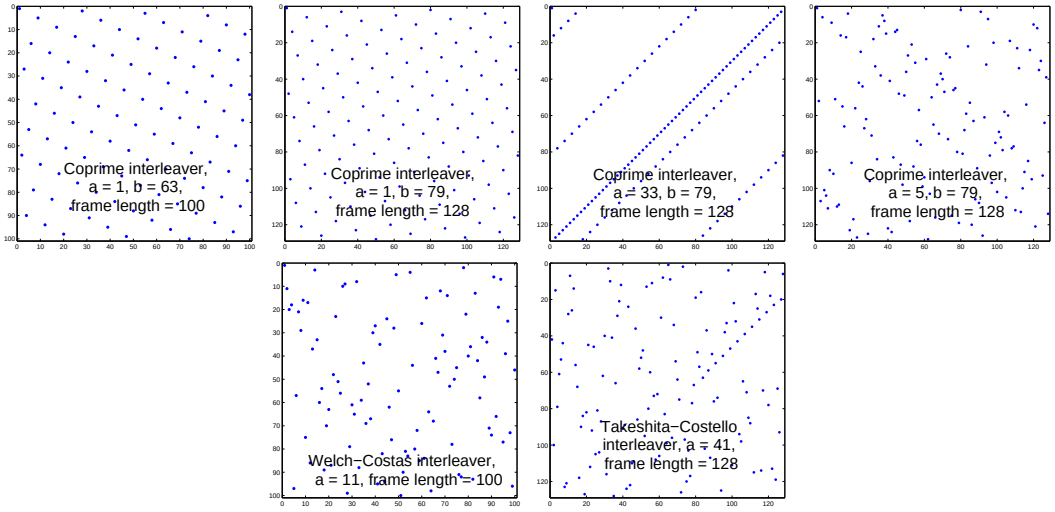


Figure 2.3: Scatter-plot representation for interleavers with $N = 100$ and 128.

To summarize, we have the following major results:

1. The ensemble of coprime interleavers comprises different subclasses parameterized by a. In general, the interleaver performances in each subclass are also dependent on b. However, some subclasses exhibit a quite strong dependence while some others appear rather insensitive.

2. One important subclass is the linear coprime interleavers where $a=1$. Despite its simplicity, it consists of some of the best coprime interleavers which can outperform random interleavers and S-random interleavers (for short lengths) [28] (e.g. the Golden prime interleaver). Since it also consists of some of the worst interleavers, the value of b should therefore be chosen with caution.

3. There exist several subclasses of coprime interleavers, referred to as *regular coprime interleavers*, which perform as well as random interleavers. Regular coprime interleavers are attractive for their random-like behavior and cheap implementation. For $N=2^m$, the following parameters lead to regular coprime interleavers:

$$\begin{cases} a &= 8k-3, \quad k=1,2,...,N/8 \\ b &= 2t-1, \quad t=1,2,...,N/2 \end{cases} \tag{2.11}$$

2.3.2 Graph Representation and Simulations

As a complement to the CCS evaluation, we visualize the randomness of some interleavers using graphs. As shown in Figure 2.3, a length-N interleaver can be represented using an $N \times N$ grid or lattice where the y-axis represents the original

sequence i and the x-axis indicates the interleaved sequence $\pi(i)$.

The coprime interleavers with $N = 100$, $a = 1$, $b = 63$ (top-left) and $N = 128$, $a = 1$, $b = 79$ (top-right) are the Golden prime interleavers. Despite their regularity which may lead to repeated and periodic error patterns, Golden prime interleavers offer quite good performances especially at short lengths.

The coprime interleaver with $N = 128$, $a = 33$, $b = 79$ (mid-left) is an example of a poor interleaver. The undesirable interleaving pattern is obvious from the existence of many repeated (error) patterns and in particular the many vulnerable pairs with very short Euclidean distances [28].

The three other interleavers, the regular coprime interleaver with parameters $N = 100$, $a = 5$, $b = 79$ (mid-right), the Welch-Costas interleaver with $N = 100$, $a_1 = 11$ (bottom-left), and the Takeshita-Costello interleaver with $N = 128$, $a_2 = 41$ (bottom-right), are clearly examples of algebraic interleavers that are constructed using structure yet exhibit random-like behavior.

Further, it is interesting to compare the three interleavers on the top-right, mid-left and mid-right, all of which have $b = 79$, the Golden section. Depending on a, they exhibit very different properties: regular but still good, regular and bad, and random-like and hence good. This points out the importance to understand the classification of coprime interleavers and the impact of the parameters on their behavior, and to subsequently make informed choices.

Finally, we provide the SNR-vs-BER performance of regular coprime interleavers in Figure 5.13, and compare it with that of the Takeshita-Costello interleavers and random interleavers. Two different interleaver lengths of 128 bits and

2048 bits are simulated for a turbo code with two identical component codes of generator polynomial $[1, 5/7]$. The simulation results confirm that regular coprime interleavers perform as well as random interleavers and the Takeshita-Costello interleavers.

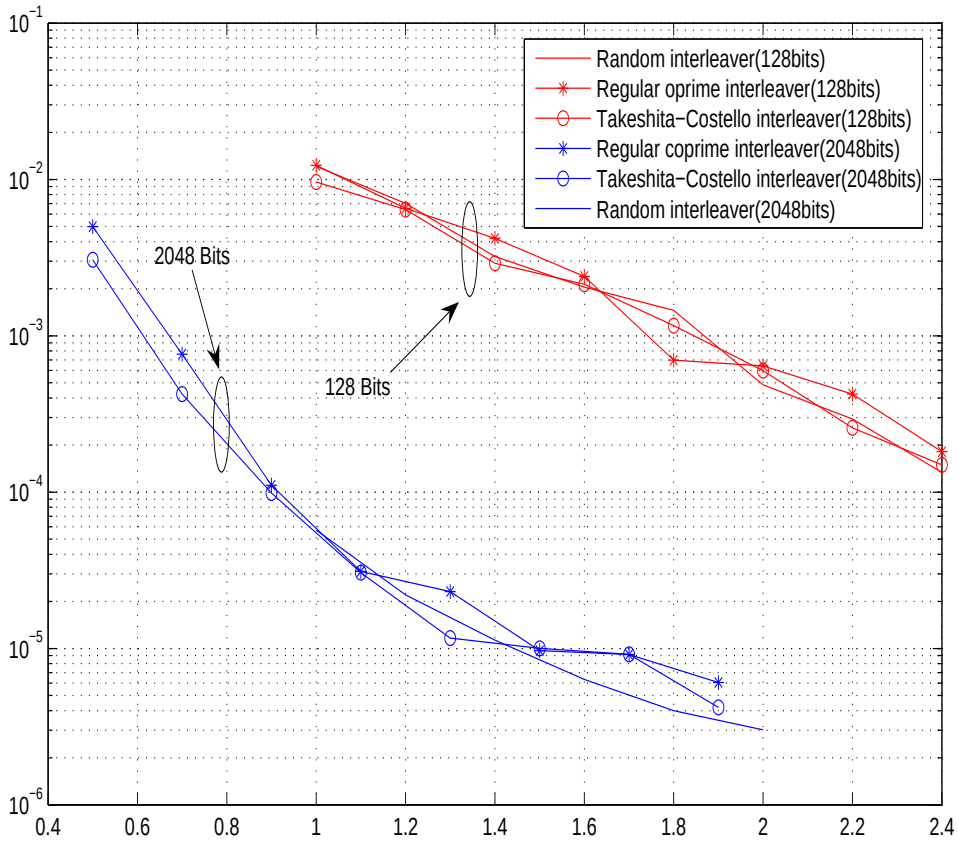


Figure 2.4: BER performance of the random-like interleaver.

To summarize, algebraic interleavers are preferable due to practical concerns such as reduction of hardware requirements and interleaving/deinterleaving operations. We have investigated the behavior of random interleavers and random-like algebraic interleavers using the CCS metric. With the above investigation, we

found that random interleavers and S-random interleavers fall into the fixed regions in the CCS plane. The Welch-Costas interleavers, the Takeshita-Costello interleavers and certain subclasses of coprime interleavers will also stay in the random interleaver region. Following this observation, we propose a bank of good interleavers, termed *regular coprime interleavers*, and formulate their parameters for interleaver lengths of power of 2. Graph representation and BER simulations further confirm the randomness and the good performance exhibited by regular coprime interleavers. In addition, we found that the subclass of linear coprime interleavers ($a = 1$), although simple, contain some of the best interleavers. However, caution should be taken in choosing parameter b, since the same subclass also contain some of the worst interleavers.

We therefore propose the regular coprime interleavers as a strong candidate for practical turbo codes. They offer similar performance as the Welch-Costas interleavers, the Takeshita-Costello interleavers, and random interleavers, but are simpler, more storage efficient and easily parallelizable.

In the next section, we will prove that regular coprime interleavers (c_{power}=0) have the largest randomness degree among all the coprime interleavers and that the degree of randomness decreases with the increase of c_{power}. Hence, it appears that randomness and spread can not be optimized at the same time, and one must strike a good trade off between the two aspects.

2.4 Metric 2: Variance of the second-order spread spectrum (VSSS)

We first introduce the concept of variance of the second-order spread spectrum (VSSS) to characterize the degree of randomness for an interleaver [29]. We next show that regular coprime interleavers have the maximal degree of randomness.

Definition 2.1: Let i and j be the input bit-pair of an interleaver with length N, and $\pi(i)$ and $\pi(j)$ be the corresponding interleaved bit-pair. Let u and v be the distances where $1 \leq u = |i-j| \leq N-1$ and $1 \leq v = |\pi(i)-\pi(j)| \leq N-1$. Let $S_{u,v}$ be the number of the weight-2 patterns (i-j pairs) with the same u and v. Then the set of $S_{u,v}$ forms an $(N-1)$-by-$(N-1)$ matrix (termed the *second-order spread spectrum matrix* [29]), with u indexing the rows and v indexing the columns. $VSSS$ is defined as $\sum_u(var(S_{u,:}))/(N-1)$, where $var(S_{u,:})$ stands for the variance of u-th row in the spread matrix. [29] shows that a smaller $VSSS$ indicates a larger degree of randomness of the interleaver.

Definition 2.2: Consider a function $z = F(x, y)$, $0 < x < m$, $0 \leq y < m-1$. Let $m_{z,x}$ denote the number of y which generates the same z, for a given x. We define the matrix M_F, whose entries are $m_{x,z}$ with x and z representing the row index and column index respectively, as the *input-output-matrix* of function $F(x, y)$. The variance of M_F is defined as $\sum_x(var(m_{x,:}))/(m-1)$.

Lemma 2.1: For a given $u \in \{1, 2, ..., N-1\}$, the elements contained in the set

$\{A(i)\}$, where

$$A(i_1) = \mathrm{mod}(b\frac{a^u - 1}{a - 1}a^{i_1}, N), \quad i_1 = 0, 1, ...N - 1, \tag{2.12}$$

will not co-exist in the set $\{B(i)\}$, where

$$B(i_2) = N - \mathrm{mod}(b\frac{a^u - 1}{a - 1}a^{i_2}, N), \quad i_2 = 0, 1, ...N - 1. \tag{2.13}$$

where N is a power of 2, both $a, b < N$. In addition, $a = 4c + 1$ where c is integer, and b is relative prime with N.

∇ *Proof:* (Proof by contradiction) If an element $A(i_1)$ in sequence (2.12 equals to an element $B(i_2)$ in sequence 2.13, we have

$$\mathrm{mod}(b\frac{a^u - 1}{a - 1}a^{i_1}, N) + \mathrm{mod}(b\frac{a^u - 1}{a - 1}a^{i_2}, N) = N. \tag{2.14}$$

Without loss of generality, we assume $i_1 \leq i_2$ and let $t = |i_2 - i_1|$. We can rewrite the previous equation as

$$\mathrm{mod}(b\frac{a^u - 1}{a - 1}a^{i_1}(1 + a^t), N) = 0. \tag{2.15}$$

Consider that $\frac{a^u - 1}{a - 1} = 2^{q_1}q_2$ (where q_2 is odd).

If $N \leq 2^{q_1}$, since N is the power of 2, then $A(i_1)$ is always 0 and $B(i_2)$ is always N.

If $N > 2^{q_1}$, we have

$$\mathrm{mod}(bq_2a^{i_1}(1+a^t), N/2^{q_1}) = 0. \tag{2.16}$$

and subsequently

$$\mathrm{mod}((1+a^t), N/2^{q_1}) = 0. \tag{2.17}$$

Following the definition of coprime interleavers and substituting a with $a = 4c+1$, we expand this equation to:

$$\mathrm{mod}(\sum_{k=1}^{t}\binom{t}{k}(4c)^k + 2, N/2^{q_1}) = 0. \tag{2.18}$$

It is easy to see that $\sum_{k=1}^{t}\binom{t}{k}(4c)^k + 2$ is a multiple of 2, and the quotient is odd. This makes

$$\mathrm{mod}((2B), N/2^{q_1}) = 0, \tag{2.19}$$

where B is odd. Since $N > 2^{q_1+1}$, (2.19) can not hold. Contradiction. $\triangle$

Theorem 2.2: If $F(x,y)$ is in the form of

$$F(x,y) = \mathrm{mod}(b\frac{a^x-1}{a-1}a^y, N),$$

then the $VSSS$ of the length-N coprime interleaver generated with parameters a and b is smaller than the variance of M_F for $F(u,i)$, where i and j are any input pair of the interleaver, and $u = |i-j|$.

∇ *Proof:* Given a pair (u, v), we can find a set C_i of i satisfying $v = F(u, i)$, then $m_{u,v}$ equals the size of C_i.

On the other hand, according to the definition of coprime interleavers, we have

$$v = \begin{cases} \text{mod}(b\frac{a^u-1}{a-1}a^i, N), & \pi(j) > \pi(i), \\ \\ N - \text{mod}(b\frac{a^u-1}{a-1}a^i, N), & \pi(j) \leq \pi(i). \end{cases} \tag{2.20}$$

We divide C_i into two subsets: $C_i^{(1)}$ for $\pi(j) > \pi(i)$ and $C_i^{(2)}$ for $\pi(j) < \pi(i)$, such that $C_i^{(1)} \bigcup C_i^{(2)} = C_i$ and $C_i^{(1)} \bigcap C_i^{(2)} = \phi$. Hence the size of $C_i^{(1)}$ is not larger than the size of C_i, which equals to $m_{u,v}$.

For a coprime interleaver π, using (2.20), and given u and $C_i^{(1)}$, we get the unique output v_1. Now from Lemma 1, given u, $C_i^{(1)}$ contains all the is that will generate v_1. Hence, S_{u,v_1} equals the size of $C_i^{(1)}$, and it is smaller than $m_{u,v}$.

Following the same line of derivation, when we assume that the set $C_i^{(2)}$ will generate v_2 under u, we will get that $m_{u,v} \leq S_{u,v_2}$.

Hence, the value of each element in M_F is divided into two parts which correspond to two elements in VSSS. Therefore, $VSSS$ is less than the variance of M_F of $F(u, i)$. $\triangle$

Additionally, because $F(u, i)$ is periodic for a given i, we can convert the problem of maximizing the $VSSS$ of a coprime interleaver to one of increasing the period of $F(u, i)$.

Theorem 2.3: If we break $\frac{a^u-1}{a-1}$ down to the product of 2^q (even component) and l (odd component), where l is odd and q is a nonnegative integer, then $F(u,i)$ and sequence $\text{mod}(a^i, N/2^q)$ have the same period, where N is the power of 2 and $N > 2^q$.

∇ *Proof:* Assume the period of $F(u,i)$ is P_f, then

$$(F(u, i+P_f) - F(u,i)) \equiv 0. \tag{2.21}$$

From the definition of $F(u,v)$'s definition, we get

$$\text{mod}((b\frac{a^u-1}{a-1}(a^{i+P_f} - a^i)), N) \equiv 0. \tag{2.22}$$

Under the assumption in Theorem 2.3, i.e. $\frac{a^u-1}{a-1} = 2^q l$, where l is odd, (2.22) becomes

$$\text{mod}((b2^q l(a^{i+P_f} - a^i)), N) \equiv 0. \tag{2.23}$$

Since both b and l are odd, we have

$$\text{mod}((a^{i+P_f} - a^i), N/2^q) \equiv 0, \tag{2.24}$$

This essentially states that P_f is also the period of sequence $\text{mod}(a^i, N/2^q)$. $\triangle$

Theorem 2.4: For a coprime interleaver with length $N = 2^m \geq 4$ and parameters a and b, if $c = (a-1)/4$ is odd, then the period of sequence $\text{mod}(a^i, N/2^q)$ is maximized.

∇

Proof: Since N is a multiple of 4 and $a = 4c + 1$ (see the definition of coprime interleavers), (2.24) can be simplified to

$$\mathrm{mod}((4c+1)^{P_f} - 1, N/2^q) \equiv 0. \tag{2.25}$$

Expanding (2.25), we get

$$\mathrm{mod}((\sum_{k=1}^{P_f} \binom{P_f}{k} (4c)^k), N/2^q) \equiv 0, \tag{2.26}$$

which can be re-written as

$$\mathrm{mod}(P_f(4c) + \binom{P_f}{2}(4c)^2 + ... + (4c)^{P_f}), N/2^q) \equiv 0. \tag{2.27}$$

Similarly, we can factorize P_f into a product of an odd component O_p and an even component E_p. Observing that all the terms in $\sum_{k=1}^{P_f} \binom{P_f}{k}(4c)^k$ contain $4cE_p$, we can extract it and move it before the summation. The remainder can be denoted by an odd number A. Finally, (2.27) becomes

$$\mathrm{mod}(A4E_pc, N/2^q) \equiv 0. \tag{2.28}$$

On the other hand, since P_f is the period, it is the smallest number satisfying (2.24). Thus the smallest possible value P_f is E_p (when $O_p = 1$). More important, consequently, only when c is odd, E_p can be maximized as $N/2^{(q+2)}$. Finally, $P_f = N/2^{(q+2)}$ is the largest period possible, obtained when c is odd. $\triangle$

Corollary 2.5: Among all the coprime interleavers, the regular coprime interleavers ($c_{power=0}$) provide the minimal $VSSS$.

Corollary 2.6: Let $c = 2^{c_{power}} c_{odd}$. The degree of randomness as measured by VSSS decreases with the increase of c_{power}. The degrees of randomness of all the coprime interleavers remain at the same level for the same c_{power}.

2.5 Interleaver Design and Simulations for coprime interleaver

As shown in the previous discussion, the largest degree of randomness and the largest spread can not be achieved at the same time for a coprime interleaver. As c_{power} increases, the best spread (indicated by the lowest CCS) in the category of coprime interleavers increases, but the degree of randomness (indicated by VSSS) reduces. Additionally, they both stay at the same level for the same c_{power}, irrelevant to c_{odd}. Hence for convenience we can take $c_{odd} = 1$.

When we design a coprime interleaver, we need to first carefully select the parameter $a = 4c + 1$, where $c = 2^{C_{power}} C_{odd}$, and then select b.

- At first, we will choose c_{power} which determines both the randomness character and the range of spread character. For code length $N = 2^k$, c_{power} could be any integer from 0 to $k - 3$, since $a \leq N$. To balance the spread and randomness, we might take some middle value of c_{power}. On the other hand, for convenience, we also let $c_{odd} = 1$. Then we can obtain the parameter a

and $a = 4 \cdot 2^{c_{power}} + 1$.

- After choosing the parameter a that strikes a good balance between the randomness and the spread, we can search the parameter b to obtain the maximal spread by minimizing the CCS.

As an example, consider $N = 2048$. We have 10 categories of coprime interleavers, exemplified by $a = \{1, 5, 9, 17, 33, 65, 129, 257, 513, 1025\}$, each associated with a different c_{power}: $c_{power} = \infty, 0, 1, ..., 7, 8$. To get a good balance between the spread and the randomness, we choose $c_{power} = 5$, that is $a = 129$. Then we will search the parameter b to obtain the lowest CCS associated with $a = 129$. Finally we get $a = 129$ and $b = 161$.

We compare the BER performance of the coprime interleaver ($a = 129, b = 161$) and two S-random interleavers (spread $s = 10, 20$) based on a turbo code with code length 2048 on AWGN channels. We use [5,7] as the component code of the turbo code, and the code rate is 1/3. For each frame, we performed 8 rounds of iterations. From Figure 2.5, we see that our optimized coprime interleaver outperforms the S-random interleaver with $s = 10$ by $0.2db$. It provides a performance better than the S-random interleaver with $s = 20$ at low to medium SNRs and a comparable performance at high SNR.

2.6 Conclusion

Algebraic interleavers are preferable due to their simplicity in hardware implementation and economy in storage. Algebraic interleavers with good randomness and

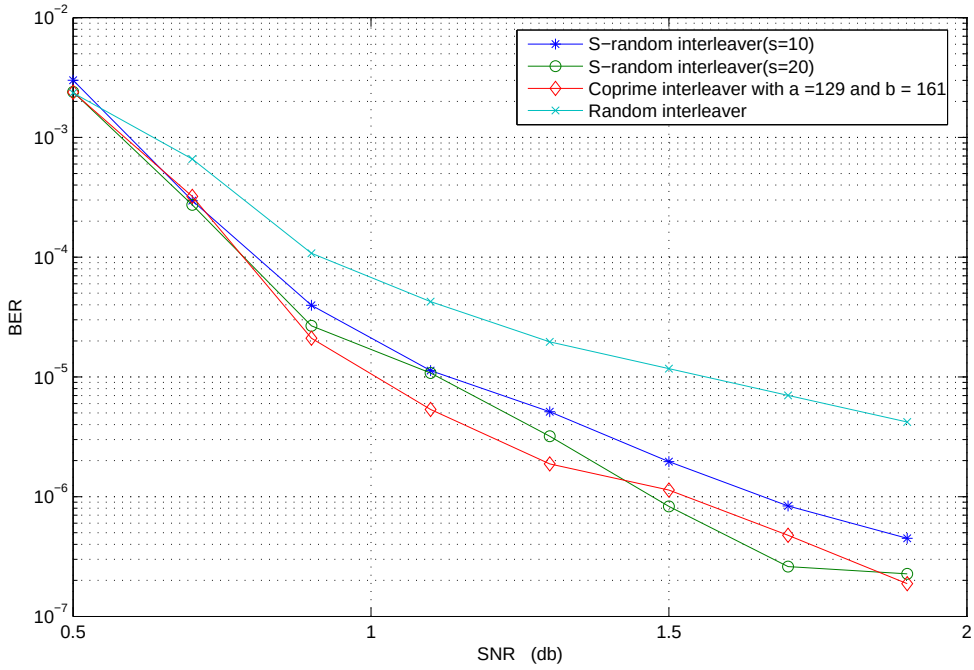


Figure 2.5: BER of the optimized coprime interleaver ($a = 129, b = 161$) and S-random interleavers (s=10,20) for $N = 2048$.

spread properties promise great performances at low cost.

In this chapter, we investigates the coprime interleavers, a rich subset of algebraic interleavers. For interleavers whose lengths are powers of 2, we formulated a critical parameter c_{power}, which captures some important behavioral properties of coprime interleavers. We used the cycle correlation sum criterion (CCS), to measure the minimum spread, and the variance of the second order spread spectrum, to measure the degree of randomness, for coprime interleavers. With the increasing c_{power}, the CCS property becomes better, while the randomness property becomes worse. Since the optimal degree of randomness and the optimal spread cannot be simultaneously achieved for coprime interleavers, we formulated a rule to find interleaver parameters that strike the a good balance between them. Simulations

confirm the effectiveness of our rule by demonstrating that optimized coprime interleavers perform as well as or better than S-random interleavers.

Chapter 3

Gaussian Assumption of LDPC Codes

The breakthrough of turbo codes and low-density parity-check (LDPC) codes has revolutionized the coding research with new concepts for successful error correction: a paradigm of constructing long, powerful codes using short, weak component codes and decoding them using soft, iterative decoders with manageable complexity. To fully understand soft-iterative decoding, researchers explored stochastic approaches which model the input and output of a soft decoder as random processes and track the evolution of their statistic characteristics through iterations. This resulted in the renowned method of *density evolution* (DE) [2], a useful method for asymptotic performance analysis and optimization of sparse-graph codes [5]. The original DE tracks the complete probability density function (pdf) of the messages, which unavoidably involves infinite or huge dimensional algebra, and is therefore computationally tedious. Remedy was then proposed to approximate the log-likelihood

(LLR) messages exchanged between component decoders by a Gaussian distribution [3] [4]. This *Gaussian assumption* (also termed *Gaussian approximation* (GA)), when combined with the *symmetry condition* (known to preserve in any message-passing algorithm [5]), leads to a remarkable doubling relation, namely, the variance equals twice the mean. $\sigma^2 = 2\mu$. The results is a significant simplification of DE, since it now suffices to track a single scalar parameter (i.e. mean or variance or some function of them) rather than the entire message pdf.

Building upon the successful idea of DE, [6] proposed to use extrinsic information transfer (EXIT) charts to characterize the behavior of iterative decoding as the temporal evolution of the extrinsic mutual information (MI) exchanged between different computational units. Although they were initially proposed largely as a visualization tool, recent studies have revealed surprisingly elegant and useful properties of EXIT charts, including, for example, the convergence property, the area property, and code optimization through curve fitting [30] [31].

Despite the significant role the Gaussian assumption has played in simplifying and popularizing these methods, the justification of this assumption is largely pragmatic. This seems-to-work philosophy has underlined the analysis of the iterative decoding for much of its short history, and it is only recently that [7] provided a statistical analysis on the accuracy of the Gaussian assumption for turbo codes. As a parallel to [7], this work provides a statistical justification for LDPC codes.

We investigate when and how well the Gaussian distribution approximates the real message density, and the far subtler why. We will show that the Gaussian assumption is statistically sound (i) when the LLR messages extracted from the channel are reasonably reliable to start with, and (ii) when the check node degrees

of the LDPC code are not very high, but the assumption is much less accurate when one or both conditions are violated. Extensive simulation results are provided to exemplify and verify the discussion.

The analysis of GA naturally leads to the study of EXIT charts for LDPC codes. We differentiate two cases: (1) When GA is (reasonably) accurate and so are EXIT charts, we consider simplifying the computation of EXIT charts by avoiding the many cumbersome integrations involved in the EXIT formulation. A previous effort was made in [30], but the model therein involves many parameters. The new formulation developed here is much simpler, and works well for both regular and irregular LDPC codes. As part of the investigation, we also derive several simple and good closed-form approximations for evaluating the mutual information between bits and their LLR messages. These approximations can be applied to codes beyond LDPC codes. (2) When the Gaussian approximation is less accurate and so the doubling relation is less accurate, we compare several mutual information formulas computed using either the mean, the variance or both parameters. We show that different choices of formulations will lead to different levels of accuracy in EXIT charts, and with the right choice, a good level of EXIT accuracy is achievable (dispute the discrepancy between the true message density and the Gaussian distribution). A similar observation was noted previously [6], but we have included more comparison cases. Hence, for practical purpose, the Gaussian assumption and the doubling relation may still be used for EXIT analysis. It then follows that the simple EXIT model we developed for Case (1) is also practically useful in Case (2) where Gaussianity actually breaks. We also show that it is possible, but rather tedious, to simultaneously track the evolution of the message mean and variance, and to use them to obtain a very close approximation

to the true EXIT curves.

The remainder of the chapter is organized as follows. Section 3.1 briefs the the background of LDPC decoding and the notations used in the chapter. Section 3.2 discusses lognormal distributions and establishes several properties useful for our analysis. Section 3.3 discusses the accuracy and the applicable region of the Gaussian assumption. Section 3.4 proposes a new simple EXIT formulation. Section 3.5 discusses accuracy of different mutual information formulations for EXIT chart when Gaussian assumption is less accurate. Finally, Section 3.6 concludes the chapter.

3.1 Background and Notations

An (n, k) LDPC code is a linear channel code characterized by a sparse parity check matrix $H = \{h_{i,j}\}$ with n columns representing all the bits in the codeword and $m \geq n-k$ rows representing the parity constraints imposed on the coded bits. Practical decoding of LDPC codes makes essential use of bipartite graphs, known as *Tanner graphs* (or *factor graphs* which are generalization of Tanner graphs), to represent codes, and to pass probabilistic messages along the edges of the graph. The Tanner graph for an (n, k) LDPC code consists of n variable nodes corresponding to the columns in H, m check nodes corresponding to the rows in H, and multiple edges connecting the two types of nodes. An edge connects ith variable node and jth check node if and only if $h_{ij} = 1$. The number of the edges connected to a node is termed the *degree* of this node. We will use d_v and d_c to represent the degree of a variable node and a check node, respectively.

Consider message-passing decoding over an LDPC Tanner graph, where soft extrinsic information iterates between variable nodes and check nodes, and updates itself after each iteration. Let superscript ℓ denote the number of decoding iterations, and subscripts i and j denote, respectively, variable nodes and check nodes. At ℓ-th iteration, the extrinsic information passed from variable node i to check node j, denoted as m_{ij}^{ℓ}, and the extrinsic information passed from check node j to variable node i, denoted as m_{ji}^{ℓ}, are updated as follows:

$$m_{ij}^{\ell} = \begin{cases} m_i, & \ell = 0, \\ m_i + \sum\limits_{j' \in N_c(i) \setminus \{j\}} m_{j'i}^{\ell}, & \ell > 0. \end{cases} \tag{3.1}$$

$$m_{ji}^{\ell} = \ln\left(\frac{1 + \prod_{i' \in N_v(j)\setminus\{i\}} \tanh(m_{i'j}^{\ell-1}/2)}{1 - \prod_{i' \in N_v(j)\setminus\{i\}} \tanh(m_{i'j}^{\ell-1}/2)}\right), \tag{3.2}$$

$$= 2\tanh^{-1}\Big(\prod_{i' \in N_v(j)\setminus\{i\}} \tanh\frac{m_{i'j}^{\ell-1}}{2}\Big), \tag{3.3}$$

$$= \Big(\prod_{i' \in N_v(j)\setminus\{i\}} \operatorname{sign}(m_{i'j}^{\ell-1})\Big) \cdot \Phi\Big(\sum_{i' \in N_v(j)\setminus\{i\}} \Phi\big(m_{i'j}^{\ell-1}\big)\Big), \tag{3.4}$$

where $N_c(i)$ is the set of check nodes connected with i-th variable node, $N_v(j)$ is the set of variable nodes connected with j-th check node, and m_i is the log likelihood ratio (LLR) of signal s_i, extracted from ith channel output r_i:

$$m_i = \ln\frac{\Pr(r_i|s_i = +1)\Pr(s_i = +1)}{\Pr(r_i|s_i = -1)\Pr(s_i = -1)} \tag{3.5}$$

For equally-probable input and additive white Gaussian noise (AWGN) with a zero mean and a variance σ_n^2, we have $m_i = 2r_i/\sigma_n^2$, and m_i follows a (conditional)

Gaussian distribution: $m_i|s_i \sim \mathcal{N}(2s_i/\sigma_n^2, 4/\sigma_n^2)$. The function $\Phi(\cdot)$ in (3.4) is defined as:

$$\Phi(x) = \ln\left(\frac{e^{|x|}+1}{e^{|x|}-1}\right), \tag{3.6}$$

where for convenience we let $\Phi(0) = \ln(2/0) = \infty$. The formulations in (3.2), (3.3) and (3.4) present three different forms describing the same check update operation. Our Gaussian analysis in the below will be performed based on (3.4).

3.2 Lognormal Distributions

This section establishes a few useful properties of lognormal distributions, upon which our analysis of Gaussian approximation is based.

Definition 3.1: [Lognormal distribution] A positive random variable X is said to be *lognormal* distributed if its logarithm value $\ln(X)$ follows a Gaussian distribution. Using the Jacobian rule, the lognormal pdf for X follows:

$$f_X(x) = \frac{1}{\sqrt{2\pi}x\sigma}\, e^{-\frac{(\ln(x)-\mu)^2}{2\sigma^2}}, \quad \text{for } x > 0, \tag{3.7}$$

where μ and σ^2 are the mean and the variance of $\ln(X)$.

To provide a visual impression of how lognormal densities look like, Figure 3.1 plots the pdf curves for 4 lognormal distributions with $\mu = 0$ and $\sigma = 0.5, 1.0, 1.5, 3.0$, respectively.

A long-recognized fact in statistics is that *the sum of two lognormal random variables is also lognormal*. This statistic rule has been widely applied in many

science and engineering fields. For example, it was used to derive the coherent channel interference model in wireless communications [32] [33] [34] [35] [36] [37] [38] and to perform risk measuring in finance [39]. Recently, it was also used to justify the Gaussian assumption in the BCJR decoding algorithm [7]. Here we exploit this rule to evaluate the Gaussian assumption in the LDPC message-passing algorithm. Notice that repeated application of this rule tends to suggest that the lognormal property will preserve even when the number of additive terms becomes large, which will conflict with the central limit theorem. Below we restate this statistical rule in a more accurate way by differentiating between correlated and uncorrelated random variables and between finite and infinite number of terms.

Proposition 3.1: [Sum of Lognormal Variables] The sum of a set of *correlated* lognormal random variables follows a lognormal distribution, regardless of whether the set is finite or countably infinite. The sum of a set of independent lognormal random variables approximates the lognormal distribution when the set is small, transforms from lognormal to Gaussian as the set size increases, and eventually becomes Gaussian in the limit of infinite set size.

The case of correlated random variables can refer to [36]- [38], [39], [7] and may be verified by simulations (see Figure 3.2 and the related discussion later on). For independent random variables, the investigation in [40] confirmed that the lognormal approximation is quite accurate for a set size of 10 or smaller. Gaussianity of the sum in the limiting case follows from the central limit theorem[1]

Proposition 3.2: [Power Sum of Lognormal Variables] Let X be a lognormal

[1]Strictly speaking, the application of the central limit theorem requires that no one or few terms in the set are dominant.

random variable, its power sum, defined as

$$S = \sum_{i=1}^{k} a_i X^{b_i}, \tag{3.8}$$

follows a lognormal distribution, where $\{a_i\}$ and $\{b_i\}$ are sets of arbitrary non-zero constants and k may be either finite or infinite.

∇ *Proof:* Since X follows a lognormal distribution, there exists a Gaussian random variable Z that satisfies the equality $X = e^Z$. Rewrite $a_i X^{b_i}$ as $e^{b_i Z + c_i}$, where c_i is a constant and $c_i = \ln(a_i)$. Since $b_i Z + c_i$'s satisfy the correlated Gaussian distribution (for $b_i \neq 0$), according to the definition of the lognormal distribution, $e^{b_i Z + c_i}$'s, and hence $a_i X^{b_i}$'s for $a_i \neq 0$, form a set of correlated lognormal random variables. Following Proposition 3.1, their sum S will also be lognormal. $\triangle$

For LDPC analysis, we are most interested in negative integer values of b_i's. Since the proof of Proposition 3.2 uses Proposition 3.1 which is a statistical rule-of-thumb, we perform experimental tests to verify Proposition 3.2. Figure3.2 presents the histograms, each collected over 10000 test samples, for $\ln(S)$ with set size $k = 2, 5, 10, 100$ and randomly selected negative integers b_i's. The histograms demonstrate that $\ln(S)$ consistently behaves like a Gaussian variate regardless of the set size, which confirms the validity of the lognormal approximation for S.

To provide a quantifiable evaluation of how close the empirical data matches the true Gaussian distribution (and hence to what accuracy Proposition 3.2 stands), we resort to a goodness-of-fit tool named Kolmogorov-Smirnov (KS) test [41]. The KS test compares the cumulative frequency of empirical data (normalized by the sample size) with the cumulative density function (cdf) of a Gaussian distribu-

tion by measuring the greatest discrepancy between the two cdf's. This greatest discrepancy, termed the $D_{statistic}$ [41], is mathematically formulated as expressed as:

$$D_{statistic} = \max_{x} \left(|F(x) - G(x)|\right), \tag{3.9}$$

where $F(x)$ represents the normalized cumulative frequency of the observations that are equal to or small than x, and $G(x)$ represents the standard Gaussian cdf evaluated at x. For the experimental data in Figure 3.2, the KS tests show that for $k = 2, 5, 20, 100$, $D_{statistic} = 0.0047345$, 0.0048123, 0.0087541, 0.0077152, respectively. The uniformly very small values of $D_{statistic}$ confirm that $\ln(S)$ is very close to Gaussian and hence S is very close to lognormal.

Proposition 3: [Distribution of $\Phi(x)$ with Gaussian Input] If $|X|$ follows an (approximate) Gaussian distribution, then $\Phi(X)$ in Equation (3.6) follows an (approximate) lognormal distribution.

∇

Proof: Consider an auxiliary function $\xi(z)$ defined for $z \geq 1$ as

$$\xi(z) = \ln(\frac{z+1}{z-1}), \quad z \geq 1, \tag{3.10}$$

Using the Tailor series expansion, $\xi(z)$ can be expressed as

$$\xi(z) = 2\sum_{k=1}^{\infty} \frac{z^{1-2k}}{(2k-1)}. \tag{3.11}$$

Since $e^{|x|} \geq 1$, we substitute for z in (3.11) with $e^{|x|}$ and get

$$\Phi(X) = \xi(e^{|X|}) = 2\sum_{k=1}^{\infty} \frac{e^{(1-2k)X}}{(2k-1)} \tag{3.12}$$

$$= 2\,(e^{|X|})^{-1} + \frac{2}{3}\,(e^{|X|})^{-3} + \frac{2}{5}\,(e^{|X|})^{-5} + ... \tag{3.13}$$

Since $|X|$ is (approximately) Gaussian, $e^{|X|}$ satisfies an (approximate) lognormal distribution. Hence, according to Proposition 3.2, $\Phi(X)$, the power sum of lognormal variables $e^{|X|}$ follows an (approximated) lognormal distribution. $\triangle$

Comment 3.1: Since $|X| \geq 0$, $|X|$ cannot be exactly Gaussian. If X is a Gaussian variable such that $\Pr(X \geq 0) >> \Pr(X < 0)$ (or $\Pr(X \leq 0) >> \Pr(X > 0)$), then $|X|$ equals X (or $-X$) most of the time and will follow the Gaussian distribution closely. Hence, when we let a Gaussian random variable X, whose probability mass is heavily concentrated on one side of the origin, be the input to $\Phi(\cdot)$, then the output, $\Phi(X)$, will follow an approximate lognormal distribution.

Proposition 3.4: [Distribution of $\Phi(X)$ with Lognormal Input] If X ($X \geq 0$) follows a lognormal distribution, then $\Phi(X)$ will follow a Gaussian distribution.

∇

Proof: Let $\Phi^{-1}(x)$ denote the inverse function for $\Phi(x)$. It is easy to verify that

$$\Phi^{-1}(x) = \ln(\frac{e^{|x|}+1}{e^{|x|}-1}) = \Phi(x). \tag{3.14}$$

Since $\Phi(X)$ is self-inversed, and since a Gaussian distribution at the input

to $\Phi(X)$ will produce a lognormal distribution at the output (Proposition 3.3), it follows that a lognormal distribution at the input to $\Phi(X)$ will produce a Gaussian distribution at the output. $\triangle$

3.3 Accuracy of Gaussian Approximation

This section provides a statistical analysis of when and how well the messages exchanged in the message-passing decoding of LDPC codes approximate the Gaussian distribution.

3.3.1 Validation of Gaussian Assumption in Message-Passing Decoding

Consider AWGN channels which are symmetric and memoryless: $\Pr(r_i = q|s_i = +1) = \Pr(r_i = -q|s_i = -1)$. Since LDPC codes are linear codes, without loss of generality, we take the all-zero codeword, mapped to $s_i = +1$ for all i, as the reference codeword. Further consider belief propagation on a Tanner graph with asymptotically unbounded girth, such that the probabilistic messages passed through different edges from variable nodes to check nodes (as well as from check nodes to variable nodes) follow an independent and identical distribution.

The variable node update and the check node update are formulated in (3.1) and (3.4), respectively. Initially, $m_{ij}^0 = m_{ji}^0 = 0$ for all i and j, and the LLR information m_i extracted from the Gaussian channel is Gaussian distributed. Thus, the first set of messages, $m_{i,j}^1$, passed from variable nodes to check nodes, follow a Gaussian

density.

Now suppose that the messages exchanged at $(\ell-1)$th iteration are Gaussian distributed. We wish to show whether or when Gaussianity is preserved through the variable node update and the check node update in ℓth iteration. The main result of this Section is stated in the below.

Theorem 3.1: [Gaussianity of Messages from Check Nodes] The outbound messages from check nodes to variable nodes at the ℓth iteration, m_{ji}^{ℓ}, can preserve Gaussianity from the previous iteration, provided that (i) the inbound messages, $m_{i'j}^{\ell-1}$,are reasonably reliable and that (ii) the degree of the check nodes is small.

∇

Proof: Consider the check node update in (3.4). Since +1's are transmitted, the condition that the inbound messages are reasonably reliable implies that the majority of $m_{i'j}^{\ell-1}$'s take positive values. From Proposition 3.3 and Comment 3.1, $|m_{i'j}^{\ell-1}|$ will then approximate a Gaussian distribution and so $\Phi(m_{i'j}^{\ell-1})$ will follow an (approximate) lognormal distribution. Further, $\Phi(m_{i'j}^{\ell-1})$'s are independent from each other because of the independent assumption for $m_{i'j}^{\ell-1}$'s. Now Proposition 3.2 states that only the sum of a small set of independent lognormal random variables will continue to be lognormal. Hence, $\sum_{i'\in N_v(j)\backslash\{i\}}\Phi(m_{i'j}^{\ell-1})$ will be lognormal when (and only when) the check node degree, $d_c(j)=|N_v(j)|$, is small, where $|.|$ means the size of a set. Finally, from Proposition 3.4 that a lognormal distribution at the input to $\Phi(\cdot)$ makes the output Gaussian, we get that $\Phi(\sum_{i'\in N_v(j)\backslash\{i\}}\Phi(m_{i'j}^{\ell-1}))$, and subsequently m_{ji}^{ℓ} follow Gaussian distributions.

The proof is best summarized as

$$m_{ji}^{\ell} = \Big[\prod_{i' \in N_v(j) \setminus \{i\}} \operatorname{sign}(m_{i'j}^{\ell-1}) \Big] \cdot \underbrace{\Phi\Big(\overbrace{\sum_{i' \in N_v(j) \setminus \{i\}} \overbrace{\Phi\big(\underbrace{m_{i'j}^{\ell-1}}_{Gaussian\ 1} \big)}^{lognormal\ 2}}^{lognormal\ 3} \Big)}_{Gaussian\ 4}, \tag{3.15}$$

where from "Gaussian 1" to "lognormal 2", it requires $m_{i'j}^{\ell-1}$ to be a Gaussian random variable with a small tailing probability (which asks for reliable messages to start with), and from "lognormal 2" to "lognormal 3" it requires the terms in the summation to be small (which corresponds to small check degrees). $\triangle$

Theorem 3.2: [Gaussianity of Messages from Variable Nodes] The outbound messages from variable nodes to check nodes ℓth iteration, m_{ij}^{ℓ}, preserves Gaussianity from the previous iteration.

∇

Proof: The result follows directly from the independence assumption and the fact the sum of independent Gaussian random variables is also a Gaussian random variable. $\triangle$

Comment 3.2: Theorem 3.2 is rather obvious, and is stated here solely for completeness. A comment is that, when the variable node degree $d_v(i) = |N_c(i)|$ is very large, according to the central limit theorem, the outbound messages from the variable nodes, $\sum_{j' \in N_c(i) \setminus j} m_{j'i}^{\ell}$, will behave like Gaussian even if the inbound messages, $m_{j'i}^{\ell}$, are not. In other words, when the code rate is very small, the discrepancy between the true message density and the Gaussian distribution, caused by the check node operation in a decoding iteration, can be mitigated by the succeeding

variable node operation.

Gathering Theorem 3.1 and Theorem 3.2, we have:

Corollary 3.1: [Validity of Gaussian Assumption] The LLR messages passed between the check nodes and the variable nodes of an LDPC code during the decoding iterations, as well as those produced at the output of the decoder, can be well approximated by Gaussian distributions, if (i) the input LLRs are reasonably reliable; and (ii) the check node degrees are not large.

3.3.2 Additional Comments and Simulation Verifications

This section further investigates the accuracy and applicable region of the Gaussian assumption in the iterative analysis for LDPC codes. From Theorem 3.1 and Theorem 3.2, two conditions need to be satisfied in order for the message density to approximate the Gaussian distribution well.

First, the messages passed along the edges need be reasonably reliable to start with. In general, the message reliability improves with iterations, but to ensure reliability in the first few iterations, the AWGN channel (or the "virtual" AWGN channel) on which the LDPC code operates needs to have a reasonably high SNR. To demonstrate the impact of channel SNR on the message density, we show in Figure 3.3 the histograms of messages passed from check nodes to variable nodes during the first iteration, L_{ji}^1, for a LDPC code whose check nodes degree is 4. As evident from the figure, the message density is very close to Gaussian at high SNRs (e.g. ≥ 1 db), but starts to deviate noticeably from Gaussian as the SNR drops low (e.g. ≤ 0 db).

Second, the degrees of the check nodes should not be large. The check node degree of a regular LDPC code relates to the variable node degree and the code rate by $d_c = d_v/(1-R)$ (assuming all the rows in the parity check matrix are linearly independent). For example, a constant variable node degree of 3 will require a constant check node degree to be 6 for rate 1/2, 9 for rate 2/3, 12 for rate 3/4, 15 for rate 4/5 and so on. This implies that the Gaussian approximation does work well for high-rate codes (such as rates above 0.8). For verification, Figure 3.4 provides the histograms of the messages m_{ji}^1 for a set of LDPC codes having the same variable node degree of 3 but different check node degrees. To eliminate the impact of channel SNR, we consider a sufficiently high SNR of 3 db. We observe that a check node degree of 30 and above (corresponding to rate 9/10 and above) has caused a large discrepancy from Gaussian density (at this SNR).

It should be noted that the two conditions we just discussed speak for different dimensions of the problem, and a favorable condition for one may mitigate the negative impact of the other. To evaluate the effect with both conditions combined, we show in Figure 3.5 the KS test values of the check node messages in the first iteration, m_{ji}^1, for different channel SNRs and check node degrees. The $D_{statistics}$ smaller than a critical value of 0.04, marked out in a solid horizontal line, indicate a close approximation to the Gaussian distribution. Not surprisingly, "a high SNR" points to different db values for codes with different check node degrees. For a regular LDPC code with check degree of 4, 0 db appears to be adequate, whereas for a code with check degree of 30, it requires some 4.5 db before the channel SNR is considered sufficient.

It is possible to combine the two conditions in one metric by defining the *error*

rate of check nodes as

$$P_E^{\ell} = \int_{-\infty}^{0^-} p_{check}^{\ell}(m)dm + \frac{1}{2}\int_{0^-}^{0^+} p_{check}^{\ell}(m)dm, \tag{3.16}$$

where $p_{check}^{\ell}(m)$ is the pdf of check nodes at ℓ-th iteration, which can be approximated using statistical histogram. The error rate of the check nodes indicates the accuracy of the Gaussianity through a threshold of around 0.2. For example, in Fig. 3.5, all the test points above the $D_{statistics} = 0.04$ horizontal line have corresponding P_E^{ℓ} larger than 0.2, and those below the line have P_E^{ℓ} smaller than 0.2. This observation also implies that the two conditions which ensure the Gaussianity are also indications of good error rate performance, or, the accuracy of the Gaussian message density is in line with the good performance of the LDPC code. Since the (average) variable node degree and the (average) check node degree are linearly proportional to each other for a given code rate ($d_v = d_c(1 - R)$), that the check node degree should not be large (Condition 2) suggests that the variable node degree should also be kept small. This is again in agreement with the empirical results that good LDPC ensembles generally have relatively small (average) variable node degrees between 3 and 6.

To summarize, the Gaussian assumption holds better for codes with low rates than with high rates, and for channels with high SNRs than with low SNRs. When evaluating practical scenarios where the code rate is generally smaller than $R = 0.85$ and the operating (Gaussian) channel has a reasonable quality, the Gaussian assumption holds good fidelity. However, when computing the asymptotic threshold which typically concerns a rather low SNR, the Gaussian approximation is less accurate and will likely affect the accuracy of the analytical result.

3.4 A New LDPC EXIT Formulation When Gaussian Assumption is Accurate

Having provided a statistical analysis of the accuracy and applicability of the Gaussian assumption for LDPC codes, we now discuss a few simplifications for tracking the message evolution and plotting the EXIT charts. Although an EXIT chart is essentially repeated application of density evolution on the two-part iterative decoder, its ability to visualize the trajectory of the probabilistic evolution as well as its elegant properties (such as the area property) make it extremely popular. At the emergence of the EXIT technique, a number of quantities, including the mean of the messages, the equivalent SNR, and the corresponding error probability, were used to describe the EXIT curves, until [6] showed that mutual information, arguably the single most important metric in information theory, is least sensitive to numerical artifacts and hence most accurate for characterizing EXIT curves.

Below we will first formulate a few simple, closed-form approximations to compute mutual information from LLR messages (Subsection 3.4.1), and further develop a simple new model to compute the EXIT curves (Subsection 3.4.2). The discussion throughout this section uses the Gaussian assumption.

3.4.1 Simplifying Computation of Mutual Information

Let $X \in \{+1, -1\}$ be a (coded) bit, and L_x be its associated LLR message with pdf $p_L(y)$. Since $p_L(y|X = +1) = p_L(-y|X = -1)$, the mutual information between

X and L_x can be computed using

$$I(X;L_x) = \int_{-\infty}^{\infty} p_L(y|X=+1) \cdot \log_2 \frac{2p_L(y|X=+1)}{(p_L(y|X=+1)+p_L(-y|X=+1))} dy. \tag{3.17}$$

Now suppose that the message L_x follows a Gaussian distribution with mean μ and variance σ^2, the mutual information can be simplified to

$$I(X;L_x) = I_{\mu,\sigma}(\mu,\sigma) \triangleq 1 - \frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{\infty} e^{-(y-\mu)^2/2\sigma^2} \log_2(1+e^{-y}) dy, \quad \text{(bit)}. \tag{3.18}$$

Considering $\sigma^2 = 2\mu$, we can define $I_\mu(\mu) \triangleq I_{\mu,\sigma}(\mu, \sqrt{2\mu})$ and $I_\sigma(\sigma) \triangleq I_{\mu,\sigma}(\sigma^2/2, \sigma)$, and rewrite the mutual information as

$$I(X;L_x) = I_\mu(\mu) = I_\sigma(\sigma). \tag{3.19}$$

Researchers have investigated simplifying mutual information computation and LDPC code design such as curve fitting [42], but most published results are for binary erasure channels (BEC), and do not easily extend to Gaussian channel due to the integral involved in the latter. One available result for AWGN, proposed in [42], approximates the mutual information in (3.19) as

$$I_\sigma(\sigma) = J(\sigma) \approx \begin{cases} a_{J,1}\sigma^3 + b_{J,1}\sigma^2 + c_{J,1}\sigma, & 0 \le \sigma \le \sigma^* \\ 1 - e^{a_{J,2}\sigma^3 + b_{J,2}\sigma^2 + c_{J,2}\sigma + d_{J,2}}, & \sigma^* \le \sigma \le 10 \\ 1, & \sigma \ge 10 \end{cases} \tag{3.20}$$

where $\sigma^* = 1.6363$, $a_{J,1} = -0.0421061$, $b_{J,1} = 0.209252$, $c_{J,1} = -0.00640081$,

$a_{J,2} = 0.00181491$, $b_{J,2} = -0.141675$, $c_{J,2} = -0.0822054$, $d_{J,2} = 0.0549608$, and approximates its inverse function as

$$J^{-1}(I) \approx \begin{cases} a_{sigma,1} I^2 + b_{\sigma,1} I + c_{\sigma,1}\sqrt{I}, & 0 \le I \le I^* \\ -a_{\sigma,2} \ln[b_{\sigma,2}(1-I)] - c_{\sigma,2} I, & I^* < I < 1 \end{cases} \tag{3.21}$$

where $I^* = 0.3646$, $a_{\sigma,1} = 1.09542$, $b_{\sigma,1} = 0.214217$, $c_{\sigma,1} = 2.33727$, $a_{\sigma,2} = 0.706692$, $b_{\sigma,2} = 0.386013$, $c_{\sigma,1} = -1.75017$.

Below we derive a few approximations for mutual information which are simpler (and more accurate) than (3.20). Our discussion herein is not limited to LDPC codes, but applicable to any code that permits EXIT analysis. We start by changing the logarithm in (3.19) from base 2 to base e and separating the mutual

information in several terms:

$$
\begin{aligned}
I_\sigma(\sigma) =& 1 - \frac{\log_2 e}{\sqrt{2\pi}\sigma}\left(\int_0^\infty e^{-(y-\frac{\sigma^2}{2})^2/2\sigma^2}\ln(1+e^{-y})dy + \right. \\
& \left. \int_{-\infty}^0 e^{-(y-\frac{\sigma^2}{2})^2/2\sigma^2}\ln(1+e^{-y})dy\right) \\
=& 1 - \frac{\log_2 e}{\sqrt{2\pi}\sigma}\left(\int_0^\infty e^{-(y-\frac{\sigma^2}{2})^2/2\sigma^2}\ln(1+e^{-y})dy + \right. \\
& \left. \int_{-\infty}^0 e^{-(y-\frac{\sigma^2}{2})^2/2\sigma^2}(\ln(1+e^{y})-y)dy\right) \\
=& 1 - \frac{\log_2 e}{\sqrt{2\pi}\sigma}\left(\underbrace{\int_0^\infty e^{-(y-\frac{\sigma^2}{2})^2/2\sigma^2}\ln(1+e^{-y})dy}_{Part_A} + \right. \\
& \left. \underbrace{\int_{-\infty}^0 e^{-(y-\frac{\sigma^2}{2})^2/2\sigma^2}\ln(1+e^{y})dy}_{Part_B} - \underbrace{\int_{-\infty}^0 e^{-(y-\frac{\sigma^2}{2})^2/2\sigma^2}ydy}_{Part_C}\right)
\end{aligned}
\tag{3.22}
$$

Next simplify the three parts one by one.

For $Part_A$, since $y \geq 0$, we have $e^{-y} \leq 1$. Using the Taylor series expansion, we get

$$
\ln(1+e^{-y}) = \sum_{k=1}^{\infty}(-1)^{k+1}\frac{e^{-ky}}{k} \tag{3.23}
$$

Plugging in (3.23) leads to

$$
\begin{aligned}
Part_A &= \int_0^\infty e^{-(y-\frac{x^2}{2})^2/2x^2}\sum_{k=1}^{\infty}(-1)^{k+1}\frac{e^{-ky}}{k}dy & (3.24)\\
&= \sum_{k=1}^{\infty}\frac{(-1)^{k+1}}{k}\int_0^\infty e^{-\frac{y^2+(2k-1)x^2y+\frac{x^4}{4}}{2x^2}}dy. & (3.25)
\end{aligned}
$$

Since

$$\int e^{-(ay^2+2by+c)}dy = \frac{1}{2}\sqrt{\frac{\pi}{a}}e^{\frac{b^2-ac}{a}}erf(\sqrt{a}y+\frac{b}{\sqrt{a}}), \tag{3.26}$$

where $erf(\cdot)$ is the error function, we get

$$\begin{aligned} Part_A &= \sum_{k=1}^{\infty}\frac{(-1)^{k+1}}{k}\frac{\sqrt{2\pi}\sigma}{2}\ e^{\frac{k^2-k}{2}\sigma^2}erf\Big(\frac{\sqrt{2}}{2\sigma}y+\frac{2k-1}{4}\sqrt{2}\sigma\Big)\Big|_0^{\infty} &(3.27)\\ &= \sum_{k=1}^{\infty}\frac{(-1)^{k+1}}{k}\frac{\sqrt{2\pi}\sigma}{2}\ e^{\frac{k^2-k}{2}\sigma^2}\left(1-erf\Big(\frac{2k-1}{4}\sqrt{2}\sigma\Big)\right) &(3.28)\end{aligned}$$

For $Part_B$, we have $y \leq 0$ and hence $\Rightarrow e^y \leq 1$. Following a similar procedure as with $Part_A$, we get

$$Part_B = \sum_{k=1}^{\infty}\frac{(-1)^{k+1}}{k}\frac{\sqrt{2\pi}\sigma}{2}\ e^{\frac{k^2+k}{2}\sigma^2}\left(1-erf\Big(\frac{2k+1}{4}\sqrt{2}\sigma\Big)\right). \tag{3.29}$$

For $Part_C$, we replace y with $y = t + \frac{x^2}{2}$:

$$Part_C = -\int_{-\infty}^{-\sigma^2/2} e^{-t^2/2\sigma^2}t\ dt - \frac{\sigma^2}{2}\int_{-\infty}^{-\sigma^2/2} e^{-t^2/2\sigma^2}dt \tag{3.30}$$

We again apply (3.26) and arrive at

$$Part_C = \sigma^2 e^{-\frac{\sigma^2}{8}} - \frac{\sqrt{2\pi}}{4}x^3(1-erf(\frac{\sqrt{2}}{4}x)). \tag{3.31}$$

To help combine $Part_A$ and $Part_B$, we further rewrite $Part_A$ in (3.25) by

separating the term $k = 1$ from all the others:

$$Part_A = \frac{\sqrt{2\pi}\sigma}{2}\left(1 - erf\Big(\frac{\sqrt{2}}{4}\sigma\Big)\right) + \sum_{k=1}^{\infty} \frac{(-1)^k}{k+1}\frac{\sqrt{2\pi}\sigma}{2} e^{\frac{k^2+k}{2}\sigma^2}\left(1 - erf\Big(\frac{2k+1}{4}\sqrt{2}\sigma\Big)\right). \tag{3.32}$$

Substituting for $Part_A$, $Part_B$ and $Part_C$ in (3.22) with (3.32), (3.29) and (3.31), we get

$$\begin{aligned} I_\sigma(\sigma) &= 1 - \big(\log_2 e\big)\left[\frac{\sigma}{\sqrt{2\pi}} e^{-\frac{\sigma^2}{8}} + \frac{2-\sigma^2}{4}\left(1 - erf\Big(\frac{\sqrt{2}}{4}\sigma\Big)\right) + \right. \\ &\left. \frac{\sqrt{2\pi}\sigma}{2}\sum_{k=1}^{\infty}\frac{(-1)^{k+1}}{k(k+1)} e^{\frac{k^2+k}{2}\sigma^2}\left(1 - erf\Big(\frac{2k+1}{4}\sqrt{2}\sigma\Big)\right)\right], \end{aligned} \tag{3.33}$$

which contains only the standard functions. As the number of terms in the summation approaches infinity, (3.33) converges to $I(\sigma)$. Since the impact of the terms become negligibly small for large k, it is a standard practice to truncate after a few terms to sacrifice a little accuracy in exchange for a significant reduction in complexity. Specifically, our experiments show that keeping only the first four terms ($k \leq 4$) suffices to give a close approximation. Table 3.1 lists the simulated distortion and more discussion will follow shortly.

Further, although $erf(\cdot)$ is considered a standard function, it nevertheless involves an integral. In the case when computing $erf(\cdot)$ is cumbersome, we can resort to curve fitting without going through the Taylor series expansion. Observing that $I_\sigma(\sigma)$ has a curve whose shape is very similar to the exponential function $1 - e^{-x^2}$, we hereby propose three possible forms of approximation:

1. Form 1: $\hat{I}(\sigma) = 1 - e^{-a\sigma^2}$;

2. Form 2: $\hat{I}(\sigma) = 1 - ae^{-b\sigma^2} - (1-a)e^{-c\sigma^2}$;

3. Form 3: $\hat{I}(\sigma) = 1 - e^{-a\sigma^b}$.

To help determine the parameters a, b and c, we use the mean squared error (MSE) as the figure of merit. The MSE is defined as

$$MSE = \frac{1}{N}\sum_{k=1}^{N}\left(\hat{I}_\sigma(\sigma_k) - I_\sigma(\sigma_k)\right)^2, \tag{3.34}$$

where $I(\cdot)$ is the true mutual information given in (3.19), $\hat{I}(\cdot)$ is the approximation, N is the number of test samples, and σ_k's are randomly generated positive real number. Through a computer-assisted search, we have found the following to be good parameters:

$$\text{Form 1: } a = 0.16, \tag{3.35}$$

$$\text{Form 2: } a = 0.83,\ b = 0.141,\ c = 0.355, \tag{3.36}$$

$$\text{Form 3: } a = 0.178,\ b = 1.894. \tag{3.37}$$

The MSE distortion of these three exponential forms and two truncated Taylor series that are truncated after $k = 1$ term and $k = 4$ terms are evaluated and listed in Table 3.1. For comparison purpose, the approximation proposed in [42] is also listed. The results are obtained over $N = 1000$ test samples with each σ randomly generated between 0 and 10. The reason we direct more attention to $\sigma \leq 10$ is because the mutual information becomes extremely close to 1 for $\sigma \geq 10$.

We see that Form 2 and Form 3 provide the best approximation with the least distortion. It is also worth noting that Form 3 has a simple inverse function, which is particularly useful as we derive a new EXIT formulation in the next subsection. The theorem below summarizes the results discussed in this subsection:

Theorem 3.3: [Closed-Form Approximation for Mutual Information $I_\sigma(\sigma)$] Under the Gaussian assumption, the mutual information between the (coded) bits and their LLR messages, as defined in (3.19), can be closely approximated by

$$\begin{aligned} I(X;L_x) = I_\sigma(\sigma) &\approx 1 - \left(\log_2 e\right) \left[\frac{\sigma}{\sqrt{2\pi}} e^{-\frac{\sigma^2}{8}} + \frac{2-\sigma^2}{4}\left(1 - erf\left(\frac{\sqrt{2}}{4}\sigma\right)\right) + \right. \\ &\quad \left. \frac{\sqrt{2\pi}\sigma}{2} \sum_{k=1}^{4} \frac{(-1)^{k+1}}{k(k+1)} e^{\frac{k^2+k}{2}\sigma^2} \left(1 - erf\left(\frac{2k+1}{4}\sqrt{2}\sigma\right)\right)\right] \quad (3.38) \\ &\approx 1 - 0.83e^{-0.141\sigma^2} - 0.17e^{-0.355\sigma^2}, \quad (3.39) \\ &\approx 1 - e^{-0.178\sigma^{1.894}}. \quad (3.40) \end{aligned}$$

Substituting $\mu = 2\sigma^2/2$ in (3.40), we get

Corollary 3.2: [Approximation for Mutual Information $I_\mu(\mu)$]

$$I(X;L_x) = I_\mu(\mu) \approx 1 - e^{-0.343\mu^{0.947}} \quad (3.41)$$

3.4.2 A New Formulation for Computing EXIT Charts

Having simplified the computation of mutual information, we now proceed to simplifying the computation of LDPC EXIT charts. The new formulation makes

explicit use of Form 3 which has a low distortion and a simple inverse function.

For notational convenience, define (i.e. rewrite (3.41))

$$\eta(x) \triangleq 1 - e^{-0.343x^{0.947}}, \quad x \geq 0, \tag{3.42}$$

whose inverse function is

$$\eta^{-1}(x) \triangleq -3.094\,(\ln(1-x))^{1.056}, \quad 0 \leq x \leq 1, \tag{3.43}$$

where $\ln(0) = -\infty$.

Let μ_A and μ_E be the mean of *a priori* and extrinsic LLRs, and I_A and I_E be the *a priori* and extrinsic mutual information, respectively. According to (3.41) in Corollary 3.2,

$$I_A = \eta(\mu_A), \quad I_E = \eta(\mu_E). \tag{3.44}$$

If the relation between μ_A and μ_E can be explicitly formulated for a given LDPC code, then its EXIT curves can be computed efficiently, obviating lengthy Monte Carlo simulations or complicated calculations. Further, as an immediate implication of the area property and the convergence property of EXIT charts, a channel code needs to be optimized such that the EXIT curves corresponding to the (two) local computing units match closely to each other [30]. Hence, a simpler EXIT formulation, when combined with curve fitting, also facilitates the design of the optimal degree profiles for LDPC codes.

Consider an LDPC code with check node degree d_c and variable node degree

d_v, the relation between the message mean of variable nodes, μ_v, and the message mean of check nodes, μ_c, is given by [4]

$$\mu_v = \mu_0 + (d_v - 1)\mu_c; \tag{3.45}$$

$$\mu_c = \varphi^{-1}(1 - (1 - \varphi(\mu_v))^{d_c - 1}), \tag{3.46}$$

where $\mu_0 = 4\text{SNR}$ (SNR here is not in the logarithm domain and does not use *db* as the unit) is the message mean from the Gaussian channel. The authors of [4] provided the definition of $1 - \varphi(x)$, as well as a closed-form approximation for $x \leq 10$. Our study leads to a similar form to that in [4], but uses different parameters and works well for the entire region of x:

$$\varphi(x) \triangleq \begin{cases} \dfrac{1}{\sqrt{4\pi x}} \displaystyle\int_{-\infty}^{\infty} \tanh \frac{u}{2} e^{-\frac{(u-x)^2}{4x}} du, & if\ x\ >\ 0; \\ 0, & if\ x\ =\ 0, \end{cases} \tag{3.47}$$

$$\approx 1 - e^{-0.432x^{0.88}}, \qquad x \geq 0, \tag{3.48}$$

Gathering (3.42), (3.45) (3.46) and (3.48), we obtain a simpler model for computing the EXIT charts:

Theorem 3.4: [EXIT Model for Regular LDPC Codes] Under the Gaussian assumption, the EXIT curves for a (d_v, d_c)-regular LDPC code with unbounded girth can be computed directly using the following relations between the *a priori* and

extrinsic mutual information:

$$\text{Variable nodes: } I_{E,v}(I_{A,v}, d_v) = 1 - e^{-0.343[4\gamma + 3.094(d_v-1)(-\ln(1-I_{A,v}))^{1.056}]^{0.947}}, \tag{3.49}$$

$$\text{Check nodes: } I_{E,c}(I_{A,c}, d_c) = \zeta_2([\zeta_1(I_{A,c})]^{d_c-1}), \tag{3.50}$$

where γ is the SNR of the underlying AWGN channel, and ζ_1 and ζ_2 are defined as

$$\zeta_1(x) = 1 - e^{-1.1671(-\ln(1-x))^{0.9293}}, \tag{3.51}$$

$$\zeta_2(x) = 1 - e^{-0.8468(-\ln(1-x))^{1.0761}}. \tag{3.52}$$

The new EXIT model in Theorem 3.4 is much less complex than the conventional model. To demonstrate its accuracy, we compare in Figure 3.6 the EXIT curves of a $(3, 6)$-regular LDPC code computed using Theorem 3.4 and the conventional Gaussian-approximated density evolution model (i.e. Equations (3.45) (3.46) and (3.19)). The X-axis denotes the extrinsic (*a priori*) mutual information for the check (variable) nodes, and the Y-axis denotes the extrinsic (*a priori*) mutual information for the variable (check) nodes.

From eye observation, the curves resulted from the new model agree extremely well with those from the conventional model. Applying the mean-squared-error test, we find that the MSE distortion between these EXIT curves is 1.5231×10^{-6} and 1.3459×10^{-4} for the variable nodes and the check nodes respectively, which further confirms the accuracy of the proposed new EXIT model.

Theorem 3.5: [EXIT Model for Irregular LDPC Codes] Consider an irregular LDPC code with unbounded girth. Let $\lambda(x) = \sum_i \lambda_i x^i$ and $\rho(x) = \sum_i \rho_i x^i$ be the respective generating function of the degree distributions for the variable nodes and check nodes, such that λ_i and ρ_i are the percentage of the degree-i variable nodes and check nodes. Assuming that the Gaussian assumption holds, the EXIT curves of this code can be computed using

$$\text{Variable nodes:} \qquad I_{E,v}(I_{A,v}, \lambda(x)) = \sum_i \lambda_i I_{E,v}(I_{A,v}, d_v = i), \tag{3.53}$$

$$\text{Check nodes:} \qquad I_{E,c}(I_{A,c}, \rho(x)) = \sum_i \rho_i I_{E,c}(I_{A,c}, d_c = i), \tag{3.54}$$

where $I_{E,v}(I_{A,v}, d_v = i)$ and $I_{E,c}(I_{A,c}, d_c = i)$ are given in (3.49) and (3.50), respectively.

∇

Proof: Theorem 3.5 follows from Theorem 4 and the fact that, for an irregular LDPC code, the EXIT curves are evaluated by averaging the extrinsic mutual information from all the variable nodes and the check nodes, respectively [30]. Specifically, let the irregular LDPC code has m variable nodes and n check nodes, and let $I_{E,v,i}(I_A)$ and $I_{E,c,j}(I_A)$ be the extrinsic mutual information associated with the ith variable node and the jth check node given *a priori* mutual information I_A ($1 \leq i \leq n$, $1 \leq j \leq m$). The average extrinsic mutual information exchanged

between the variable nodes and the check nodes is given by [30]

$$\text{Variable nodes:} \quad I_{E,v}(I_{A,v}) = \frac{1}{m}\sum_{i=1}^{m} I_{E,v,i}(I_{A,v}), \tag{3.55}$$

$$\text{Check nodes:} \quad I_{E,c}(I_{A,c}) = \frac{1}{n}\sum_{j=1}^{n} I_{E,c,j}(I_{A,c}). \tag{3.56}$$

Notice that $I_{E,v,i}(I_{A,v})$ is only dependent on the degree of the ith variable node and $I_{A,v}$, the (average) *a priori* mutual information available to variable nodes. Since the latter is the same for all the variable nodes, we can group the variables nodes having the same degree in (3.55), which immediately gives rise to (3.53). A parallel argument holds for the check nodes. $\triangle$

Theorem 3.5 complements Theorem 3.4 by making the same simple EXIT model work for irregular LDPC codes also. An exemplary EXIT chart computed using Theorem 3.5 is presented in Figure 3.7, where the irregular LDPC code has variable node and check node degree profile

$$\begin{cases} \lambda(x) = 0.6x^3 + 0.3x^4 + 0.1x^6; \\ \rho(x) = 0.7x^7 + 0.3x^8, \end{cases} \tag{3.57}$$

and the SNR being evaluated includes -2db, -1db and 2db.

The EXIT model presented in Theorems 3.4 and 3.5 speaks for the context of a single LDPC code operating on an AWGN channel, where check nodes and variable nodes are each viewed as a sub computing unit and each generates an EXIT curve. When an LDPC code is used in serial concatenation with another code or a modulation scheme, then the entire LDPC code is viewed as a sub computing unit in the global iterative system. The proposed new EXIT model can be easily

adapted to those cases by evaluating mutual information after a full variable- and check-node decoding iteration (instead of a half iteration), and by accounting for the Turbo principle.

Corollary 3.3: [EXIT Model for Concatenated LDPC codes] Consider an LDPC code being a component code in a serially concatenated system. Assume that the LDPC code has unbounded girth and that the LLR messages extracted from the channel and exchanged between different parts of the system follow the Gaussian distribution. Let $\lambda(x)$ and $\rho(x)$ be the variable node and check node degree profiles of the LDPC code, μ_0 be the mean of the LLR messages extracted directly from the channel, I_A be the extrinsic mutual information passed from the other component code to the LDPC code, and I_E be the extrinsic mutual information passed from the LDPC code to the other component code. The EXIT curve of this LDPC code, i.e. I_E as a function of I_A, can be derived using the following steps:

1. Before decoding: the *a priori* mutual information available to (the variable nodes of) the LDPC code is

$$I_1 = \begin{cases} \eta\big(\eta^{-1}(I_A) + \mu_0\big), & \text{LDPC is inner code} \\ I_A, & \text{LDPC is outer code} \end{cases}, \tag{3.58}$$

   where $\eta(\cdot)$ is defined in (3.42).

2. After check node update: the mutual information passed from the check nodes to the variable nodes inside the LDPC decoder is

$$I_2 = I_{E,c}(I_{A,c} = I_1,\, \rho(x)), \tag{3.59}$$

where $I_{E,c}(\cdot,\cdot)$ is defined in (3.54).

3. After variable node update: the total mutual information at the output of the variable nodes is

$$I_3 = I_{E,v}(I_{A,v} = I_2,\, x\lambda(x)), \tag{3.60}$$

where $I_{E,v}(\cdot,\cdot)$ is defined in (3.53).

4. Outbound message in accordance to the Turbo principle: the extrinsic mutual information passed from the LDPC code to the other component code is

$$I_E = \eta\big(\eta^{-1}(I_3) - \eta^{-1}(I_A)\big) \tag{3.61}$$

3.5 Evaluating EXIT Formulations When Gaussian Assumption is Less Accurate

The previous section has formulated a new LDPC EXIT model under the Gaussian assumption. As discussed in Section 3.3, when some conditions are not satisfied, the extrinsic message densities do not match well with the Gaussian distribution. This gives rise to several questions. First, how to properly track the evolution of the probabilistic information in order to perform iterative analysis for LDPC codes? The pioneering work of density evolution [4] and EXIT charts [42] recommends using Gaussian approximation (and the doubling rule $\sigma^2 = 2\mu$ that follows after) regardless, since the great simplicity more than outweighs the small degradation in accuracy. In practice, if the LLR messages are the *probabilistic information*

of interest, then the Gaussian assumption could lead to noticeable discrepancy, whereas the metric of mutual information is much less sensitive to pdf mismatch [42].

In the actual message passing, the LLR variance σ^2 may be either more or less than twice of the mean value μ. This observation, also reported in [7] for turbo codes, is not due to any violation of the symmetry condition, but rather the inaccuracy of the Gaussian assumption. When the doubling relation holds, the mutual information between data and their LLR messages may be evaluated using either $I_\mu(\mu)$ or $I_\sigma(\sigma)$ in (3.19), and both equal $I_{\mu,\sigma}(\mu, \sigma)$ in (3.18). Otherwise, these formulations lead to different values, causing different levels of discrepancy from the true value. Now what formulation to use? Do we have to track both μ and σ?

To answer these questions, in what follows, we provide a comparison study on the accuracy of different mutual information formulations. Please note that, unless otherwise stated, the assumption here is that LLR messages follow some Gaussian distribution with mean μ and variance σ^2, where μ and σ^2 do not necessarily relate to each other by a factor of 2. This assumption may appear self-contradicting, since, given that the symmetry condition [5] always holds, admitting Gaussian pdf implies $\sigma^2 = 2\mu$. The reason behind our assumption is two-fold. First, although less accurate than desired, Gaussian is nevertheless the best and simplest distribution to model the message density. Second, as we will show shortly, relaxing the doubling relation (but still holding on to the Gaussianity) may improve the accuracy of the results in certain cases. The true EXIT chart without any assumption, computed using (3.17) with the histogram serving as the pdf, will also be provided as the benchmark.

It should be noted that although $I_{\mu,\sigma}(\mu,\sigma)$ appears to involve one additional degree of freedom than $I_{\mu}(\mu)$ or $I_{\sigma}(\sigma)$, in essence, they are all one-dimensional functions. This is because $I_{\mu,\sigma}(\cdot,\cdot)$ can be rewritten as a function of a single parameter $t = \mu/\sigma$:

$$I_{\mu,\sigma}(\mu,\sigma) = I_{\mu/\sigma}(\mu/\sigma = t) \triangleq 1 - \int_{-\infty}^{\infty} \frac{e^{-(y-(t/\sqrt{2}))^2}}{\sqrt{\pi}} \log_2(1 + e^{-2\sqrt{2}yt}) dy. \quad (3.62)$$

From the discussion in Section 3.3, the Gaussian assumption and the doubling relation do not hold well in the low SNR region. Although hard to prove, as shown in the example in Fig. 3.8, the probability mass of the extrinsic LLR messages in this region tends to lean toward the left, causing $\sigma^2/\mu > 2$; see, for example, Fig. 3.8. Similar phenomena were also observed in [4] [6].

Theorem 3.6: Let μ and σ^2 be the mean and variance of the (*a priori* or extrinsic) LLR messages of any channel code at any decoding stage. If $\sigma^2/\mu > 2$, then

$$I_{\mu/\sigma}(\mu/\sigma) < I_{\mu}(\mu) < I_{\sigma}(\sigma). \quad (3.63)$$

Proof: By definition,

$$I_{\sigma}(\sigma) = I_{\mu,\sigma}(\sigma^2/2, \sigma) = I_{\mu/\sigma}(\sigma/2), \quad (3.64)$$

$$I_{\mu}(\mu) = I_{\mu,\sigma}(\mu, \sqrt{2\mu}) = I_{\mu/\sigma}(\sqrt{\mu/2}). \quad (3.65)$$

From $\sigma^2/\mu > 2$, we get

$$\sigma/2 > \sqrt{\mu/2}, \tag{3.66}$$

$$\text{and} \quad \sqrt{\frac{\mu}{2}} > \sqrt{\frac{\mu}{\sigma^2/\mu}} = \mu/\sigma. \tag{3.67}$$

Since $I_{\mu/\sigma}(\cdot)$ is a strictly monotonically increasing function,

$$I_{\mu/\sigma}(\mu/\sigma) < I_{\mu/\sigma}(\sqrt{\mu/2}) < I_{\mu/\sigma}(\sigma/2). \quad \square \tag{3.68}$$

Comment 3.3: When $\sigma^2/\mu < 2$, the order of the three mutual information results will simply reverse. It appears, however, that the case of $\sigma^2/\mu < 2$ does not occur with LDPC codes: at low SNRs, $\sigma^2/\mu > 2$, and at high SNRs, σ^2/μ converges to 2 from above.

To assess the relative accuracy of $I_{\mu/\sigma}$, I_μ and I_σ requires the knowledge of the true mutual information. The true mutual information between data X and their LLR messages L_x is evaluated using (3.17), where the message pdf $p_L(y)$ satisfies the symmetry condition but not necessarily the Gaussian distribution. Due to the lack of formal methods to track the exact $p_L(y)$, the assessment here has to rely on the statistics collected from extensive experiments.

We have tested a vast number of cases, each at a relatively low SNR, and evaluated the extrinsic mutual information from the check nodes with $d_c = 6$ after the first decoding iteration. In each test, $200,000$ samples are collected to form the pdf histogram from which μ, σ^2, and the true mutual information $I(X; L_x)$ are derived. μ and σ^2 are then used to evaluate the approximated values $I_{\mu/\sigma}$, I_μ

and I_σ. To minimize the distortion associated with the experiments, 50 tests are performed for each case which represents a particular combination of an LDPC code and an operating SNR, and the average results over these tests are recorded. Some of these average results are provided in Table 3.2. For clarity, we list the true mutual information $I(X; L_x)$ and the distortion introduced by $I_{\mu/\sigma}$, I_μ, I_σ and $(I_{\mu/\sigma} + I_\mu)/2$, respectively.

The experimental results are very consistent. The true mutual information $I(X; L_x)$ largely locates between $I_{\mu/\sigma}$ and I_μ. In general, I_σ introduces a higher distortion than either I_μ or $I_{\mu/\sigma}$, and should therefore be avoided. Both I_μ and $I_{\mu/\sigma}$ provide quite accurate approximations, with distortion of less than a couple of percent. Since $I_{\mu/\sigma}$ tends to be slightly pessimistic while I_μ tends to be slightly optimistic[2], taking an average on them effectively cancels out the distortion on both sides, resulting in a truly good approximation whose distortion is only a fraction of a percent. Finally, although not shown, we have performed similar tests on turbo codes and found that $\frac{1}{2}(I_\mu + I_{\mu/\sigma})$ therein also provides very close approximation to $I(X; L_x)$.

The examples in Table 3.2 are evaluated in the first decoding iteration. To see how the accuracy of different mutual information formulas evolve with iterations, we compare a set of EXIT curves resulted from different methods. The true EXIT curves without any assumption (not even the Gaussian assumption on the in-edge messages), computed using the discretized density evolution with the histogram serving as the pdf, is provided as the benchmark to the approximated values. The complete EXIT chart and its close-up shot are shown in Figure 3.9.

[2]Exceptions exist, e.g., the last case in the table. In the exception case, both I_μ and $I_{\mu/\sigma}$) cause only a few thousandths in distortion.

Table 3.1: MSE Distortion for different approximations to compute mutual information.

Name of Approximation	MSE Distortion
Form 1	0.10358
Form 2	1.5913×10^{-7}
Form 3	1.5767×10^{-6}
Truncated Taylor $k = 1$	2.9424×10^{-4}
Truncated Taylor $k \leq 4$	4.7839×10^{-6}
Approximation in [42]	1.759×10^{-3}

Table 3.2: The extrinsic mutual information generated from the check nodes with $d_c = 6$ at the first decoding iteration.

SNR(db)	$\frac{variance}{mean}$	$I = I(X; L_x)$	$I_{\mu/\sigma} - I$	$I_\mu - I$	$I_\sigma - I$	$\frac{1}{2}(I_{\mu/\sigma} + I_\mu) - I$
-1	2.1899	0.0929	-0.0058	0.0020	0.0103	-0.0019
-0.5	2.1990	0.1264	-0.0072	0.0036	0.0153	-0.0018
0	2.2009	0.1685	-0.0088	0.0054	0.0206	-0.0017
0.5	2.1873	0.2208	-0.0103	0.0064	0.0243	-0.0019
1	2.1641	0.2820	-0.0108	0.0072	0.0260	-0.0018
1.5	2.1217	0.3524	-0.0096	0.0063	0.0226	-0.0017
2	2.0677	0.4294	-0.0058	0.0043	0.0146	-0.0007
2.5	2.0032	0.5105	0.0005	0.0010	0.0015	0.0007

For the variable-node EXIT curve, different mutual information formulas exhibit negligibly small differences and all can be considered sufficiently accurate. For the check-node EXIT curve, observations similar to those from Table 3.2 can be noted, with I_σ being the least accurate, $(I_{\mu/\sigma} + I_\mu)/2$ being the most accurate, and $I_{\mu/\sigma}$ and I_μ each being reasonably accurate. Considering the complexity, we recommend computing EXIT curves using either I_μ or $(I_{\mu/\sigma} + I_\mu)/2$, where the latter trades a little more complexity for an even higher accuracy.

In Section 3.4, we developed a simple EXIT model based on $I_\mu(\mu)$. The close proximity of I_μ to the real mutual information, insensitive to the accuracy the Gaussianity, is thus clear indication that the new EXIT model can also find useful application where the Gaussian assumption holds less well. A verifying example is provided in Fig. 3.10, whose EXIT curves are computed using the new model and compared to the exact density evolution (without any assumption). Although the $(3, 6)$-regular LDPC code is evaluated at an SNR of -1 and -2 db, a region where the LLR messages deviate noticeably from the Gaussian distribution (at least in the first few iterations), the two EXIT curves nevertheless match very well.

3.6 Conclusion

While the prevailing assumption that LLR messages follow Gaussian distributions, and the simplicity it brings to density evolution and EXIT analysis, contribute substantially to the flourishing of iterative analysis, its theoretical justification is largely lacking. This work fills the gap for LDPC codes by performing a statistical analysis for when, how and how well the Gaussian distribution approximates

the real message densities in the iterative decoding. The impact of the Gaussian assumption on the accuracy of the EXIT charts is then investigated, and new approximations for mutual information and a new EXIT model are developed. The major contributions are summarized as follows:

1. We performed statistical analysis and showed that the Gaussian assumption is accurate when the initial LLR messages from the channel (or the front-end modulator/detector) have reasonable reliability *and* when the check node degrees are not large.

2. In the cases when the LLR messages deviate noticeably from the Gaussian distribution, we evaluated the accuracy of the mutual information between bits and their LLR messages (and subsequently the EXIT curves), computed using different formulations under the Gaussian assumption. We showed that $(I_\mu + I_{\mu/\sigma})/2$ provides an extremely close match to the true mutual information, the simpler form I_μ is also quite accurate, but I_σ results in a large discrepancy and should be avoided. Hence, EXIT analysis can be accurate even when the underlying Gaussian assumption is not, provided that one uses the right formulation for mutual information.

3. We derived several simple but good approximations to compute the mutual information using I_μ. We also developed a simple analytical model, consisting of closed-form expressions only, to compute the EXIT charts for regular and irregular LDPC codes. The new EXIT model provides an efficient alternative to the conventional model to analyze LDPC code ensembles.

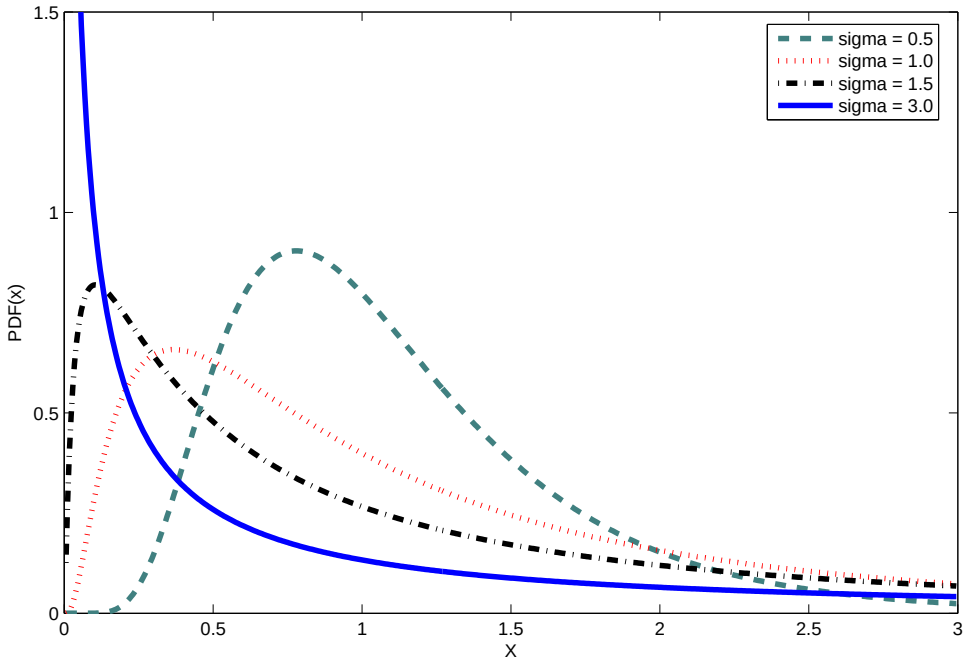


Figure 3.1: Illustration of lognormal pdf's $\mu = 0$ and $\sigma = 0.5, 1.0, 1.5, 3.0$.

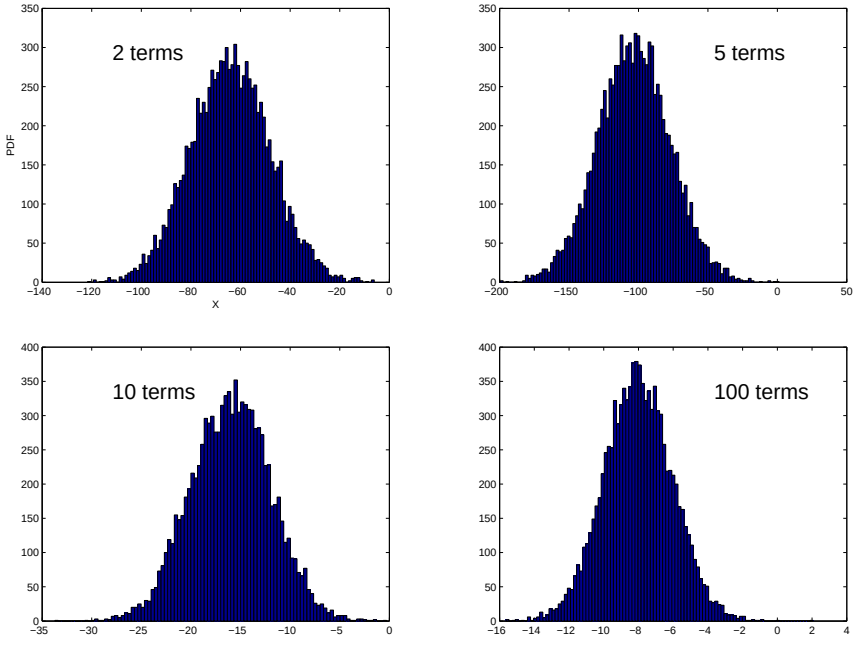


Figure 3.2: Histograms for $\ln(S)$ with $k = 2, 5, 10, 100$.

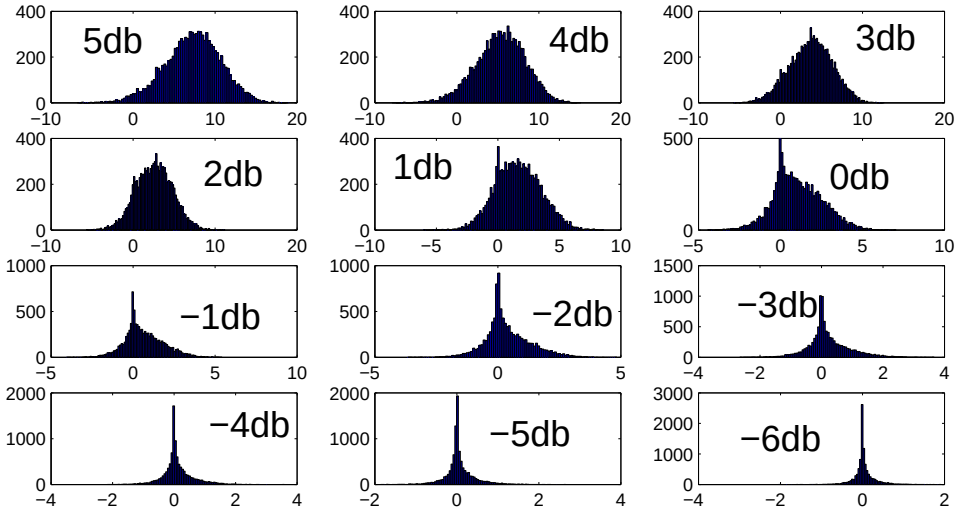


Figure 3.3: Histogram of messages m_{ji}^1 for a LDPC code with $d_c = 4$ at different SNRs.

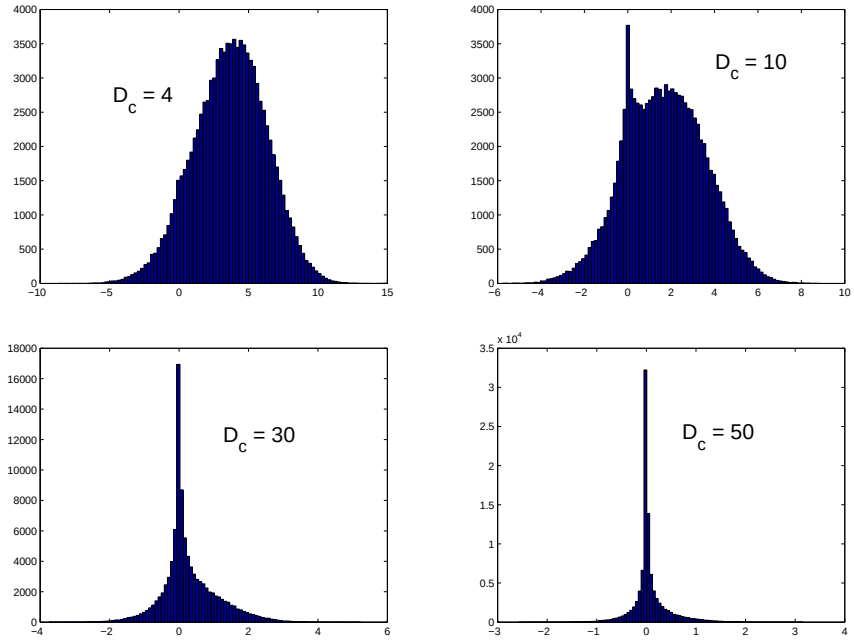


Figure 3.4: Histograms of message m_{ji}^1 for regular LDPC codes with variable node degree 3 and different check node degrees (SNR=3db).

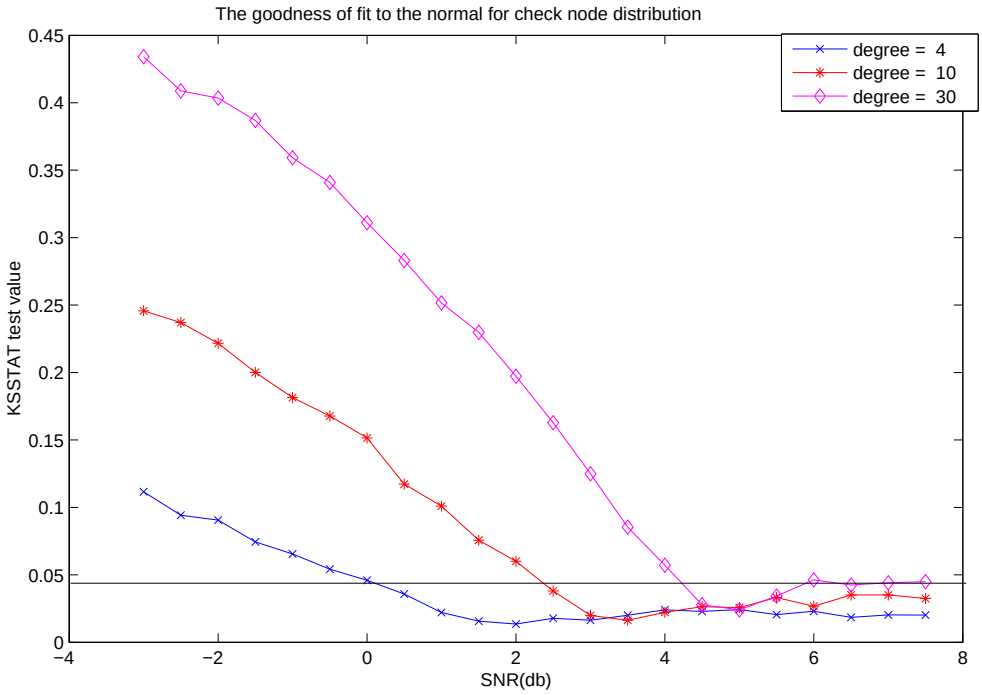


Figure 3.5: D-statistic collected from the KS test for codes with different check node degrees operating on different channel SNRs. $d_v = 3$, $d_c = 4, 10, 30$. D-statistics below the solid horizontal line correspond to cases where the Gaussian assumption holds well.

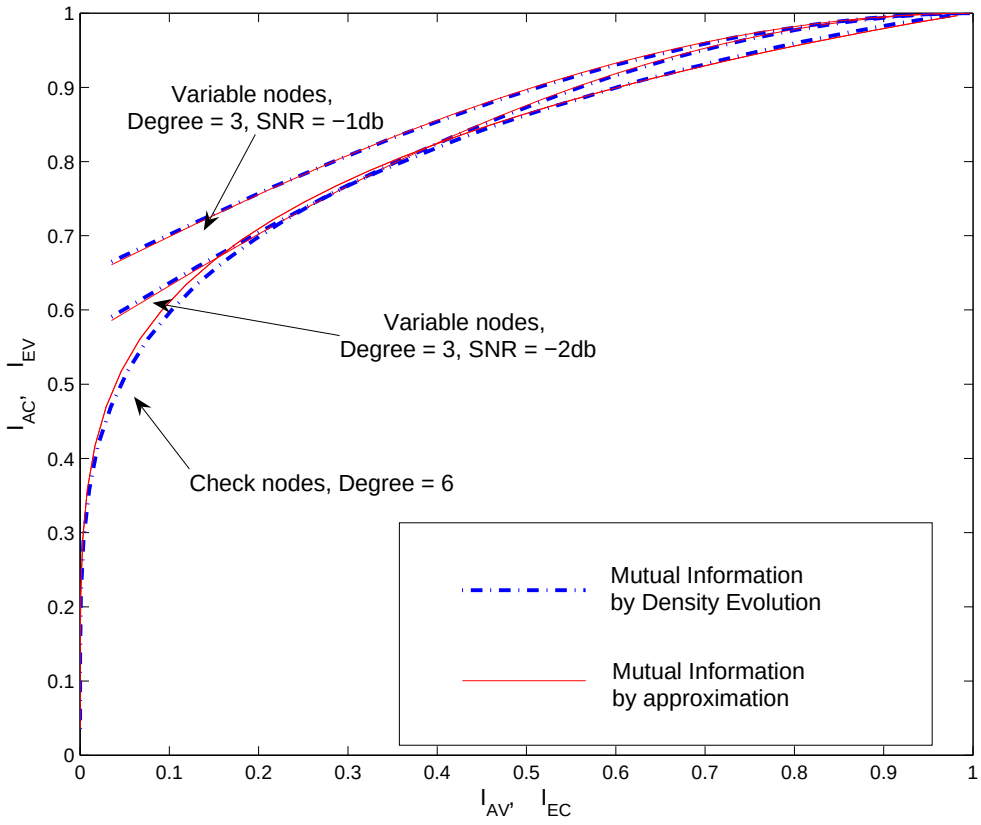


Figure 3.6: Comparison of EXIT charts of a $(3, 6)$-regular LDPC code computed by Theorem 3.4 and the conventional density evolution. SNR=$\{$-1, -2$\}$ db.

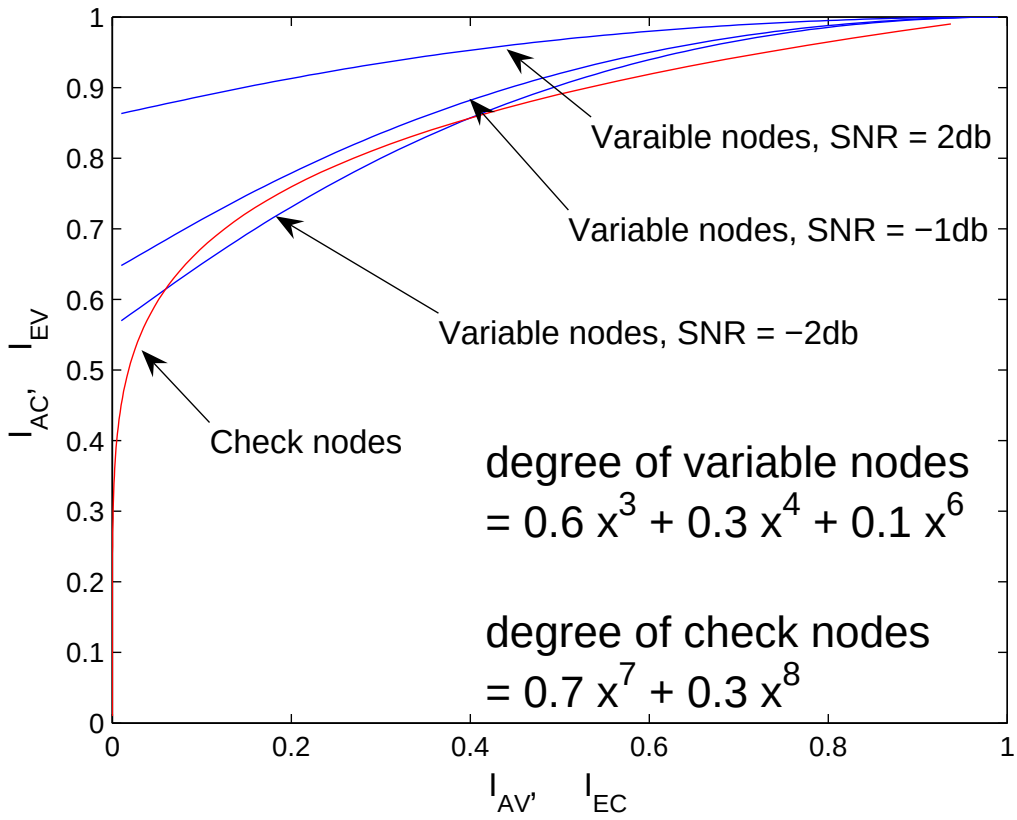


Figure 3.7: EXIT chart of irregular LDPC code at $SNR = \{-2db, -1db, 2db\}$

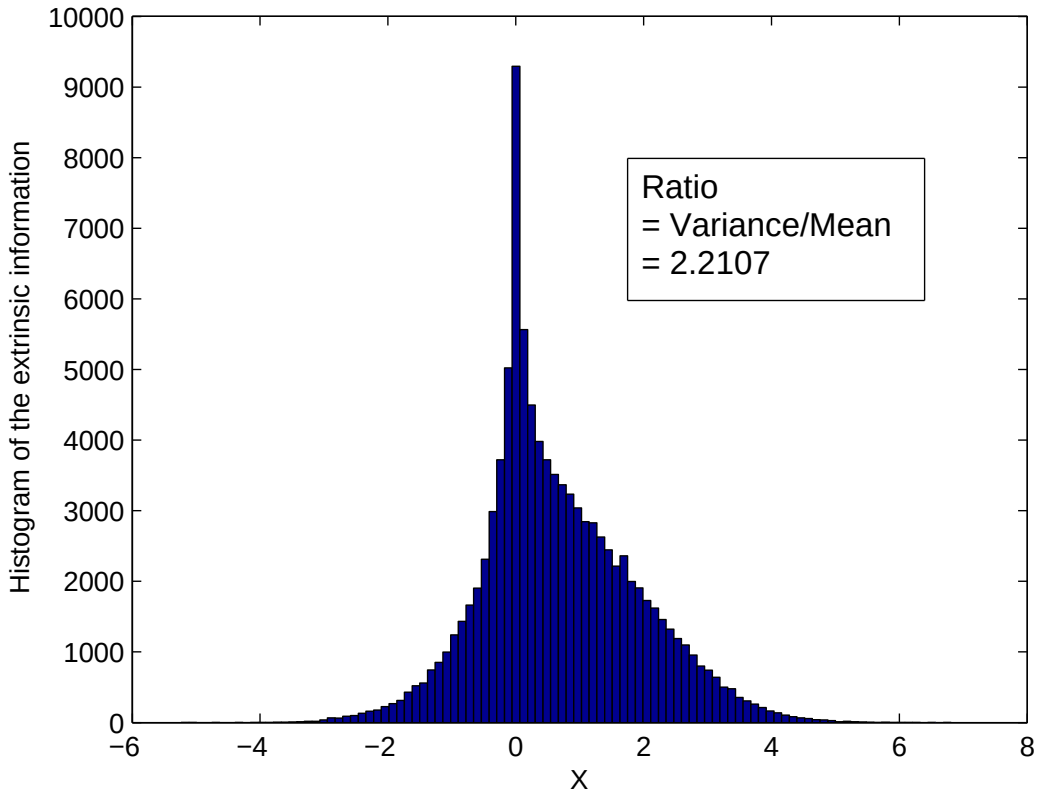
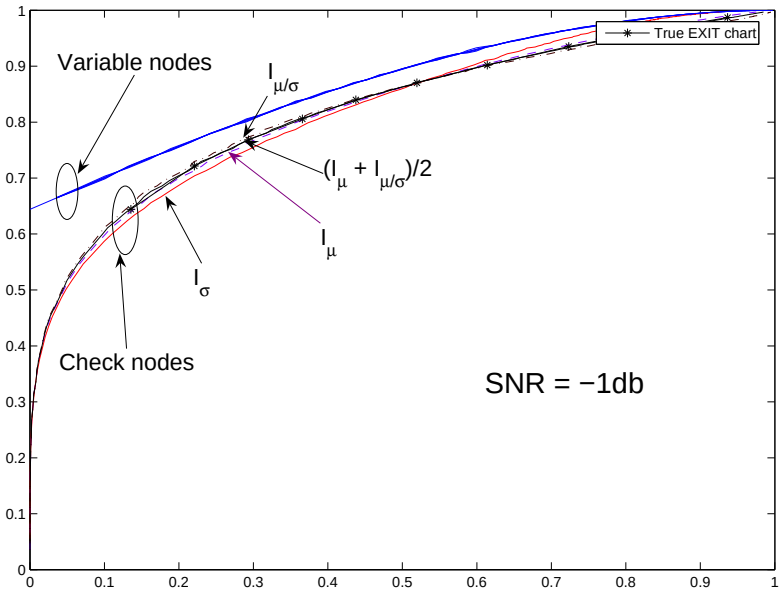
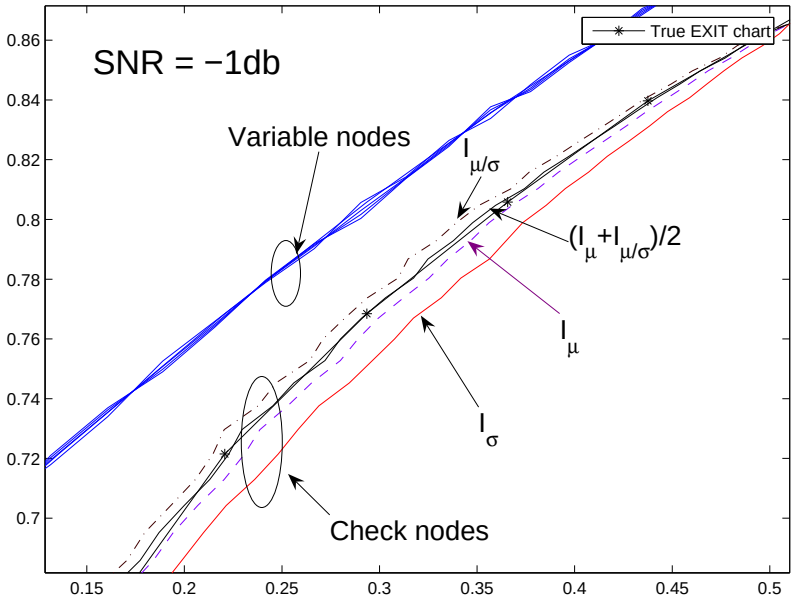


Figure 3.8: The pdf of the extrinsic LLR messages from the check nodes to the variable nodes, after one decoding iteration on an AWGN channel of 0.5 db. The check nodes have degree 6.



(A)



(B)

Figure 3.9: EXIT curves computed using different formulations. (A) The complete EXIT chart. (B) The zoomed-in EXIT chart.

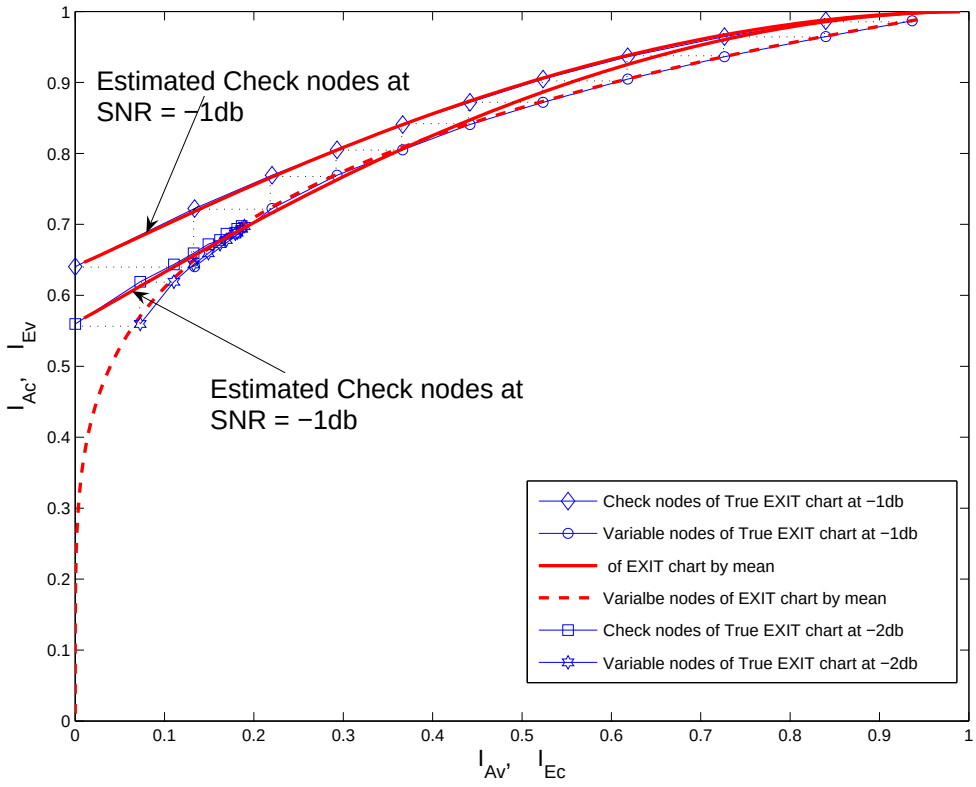


Figure 3.10: Comparison of the EXIT curves computed using the proposed new model and using the exact density evolution (without any assumption) in regions where the Gaussian assumption is not accurate. $(3, 6)$-regular LDPC codes. Channel SNR is -1 db and -2 db.

Chapter 4

Analog Coding and Linear analog coding

The first work of *analog error correction coding* traces back to the early 80's, when Marshall and Wolf independently introduced the concept [43–46]. It was termed *real number coding* in Marshall's work and *analog coding* in Wolf's work. Early ideas of analog coding are a natural outgrowth of digital error correction coding, by extending conventional digital error correction codes from the finite field to the real-valued or the complex-valued field (i.e. symbols from a very large finite field can approximate real values). Hence, linear codes prevail the short literature of analog coding, just as they do in digital coding. There have also been proposals of nonlinear analog codes, but the study is rather limited.

The fundamental idea of error correction coding is to enlarge the distances among the codewords by mapping, for example, a small space to a larger space.

The notion of distance is therefore of critical importance to coding theory and code evaluation. Since much of the development of analog codes follows a similar path as digital codes, Hamming distance, a key metric in digital codes, was also taken as a figure of merit in analog codes; namely, the distance between two (analog) codewords was also measured by the number of different symbols between them. While the adoption of Hamming distance has also lead to the adoption of related concepts such as maximum distance separability, Hamming distance is not nearly as indicative in the analog domain as in the digital domain.

Aiming at advancing the theory and practice of analog codes, in this dissertation, we develop several new concepts for analyzing and understanding analog codes, including the encoding power gain, minimum (Euclidean) distance/squared weight ratio and its achievable upper bound, and the minimum MSE distortion and its achievable lower bound. For linear analog codes, we define a concept of maximum distance ratio expansible (MDRE) (with respect to squared Euclidean distance), a concept similar in spirit to maximum distance separable (MDS) (with respect to Hamming distance). We show that MDRE codes can achieve the best (i.e. largest) squared Euclidean distance ratio and the best (i.e. smallest) average Euclidean distortion among all linear analog codes. In this, we show that all MDRE codes perform exactly the same on AWGN channels when evaluated by mean square error (MSE) metric. We identify linear analog codes that are MDRE, as well as codes that are MDS. We show that MDRE codes and MDS codes, although evaluated against different distance metrics, do not have to conflict each other, and can actually be unified in the code design. We also proposed the concept of maximum squared Euclidean distance ratio to analog codes, and show that it is a rather effective tool in indicating the performance of an analog code.

On the more practical side, we also study the two categories of analog codes: linear codes and nonlinear codes. We summarize the existing analog codes and demonstrate a few new codes we designed. We apply our newly-developed concepts and tools on these codes to reveal useful properties. We also study maximum likelihood (ML) decoding algorithms for these codes. One important conjecture that results from our study is that, although linear digital codes are sufficient in achieving the channel capacity of additive white Gaussian noise (AWGN) channels, linear analog codes are inadequate in handling Gaussian noise. The majority of the existing linear analog codes, such as analog Bose-Chaudhuri-Hocquenghem (BCH) codes, grow out of their digital counterparts by extending the supporting finite field to an infinite size. Hence, when they are decoded through a BCH-like decoder (e.g. a modified Berlekamp-Massey and Forney algorithm), they can only survive pulse noise that occurs only to a limited positions in the codeword. Even with an ML decoder, these analog linear codes are rather weak in handling ambient noise that occurs everywhere in the codeword. We provide a geometric method to illustrate how and why, and point to nonlinear codes as the solution for analog error control. Specifically, we show that chaotic systems, an special class of nonlinear dynamical systems, can play an important role in analog coding. We demonstrate a few novel designs for nonlinear chaotic analog codes, whose "butterfly effect" can lead to surprisingly good performance.

4.1 Theory and Concepts for Analog Codes and Linear Analog codes

In what follows, we will always use bold fonts, such as $\mathbf{G}$ and $\mathbf{u}$, to denote vectors or matrices, and use regular fonts, such as n and R_w to denote scalars. Further, superscript ^{T} denotes simple transpose of a vector or matrix, while superscript ^{H} denotes Hermitian transpose. By default, all the codes have parameters (n, k), and map a length-k discrete-time analog source sequence $\mathbf{u} = (u_0, u_1, \cdots, u_{k-1})^T$ to a length-n discrete-time analog codeword $\mathbf{v} = (v_0, v_1, \cdots, v_{n-1})^T$.

4.1.1 Definition of Analog Error Correction Codes

One difference between an analog code and a digital code is error tolerance. A digital error correction code has a discrete codeword space, and hence a small (analog) distortion on each finite-field symbol can usually be rounded off. In comparison, an analog symbol takes a value in a continuum of real or complex space, and therefore has zero tolerance to analog noise. Hence, unlike a digital error correction code, an analog error correction code can never achieve the error-free transmission of a codeword across an AWGN channel. In general, an analog code will always result in some non-zero distortion on AWGN channels. As long as the distortion is controlled under a desirable level, then the analog code can still find good use in transmitting analog sources (e.g. audio and video signals) as well as digital signals (e.g. the first five digits after the decimal are correct with a high probability).

Before introducing new concepts and tools for analog codes, below we first

provide a few basic definitions of an analog error correction code.

Definition 4.1: [Analog codes] Consider a mapping $\mathbf{u} = (u_0, u_1, \cdots, u_{k-1})^T \xrightarrow{\mathbb{C}} \mathbf{v} = (v_0, v_1, \cdots, v_{n-1})^T$, which transforms a length-k sequence $\mathbf{u}$ belonging to the space $\mathbb{U}^k$ to a length-n sequence $\mathbf{v}$ belonging to the space $\mathbb{V}^n$, where $u_i \in \mathbb{U}$ for $0 \leq i < k$, and $v_j \in \mathbb{V}$ for $0 \leq j < n$. If the sets $\mathbb{U}$ and $\mathbb{V}$ are both continuums (or the union of a finite number of continuums) of real or complex values, then the above mapping defines an *analog code* $\mathcal{C}(n, k)$. Here, k/n is termed the rate of the code, and the vector $\mathbf{v}$ is termed a codeword. If any source sequence $\mathbf{u}$ is also part of its corresponding codeword $\mathbf{v}$, then the analog code is said to be systematic.

From the Nyquist sampling theorem, any continuous-time waveform can be represented by a sequence of discrete-time samples without any loss of information, provided that the sampling rate is at least twice as fast as the bandwidth of the original waveform. Hence, any continuous-time waveform can be sampled, encoded through an analog code $\mathcal{C}(n, k)$, and subsequently decoded and interpolated to form another continuous-time waveform, and k/n would be the ratio between the bandwidths of the original waveform and the transformed waveform.

Similar to digital codes, analog codes can also be categorized into two groups with code rate $k/n > 1$ and $k/n \leq 1$ respectively. The former corresponds to compression or source coding, and the latter corresponds to error correction or channel coding, which is the subject of this work. In what follows, unless otherwise stated, the term analog coding/codes (and similarly digital coding/codes) refers to analog error correction coding/codes (digital error correction coding/codes).

In what follows, we will discuss both linear analog codes and nonlinear analog

codes. Since all linear analog codes can be expressed in the form of linear analog block codes, our discussion on linear analog codes will therefore focus on block codes. We now specify a few definitions and notations that will be used in the discussion. Some of the concepts and theorems developed in this section apply to both linear and nonlinear analog codes, while others are specific to linear (block) analog codes.

Definition 4.2: [Linear analog block codes] A linear analog block code $\mathcal{C}(n,k)$ can be defined by its generator matrix $\mathbf{G}$. A discrete-time analog source stream, whose values may either be real or complex, is fed into the analog encoder in blocks of k symbols each. Each block $\mathbf{u} = \{u_0, u_1, ..., u_{k-1}\}^T$ of length k analog symbols is encoded to a codeword $\mathbf{v} = \{v_0, v_1, ..., v_{n-1}\}^T$ of length n analog symbols, through a linear matrix operation:

$$\mathbf{v} = \mathbf{G}^{\mathbf{H}}\mathbf{u}. \tag{4.1}$$

where $\mathbf{G}$ is a $k \times n$ matrix of rank k. All the elements u_i and v_i may be complex numbers or real numbers: $u_i, v_i \in \mathbb{C}$ or $\mathbb{R}$. The support sets for $\mathbf{u}$ and $\mathbf{v}$ are called the source space $\mathbb{U}^k$ and the codeword space $\mathbb{V}^n$ respectively.

The codeword $\mathbf{v}$ is put onto a channel with additive noise $\mathbf{w}$, which results in a noisy vector $\mathbf{r}$ at the receiver,

$$\mathbf{r} = \mathbf{v} + \mathbf{w} \tag{4.2}$$

The decoder produces an estimate $\tilde{\mathbf{u}}$ of the original source vector $\mathbf{u}$. The system model is shown in Fig. 4.1.

An analog code $\mathcal{C}(n,k)$ can be regarded as a mapping from a subspace $\mathbb{U}^k$ of

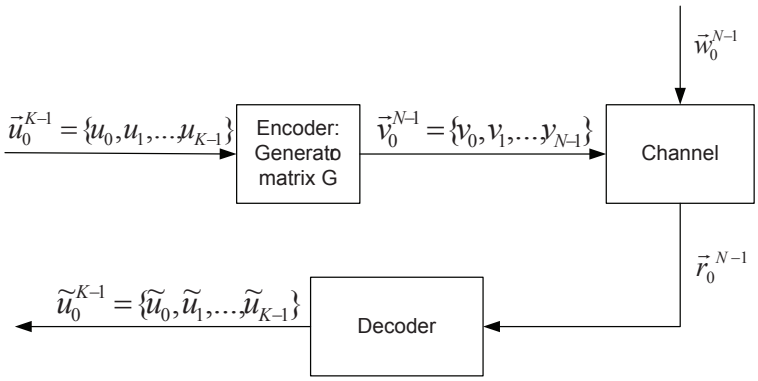


Figure 4.1: The system model of a general analog code.

the k-dimensional real-valued space $\mathbb{R}^k$ or complex-valued space $\mathbb{C}^k$ to a subspace $\mathbb{V}^n$ of the n-dimensional space $\mathbb{C}^n$ or $\mathbb{R}^n$ with a transform matrix $\mathbf{G}$. With the expanded distance, the distortion due to additive noise can be reduced by finding the closest or the most likely vector within the subspace $\mathbb{V}^n$.

Similar to digital error correction codes, a parity check matrix $\mathbf{H}$ can be defined for linear analog block codes, where $\mathbf{HG}^H = 0$. The syndrome $\mathbf{s}$ is computed as $\mathbf{s} = \mathbf{Hr}$, where $\mathbf{r} \in \mathbb{C}^n$ or $\mathbb{R}^n$. For any valid codeword $\mathbf{v}$, its syndrome satisfies $\mathbf{s} = \mathbf{Hv} = \mathbf{0}$.

4.1.2 Euclidean Weight and Squared Euclidean Weight Ratio

In digital error correction, the separation between two codewords is typically measured by Hamming distance, and a concept of minimum (Hamming) distance is defined to evaluate the "space expansion" capability of a code. If the minimum Hamming distance achieves the Singleton bound, then the corresponding code is

called a maximum distance separable code or Hamming-distance-optimal. When the digital code operates on a Gaussian noise channel, one can also define the Euclidean distance for the digital error correction code, and use a similar concept of "minimum Euclidean distance" to measure the code performance. In the digital context, both Hamming distance and Euclidean distance are equivalent in essence.

However, the situation becomes rather different in the analog context. First, although a similar concept of minimum Hamming distance can be defined for analog codes, the metric is not really useful in evaluating the code performance. Second, Hamming distance and Euclidean distance no longer have a one-to-one correspondence in the analog domain. In fact, the minimum Euclidean distance of a linear analog code always approaches 0 (please refer to Theorem 4.2), making this metric useless in the analog coding context. This calls for the development of new and more appropriate metrics for evaluating analog codes.

The structural goodness of a code is determined by the codeword space (termed the "code book" in the DECC literature) and the mapping between the source space and the codeword space (termed the "encoding function" in the DECC literature). For linear block codes, that is completely determined by the generator matrix $\mathbf{G}$. Consider two generator matrices $\mathbf{G}'$ and $\mathbf{G}$, where one is a scaled version of the other (i.e. $\mathbf{G}' = a\mathbf{G}$, where $a > 1$ is a real-value scalar). Apparently, these two codes have essentially the same code structure, and, although one may appear to have expanded the distances more than the other, the gain comes with a comparable cost of a higher (average) transmission power. For fair comparison and analysis, one should constrain the transmission power of all the analog codes at the same level. That is, the ratio between the average codeword power and the

source vector power should be limited to the same number or be normalized to 1. With this, we first define the power gain of a generator matrix $\mathbf{G}$.

Definition 4.3: [Encoding power gain] The encoding power gain Γ of an analog code is defined as the ratio between the average codeword power and the average source vector power:

$$\Gamma = \frac{\int P(\mathbf{u})\mathbf{v}^H\mathbf{v}d\mathbf{u}}{\int P(\mathbf{u})\mathbf{u}^H\mathbf{u}d\mathbf{u}} \tag{4.3}$$

where $P(\mathbf{u})$ is the probability density function (pdf) of the source vector $\mathbf{u}$, and $\int f(\mathbf{v})d\mathbf{v}$ represents the multiple integrals

$$\int\int\cdots\int f(v_0, v_1, \cdots, v_{n-1})dv_{n-1}\cdots dv_1 dv_0. \tag{4.4}$$

Theorem 4.1: For a given linear analog codes, suppose that $\mathbf{u}$ is uniformly distributed in the source space $\mathbb{R}^k$ and encoded by the generator matrix $\mathbf{G}$, then the encoding power gain is given by $\Gamma = trace(\mathbf{G}\mathbf{G}^H)/k$.

Proof:

$$\begin{aligned}
\Gamma &= \frac{\int P(\mathbf{u})\mathbf{v}^H\mathbf{v}d\mathbf{u}}{\int P(\mathbf{u})\mathbf{u}^H\mathbf{u}d\mathbf{u}} && (4.5)\\
&= \frac{\int \mathbf{u}^H\mathbf{G}\mathbf{G}^H\mathbf{u}d\mathbf{u}}{\int \mathbf{u}^H\mathbf{u}d\mathbf{u}} && (4.6)\\
&= \frac{\int trace(\mathbf{G}\mathbf{G}^H)/k\mathbf{u}^H\mathbf{u}d\mathbf{u}}{\int \mathbf{u}^H\mathbf{u}d\mathbf{u}} && (4.7)\\
&= trace(\mathbf{G}\mathbf{G}^H)/k && (4.8)
\end{aligned}$$

An error correction code provides error protection by expanding the distances among codewords. On AWGN channels, Euclidean distance is a very relevant con-

cept, since it constitutes the exponential part of the Gaussian distribution, and is closely related to the likelihood (probabilistic) test. It should also be noted, although the term generally goes as the "Euclidean distance," for computational convenience, most of the expressions actually involve the "squared Euclidean distance."

Definition 4.4: [Squared Euclidean distance of analog codes] For an analog code $\mathbb{C}(n,k)$, consider two source sequences $\mathbf{u}$ and $\mathbf{u}'$ being encoded to two codewords $\mathbf{v}$ and $\mathbf{v}'$, respectively, the squared Euclidean distance between them is given by

$$D_{E^2}(\mathbf{v},\mathbf{v}') = \sum_{i=0}^{n-1} |v_i - v_i'|^2. \tag{4.9}$$

For linear codes, the concepts of "Hamming distance" and "Hamming weight" can be used inter-changeably in many cases. For example, the Hamming distance spectrum is the same as the Hamming weight spectrum, and the minimum Hamming distance of a code is the same as the minimum non-zero Hamming weight of the code. Here we define a similar concept of Euclidean weight for linear (analog) codes.

Definition 4.5: [Squared Euclidean weight and Minimum Euclidean weight of analog codes] Let $\mathbf{v}$ be a codeword of an analog code $\mathbb{C}$. The squared Euclidean weight of this codeword is given by

$$W_{E^2}(\mathbf{v}) = \sum_{i=0}^{n-1} |v_i|^2. \tag{4.10}$$

The smallest non-zero squared Euclidean weight of all the valid codewords, $W_{E^2,min}$,

is called the *minimum squared Euclidean weight* of $\mathbb{C}$.

Since the all-zero sequence is always a valid codeword in a linear analog code, the Euclidean weight of any codeword is also the Euclidean distance between itself and the all-zero codeword. Hence, the minimum Euclidean distance of a linear analog (block) code is equivalent to its minimum Euclidean weight. In what follows, when applicable, we will study the Euclidean weight instead of Euclidean distance.

Theorem 2 Any linear analog block code $\mathbb{C}(n,k)$ has a minimum Euclidean weight that is approaching 0. That is, for any small positive value ε, there always exists a codeword $\mathbf{v} = \mathbf{G}^H\mathbf{u}$ whose squared Euclidean weight $W_{E^2}(\mathbf{v}) = \sum_{i=0}^{n-1} |v_i|^2 < \varepsilon$.

Proof: Consider a source sequence $\mathbf{u} = \{u_0, 0, ..., 0\}$ with only one non-zero element u_0. After encoding, we have

$$\begin{aligned} D_{E^2}(\mathbf{v}) &= \sum_{i=0}^{n-1} |v_i|^2 && (4.11)\\ &= ||\mathbf{G}^H\mathbf{u}||^2 && (4.12)\\ &= u_0^2 \sum_{i=0}^{n-1} |g_{i0}|^2 && (4.13) \end{aligned}$$

where $||\cdot||$ represents the p-2 norm, and g_{ij} is an element in the ith row and the jth column of the generator matrix $\mathbf{G}$.

Clearly, if we select

$$u_0 < \frac{\sqrt{\varepsilon}}{\sqrt{\sum_{i=0}^{N-1} |g_{i0}|^2}}, \tag{4.14}$$

then the codeword $\mathbf{v} = \mathbf{G}^H\mathbf{u}$ has a squared Euclidean weight D_{E^2} smaller than ε.

Since the minimum Euclidean weight of any analog linear code can be arbitrarily small, it can no longer indicate the spacial goodness of an analog code. Instead, we introduce a new concept, the *squared distance ratio*.

Definition 4.6: [Distance square ratio (DR) and squared weight ratio (WR)] Given an analog code $\mathbb{C}(n, k)$, consider any two source sequences $\mathbf{u}$ and $\mathbf{u}'$ and their respective codewords $\mathbf{v}$ and $\mathbf{v}'$. The squared distance ratio between them is defined as

$$\mathcal{R}_D(\mathbf{u}, \mathbf{u}') = \frac{D_{E^2}(\mathbf{v}, \mathbf{v}')}{D_{E^2}(\mathbf{u}, \mathbf{u}')}. \tag{4.15}$$

The smallest squared distance ratio among all the source pairs is termed the minimum squared (Euclidean) distance ratio of the code $\mathbb{C}$.

For a linear analog (block) code, the Euclidean weight (square) ratio for any non-zero sequence is defined as

$$\mathcal{R}_W(\mathbf{u}) = \frac{W_{E^2}(\mathbf{v})}{W_{E^2}(\mathbf{u})}, \tag{4.16}$$

and the smallest non-zero squared weight ratio is termed the minimum (Euclidean) squared weight ratio.

4.1.3 Maximum squared distance ratio Expansible (MDRE) Codes

Having defined the squared Euclidean distance ratio and the squared Euclidean weight ratio, we now perform analysis on the squared weight ratio of linear analog codes.

Theorem 4.3: [Upper bound for squared distance ratio] For an $\mathbb{C}(n,k)$ linear analog code with a fixed power gain Γ, its minimum weight ratio (squared distance ratio) is upper bounded by Γ, and the upper bound is achieved when all the k eigenvalues of $\mathbf{G}\mathbf{G}^H$ are identical.

Proof:

$$\begin{aligned} \mathcal{R}_W(\mathbf{u}) &= \frac{W_{E^2}(\mathbf{v})}{W_{E^2}(\mathbf{u})} && (4.17)\\ &= \frac{\mathbf{u}^H\mathbf{G}\mathbf{G}^H\mathbf{u}}{\mathbf{u}^H\mathbf{u}} && (4.18)\\ &= \frac{\mathbf{u}^H\mathbf{G}\mathbf{G}^H\mathbf{u}}{\mathbf{u}^H\mathbf{u}} && (4.19) \end{aligned}$$

Since $\mathbf{G}\mathbf{G}^H$ is a Hermitian matrix and a positive definite matrix, it is possible to perform a singular value decomposition, such that $\mathbf{G}\mathbf{G}^H = \mathbf{A}^H\mathbf{D}\mathbf{A}$, where $\mathbf{A}$ is unitary matrix and $\mathbf{D}$ is a diagonal matrix with all the positive (real-valued) diagonal elements $\{d_0, d_1, ..., d_{k-1}\}$. Without loss of generality, we can assume that d_{min} is the minimum value of all the element: $d_{min} = \min\{d_0, d_1, ..., d_{k-1}\} > 0$.

Let $\mathbf{u}' = \mathbf{A}\mathbf{u}$, Equation (4.19) can be further simplified as

$$\begin{aligned} \mathcal{R}_W(\mathbf{u}) &= \frac{\mathbf{u}'^H\mathbf{D}\mathbf{u}'}{\mathbf{u}^T\mathbf{u}} && (4.20)\\ &\geq d_{min}\frac{\mathbf{u}'^H\mathbf{I}\mathbf{u}'}{\mathbf{u}^H\mathbf{u}} && (4.21) \end{aligned}$$

where $\mathbf{I}$ is an identical matrix. The equality in (4.21) is achieved when $\mathbf{u}' = (0, 0, \cdots, u_i, \cdots, 0)$ where i is the location for d_{min}.

Since $\mathbf{u'}^H\mathbf{u'} = \mathbf{u}^H\mathbf{A}^H\mathbf{A}\mathbf{u} = \mathbf{u}^H\mathbf{u}$, we have

$$\min(\mathcal{R}_W(\mathbf{u})) = d_{min}, \tag{4.22}$$

which gives rise to

$$\begin{aligned} \min(\mathcal{R}_W(\mathbf{u})) &= d_{min} && (4.23)\\ &\leq \frac{\sum_{i=0}^{k-1} d_i}{k} && (4.24)\\ &= \frac{trace(\mathbf{G}\mathbf{G}^H)}{k} && (4.25)\\ &= \Gamma && (4.26) \end{aligned}$$

The equality in (4.24), i.e., the upper bound of the minimum squared weight (distance) ratio, is achieved when all the eigenvalues of $\mathbf{G}\mathbf{G}^H$ are identical, i.e. $d_0 = d_1 = \cdots = d_{k-1} = d_{min}$.

Corollary 4.4: Given a linear analog code $\mathcal{C}(n, k)$ with generator matrix $\mathbf{G}$, its minimum squared Euclidean distance ratio is d_{min}, where d_{min} is the minimum eigenvalue of matrix $\mathbf{G}\mathbf{G}^H$.

Definition 4.7: [Maximum squared distance ratio expansible (MDRE) codes] Consider all the linear analog codes $\mathcal{C}(n, k)$ with the fixed encoding power gain Γ. A code is called maximum squared distance ratio expansible or MDRE, if its minimum squared Euclidean distance ratio achieves the upper bound Γ.

Corollary 4.5: An analog linear block code $\mathcal{C}(n, k)$ with generator matrix $\mathbf{G}$ is MDRE, if and only if the eigenvalues of $\mathbf{G}\mathbf{G}^H$ are all identical.

Definition 4.8: [Analog unitary Codes] A linear analog code $\mathcal{C}(n,k)$ is called an analog unitary code, if its generator matrix is formed by $\mathbf{G} = a\mathbf{\Xi}$, where a is non-zero real-value scalar and $\mathbf{\Xi}$ is formed of a set of k columns selected from a unitary matrix $\mathbf{U}$.

Theorem 4.6: Analog unitary codes are MEDR codes.

Proof: Let $\mathbf{G}$ be the generator matrix of an analog unitary code. Clearly, $\mathbf{G}\mathbf{G}^H = a^2\mathbf{I}$, where $\mathbf{I}$ is an identical matrix. Hence $\mathbf{G}\mathbf{G}^H$ has identical eigenvalues and the code is therefore MEDR.

4.1.4 ML Decoding and Distortion

In the following, we will first discuss the maximum likelihood decoder of a general analog code (linear or nonlinear), and then focus on linear codes.

Definition 4.9: [Basic space] Given an analog code $\mathcal{C}$ with mapping $\mathbb{U}^k \xrightarrow{\mathcal{C}} \mathbb{V}^n$. The source space $\mathbb{U}^k$ comprises a finite number of t subspaces, denoted as $\mathbb{B}_i$ for $0 \leq i \leq t-1$, such that the function $\mathcal{C}$ is continuous and differentiable in each subspace. We call each subspace $\mathbb{B}_i$ as a basic space of the code $\mathcal{C}$. Each $\mathbb{B}_i$ is indexed by $I_i, 0 \leq i \leq t-1$, named as the basic index.

Consider the sequence $\mathbf{r}$ at the output of a noise channel. We can define the ML decoder for a general analog code as

$$\tilde{\mathbf{u}} = \arg\max_{0\leq i\leq t-1}(\arg\max_{\tilde{\mathbf{u}}\in\mathbb{B}_i} \Pr(\mathbf{r}|\mathbf{u})), \tag{4.27}$$

where $\tilde{\mathbf{u}}$ is the estimation of source $\mathbf{u}$.

Since the function $\mathcal{C}$ is continuous and differentiable in each subspace $\mathbb{B}_i, 0 \leq i \leq t-1$, if the channel transfer function $p(\mathbf{y}|\mathbf{v})$ is differentiable (a condition that is generally satisfied for channels), then $p(\mathbf{y})|\mathbf{u} \in \mathbb{U}_i)$ is also differentiable. Suppose there are only a finite number of local maximums (again a condition that is generally satisfied for linear and nonlinear mapping), then we will have a finite number of candidates for possible $\mathbf{u}$. The ML decoder can compare all of these candidates to identify the best $\mathbf{u}$ with the largest probability. The complexity of the ML decoder will be linear to the number of candidates.

Below we prove that MEDR codes are the best linear analog codes on AWGN channels in terms of mean square error (MSE) distortion. To show that, we first discuss the optimal decoder for linear analog codes on AWGN channels.

The maximum likelihood decoding of a linear analog code on an AWGN channel can be modeled as an unconstrained convex optimization problem:

$$\min ||\mathbf{r} - \mathbf{G}^H\mathbf{u}||^2 \tag{4.28}$$

where $||\cdot||^2$ is the square of the p-2 norm.

This problem can be solved analytically by expressing the objective function as the convex quadratic function

$$f(\tilde{\mathbf{u}}) = \mathbf{u}^H\mathbf{G}\mathbf{G}^H\mathbf{u} - 2\mathbf{r}^H\mathbf{G}^H\mathbf{u} + \mathbf{r}^H\mathbf{r}. \tag{4.29}$$

The minimum value of $f(\tilde{\mathbf{u}})$ is obtained when

$$\tilde{\mathbf{u}} = (\mathbf{G}\mathbf{G}^H)^{-1}\mathbf{G}\mathbf{r} \tag{4.30}$$

Theorem 4.7: [ML decoder of linear analog codes] The maximum-likelihood decoder of a linear analog code on an AWGN channel produces:

$$\tilde{\mathbf{u}} = (\mathbf{G}\mathbf{G}^H)^{-1}\mathbf{G}\mathbf{r}, \tag{4.31}$$

where $\mathbf{G}$ is the generator matrix of the code, and $\mathbf{r}$ is the noisy codeword observed from the AWGN channel.

Definition 4.10 [MSE distortion] Consider an analog code $\mathcal{C}(n, k)$ transmitted over a noisy channel with an additive noise $\mathbf{w}$. The mean square error distortion for a particular decoder on this channel is defined as

$$\Delta = \int p(\mathbf{u}) \int ||\tilde{\mathbf{u}} - \mathbf{u}||^2 p(\mathbf{w}) d\mathbf{w} d\mathbf{u}, \tag{4.32}$$

where $\mathbf{u}$ is the analog source vector, the $\tilde{\mathbf{u}}$ is the decoder estimate for $\mathbf{u}$, $p(\mathbf{u})$ is the pdf for the noise vector and $p(\mathbf{u})$ is the pdf for the source vectors. Specifically, for linear analog codes, because of the geometric uniformity, instead of evaluating over all the possible source vectors $\mathbf{u}$, the all-zero source vector can serve as the representative. Hence the MSE distortion can be simplified to:

$$\Delta = \int ||\tilde{\mathbf{u}}||^2 p(\mathbf{w}) d\mathbf{w}, \tag{4.33}$$

where $\tilde{\mathbf{u}}$ is the decoder estimate for the all-zero codeword.

Theorem 4.8: [Lower bound of MSE distortion for linear analog codes] Consider an (n, k) linear analog code with encoder power gain Γ operating on an AWGN channel with noise $\mathbf{w}$, where $w_i \sim \mathcal{N}(0, \sigma^2)$. The mean square error distortion Δ after ML coding is lower bounded by

$$\Delta \geq \Delta_{min} = \frac{k\sigma^2}{\Gamma}. \tag{4.34}$$

The lower bound is achieved by $s_0^2 = s_1^2 = ...s_{k-1}^2 = \Gamma$, where $\{s_0, s_1, ...s_{k-1}\}$ are the set of singular values of $\mathbf{G}$.

Proof: Without loss of generality, assume that the all-zero codeword is transmitted. Substituting $\mathbf{r} = \mathbf{w}$ and (4.30) in (4.31):

$$\begin{aligned}
\Delta &= \int ||(\mathbf{G}\mathbf{G}^H)^{-1}\mathbf{G}\mathbf{w}||^2 P(\mathbf{w}) d\mathbf{w} \\
&= \int ||(\mathbf{G}\mathbf{G}^H)^{-1}\mathbf{G}\mathbf{w}||^2 \prod_{i=0}^{n-1} (\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{w_i^2}{2\sigma^2}}) d\mathbf{w} \\
&= \frac{1}{(2\pi\sigma^2)^{n/2}} \int ||(\mathbf{G}\mathbf{G}^H)^{-1}\mathbf{G}\mathbf{w}||^2 e^{-\frac{\sum_i w_i^2}{2\sigma^2}} d\mathbf{w} \\
&= \frac{1}{(2\pi\sigma^2)^{n/2}} \int \mathbf{w}^H \mathbf{G}^H (\mathbf{G}\mathbf{G}^H)^{-1} (\mathbf{G}\mathbf{G}^H)^{-1} \mathbf{G}\mathbf{w} e^{-\frac{\sum_i w_i^2}{2\sigma^2}} d\mathbf{w} \\
&= \frac{1}{(2\pi\sigma^2)^{n/2}} \int \mathbf{w}^H \mathbf{B}\mathbf{w} e^{-\frac{\sum_i w_i^2}{2\sigma^2}} d\mathbf{w}
\end{aligned} \tag{4.35}$$

where $\mathbf{B} = \mathbf{G}^H(\mathbf{G}\mathbf{G}^H)^{-1}(\mathbf{G}\mathbf{G}^H)^{-1}\mathbf{G}$. Equation (4.35) calculates the variance of a weighted summation of an n-dimension i.i.d. Gaussian vector. Therefore, (4.35) can be further simplified as

$$\begin{aligned}\Delta &= trace(\mathbf{B})\sigma^2 \\ &= \sum_{i=0}^{k-1} \frac{1}{s_i^2}\sigma^2, \end{aligned} \tag{4.36}$$

where $\{s_0, ...s_{k-1}\}$ are the set of singular values of $\mathbf{G}$, since $(\mathbf{G}\mathbf{G}^H)^{-1}\mathbf{G}$ is the pseudo inverse matrix of $\mathbf{G}$.

To minimize Δ is to minimize $\sum_{i=0}^{k-1} \frac{1}{s_i^2}$, subject to $\sum_{i=0}^{k-1} s_i^2 = k\Gamma$, which leads to:

$$s_1^2 = s_1^2 = ... = s_{k-1}^2 = \Gamma. \tag{4.37}$$

Hence we have

$$\Delta \geq \frac{k\sigma^2}{\Gamma} \tag{4.38}$$

Corollary 4.9: An MDRE code can achieve the minimum bound of the average MSE distortion on AWGN channel, and is therefore distortion optimal.

The above analysis shows that the minimum distance (weight) ratio can provide an effective indication of the code performance (i.e. MEDR and minimum MSE distortion) for linear analog codes. However, as will be shown later in Chapter 5, this metric is much less useful for nonlinear analog codes. One way to explain this is that a linear mapping scales a source vector exactly the same way as it does to another source vector, but it is not the case for nonlinear mapping. Hence, we introduce the concept of the average squared distance ratio for nonlinear codes.

Definition 4.11: [Average squared distance ratio] The average squared distance ratio of an analog code $\mathcal{C}$ is defined as

$$\mu(\mathcal{R}_D) = \int_{\mathbf{u}} \int_{\mathbf{u}'} p(\mathbf{u}, \mathbf{u}') \mathcal{R}_D(\mathbf{u}_0^{k-1}, \mathbf{u}'^{k-1}_0)) d\mathbf{u} d\mathbf{u}', \tag{4.39}$$

where $\mathbf{u}$ and $\mathbf{u}'$ are source vectors of the code.

Further, to fairly compare the code performances of two codes (linear or nonlinear) having possibly different source space, we introduce the metric of normalized MSE distortion.

Definition 4.12: [Normalized MSE distortion] The normalized MSE distortion for the performance measurement of an analog code is defined as

$$\Delta^* = \frac{\Delta}{\sigma_u^2}, \tag{4.40}$$

where σ_u^2 is the variance of the source $\mathbf{u}$ and Δ is the averaged MSE distortion of the code.

4.2 Analysis of Linear Analog Codes

We have previously developed several concepts and theorems for analog codes and linear analog codes.

In this section, we first provide a brief overview of the existing linear analog codes in general, and then focus the discussion on two important classes, the analog discrete Fourier transform (DFT) codes and the analog discrete cosine transform

(DCT) codes. We will apply the concepts and tools we developed earlier to analyze these codes and to advance our understanding of linear analog codes.

4.2.1 A Brief Overview

The first analog code, due to Marshall [43] and Wolf [44], is called the *discrete Fourier transform* (DFT) code. To achieve a desirable Hamming distance t, the DFT code extracts $r = 2t$ columns from the inverse discrete Fourier transform (IDFT) matrix to form the generator matrix $\mathbf{G}$. When the extracted columns follow certain structural formalism, the resultant complex DFT code can be viewed as an analog Bose-Chaudhuri-Hocquenghem (BCH) code and at the same time also satisfies the maximum distance separable condition in terms of Hamming distance [46]. In other words, there exist a subclass of DFT codes that are by nature analog Read-Solomon (RS) codes and optimal in the MDS sense.

Another important class of MDS-optimal analog codes was proposed by Wu and Shiu and named *discrete cosine transform* (DCT) codes [47]. Unlike DFT codes, DCT codes are not analog BCH codes or even cyclic codes. However, Wu and Shiu showed that a specific subclass of DCT codes can be expressed in a BCH-like structure and decoded by a modified Berlekamp-Massey and Forney algorithm [47]. This BCH-like DCT structure was later generalized by Rath and Guillemot, which gave rise to *discrete sine transform* (DST) codes [9]. A subspace-based decoding algorithm was also developed for these codes [9]. However, the work in [47] and [9] only discussed a special case of DCT and DST codes, namely DCT and DST codes with a BCH-like structure. A general decoding based on subspace methods is proposed in [48].

Following the same line of development of analog block codes, there have also been studies of analog convolutional codes, and their encoding and decoding mechanisms [46] [49].

4.2.2 Discrete Fourier Transform Codes and Analog BCH Codes

We now discuss discrete Fourier transform codes, one of the most important class of linear analog codes in literature. An (n, k) DFT code is an analog linear block code whose Hermitian transpose of the generator matrix consists of a set of k columns from the DFT matrix $\mathbf{\Psi}$ of order n, where

$$\mathbf{\Psi} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \phi & \phi^2 & \cdots & \phi^{n-1} \\ 1 & \phi^2 & \phi^4 & \cdots & \phi^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \phi^{n-1} & \phi^{2(n-1)} & \cdots & \phi^{(n-1)^2} \end{bmatrix} \tag{4.41}$$

where $\phi = e^{-j2\pi/n}$.

The remaining $(n - k)$ columns of the DFT matrix forms parity check matrix of the code

For example, we can take the first k columns to form the generator matrix, and the remainder $(n-k)$ columns to form the parity check matrix, as shown in Fig. 4.2.

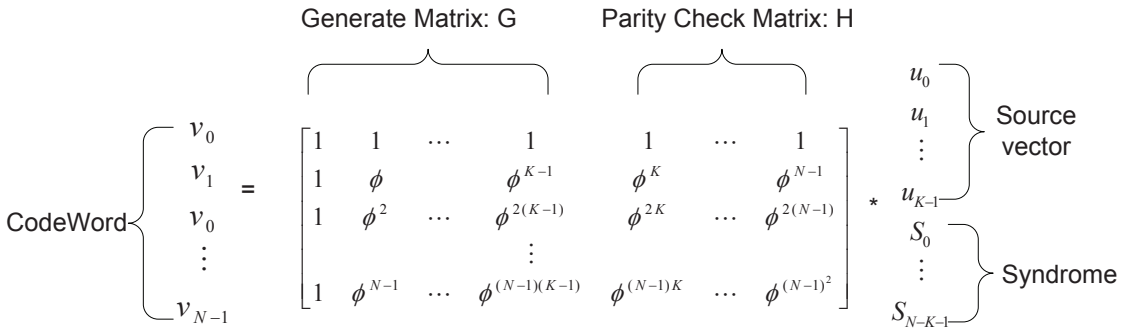


Figure 4.2: The structure of DFT codes

The DFT matrix $\mathbf{\Psi}$ consists of Hermitian transpose of the generator matrix $\mathbf{G}^H$ and Hermitian transpose of the parity check matrix $\mathbf{H}^H$. The n-dimensional vector on the right hand side of the equation in Fig. 4.2, consists of a k-dimensional source vector $\mathbf{v}$ and a $(n-k)$-dimensional syndrome $\mathbf{s}$; and the left hand side of the equation corresponds to some n-dimensional coded vector $\mathbf{v}$. Since the DFT matrix is a unitary matrix, it is easy to prove that $\mathbf{v}$ is the valid codeword for the source $\mathbf{u}$ if and only if the syndrome $\mathbf{s} = \mathbf{0}$.

Encoding of the DFT code follows the usual matrix multiplication process (between the source vector and the generator matrix) of a linear block code. Alternatively, it may also be treated as a matrix multiplication between the square unitary matrix and the zero-stuffed source vector. The later viewpoint draws close parallelism to OFDM with redundant information bits, and DFT therefore finds a useful application in OFDM transmission (see, for example, [50] [51]).

If the parity check matrix of the DFT code can be expressed in the form of

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \phi^{\alpha} & \phi^{2\alpha} & & \phi^{(n-1)\alpha} \\ 1 & \phi^{2\alpha} & \phi^{4\alpha} & & \phi^{2(n-1)\alpha} \\ & & \vdots & & \\ 1 & \phi^{(n-k-1)\alpha} & \phi^{2(n-k-1)\alpha} & \cdots & \phi^{(n-k-1)(n-1)\alpha} \end{bmatrix} \tag{4.42}$$

where α is an integer relatively prime to n, then the resultant code becomes a BCH DFT code. Further, since α is relatively prime to n, all the elements of the second column in $\mathbf{H}$ are different due to $mod(i\alpha, n) \neq mod(j\alpha, n)$, $0 \leq i, j \leq n{-}k{-}1$. Then according to the properties of Vandermonde matrices, any $(n{-}k)$-by-$(n{-}k)$ sub-matrix of $\mathbf{H}$ is full rank. Thus, this code is also a complex-valued MDS code in terms of Hamming distance, and is therefore also termed the analog RS code. It has been shown that the traditional decoder (such as Peterson-Gorenstein-Zierler (PGZ) decoder, Berlekamp-Massey algorithm and Forney algorithm) of digital BCH codes can be applied to analog BCH codes.

4.2.3 Discrete Cosine Transform (DCT) Codes and Analog BCH-like codes

Similar as the DFT codes, the generator matrix $\mathbf{G}$ of the discrete cosine transform code comprises k selected columns from a matrix $\mathbf{\Xi}$, where each element $\xi(i, j) \in \Xi$ is defined as

$$\xi(i,j) = \begin{cases} 1/\sqrt{n} & j = 0 \\ \frac{2}{\sqrt{n}} \cos \frac{(2i+1)j\pi}{2n} & j = 1, 2, ..., n-1 \end{cases} \tag{4.43}$$

Unlike DFT codes, the DCT codes are not analog BCH codes. However, the parity check matrix $\mathbf{H}$ of the DCT code can be decomposed into [9] $\mathbf{AXU}$, where $\mathbf{A}$ is an $(n-k)$-by-$(n-k)$ full rank matrix, $\mathbf{U}$ is an n-by-n diagonal matrix and $\mathbf{X}$ is an $(n–k)$-by-n Vandermonde matrix. For an arbitrary $(n-k)$-by-$(n-k)$ submatrix $\mathbf{X}'$ of $\mathbf{X}$, interacting with full-rank matrices $\mathbf{A}$ and $\mathbf{U}$ does change its rank. Therefore, the parity check matrix will preserve the properties of Vandermonde matrix which in turn defines an MDS code.

4.2.4 Linear Analog Codes on Pulse Channels

Early work of analog codes has primarily used a special channel model where *pulse noise* will either occur or not occur upon any (analog) symbol transmitted through the channel. For instance, consider an analog codeword $\mathbf{A} = \{a_0, a_1, ..., a_{n-1}\}^T \in \mathcal{R}^n$ that is being transmitted. If m pulse errors occur to this codeword, that means only m analogy symbols will be distorted, and the remainder $n - m$ analog symbols must remain perfectly intact, free from any noise or interference, including thermal noise, circuitry noise, media noise, detection distortion and the like. Any slightest change on any additional symbol will cause the decoder to malfunction or completely fail.

Since all real-world devices are subject to imperfection or noise of some kind, researchers have also considered more realistic channel models with additive white Gaussian noise (AWGN) or a mixture of white Gaussian noise and pulse noise, and studied the code design and decoding strategies in this context. For example, [52] and [53] introduced an iterative decoding strategy for analog product codes and analog component codes on an AWGN channel. In [54], Redinbo proposed a Weiner

estimator for on the mixed Gaussian-and-pulse noise channel by using a modified Berlekamp-Massey algorithm as the error position detector. A more robust decoder for the DFT analog codes is further developed in [55] by modifying the error location polynomial. In [56], Takos and Hadjicostis proposed a strategy to estimate the number of errors in the DFT codes in the presence of low-level quantization noise [56]. These studies have certainly demonstrated an encouraging step forward compared to the pure pulse noise channel, but the Gaussian noise thereof is all very small.

4.2.5 Analysis of Existing Linear Analog Codes on AWGN Channels

Since DFT codes and DCT codes are both unitary codes, and unitary codes are MEDR codes, we have:

Corollary 4.10: The DFT codes and DCT codes are MEDR codes.

Further, since the $(n, 1)$ repetition code, whose generator matrix is all ones, is also a unitary code, we have

Corollary 4.11: The repetition codes are MEDR codes.

Additionally, the DFT codes, the DCT codes, and the repetition codes are also MDS codes in terms of Hamming distance.

Corollary 4.9 states that MEDR codes can achieve the biggest squared Euclidean distance ratio and the smallest average MSE distortion. Hence they exhibit

the same, best MSE performance on AWGN channels. This conclusion can also be verified by computer simulations. In Fig. 4.3, we compare four linear analog codes with parameters, $n = 60$ and $k = 30$ with the same given power gain of $\Gamma = 60$ on the same AWGN channel. They are: a DCT code, a repetition code and two randomly-constructed generator matrices with minimum squared Euclidean distance ratio of 0.0235 and 0.0514, respectively. Since the DCT code and the repetition code are all MDRE codes, they have the same best minimum squared weight ratio of 2, and achieve the smallest average MSE distortion . Simulations confirm that they perform the same. The two random analog codes have much lower minimum squared weight ratios which in turn lead to considerably larger average MSE distortions, with the one having a higher minimum squared weight ratio of 0.0514 exhibiting a sightly smaller average distortion.

Our analysis and simulations indicate that the minimum distance (weight) ratio can provides a simple and good metric for the design and evaluation of linear analog codes. Specifically linear analog codes that achieve the minimum squared weight ratio possible also achieve the smallest MSE ratio possible on AWGN channels. In the next subsection, we will discuss the design of linear analog codes for AWGN channels. We note, however, that that linear analog codes are not the best codes for AWGN channels, and that nonlinear analog codes can easily exceed the performance bound of linear analog codes. In the discussion of the nonlinear analog codes in the next chapter, we will also show that the easy-to-calculate minimum squared distance ratio serves only as a good metric for linear analog codes, and not for nonlinear analog codes. Instead, the union bound serves both cases well.

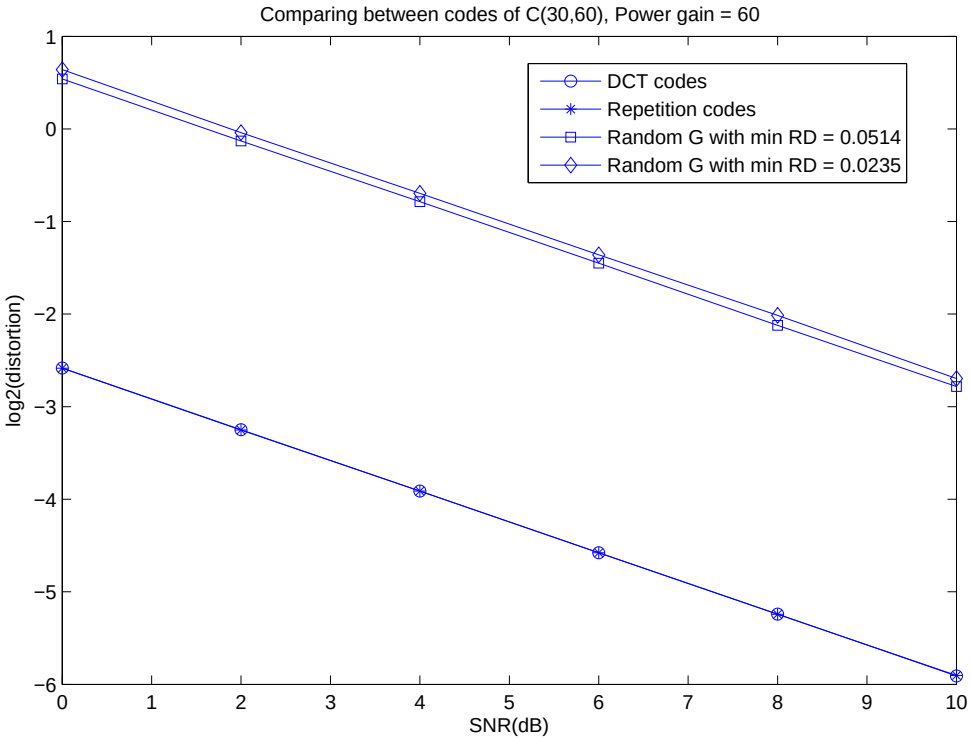


Figure 4.3: Performance of linear analog code with AWGN.

4.3 Design of Linear Analog Block Codes on AWGN Channels

BCH codes and BCH-like codes primarily consider the Hamming distance. To satisfy certain decoding capability, the Hamming weight of the noise vector must be limited. That is, in the system model, the additive noise $\mathbf{w}$ can only be modeled as the pulse function that happen in a limited number of positions. In all the other positions, the distortion must be zero in order for the BCH-like decoder to work. Since such a channel model is too idealistic and artificial, there has been some recent work studying the decoder design and the performance of linear analog block

codes on AWGN or mixed Gaussian-and-pulse noise channels [52] [53] [54], [55] [56]. However, most of these studies have not really departed from the BCH and the pulse noise model, and underpinning theory of designing linear analog codes for AWGN channels is far from mature.

Below we provide a geometric view for linear analog codes and propose engineering rules to design good linear analog codes.

4.3.1 Geometric explanation of linear analog codes

From the geometric point of view, to encode a linear analog code is to linearly transform some subspace in the n-dimensional space. The source vectors span a k-dimensional subspace, which, when represented in the n-dimensional space, is like the n-dimensional vectors having $(n-k)$ zeros in the last $n-k$ dimensions. After encoding, the k-dimensional subspace is linearly transformed to a new n-dimensional subspace.

A linear transform can de decomposed to a set of basic linear transformations: rotation, scaling, shearing, and reflection. It has been known that an arbitrary n-dimensional matrix $\mathbf{G}$ can be decomposed to $\mathbf{G} = \mathbf{UDV}$ by singular value decomposition, where $\mathbf{U}$ and $\mathbf{V}$ are two square unitary matrices and $\mathbf{D}$ is a diagonal matrix whose diagonal elements are the eigenvalues of the matrix $\mathbf{G}$. In other words, any linear transform can be implemented in three steps: rotating via the rotation matrix $\mathbf{U}$, followed by scaling via the scale matrix $\mathbf{D}$, and followed by a second rotating via the rotation matrix $\mathbf{V}$. Fig.4.4 uses a simple case of $n = 3$ and $k = 2$ to illustrate how the rotation and scaling operations will impact the

Hamming distance and the Euclidean distance. The general idea holds for any n dimensions.

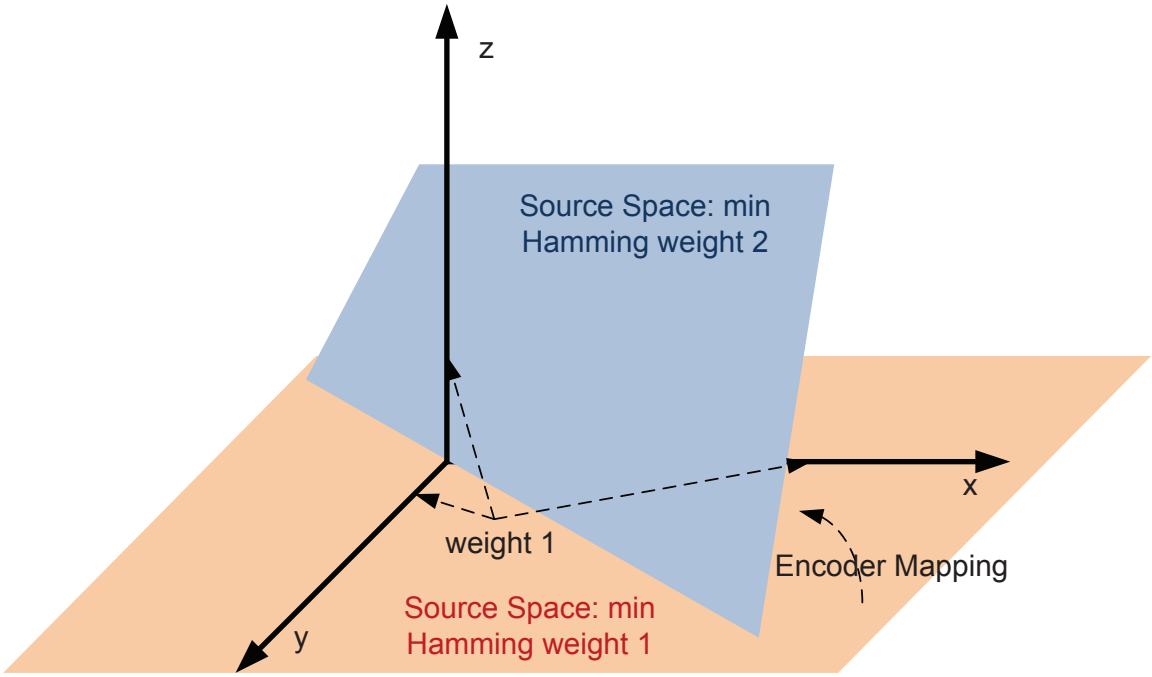


Figure 4.4: Geometric explanation of linear analog code.

In Fig 4.4, a 2-dimensional source space, namely, the x-y plane, is mapped to the codeword space in the 3-dimensional space. Since it is a linear mapping, the codeword space is also a plane, and also passes through the origin, the $(0, 0, 0)$ point. In terms of Hamming weight in a 3-dimension space, the origin $(0, 0, 0)$ has the lowest weight of 0. Other points on the x axis, the y axis or z axis have the second lowest weight of 1. The points located in the x-y plane, the y-z plane or the x-z plane excluding the three axises have Hamming weight of 2. All the other points will have Hamming weight of 3. Since the original source space, the x-y plane, contains the x axis and the y axis, so the minimum (non-zero) Hamming weight of the source space is 1. Through encoding, if the source plane is rotated to

a different plane that does not include any one of the x , y or z axises (i.e. does not intersect these axises except for the origin), then the minimum Hamming weight is increased to at least 2. In addition, since the codeword plane will inevitably have an intersecting line with either the x-y plane, the y-z plane or the x-z plane, the minimum Hamming distance is upper limited to 2. From this analysis, we can tell that the rotation operation plays a key role in expanding the Hamming weight during the encoding process. The scaling operation will only affect the Euclidean weight, but casts no impact on Hamming weight. Hence as far as Hamming weight is concerned, one can safely assume that the scaling matrix is an identity matrix (i.e. no scaling on any dimension). What this implies in terms of code design is, for a given linear analog code $\mathcal{C}(n, k)$, it is always possible to find an analog unitary code which will produces the same Hamming weight as the original linear analog code for all the source vectors. In other words, a rotation matrix suffices to achieve the upper bound of the minimum Hamming weight.

Now to put Euclidean distance in perspective, it is clear that rotation becomes irrelevant and scaling takes the determining role. From our previous analysis of maximum squared Euclidean distance ratio and the minimum MSE distortion, the best scaling should be one that is uniform across all the dimensions. This is why, for example, codes whose eigenvalues of the $\mathbf{G}\mathbf{G}^H$ are all identical are MDRE codes that can achieve the maximum squared distance ratio and the minimal MSE distortion.

To conclude, the goals of optimizing Hamming weight and optimizing Euclidean weight do not conflict with each other in the case of linear analog codes. A good code design can unify both metrics in one. For example, a carefully-selected rota-

tion matrix, such as that for an analog unitary code, can achieve both MDRE and MDS bounds at one shot.

Chapter 5

Non-Linear analog coding

A linear analog codes will do a uniform distance expansion for all the codewords. However, to have a best average distance expansion, we expect that the codeword pair which has a smaller distance in the source space is expanded more in the codeword space, therefore the nonlinear analog codes are desired.

Fig 5.1 compares the difference between two toy examples of the linear analog codes $AC(3,1)$ and nonlinear analog code (tent map codes) $NAC(3,1)$. The input space is a straight line. After encoding, the analog code $AC(3,1)$ is still a straight line. Without any transmission power gain, the space distance will remain the same as before encoding. However the nonlinear code $NAC(3,1)$ will be mapped to a folded line after encoding. Even without accurately measuring, we can tell that the folded line has a longer length than the straight line. That is, the distances between the neighboring points are extended more in nonlinear analog codes than the linear analog codes. From another point of view, the linear operation will preserve the dimension of the space during encoding, and therefore can not take

the full advantage of the changing of the space dimension. However, the codeword dimension will be increased for nonlinear operation. Further, the linear operation will evenly expand the codeword distance, while the nonlinear operation can give different pair different expanding ratios, such as giving a larger expansion to the pair of points that are closer to each other before encoding.

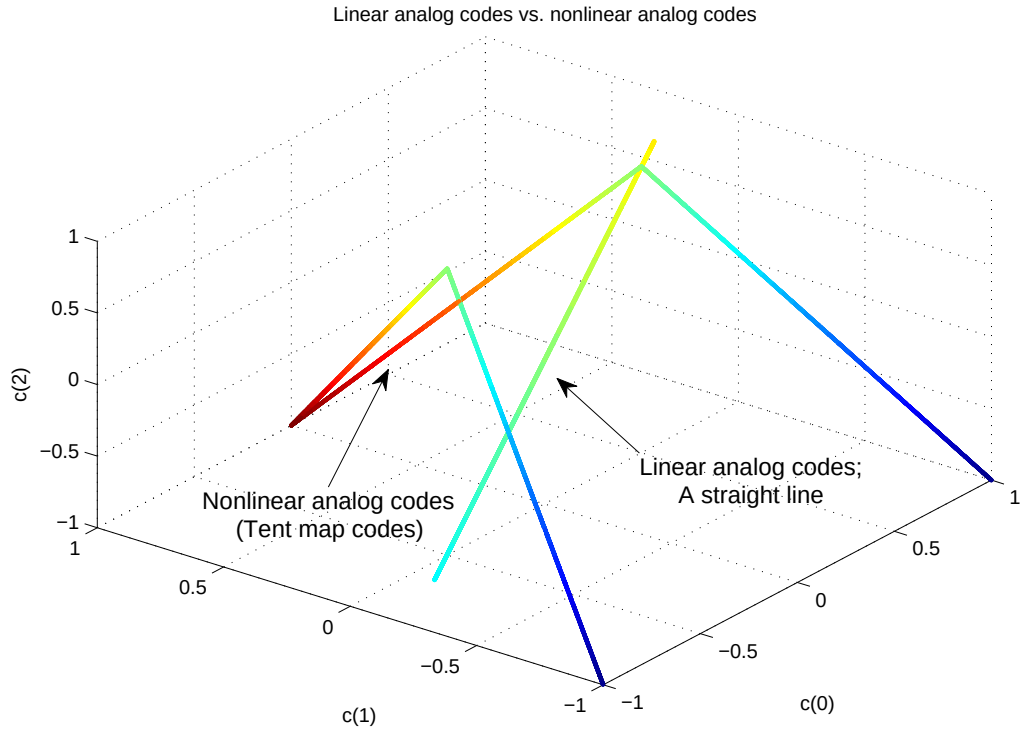


Figure 5.1: Comparing between the linear analog codes and nonlinear analog codes

The simulation can also confirm the advantage of nonlinear analog codes. Fig. 5.2 shows that the tent map codes and the mirrored baker's map codes (both simple nonlinear analog codes) can outperform the DCT codes (the best linear analog codes) with the same code rate.

From Shannon's theory, we know that, given a source $\mathbf{x}$ with distortion $d(\mathbf{x}, \tilde{\mathbf{x}})$,

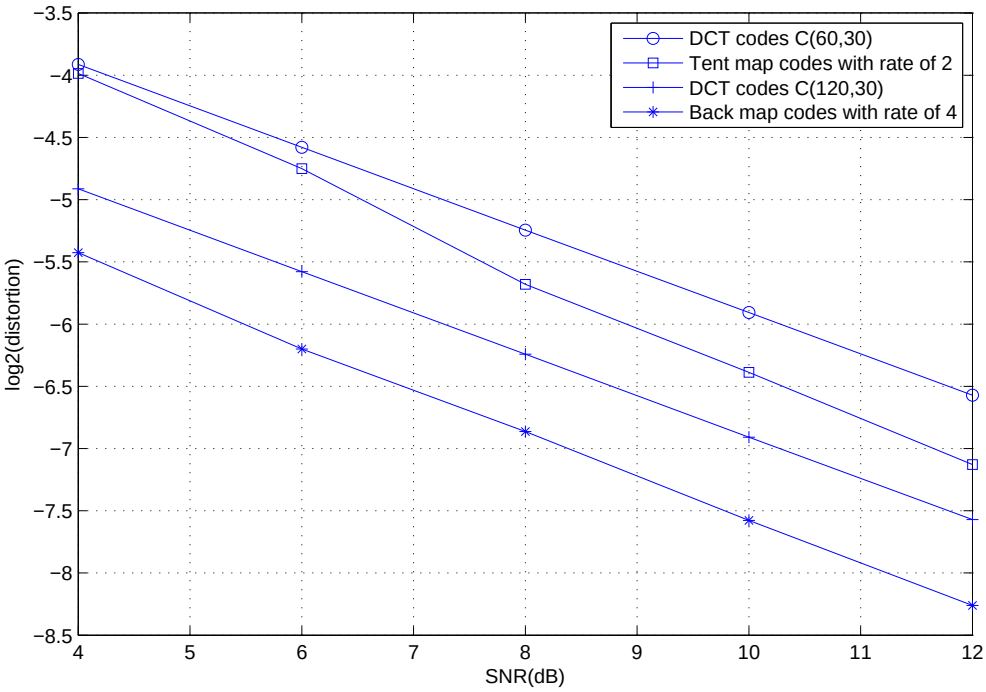


Figure 5.2: Performance comparison between the linear analog codes (DCT codes) and nonlinear analog codes(tent map codes, baker's map codes)

a channel with capacity C and coding with bandwidth expansion ratio of n/k, the minimum achievable distortion is limited to

$$R(D) \leq n/kC \tag{5.1}$$

where $R(D)$ is the rate-distortion function given by

$$R(D) = min_{p(\tilde{\mathbf{x}}|\mathbf{x}),E(d(\mathbf{x},\tilde{\mathbf{x}})<D}(I(\mathbf{x},\tilde{\mathbf{x}})) \tag{5.2}$$

For example, given a Gaussian distributed source with mean square error distortion and AWGN channel, the lower bound of the MSE distortion (5.1) is derived

as

$$\frac{1}{2}\log\frac{\sigma_x^2}{D} \leq \frac{n}{2k}\log(1+\frac{P}{\sigma_w^2}) \tag{5.3}$$

$$\frac{D}{\sigma_x^2} \geq \frac{1}{(1+\frac{P}{\sigma_w^2})^{\frac{n}{k}}}, \tag{5.4}$$

where $\frac{P}{\sigma_w^2}$ is the signal to noise ratio of AWGN channel, and σ_x^2 is the variance of the Gaussian source.

In the log domain, we have

$$log\frac{D}{\sigma_x^2} \geq -\frac{n}{k}log(1+\frac{P}{\sigma_w^2}) \tag{5.5}$$

This bound is plotted in Fig. 5.3 for different values of n/k

5.1 Chaotic analog codes

The first chaotic analog code is the tent map code, constructed by Chen and Wornell [11] using the tent map as the chaotic generator or the encoder. The tent map is a popular nonlinear chaotic mapping with simple formulation. For each input symbol, the encoder will generate a corresponding chaotic analog sequence.

Chaotic systems are nonlinear systems with bounded state spaces exhibiting a topological mixing feature. They are widely existent in the natural world and the engineering world, and many of them can be realized using simple electric circuits.

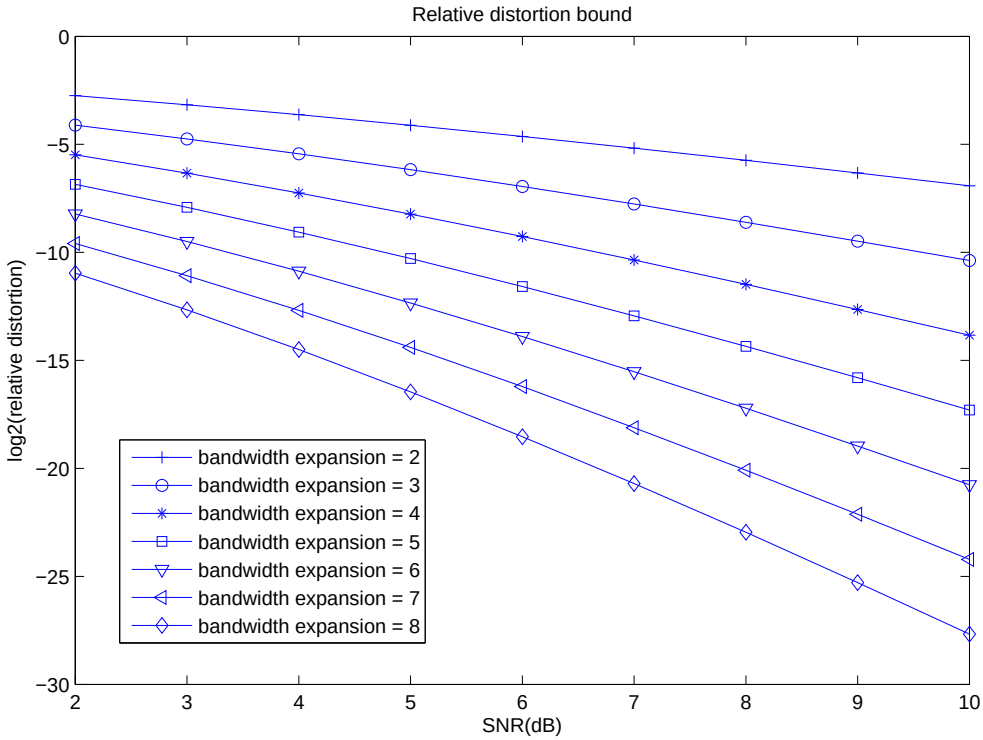


Figure 5.3: The normalized MSE Distortion Bound for Gaussian source and AWGN channels

Despite the rich variety of formalities, chaotic systems share one common property of high sensitivity to the initial state. Popularly dubbed the "butterfly effect," this property states that a small distortion on the initial states of a chaotic system will cause a big difference at the ending states. Although this butterfly effect is in general viewed as a system penalty, it can actually be cleverly exploited to satisfy the distance expansion property required by a good channel code. Specifically, if one treats the initial states of a chaotic system as the source (to be encoded), and treats some later states as the codeword (having been encoded), then the chaotic system naturally enacts a channel encoder that successfully magnifies the small difference (distance) among the source sequences.

This elegant feature was first exploited by Chen and Wornell in the late nineties. A (near) maximum likelihood (ML) detector is also developed to perform effective channel decoding [11]. In part because the tent map code performs nowhere comparable to digital codes, and in part because the chaotic theory is rather foreign to the coding community, the beautiful idea exposed in [11] slept for a decade before we recently picked it up [21]. Carrying the idea further, and leveraging useful concepts from digital coding, we propose a much better-performing code, termed the *chaotic analog turbo (CAT) code* [21]. By connecting two tent map codes in parallel and decoding it through a soft iterative decoder, the CAT code is capable of achieving a significant coding gain over a single tent map code, at the cost of a higher decoding complexity. For the coding theory, a larger dimensionality introduces a richer context and promises a better coding gain. As tent maps are 1-dimensional chaotic maps, we further explore higher dimensional maps. Specially, we study the baker's map, a 2-dimensional map and sue it to construct a mirrored baker's map codes that further outperforms the CAT codes.

5.2 Tent Map Codes

The *tent* map, a nonlinear iterative function in the shape of a tent, is widely used to generate discrete-time chaotic dynamics. It is defined as

$$F(x) = \beta - 1 - \beta|x|, \quad 1 < \beta \leq 2, \quad -1 < x \leq \beta - 1. \tag{5.6}$$

In a time evolution, all the states x and $F^n(x)$ $(n = 1, 2...)$ in the interval $(-1, \beta-1]$ tend to the attracting interval $[-(\beta-1)^2, \beta-1]$. In general, the invariant density of

the tent map cannot be described in closed form, except for the case of $\beta = 2$ where the invariant density is uniform over the region of $(-1, -1]$. For simplicity and for the purpose of transmitting over additive white Gaussian noise (AWGN) channels (symmetric source distribution is preferred on symmetric channels), below we will consider $\beta = 2$.

Wornell *et al* showed that the properties of chaotic systems are useful for channel coding, and to illustrate this point, they constructed the tent map analog codes based on the tent map [11], along with an efficient sequence decoding algorithm [11].

Suppose that we transmit a discrete-time real-valued source $x[0]$ over a stationary AWGN channel. The encoder encodes each source symbol into a sequence of $N + 1$ states, where the original source symbol $x[0]$ is the initial state followed by N subsequent parity states $x[1]$, ..., $x[N]$,

$$x[n] = F(x[n-1]) = F^n(x[0]), \tag{5.7}$$

where $F^0(x) = x$ and

$$F^n(x) = \underbrace{F \circ F \circ ... \circ F(x)}_{n}. \tag{5.8}$$

We use a different presentation for vectors throughout this part, in order to clearly identify each symbol in a vector, for instance, a length-$(N + 1)$ codeword, consisting of the information symbol $x[0]$ and the encoded parity symbols $x[1], ..., x[N]$, is denoted as $\mathbf{x}$.

5.2.1 Coding Gain of Tent Map Codes

Having discussed the encoding procedure, below we analyze the coding gain of chaotic analog codes. We start with the tent map codes, whose upsides and downsides motivate the proposed new design.

Conventional digital channel coding and analog coding both rely on space expansion and hence distance expansion to achieve coding gain. A digital code encodes an information sequence of length k ($\mathbf{U} \in \{0,1\}^k$) to a codeword sequence of length n ($c(\mathbf{U}) \in \{0,1\}^n$), or, projects a k-dimensional space onto an n-dimensional codeword space. This in general results in an increase of the Hamming distance between any pair of sequences. In the analog case, each (information) symbol takes a continuous value whose equivalent vector representation has an infinite dimensionality, and it is therefore impossible to further expand the *overall* vector space. Nevertheless coding gain is attainable through expanding a *neighborhood* vector space and hence magnifying the differences (distances) of two like symbols (vectors). The neighborhood spaces that were previously disjoint will in general overlap after expansion, giving rise to the renowned topologically mixing feature of a chaotic system.

To see this, consider the depth-1 encoding procedure shown in Fig. 5.4-a for the tent map. The depth-1 tent encoder encodes $x[0]$ to $x[1]$, and transmits $\mathbf{x}_0^1$. This is achieved by dividing the overall interval $(-1,+1]$ into two neighborhood intervals $(-1,0]$ and $(0,+1]$, and expanding and casting each neighborhood interval onto the original full interval $(-1,1]$ (see ((5.7)). Hence a pair of points in the same neighborhood interval (half-region), $x[0]$ and $x'[0]$, will see a dou-

bled Euclidean distance after being encoded to $x[1]$ and $x'[1]$. Similarly, in the depth-2 encoder shown in Fig. 5.7-b, the Euclidean distance for any pair of points within each quarter-region may potentially be quadrupled. In general, after N encoding steps, every $1/2^N$ section of $(-1, 1]$, namely, $(k2^{1-N}, (k+1)2^{1-N}]$ for $k = -2^{N-1}, -2^{N-1}+1, \cdots, 2^{N-1}-1$, will be expanded and mapped back onto the full interval $(-1, +1]$. As a result of this topological mixing, the Euclidean distance between a pair of points in the vicinity of each other (coming from the same $1/2^N$ section) may be potentially increased by 2^N times. In the case where two points are coming from distinctive sections, the sign vector $\mathbf{s}_0^N$ (signs of $\mathbf{x}_0^N$), known in the literature as *symbolic coding*, can be exploited to identify and locate each section and subsequently differentiate these two points. For instance, in Fig.5.4-b, the four quarter-sections are each associated with a unique symbolic coding $\mathbf{s}_0^1$. Suppose the symbolic coding is accurately known to the decoder. Suppose that the AWGN has variance δ^2, which means that per-symbol detection distortion has a statistical variance of δ^2. Then, a detection distortion occurred to a chaotic state at time N, $x[N]$, when mapped back to the original state at time 0, $x[0]$, will become a much smaller one with variance of only $\delta^2/2^N$. This indicates that the depth-N encoded state $x[N]$ has a noise-tolerant level that is potentially N times as large as that of the original state $x[0]$, resulting in a positive coding gain.

It should be noted, however, that the projected "n-times" gain is based on the ideal assumption the decoder gets the symbolic coding right. Clearly, symbolic coding plays a critical role in the estimation accuracy of the chaotic states. Besides, distortion error introduced by different erroneous signs has a non-uniform impact, since a mistaken sign of an earlier state will in general introduce a larger error than that of a later one. For example, in the depth-2 tent map in Fig. 5.4-b, a sign flip

of $s[0]$ may cause a detection distortion as large as 0.75, while a sign flip of $s[1]$ will cause a distortion of at most 0.5. If the symbolic coding is correctly detected for all the states involved, then the distortion would be controlled under 2^{-N} for a depth-N tent map. What this implies in the design of analog chaotic codes is the need to carefully protect symbolic coding, and the symbolic coding of the early states in particular, in order to effectively protect the information-bearing state $x[0]$. This observation motivates us to consider a parallel structure that assembles two tent maps in a turbo-like manner for a much needed double protection of the symbolic coding.

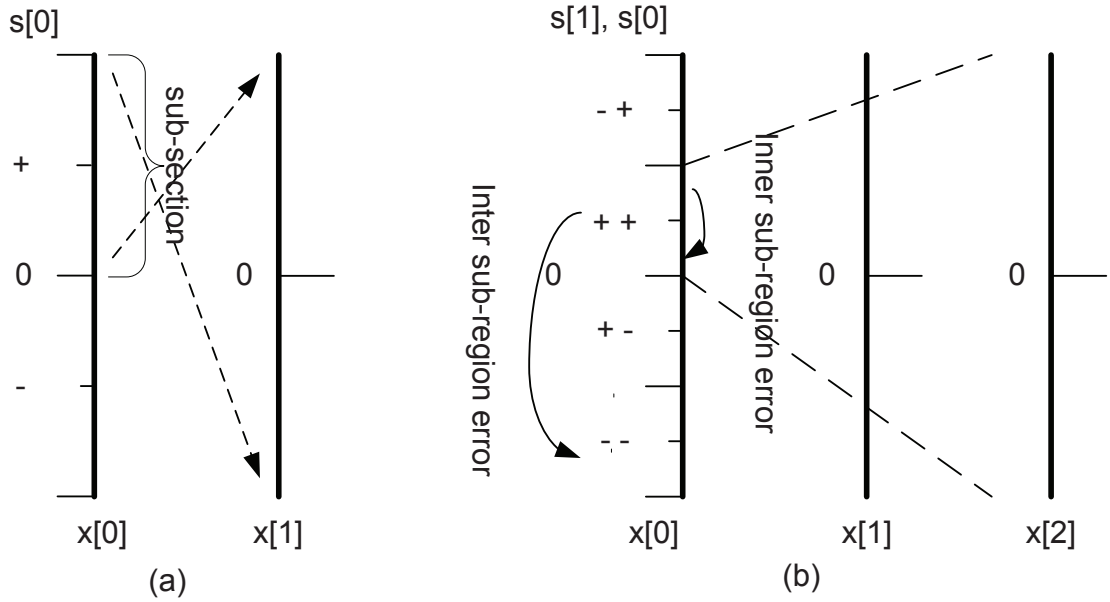


Figure 5.4: Understanding the encoding of chaotic analog code.

5.2.2 ML Decoding of Tent Map Codes

We now discuss detection of a single tent map. Let $(N+1)$ be the codeword length, $x[0]$ be the (initial) information symbol, and $\mathbf{x}_0^N$ and $\mathbf{y}_0^N$ be the codeword and the corresponding received signals respectively. An AWGN channel model with noise variance of δ^2 is considered for transmission. The ML decoder of the chaotic analog code based on the tent map can be written as:

$$\begin{aligned}\tilde{x}[0] &= \underset{-1<\tilde{x}[0]<\beta-1}{\arg\min}\ \mathbf{P}(\mathbf{y}_0^N|\tilde{x}[0]) \\ &= \underset{-1<\tilde{x}[0]<\beta-1}{\arg\min}\ \mathbf{P}(\mathbf{y}_0^N|F^0(\tilde{x}[0]),...,F^N(\tilde{x}[0])) \\ &= \underset{-1<\tilde{x}[0]<\beta-1}{\arg\min}\ \sum_{k=0}^{N}(y[k]-F^k(\tilde{x}[0]))^2, \end{aligned} \tag{5.9}$$

where $\mathbf{P}(\phi)$ denotes the probability of ϕ happening.

After replacing $F(x)$ and $F^k(x)$ with (5.6) and (5.28), one should, in theory, be able to solve $\tilde{x}[0]$ by taking derivatives. However, since $F(x)$ and $F^k(x)$ are defined piece-wise rather than a single smooth function, the complexity of the derivative operation increases exponentially with N, making the approach impractical for large N.

Papadopoulos and Wornell proposed a useful three-step decoding algorithm [11], thereafter referred to as the "backward decoding algorithm." The major steps of this detection algorithm is summarized as follows.

1) Estimate sign vector $\mathbf{s}_0^N$ via

$$\tilde{s}[n] = sgn[\sum_{k=0}^{N} \beta^{2(k-n)} F^{n-k}(y[k])], \tag{5.10}$$

where $0 \leq n \leq N$, and $\tilde{s}[n]$ is the estimated sign $s[n]$.

2) Estimate $x[N]$ via

$$\tilde{x}[0] = \frac{\sum_{k=0}^{N} \beta^{2(k-N)} F^{N-k}(y[k])}{\sum_{k=0}^{N} \beta^{2(k-N)}}. \tag{5.11}$$

3) Derive $\tilde{x}[0]$ via

$$\tilde{x}[N] = F^{-N}_{\tilde{s}[0]...\tilde{s}[N-1]}(\tilde{x}[N]), \tag{5.12}$$

where

$$F_s^{-1}(x) = \frac{1-x}{\beta} s$$

and

$$F^{-N}_{\tilde{s}[0]...\tilde{s}[N-1]}(x) = F^{-1}_{\tilde{s}[0]} \circ ... \circ F^{-1}_{\tilde{s}[N-1]}(x).$$

Compared to the ML decoding algorithm formulated in (5.30), the backward decoding algorithm has a notable advantage of lower complexity that is linear of N. Since it was not explicitly discussed in [11] the relation between backward decoding and ML decoding, below we first analyze its optimality.

Lemma 5.1: If $x[0]$ follows a uniform distribution, then the symbolic coding, $s[k] \triangleq sgn(x[k])$, is independent of $x[k+1], ..., x[N]$, $s[k+1], ..., s[N]$, and $y[k+1], ..., y[N]$ for all $k=0, ..., N$, where $sgn(x)$ denotes the sign of x.

Proof: First, the absolute value function $|x[k]|$ detaches the connection between $sgn(x[k])$ and $x[k+1], ..., x[N]$. Since $x[k]$ and $-x[k]$ have exactly the same probability to produce the same sequence $(x[k+1], ..., x[N])$, the sign of $x[k]$ is thus unrelated to $x[k+1], ..., x[N]$. Second, because $s[t]$ is solely determined by $x[t]$ (for $k+1 \leq t \leq N$), which is proved to be unrelated to $s[k]$, it follows that $s[k]$ is independent of $s[k+1], ..., s[N]$. Finally, since $y[t] = x[t] + n[t]$ where $n[t]$ is the additive independent of $x[\tau]$ or $s[\tau]$ for any τ and $x[t]$ at $t = k+1, \cdots, N$ is independent of $s[k]$, we have $y(t)$ for $k+1 \leq t \leq N$ is also independent of $s[k]$. □

Lemma 5.2: The backward decoding algorithm is suboptimal, but it approaches the performance of ML decoding in the limit of SNR going to infinity.

Proof: The exact ML estimation proceeds as

$$\begin{aligned} \tilde{x}[0] &= \underset{-1<\tilde{x}[0]<\beta-1}{\arg\min} \mathbf{P}(\mathbf{y}_0^N | \tilde{x}[0]) && (5.13) \\ &= \underset{-1<\tilde{x}[0]<\beta-1}{\arg\min} \mathbf{P}\Big(\mathbf{y}_0^N \Big| sgn(\tilde{x}[0]), F^N(\tilde{x}[0]), \\ &\qquad sgn(F(\tilde{x}[0])), \cdots, sgn(F^{N-1}(\tilde{x}[0]))\Big). && (5.14) \end{aligned}$$

Since $sgn(\tilde{x}[0])$, $F^N(\tilde{x}[0])$ and $sgn(F(\tilde{x}[0])), \cdots, sgn(F^{N-1}(\tilde{x}[0]))$ are all inde-

pendent, (5.14) can be rewritten as

$$\begin{aligned}\tilde{x}[0] = & \underset{-1<\tilde{x}[0]<\beta-1}{\arg\min} \mathbf{P}(\mathbf{y}_0^N|sgn(\tilde{x}[0]))\mathbf{P}(\mathbf{y}_0^N|F^N(\tilde{x}[0])) \\ & \mathbf{P}(\mathbf{y}_0^N|sgn(F(\tilde{x}[0])))\cdots\mathbf{P}(\mathbf{y}_0^N|sgn(F^{N-1}(\tilde{x}[0])) \\ = & \underset{-1<\tilde{x}[0]<\beta-1}{\arg\min} \mathbf{P}(y[0]|sgn(\tilde{x}[0]))\mathbf{P}(\mathbf{y}_0^N|F^N(\tilde{x}[0]))) \\ & \mathbf{P}(\mathbf{y}_0^1|sgn(F(\tilde{x}[0])))\cdots\mathbf{P}(\mathbf{y}_0^{N-1}|sgn(F^{N-1}(\tilde{x}[0])). \end{aligned} \tag{5.15}$$

Since there exists a one-to-one mapping between $x[0]$ and the sequence of $sgn(\tilde{x}[0])$, $sgn(F(\tilde{x}[0]))$, $\cdots$, $sgn(F^{N-1}(\tilde{x}[0]))$, $F^N(\tilde{x}[0])$, finding the value of $x[0]$ that minimizes (5.15) equates to finding each individual element in the sequence $\tilde{s}[0]$, $\tilde{s}[N-1]$, $\cdots$, $\tilde{x}[N]$. We have

$$\tilde{s}[k] = \underset{\tilde{s}[k]=\{+1,-1\}}{\arg\min} \mathbf{P}(\mathbf{y}_0^k|\tilde{s}[k]), \tag{5.16}$$

$$\tilde{x}[N] = \underset{-1<\tilde{x}[N]<\beta-1}{\arg\min} \mathbf{P}(\mathbf{y}_0^N|\tilde{x}[N]), \tag{5.17}$$

$$\tilde{x}[0] = F^{-N}_{\tilde{s}[0],\ldots,\tilde{s}[N-1]}(\tilde{x}[N]), \tag{5.18}$$

and $\mathbf{P}(\mathbf{y}_0^N|\tilde{x}[N])$

$$= \mathbf{P}(F^N(y[0])|\tilde{x}[N])\mathbf{P}(F^{N-1}(y[1])|\tilde{x}[N])\cdots\mathbf{P}(y[N]|\tilde{x}[N]). \tag{5.19}$$

When the SNR goes to infinity, $F^{N-k}(y[k])$ approaches a Gaussian distribution

with mean $x[N]$ and variance $\delta^2\beta^{2(N-k)}$. Replacing (5.19) with the probability density function of a Gaussian distribution and taking derivatives, we get (5.11) in the backward decoding algorithm. Further, considering that $\tilde{x}[N]$ is a weighted sum of $F^{N-k}(y[k])$ for $0 \leq k \leq N$, and that $\tilde{x}[N]$ forms a new Gaussian random variable with mean $x[N]$, we can obtain (5.10) using the same procedure shown above.

However, when the SNR is finite, albeit large, $F^{N-k}(y[k])$ is close to a Gaussian distribution with mean $x[N]$ and variance (slightly) smaller than $\delta^2\beta^{2(N-k)}$. The difference is caused by a large noise sample $n[k]$, $(0 \leq k \leq N)$ which is greater than $1/2^{N-k}$ $(0 \leq k \leq N)$. Hence, the backward decoding algorithm is in general sub-optimal, but approaches ML decoding as SNR increases without bound. □

Fig. 5.5 compares the true ML decoding and the backward decoding, and confirms this result.

The previous discussion also reveals an important fact, namely, the use of a length-k sequence $\mathbf{y}_0^{k-1}$ to estimate $s[k-1]$, for $k = 0, \cdots, N$. This becomes a problem when k is small, and in particular, when $k = 0$, we have only exploited $y[0]$ to estimate $s[0]$. This means that $s[0]$, which is most important and whose erroneous decision could cause the worst detection distortion, has actually received the weakest protection. This observation motivates us to design a new coding scheme, one that offers better protection for $s[0]$ and other symbolic coding of small indices.

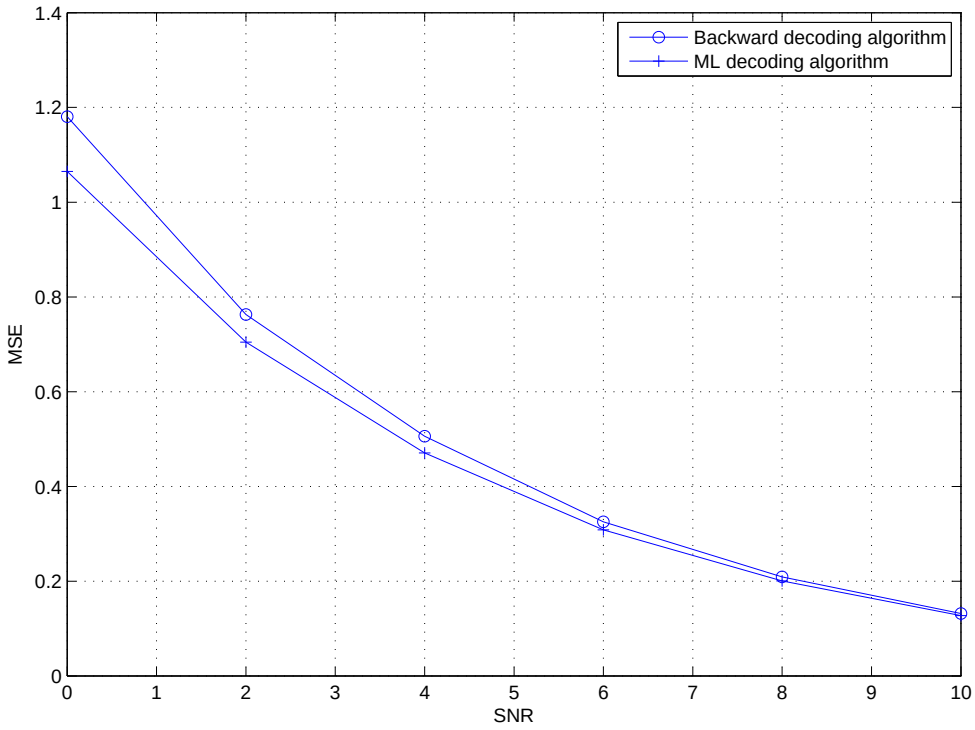


Figure 5.5: Comparison between ML decoding and backward decoding $N = 5$

5.3 CAT codes and SISO MAP decoding

We have shown that a single tent map, falls short in protecting symbolic coding $\bar{s}_0^{N-1}$, since exactly where the protection is most needed actually receives the least of it. Further, unlike digital coding coupled with the BPSK modulation, in which each codeword or symbol is transmitted with the same power, the transmit power of an analog symbol $x[k]$ is squared proportional to the magnitude of $x[k]$. Since the estimated symbolic coding $\tilde{s}[k]$ is much dominated by $y[k]$, a small absolute value of $x[k]$ (low transmit power) makes $y[k]$ less reliable and $\tilde{s}[k]$ more prone to error. This, adding to the insufficient protection of symbolic coding of small indices, could lead to rather undesirable performance loss.

To enhance and especially balance the protection of symbolic coding, we propose to construct chaotic analog turbo codes by introducing a second tent map associating it with a pre-determined symbol coding. Specifically, in this work, we let this pre-determined symbolic coding be the reverse order (reversed indices) of that from the first tent map. Thus, $s[k]$ with a small indice k, which gets little protections at the first tent encoder, is now gaining a better protection from the second tent encoder. This is close in spirit to that of the classic turbo code, one that tries to make a weakly-protected sequence (i.e. low-weight codeword) from the first component code to hopefully get better protection (i.e. become a high-weight one) from the second component code.

Fig.5.6 describes the encoding procedure of CAT codes based on two tent maps.

1. Generate the first analog sequence $x_1[0]$, ..., $x_1[N]$ by feeding the information symbol $x_1[0]$ into the first tent map.

2. Extract the symbolic coding $s_1[0]$, ..., $s_1[N-1]$.

3. Reverse the order of s_0^{N-1} to generate the symbolic coding for the second tent map: $s_2[0] = s_1[N-1], \cdots, s_2[N-1] = s_1[0]$.

4. Let $x_2[N] = x_1[N]$. Generate the second chaotic sequence $x_2[N-1]$, ..., $x_2[0]$ by using the inverse function $F^{-1}(x)$ and forcing the signs of the new chaotic sequence to be $s_2[N-1]$, ..., $s_2[0]$.

The proposed decoder follows three steps.

1. Estimate the symbolic coding $s_1[0]$, ..., $s_1[N-1]$ using both tent maps.

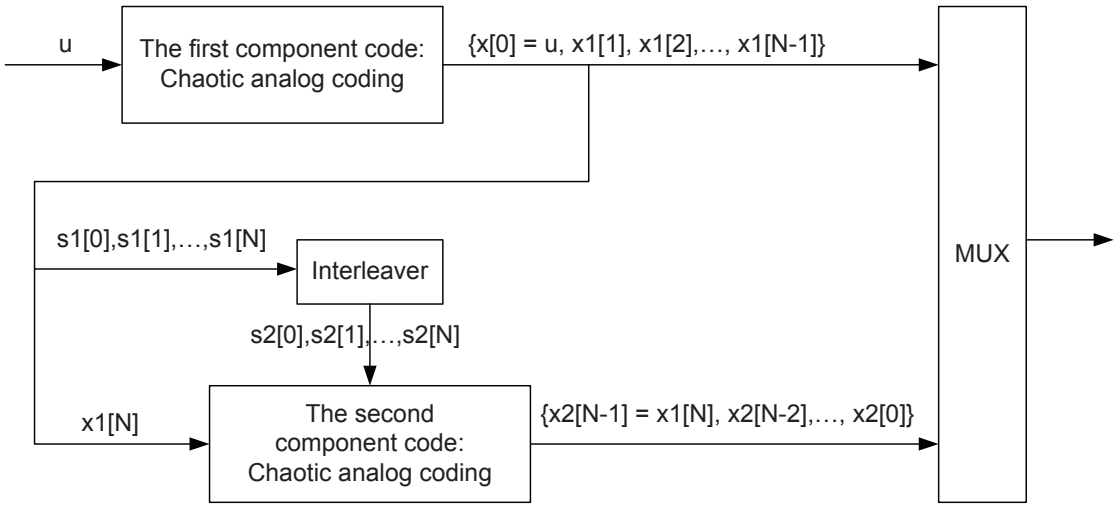


Figure 5.6: Encoding scheme of CAT codes

2. Estimate $x_1[N]$.

3. Derive $x_1[0]$ from (5.12).

Instead of using the backward decoding algorithm to estimate the symbolic coding in the first step, here we exploit the renowned soft-iterative approach based on soft-input soft-output (SISO) maximum *a posterior* (MAP) decoding of each component tent map. To facilitate SISO MAP decoding, we define a likelihood ratio (LLR) for $s[k]$.

(i) When $k = N - 1$, we have

$$\begin{aligned} LLR_{\tilde{s}[N-1]} =& log\frac{\mathbf{P}(\mathbf{y}_0^{N-1}|\tilde{s}[N-1]=+1)\mathbf{P}(\tilde{s}[N-1]=+1))}{\mathbf{P}(\mathbf{y}_0^{N-1}|\tilde{s}[N-1]=-1)\mathbf{P}(\tilde{s}[N-1]=-1))} \\ =& log\frac{e^{\frac{(G[N-1]-x_+[N-1])^2}{2\delta^2}}\mathbf{P}(\tilde{s}[N-1]=+1)}{e^{\frac{(G[N-1]-x_-[N-1])^2}{2\delta^2}}\mathbf{P}(\tilde{s}[N-1]=-1)}, \end{aligned} \tag{5.20}$$

where $G[n] = \frac{\sum_{k=0}^{n}\beta^{2(k-n)}F^{n-k}(y[k])}{\sum_{k=0}^{n}\beta^{2(k-n)}}$. We use $F_{+1}^{-1}(y[N])$ to estimate $x_{\tilde{s}[N-1]=+1}[N-1]$,

and $F_{-1}^{-1}(y[N])$ to estimate $x_{\tilde{s}[N-1]=-1}[N-1]$, and therefore get

$$LLR_{\tilde{s}[N-1]} = log \frac{e^{\frac{(G[N-1]-F_{+1}^{-1}(y[N]))^2}{2\delta^2}} \mathbf{P}(\tilde{s}[N-1]=+1)}{e^{\frac{(G[N-1]-F_{-1}^{-1}(y[N]))^2}{2\delta^2}} \mathbf{P}(\tilde{s}[N-1]=-1)}. \tag{5.21}$$

(ii) When $k < N-1$, we have

$$\begin{aligned} LLR_{\tilde{s}[k]} &= log \frac{\mathbf{P}(\mathbf{y}_0^k|++)\mathbf{P}(++) + \mathbf{P}(\mathbf{y}_0^k|+-)\mathbf{P}(+-)}{\mathbf{P}(\mathbf{y}_0^k|-+)\mathbf{P}(-+) + \mathbf{P}(\mathbf{y}_0^k|--)\mathbf{P}(--)}, \\ &= log \frac{e^{\frac{(G[k]-x_{++}[k])^2}{2\delta^2}} \mathbf{P}(++) + e^{\frac{(G[k]-x_{+-}[k])^2}{2\delta^2}} \mathbf{P}(+-)}{e^{\frac{(G[k]-x_{-+}[k])^2}{2\delta^2}} \mathbf{P}(-+) + e^{\frac{(G[k]-x_{--}[k])^2}{2\delta^2}} \mathbf{P}(--)}, \end{aligned} \tag{5.22}$$

where (X_1X_2) with $X_1, X_2 \in \{+,-\}$ represents $(\tilde{s}[k]=X_1, \tilde{s}[k+1]=X_2)$. We next use

$$\frac{F_{X_1}^{-1}(y[N]) + \beta^2 F_{X_1X_2}^{-2}(y[N])}{1+\beta^2} \tag{5.23}$$

to estimate the $x_{X_1X_2}[k]$ in (5.22).

In (5.21) and (5.22), the *a priori* probabilities $\mathbf{P}(\tilde{s}[N-1]=+1)$ and $\mathbf{P}(\tilde{s}[N-1]=-1)$, and $\mathbf{P}(++)$, $\mathbf{P}(+-)$, $\mathbf{P}(-+)$ and $\mathbf{P}(--)$ are calculated from $LLR_{\tilde{s}[k]}$, passed from the other component decoder.

The iterative decoding process starts from $LLR_{\tilde{s}[k]}=0$ for $0 \le k \le N$. The two component decoders exchange and refines $LLR_{\tilde{s}[k]}$ with iterations, and at the end of the iterations, computes the hard decision $s_1[0]$, ..., $s_1[N-1]$.

5.3.1 Performance Simulation of CAT codes

We simulate and compare CAT codes with the tent map code, for the same code length 11 ($x[0]$ and 10 parity symbols). We choose the distortion $log_2|mean((x[0]-\tilde{x}[0])^2)|$ as the metric to evaluate the performance. Fig 5.7 plots the performance curves. The CAT scheme outwins tent map codes by some 4dB gain at high SNR!

The aforementioned chaotic analog codes transmit the coded sequence $\mathbf{x}_0^N$ to protect the information-bearing symbol $x[0]$. An alternative system may transmit $x[N]$ and $\mathbf{s}_0^{N-1}$, which carry the equivalent information about $x[0]$. When transmitting $\mathbf{s}_0^{N-1}$, the binary sequence, we have the natural option of sending it through the conventional digital communication system. If a system exploits both the conventional digital system (for transmitting $\mathbf{s}_0^{N-1}$) and the analog system (for transmitting $x[N]$), we call it a "hyper system".

We simulate two hyper systems with BPSK modulation and repetition codes. The first uses (uncoded) BPSK only for $\mathbf{s}_0^{N-1}$, and the second protects $\mathbf{s}_0^{N-1}$ using a rate-1/2 repetition code followed by BPSK modulation. Fig. 5.8 compares the performance of these two hyper systems with the CAT chaotic system and Chen-Wornell system. Performance curves show that the proposed CAT chaotic system outperforms all of the other three systems.

5.4 2-D Chaotic Analog Codes: Mirrored Baker's Map Codes

The tent map code has successfully demonstrated the possibility of exploiting chaotic systems to achieve error correction [11], but its performance awaits to be desired. This is in part because it is a one-dimensional code that encodes only a single source symbol u_i at a time. From the coding theory, a larger batch involving more source symbols at a time can provide a richer context and potentially a stronger error correction capability. This suggests that high-dimensional chaotic systems may provide better candidates for coding construction. Since a high dimension also implies a high complexity, below we explore two-dimensional systems to hopefully strike a good balance between complexity and performance.

Searching over the literature, we have identified the 2-dimensional baker's map as a suitable candidate for code construction. Below we first discuss our idea of constructing the new code, and then develop the corresponding ML decoding and its simplification.

5.4.1 Encoding of Baker's Map Codes

The baker's map is a nonlinear chaotic function named after a kneading operation that bakers apply to dough. Figure 5.9 explains the entire process, and as an

example we choose the tent map with $\beta = 2$ as the "dough":

$$\begin{aligned} x[k] &= F(x[k-1]) \\ &= \begin{cases} 2x[k-1]+1, & \text{if } x[k-1] < 0 \\ 1-2x[k-1], & \text{otherwise} \end{cases} \\ -1 \leq\ x[0] &\leq 1. \end{aligned}$$

The process of the baker's map is summarized as below.

- First, the dough is compressed in vertical direction and stretched in horizon direction.
- Then, the dough is cut in half, and the two halves are stacked on one-another.
- After that, we get the mapping for y and a new 2-dimensional mapping as

$$\{x[k], y[k]\} \tag{5.24}$$

$$= F(\{x[k-1], y[k-1]\}) \tag{5.25}$$

$$= \begin{cases} \{2x[k-1]+1, \frac{y[k-1]}{2} - \frac{1}{2}\}, & \text{if } x[k-1] < 0 \\ \{1-2x[k-1], \frac{1}{2} - \frac{y[k-1]}{2}\}, & \text{otherwise} \end{cases} \tag{5.26}$$

where $-1 < x[0], y[0] < 1$.

The baker's map by itself can form an analog code, but does not perform very well, because $F(\{x[k-1], y[k-1]\})$ is not symmetrical, in the sense that the current x is used to generate the next value of y yet the conversance does not hold). Hence,

if directly applying the baker's map to analog coding, information x is embedded and protected by *two* codewords ($\mathbf{x}$ and $\mathbf{y}$), while information y is only protected by *one* codeword ($\mathbf{Y}$). Aware of this unbalanced protection, we propose a new coding scheme that makes use of *a pair* of (rather than one) baker's maps in a mirrored duplicate fashion to avoid the undesirable biased protection. The basic idea is to introduce two identical baker's maps $F(x, y)$ which work in parallel to encode the same source sequence $\mathbf{u} = \{u^{2i}, u^{2i+1}\}$. Specifically, $\{u^{2i}, u^{2i+1}\}$ and its mirror $\{u^{2i+1}, u^{2i}\}$ are fed into the first baker's map and the second baker's map respectively. We name this new 2-dimensional chaotic analog codes *the mirrored baker's map codes*. The unbalance protection is canceled out by switching the positions of inputs. Further, two parallel baker's maps enable the transformation from one-dimensional analog coding into two-dimensional coding, promising larger coding gain in theory.

Specifically, as shown in Figure 5.10, the discrete-time real-valued source sequence $\mathbf{u} = \{u^0, u^1, ...\}$ is divided into small coding frames with length 2. For the i*th* frame, encoder transforms every pair of source symbols $\{u^{2i}, u^{2i+1}\}$ into two length-N analog symbol-pair sequences $\{\mathbf{x}_1^i, \mathbf{y}_1^i\}$ and $\{\mathbf{x}_2^i, \mathbf{y}_2^i\}$, where $\{\mathbf{x}_1^i, \mathbf{y}_1^i\} = \{x_1^i[0], y_1^i[0]\}, \{x_1^i[1], y_1^i[1]\}, \ldots, \{x_1^i[N-1], y_1^i[N-1]\}$ and $\{\mathbf{x}_2^i, \mathbf{y}_2^i\} = \{x_2^i[0], y_2^i[0]\}, \{x_2^i[1], y_2^i[1]\}, \ldots, \{x_2^i[N-1], y_2^i[N-1]\}$. Note that the initial information pairs $\{x_1^i[0], y_1^i[0]\}$ and $\{x_2^i[0], y_2^i[0]\}$ equal the source pair $\{u^{2i}, u^{2i+1}\}$ and its mirrored version $\{u^{2i+1}, u^{2i}\}$ respectively. The rest parity pairs are recursively generated by

the 2-dimensional chaotic baker's map

$$\begin{cases} \{x_1^i[k], y_1^i[k]\} &= F(\{x_1^i[k-1], y_1^i[k-1]\}) \\ &= F^k(\{u^{2i}, u^{2i+1}\}) \\ \\ \{x_2^i[k], y_2^i[k]\} &= F(\{x_2^i[k-1], y_2^i[k-1]\}) \\ &= F^k(\{u^{2i+1}, u^{2i}\}), \end{cases} \tag{5.27}$$

where $F(x, y)$ is defined by baker's map as in (5.26) and

$$F^k(x,y) = \begin{cases} (x,y) & k = 0 \\ \underbrace{F \circ F \circ ... \circ F(x,y)}_{k}. & k \neq 0, \end{cases} \tag{5.28}$$

Without loss of generality, we omit the superscription i for the ith frame in this work and use $\{\mathbf{x_1}, \mathbf{y_1}\}$, $\{\mathbf{x_2}, \mathbf{y_2}\}$ to represent the codeword, in which each entry is expressed $\{x_1[i], y_1[i]\}$, $\{x_2[i], y_2[i]\}$. Suppose that the codeword $\{\mathbf{x_1}, \mathbf{y_1}\}$, $\{\mathbf{x_2}, \mathbf{y_2}\}$ passes through an AWGN channel, then a noisy codeword $\{\mathbf{Rx_1}, \mathbf{Ry_1}\}$ and $\{\mathbf{Rx_2}, \mathbf{Ry_2}\}$ is received at the receiver side.

$$\begin{cases} \mathbf{Rx_1} &= \mathbf{x_1} + \mathbf{w_1} \\ \mathbf{Ry_1} &= \mathbf{y_1} + \mathbf{w_2} \\ \mathbf{Rx_2} &= \mathbf{x_2} + \mathbf{w_3} \\ \mathbf{Ry_2} &= \mathbf{y_2} + \mathbf{w_4} \end{cases} \tag{5.29}$$

where $\mathbf{w_1}$, $\mathbf{w_2}$, $\mathbf{w_3}$ and $\mathbf{w_4}$ are the Gaussian noise sequence.

5.4.2 ML decoding of Mirrored Baker's codes

We start with the ML decoding for the proposed mirrored baker's map codes. Assuming $u = u^{2i}$ and $v = u^{2i+1}$ for each ith frame, the ML decoder will provide an estimate of $\{u, v\}$ as $\{\tilde{u}, \tilde{v}\}$ which satisfying

$$\begin{aligned} & \{\tilde{u}, \tilde{v}\} \\ = & \arg\min_{-1<\tilde{u},\tilde{v}<1} \mathbf{P}(\{\mathbf{Rx_1}, \mathbf{Ry_1}, \mathbf{Rx_2}, \mathbf{Ry_2}\}|\{\tilde{u}, \tilde{v}\}), \end{aligned}$$

where $\mathbf{P}(\mathbf{x})$ denotes the probability of x. All the discussions in this section will continue the example in (5.26), but they can be easily extended to other chaotic mapping methods.

The ML decoder of the mirrored baker's codes can be further written as:

$$\begin{aligned} & \{\tilde{u}, \tilde{v}\} \\ = & \arg\min_{-1<\tilde{u},\tilde{v}<1} \mathbf{P}(\{\mathbf{Rx_1}, \mathbf{Ry_1}\}_0^{N-1}|F_0^{N-1}(\{\tilde{u}, \tilde{v}\}))) \\ & \quad \mathbf{P}(\{\mathbf{Rx_2}, \mathbf{Ry_2}\}_0^{N-1}|F_0^{N-1}(\{\tilde{v}, \tilde{u}\}))) \\ = & \arg\min_{-1<\tilde{u},\tilde{v}<1} \prod_{i=0}^{N-1} \mathbf{P}(\{Rx_1[i], Ry_1[i]\}|F^i(\{\tilde{u}, \tilde{v}\})) \\ & \quad \mathbf{P}(\{Rx_2[i], Ry_2[i]\}|F^i(\{\tilde{v}, \tilde{u}\})), \end{aligned} \tag{5.30}$$

where $F_0^{N-1}(\{\tilde{u}, \tilde{v}\}) = \{F^0(\{\tilde{u}, \tilde{v}\}), ..., F^{N-1}(\{\tilde{u}, \tilde{v}\})\}$.

We assume Gaussian noise and rewrite $F^i(\{\tilde{u}, \tilde{v}\})$ and $F^i(\{\tilde{v}, \tilde{u}\})$ in (5.30) as

$\{\tilde{x}_1[i], \tilde{y}_1[i]\}$ and $\{\tilde{x}_2[i], \tilde{y}_2[i]\}$ respectively. Then, we have

$$\{\tilde{u}, \tilde{v}\} = \arg\min_{-1<\tilde{u},\tilde{v}<1} \sum_{i=0}^{N-1} (Rx_1[i] - \tilde{x}_1[i])^2 (Ry_1[i] - \tilde{y}_1[i])^2 \\ (Rx_2[i] - \tilde{x}_2[i])^2 (Ry_2[i] - \tilde{y}_2[i])^2. \quad (5.31)$$

All the four terms $Rx_1[i]$, $Ry_1[i]$, $Rx_2[i]$, and $Ry_2[i]$ can be expressed as a function of u and v. To get the optimal solution of $\{\tilde{u}, \tilde{v}\}$, i.e. to make (5.31) achieve the minim value, we can take the differential of function (5.31). While the function $F^i(\{\tilde{u}, \tilde{v}\})$ and $F^i(\{\tilde{v}, \tilde{u}\})$ are not differentiable over the defined range of $\{u, v\}$, both $F^i(\{\tilde{u}, \tilde{v}\})$ and $F^i(\{\tilde{v}, \tilde{u}\})$ are piece-wise linear and differentiable with respect to u and v. Take $F^i(\{\tilde{u}, \tilde{v}\})$ as an example, Figure 5.11 and 5.12 show the curves of $\tilde{x}_1[1]$, $\tilde{x}_1[n]$, $\tilde{y}_1[1]$ and $\tilde{y}_1[3]$ in terms of $\{u, v\}$. $\tilde{x}_1[n]$ is linear to $\tilde{x}_1[0]$ in any range from $\frac{i}{2^{n-1}}$ to $\frac{i+1}{2^{n-1}}$, where $-2^{n-1} \leq i < 2^{n-1}$. The same holds for $\tilde{y}_1[n]$, which is linear to $\tilde{x}_1[0]$ and $\tilde{y}_1[0]$ in the range from $\frac{i}{2^{n-1}}$ to $\frac{i+1}{2^{n-1}}$, where $-2^{n-1} \leq i < 2^{n-1}$.

Consider a length-N baker's map, the definition range of u can be divided into 2^{N-1} subregions. Within each subregion, all the function $F^i(u, v)$ with $0 \leq i < N$ are linear and differentiable. the sign sequence $\mathbf{s}_{\mathbf{1}0}^{N-2}$ of sequence $\mathbf{x}_{\mathbf{1}0}^{N-2}$ is determined. Therefore, each subregion can be represented by $\mathbf{s}_{\mathbf{1}0}^{N-2}$. By given any sign sequence $\mathbf{s}_{\mathbf{1}0}^{N-2}$,we can rewrite the expression of $F^n(\{u, v\})$ (5.26) as

$$\begin{aligned} & \{\tilde{x}_1[n], \tilde{y}_1[n]\} \\ = \; & F(\{\tilde{x}_1[n-1], \tilde{y}_1[n-1]\}) \\ = \; & \{a_1[n-1]\tilde{u} + b_1[n-1], a_2[n-1]\tilde{v} + b_2[n-1]\}, \end{aligned} \quad (5.32)$$

where $\tilde{x}_1[0] = \tilde{u}$, $\tilde{y}_1[0] = \tilde{v}$. $a_1[n]$, $b_1[n]$, $a_2[n]$, and $b_2[n]$ are the parameters and

can be obtained by the following recursive equations

$$\begin{cases} a_1[n] &= -2a_1[n-1]s_1[n-1]; \\ b_1[n] &= 1-2s_1[n-1]b_1[n-1]; \\ a_2[n] &= (-1/2)s_1[n-1]a_2[n-1]; \\ b_2[n] &= 1/2s_1[n-1]-1/2s_1[n-1]b_2[n-1]; \end{cases} \tag{5.33}$$

$$a_1[0]=1; b_1[0]=0; a_2[0]=1; b_2[0]=0. \tag{5.34}$$

The previous discussion is based solely on the first baker's map in the system $\{\tilde{x}_1[n], \tilde{y}_1[n]\}$. Similarly, we can derive $\{\tilde{x}_2[n], \tilde{y}_2[n]\}$ for the second baker's map as

$$\{\tilde{x}_2[n], \tilde{y}_2[n]\} \tag{5.35}$$

$$= F(\{\tilde{x}_1[n-1], \tilde{y}_1[n-1]\}) \tag{5.36}$$

$$= \{a_3[n-1]\tilde{v}+b_3[n-1], a_4[n-1]\tilde{u}+b_4[n-1]\}. \tag{5.37}$$

The parameter $\mathbf{A_3}_0^{N-1}$ and $\mathbf{A_4}_0^{N-1}$ can be generated by the same recursive equations as (5.33) and (5.34), but changing the sign sequence from $\mathbf{S_1}_0^{N-2}$ to $\mathbf{S_2}_0^{N-2}$.

Therefore, we can further simplify (5.31)

$$\{\tilde{u}, \tilde{v}\} \tag{5.38}$$

$$= \underset{-1<\tilde{u},\tilde{v}<1, \mathbf{S_1}_0^{N-2}, \mathbf{S_2}_0^{N-2}}{\arg\min} \sum_{i=0}^{N-1} (Rx_1[i]-a_1[i]\tilde{u}-b_1[i])^2 \\ (Ry_1[i]-a_2[i]\tilde{u}-b_2[i])^2 (Rx_2[i]-a_3[i]\tilde{v}-b_3[i])^2 \\ (Ry_2[i]-a_4[i]\tilde{u}-b_4[i])^2. \tag{5.39}$$

By taking the derivative, we have

$$\tilde{u} = \underset{\mathbf{s_1}_0^{N-2},\mathbf{s_2}_0^{N-2}}{\arg\min} \frac{T_1}{a_1^2[i] + a_4^2[i]}, \tag{5.40}$$

$$\tilde{v} = \underset{\mathbf{s_1}_0^{N-2},\mathbf{s_2}_0^{N-2}}{\arg\min} \frac{T_2}{a_2^2[i] + a_3^2[i]}, \tag{5.41}$$

where

$$T_1 = \sum_{i=0}^{N-1} Rx1[i]a_1[i] - a_1[i]b_1[i] + Ry_2[i]a_4[i] - a_4[i]b_4[i], \tag{5.42}$$

$$T_2 = \sum_{i=0}^{N-1} Ry_1[i]a_2[i] - a_2[i]b_2[i] + Rx_2[i]a_3[i] - a_3[i]b_3[i]. \tag{5.43}$$

Till now, we have proposed an ML decoding algorithm (5.30) for the mirrored baker's map codes. Under the Gaussian noise assumption, we have further simplified it and derived a close-form optimal $\{u^{2i}, u^{2i+1}\}$ in (5.40) and (5.41) with given sign sequence $\mathbf{s_1}_0^{N-2}$ and $\mathbf{s_2}_0^{N-2}$.

5.4.3 Performance of Mirrored Backer's Map codes

To verify the coding gain brought by the multi-dimension, this section compares the new 2-dimensional mirrored baker's map codes with the 1-dimensional tent map codes, at the same code rate 12 (1 systematic symbol and 11 parity symbols).

For different analog systems, each analog source symbol may take a different value range. Given the same encoder and decoder, the distortion is proportional to the source range. To provide a fair comparison of analog systems, a normalized

distortion δ is evaluated, where

$$\delta = \log_2 \frac{Avg((\mathbf{U} - \tilde{\mathbf{U}})^2)}{\max(U) - \min(U)}, \tag{5.44}$$

U denotes the source sequence and $Avg(X)$ represents the mean of vector $\mathbf{X}$.

The performance curves of the tent map codes and mirrored baker's map codes are shown and compared in Figure 5.13. A $8dB$ coding gain is observed.

5.5 Analog vs Digital systems

For much of the half-century history of modern communications, channel coding has been almost exclusively considered as a *digital-only* technology. In traditional digital communication, an analog information source will be firstly quantized to a binary sequence of 0s and 1s and subsequently be mapped from analog signal to digital signal with quantization error. Then, after some digital processing, such as channel coding, the digital sequence is modulated and mapped back from digital signal to analog signal. The introduction of analog codes can save the effort of quantization and also avoid the quantization error!

We now compare the performance of digital convolutional codes, digital turbo codes and the mirrored baker's map codes. A discrete-time real-valued source sequence $\mathbf{u}$ is fed into both digital systems and analog systems. For the digital coding systems, a uniform scalar quantizer is performed before the digital error correction codes. The output codeword of the encoder will be modulated to BPSK or 4PAM sequence. For the analog coding system, the source sequence $\mathbf{u}$ will

encoded to $\mathbf{v}$ by the chaotic analog coding directly. All the systems are evaluated by the normalized distortion in (5.44) and kept the same bandwidth ratio

$$R_w = \frac{|\mathbf{v}|}{|\mathbf{u}|}, \tag{5.45}$$

where $|\mathbf{x}|$ indicates the size of vector $\mathbf{x}$.

Figure 5.14 shows that the mirrored baker's map codes outperforms the traditional convolutional code with both BPSK and 4PAM modulation. A low-order level modulation for digital systems will likely improve the waterfall region but will also increase the quantization error floor as fewer quantization bits can be afforded. A balanced chaotic analog codes such as the mirrored baker's map code provides a larger than $5dB$ gain over the high-rate turbo codes with the component code of [15, 17] and code length of 1000 at low SNRs, and offers a smaller distortion than the low-rate turbo codes at high SNRs.

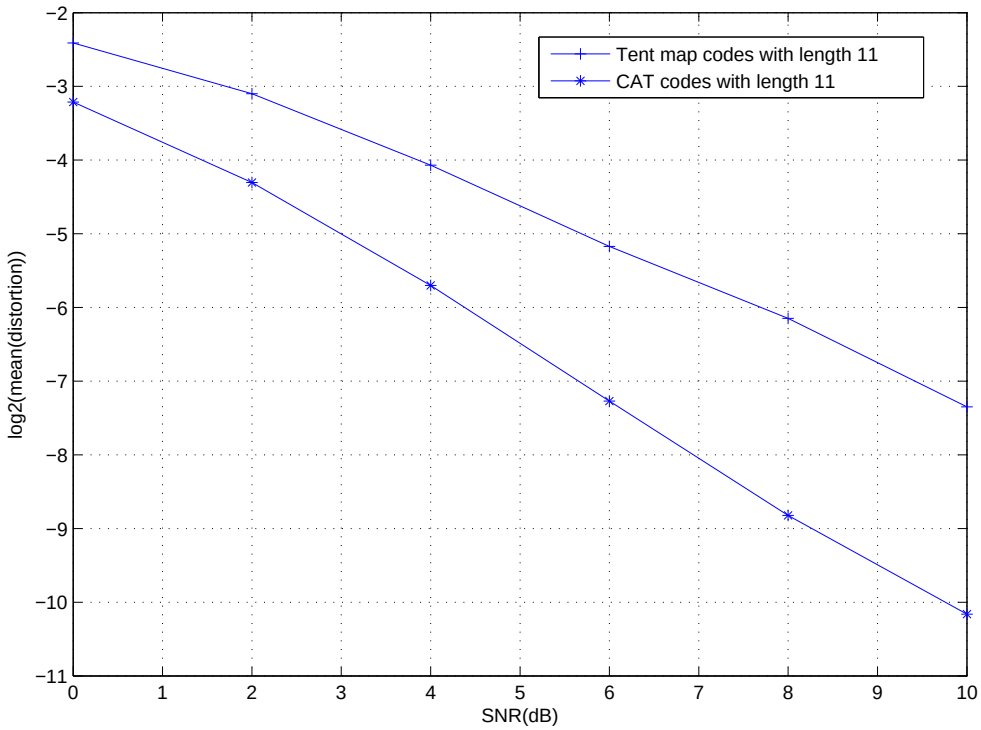


Figure 5.7: Comparison between CAT codes and tent map codes

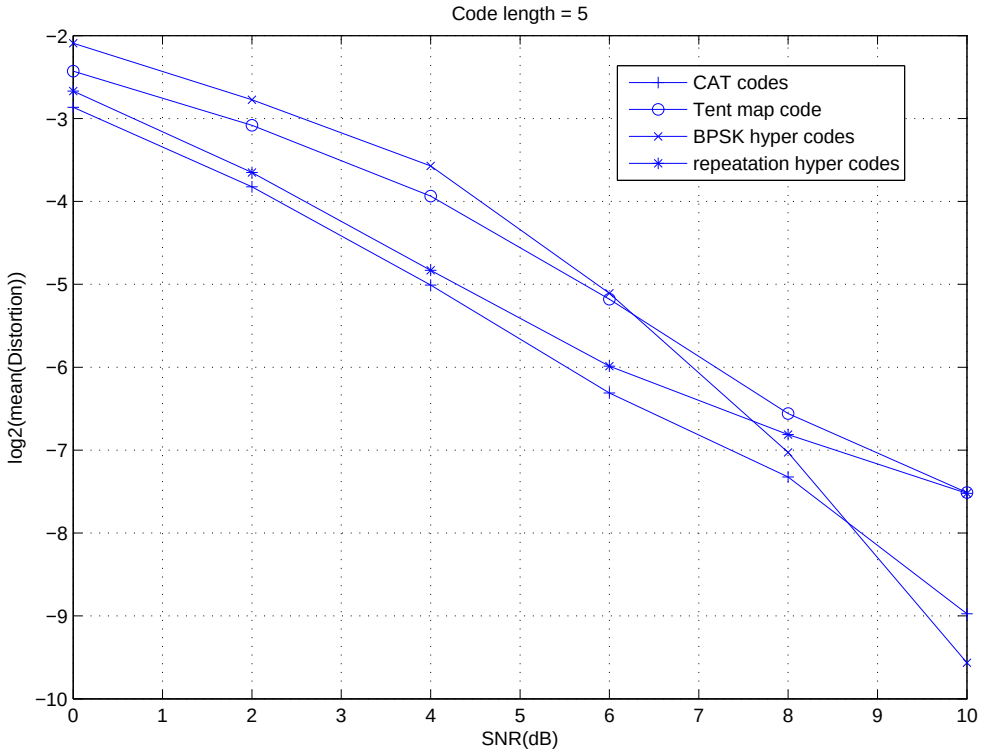


Figure 5.8: Comparison between of CAT codes and BPSK hyper codes, repetition hyper codes

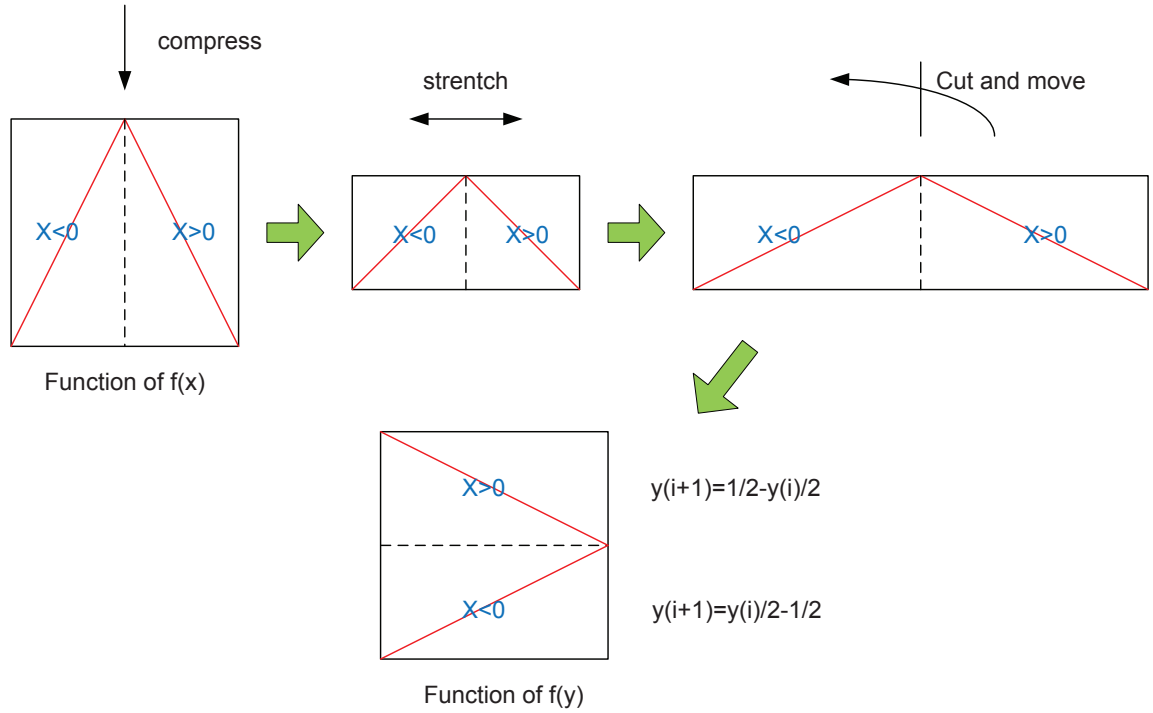


Figure 5.9: The process of baker's map

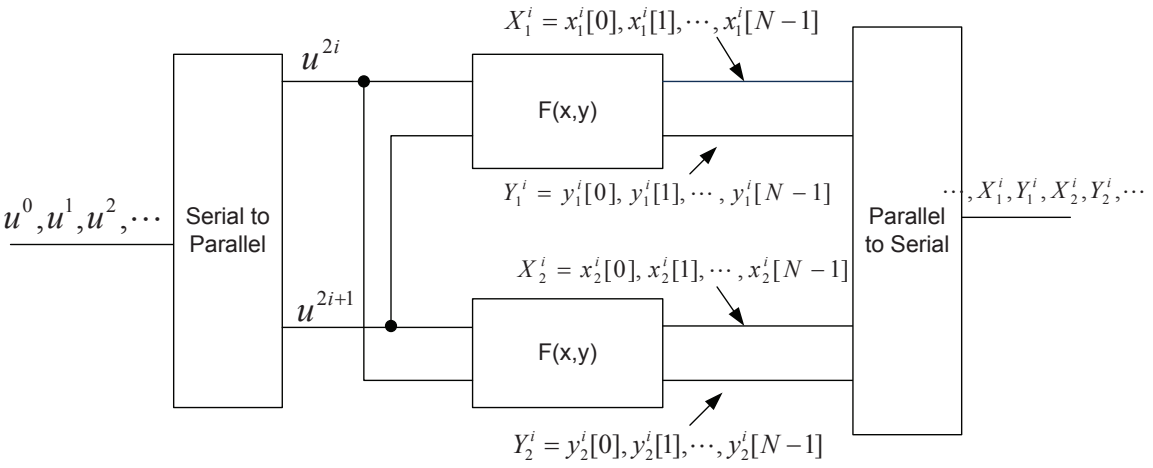


Figure 5.10: the system model of 2-D chaotic analog codes

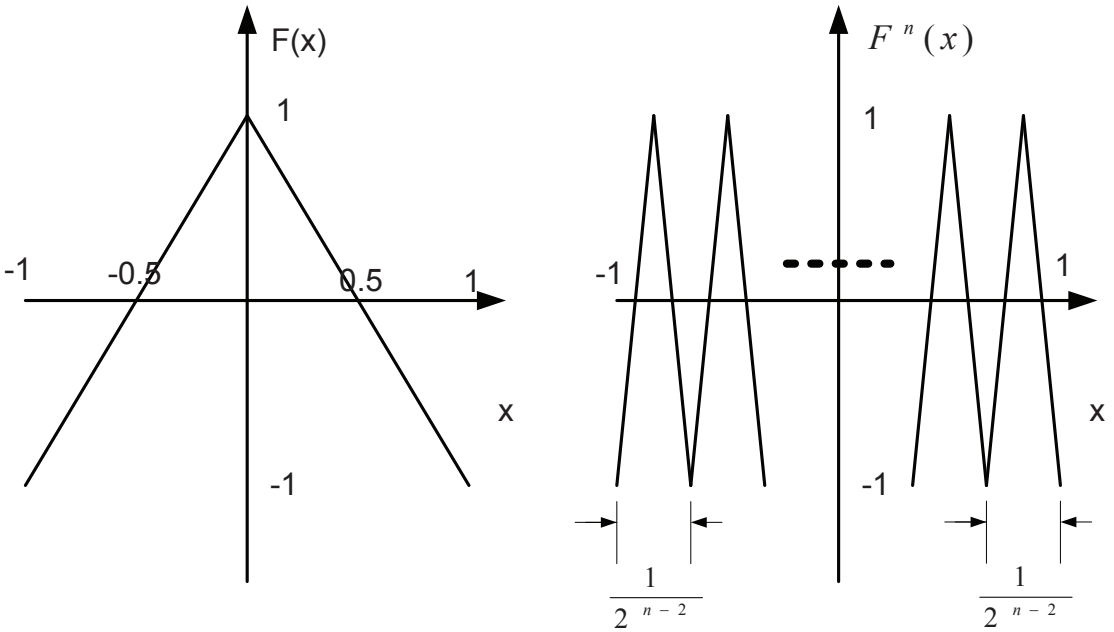


Figure 5.11: Function curve of $x_1[1]$ and $x_1[n-1]$ in terms of $\{u, v\}$.

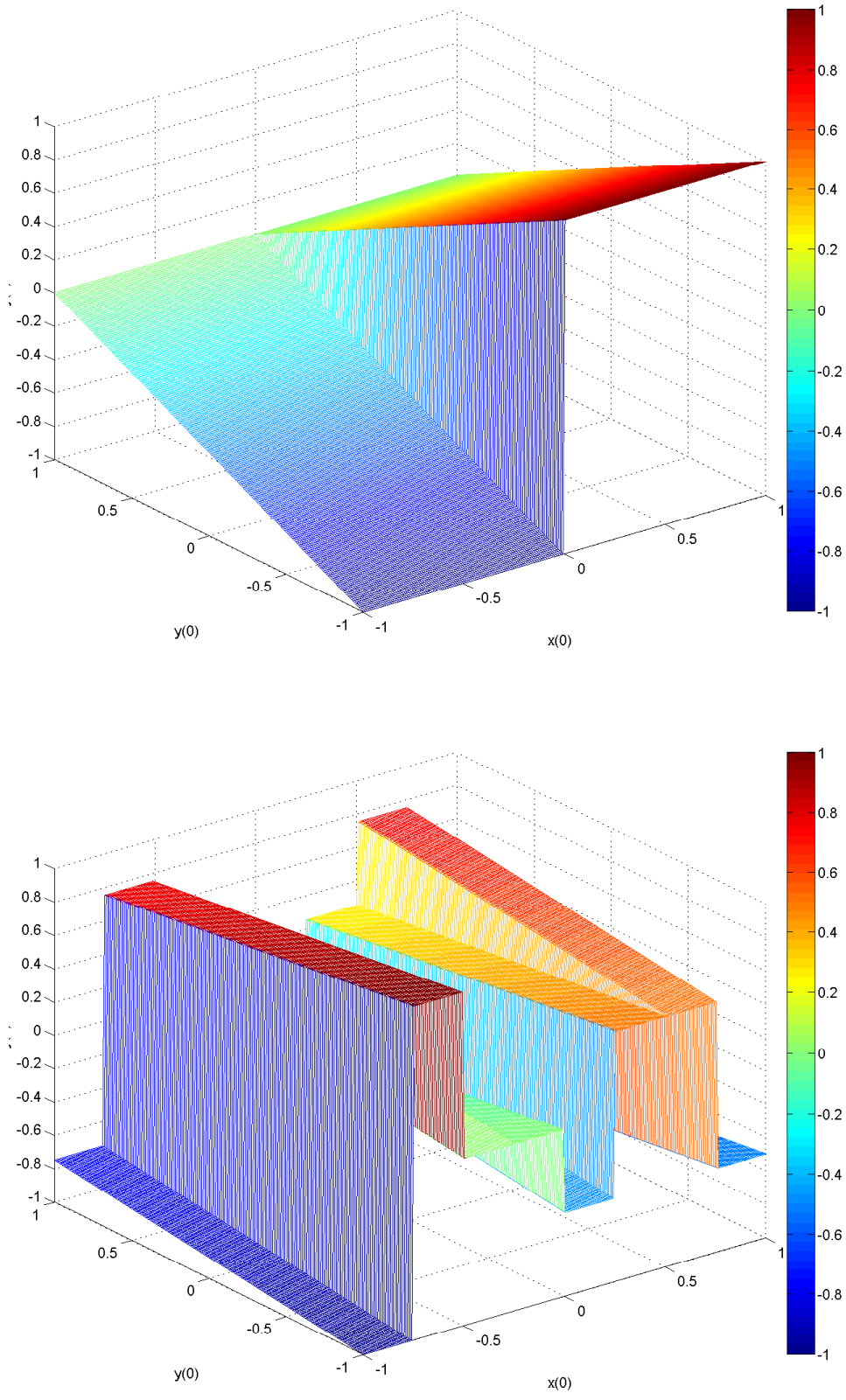


Figure 5.12: Function curve of $y[1]$ and $y[3]$ in terms of $\{u, v\}$

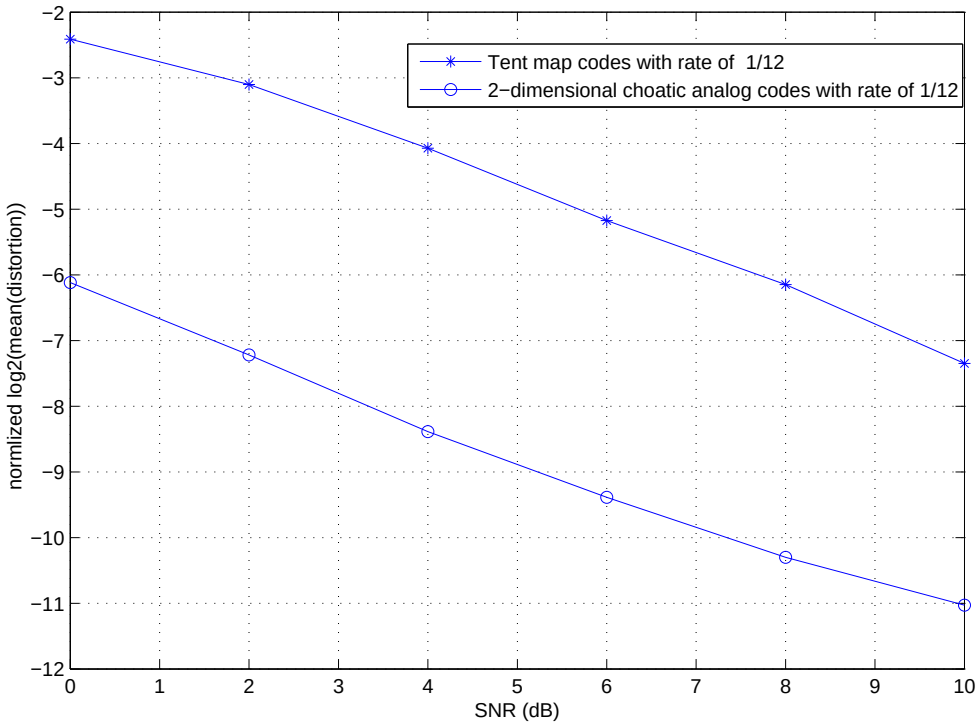


Figure 5.13: Performance comparison between baker's codes and tent map code with rate of 1/12.

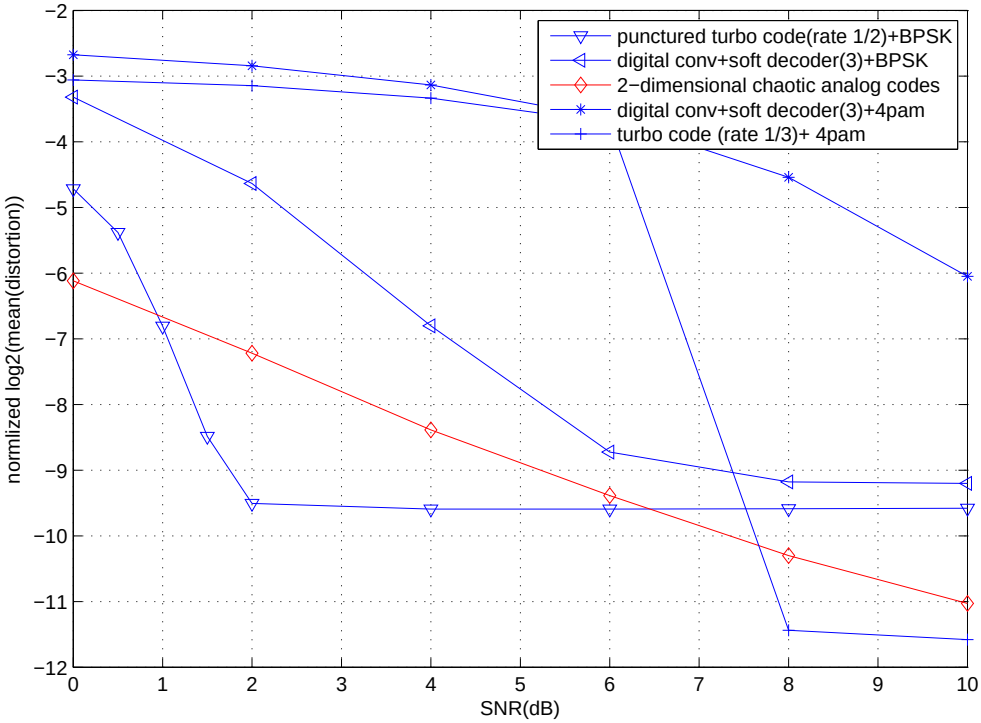


Figure 5.14: Performance comparison between analog codes and digital codes .

Chapter 6

Conclusion

Error correction coding (ECC) is widely used in wireless and wireline communications, computer architectures, and storage systems. The simple profound idea behind error correction coding is distance expansion, which could be exploited both in the digital domain and in the analog domain.

This research contains investigation in digital error correction coding and analog error correction coding. For the digital case, we study interleaver design for turbo codes and analyze iterative decoding of LDPC codes. For the analog case, we investigate the design and analysis of analog codes.

Interleavers in turbo codes assist encoders to obtain a good distance spectrum, and allow iterative decoders approximate the maximal likelihood (ML) decoder. Spread and randomness have been recognized as important aspects in interleaver design, but rigorous and efficient ways to measure these two aspects are much missing. Our study has resulted in two simple and powerful metrics to quantify

these properties. We further re-evaluate the coprime interlever (a rich class of algebraic interleavers) and provide a design guidance. Simulation results show that the new designed coprime interleaver outperforms most of the existing interleavers with a lower complexity.

In analyzing iterative decoding of LDPC codes, we investigated the Gaussian assumption. Gaussian distribution has been a popular assumption in the analysis of iterative decoding, but the justification of this assumption is largely pragmatic, except for the messages directly coming from Gaussian channels. Our research has provided a statistical justification for the Gaussian assumption in LDPC codes. We investigate when and how well the Gaussian distribution approximates the real message density and why. We have shown that the Gaussian assumption is statistically sound when the LLRs extracted from the channel are reasonably reliable to start with, and when the check node degrees of the LDPC code are not very high; but the assumption is much less accurate when one or both conditions are violated. Extensive simulation results are provided to exemplify and verify this discussion.

While distance expansion is largely performed in digital codes, we show that analog codes can potentially be a very promising extension for digital codes. Analog coding avoids quantization errors for natural analog sources (video, audio). In our research, we conducted fundamental theoretical and algorithmic study. We define a few analytical metrics and theorems, prove theoretical bounds for linear analog codes, identify classes of linear analog codes that could achieve the bounds and outline design rules for good linear analog codes. As linear analog codes do not perform well on AWGN channels, we further study nonlinear analog codes.

We cleverly exploit the butterfly effect of chaotic systems and the successful concatenation idea from digital turbo codes, and propose two new classes of chaotic analog codes: chaotic analog turbo codes and mirrored baker's map codes. For the former, we developed a SISO iterative decoder, and for the later, we developed a simple closed-form ML decoder. Both codes perform significantly better than the previous chaotic analog codes (i.e. tent map codes), and on par with the uniformly quantized digital codes. Finally we show that it is also possible to explore suitable non-chaotic nonlinear functions to construct good nonlinear analog codes.

Bibliography

[1] C. Shannon, N. Petigara, and S. Seshasai, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.

[2] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity check codes," *IEEE Transactions on Information Theory*, vol. 47, pp. 599–618, 2001.

[3] N. Wiberg and N. Wiberg, "Codes and decoding on general graphs," Ph.D. dissertation, Linkoing University Linkoing Sweden, 1996.

[4] 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 Transactions on Information Theory*, vol. 47, no. 2, pp. 657–670, 2001.

[5] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 47, pp. 619–637, 2001.

[6] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Transactions on Communications*, vol. 49, pp. 1727–1737, 2001.

[7] M. Fu, "Stochastic analysis of turbo decoding," *IEEE Transactions on Information Theory*, vol. 51, pp. 81–100, 2005.

[8] V. S. S. Nair and J. A. Abraham, "Real-number codes for fault-tolerant matrix operations on processor," *IEEE Transactions on Computers*, April 1990.

[9] G. Rath and C. Guillemot, "Characterization of a class of error correcting frames for robust signal transmission over wireless communication channels," *EURASIP Journal on Applied Signal Processing*, pp. 229–241, 2005.

[10] A. Gabay, M. Kieffer, and P. Duhamel, "Joint source-channel coding using real bch codes for robust image transmission," *IEEE Transactions on Image Processing*, pp. 1568–1583, 2007.

[11] B. Chen and G. W. Wornell, "Analog error-correcting codes based on chaotic dynamical systems," *IEEE Transactions on Communications*, vol. 46, pp. 881–890, 1998.

[12] V. A. Vaishampayan and I. R. Costa, "Curves on a sphere, shift-map dynamics and error control for continuous alphabet sources," *IEEE Transactions on Information Theory*, vol. 49, pp. 1658–1672, 2003.

[13] E. N. Lorenz, "Deterministic nonperiodic flow," *Journal of Atmospheric Sciences*, pp. 130–148, 1963.

[14] C. Pantalen, C. Pantaleon, D. Luengo, and I. Santamaria, "Optimal estimation of chaotic signals generated by piecewise-linear maps," *IEEE Signal Processing Letters*, vol. 7, pp. 235–237, 2000.

[15] C. Pantalen, L. Vielva, D. Luengo, and I. Santamaria, "Estimation of a certain class of chaotic signals: An em-based approach," 2007. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=?doi=10.1.1.15.3073

[16] C. Pantaleon, D. Luengo, and I. Santamaria, "Bayesian estimation of a class of chaotic signals," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2000, pp. 197–200.

[17] S. M. Kay, "Methods for chaotic signal estimation," *IEEE Transactions on Signal Processing*, vol. 43, pp. 2013–2016, 1995.

[18] I. Campos-Canton, E. Campos-Canton, J. S. Murguia, and H. C. Rosu, "A simple circuit realization of the tent map," *Chaos, Solitons & Fractals*, vol. 42, pp. 12–16, 2009. [Online]. Available: http://www.citeulike.org/user/uma_physics/article/3036559

[19] S. A. Barbulescu, A. Guidi, and S. S. Pietrobon, "Chaotic turbo codes," in *IEEE International Symposium on Information Theory*, 2000, p. 123.

[20] X. lin Zhou, J. bo Liu, W. tao Song, and H. wen Luo, "Chaotic turbo codes in secure communication," in *European Conference (EUROCON) International Conference on Trends in Communications*, 2001, pp. 199–201.

[21] K. Xie, P. Tan, N. B. Chong, and J. L. (Tiffany), "Analog turbo codes: A chaotic construction," *IEEE International Symposium on Information Theory*, 2009.

[22] S. N. Crozier, "New high-spread high-distance interleavers for turbo codes," in *Proceeding of Biennial Symposium of Communications*, May 2000.

[23] S. Dolinar and D. Divsalar, "Weight distributions for turbo codes using random and nonrandom permutations," in *JPL TDA Progress Report 42-122*, 1995, pp. 56–65.

[24] J. Li, "Low-complexity, capacity-approaching coding schemes: design, analysis and applications," Ph.D. dissertation, Texas A&M University, 2002.

[25] S. Crozier, J. Lodge, P. Guinand, and A. Hunt, "Performances of turbo-codes with relative prime and golden interleaving strategies," in *Proceedings of 6th International Mobile Satellite Conference*, June 1999.

[26] C. Heegard and S. B. Wicker, *Turbo Coding*. Kluwer Academic Publishers Norwell, MA, USA, 1999.

[27] O. Y. Takeshita and D. J. Costello, "New deterministic interleaver designs for turbo codes," *IEEE Transactions on Information Theory*, vol. 46, no. 6, pp. 1988–2006, 2000.

[28] K. Xie, P. Tan, J. Li, and W. Wang, "Interleaver design for short-length turbo codes," in *Proceeding of 39th Conference on Information Sciences and Systems*, March 2005.

[29] K. Xie and J. Li, "Spread spectrum properties for interleavers," *submitted to IEEE Transactions on Information Theory*.

[30] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: model and erasure channel properties," *IEEE Transactions on Information Theory*, vol. 50, pp. 2657–2673, 2004.

[31] C. Measson, A. Montanari, and R. Urbanke, "Why we cannot surpass capacity: The matching condition," in *Proceeding of Allerton Conference on Communication, Control and computing*, September 2005.

[32] L. F. Fenton, "The sum of log-normal probability distributions in scatter transmission systems," *IEE Transaction on Communication System*, vol. CS-8, pp. 57–67, 1960.

[33] S. Schwartz and Y. S. Yeh, "On the distribution function and moments of power sums with log-normal components," *Bell System Technical Journal*, vol. 6, pp. 1441–1462, 1982.

[34] C.-L. Ho, "Calculating the mean and variance of power sums with two log-normal components," *IEEE Transactions on Vehicular Technology*, vol. 44, pp. 756–762, 1995.

[35] N. C. Beaulieu, A. A. Abu-Dayya, and P. J. McLane, "Estimating the distribution of a sum of independent lognormal random variables," *IEEE Transactions on Vehicular Technology*, vol. 43, pp. 2869–2873, 1995.

[36] A. A. Abu-Dayya and N. C. Beaulieu, "Outage probabilities in the presence of correlated lognormal interferers," *IEEE Transactions on Vehicular Technology*, vol. 43, pp. 164–173, 1994.

[37] P. Pirinen, "Statistical power sum analysis for nonidentically distributed correlated lognormal signals," in *The 2003 Finnish Signal Processing Symposium*, May 2003.

[38] M. Pratesi, F. Santiccu, and F.Graziosi, "Generalized moment matching for the linear combination of lognormal RVs: application to outage analysis in

wireless systems," *IEEE Transactions on Wireless Communications*, vol. 5, pp. 1122–1132, 2006.

[39] S. Vanduffel, T. Hoedemakers, and J. Dhaene, "Comparing approximations for risk measures of sums of non-independent lognormal random variables," *North American Actuarial Journal*, vol. 9, pp. 71–82, 2005.

[40] N. C. Beaulieu and Q. Xie, "An optimal lognormal approximation to lognormal sum distributions," *IEEE Transactions on Vehicular Technology*, vol. 53, pp. 479–489, 2004.

[41] D. C. Boes, F. A. Graybill, and A. M. Mood, *Introduction to the Theory of Statistics*. McGraw-Hill New York, 1974.

[42] A. Ashikhmin, G. Kramer, and S. ten Brink, "Design of low-density parity-check codes for modulation and detection," *IEEE Transactions on Communications*, vol. 52, pp. 670–678, 2004.

[43] J. T. G. Marshall, "Real number transform and convolutional codes," *Proceedings of 24th Midwest Symposium on Circuits and Systems*, pp. 650–653, June 1981.

[44] J. K. Wolf, "Analog codes," *IEEE International Conference on Communications, Conference Record*, pp. 310–312, June 1983.

[45] ——, "Redundancy, the discrete fourier transform, and impulse noise cancellation," *IEEE Transactions on Communications*, pp. 458–461, March 1983.

[46] J. T. G. Marshall, "Coding of real-number sequenc es for error correction: A digital signal processing problem," *IEEE Journal on Selected Areas in Communications*, pp. 381–391, March 1984.

[47] J.-L. Wu and J. Shiu, "Discrete cosine transform in error control coding," *IEEE Transactions on Communications*, pp. 1857–1861, May 1995.

[48] A. A. Kumar and A. Makur, "Improved coding-theoretic and subspace-based decoding algorithms for a wider class of dct and dst codes," *IEEE Transactions on Signal Processing*, pp. 695–708, February 2010.

[49] G. R. Redinbo, "Decoding real-number convolutional codes: Change detection, kalman estimation," *IEEE Transactions on Information Theory*, pp. 1864–1876, November 1997.

[50] W. Henkel, "Analog codes for peak-to-average ratio reduction," *Proceedings of 3rd ITG Conference on Source and Channel Coding*, 2000.

[51] Z. Wang and G. Giannakis, "Complex-field coding for ofdm over fading wireless channels," *IEEE Transactions on Information Theory*, pp. 707–720, March 2003.

[52] F. Hu and W. Henkel, "Turbo-like iterative least-squares decoding of analogue codes," *Electronics Letters*, pp. 1233–1234, October 2005.

[53] ——, "An analysis of the sum-product decoding of analog compound codes," *International Symposium on Information Theory*, pp. 24–29, June 2007.

[54] G. R. Redinbo, "Decoding real block codes: Activity detection, wiener estimation," *IEEE Transactions on Information Theory*, pp. 609–623, March 2000.

[55] P. Azmi and F. Marvasti, "Robust decoding of dft-based error-control codes for impulsive and additive white gaussian noise channels communications," *IEE Proceedings Communications*, pp. 265–271, June 2005.

[56] G. Takos and C. N. Hadjicostis, "Determination of the number of errors in dft codes subject to low-level quantization noise," *IEEE Transactions on Signal Processing*, March 2008.

[57] W. Henkel, "Multiple error correction with analog codes," *Lecture Notes in Computer Science 357, Springer*, pp. 239–249, 1989.

[58] H. Poincare, "Les methodes nouvelles dela mechanique celeste," *Gauteheir-Villars*, 1892, in ENGLISH NASA Translation TTF-450/452 U.S. Federal Clearinghouse Springfield VA 1967.

[59] L. M. Pecora and T. L. Carroll, "Synchronization in chaotic systems," *Physics Review Letter*, vol. 64, no. 8, pp. 821–824, 1990.

[60] L. Kocarev, K. S. Halle, K. Eckert, L. O. Chua, and U. Parlitz, "Experimental demonstration of secure communications via chaotic synchronization," *International Journal of Bifurcation and Chaos*, pp. 709–713, 1992.

[61] U. Parlitz, L. O. Chua, L. Kocarev, K. S. Halle, and A. Shang, "Transmission of digital signals by chaotic synchronization," *International Journal of Bifurcation and Chaos*, pp. 973–977, 1992.

[62] G. Kolumban, B. Vizvari, W. Schwarz, and A. Abel, "Differential chaos shift keying: a robust coding for chaos communication," *Proceedings of the NDES*, pp. 87–92, 1996.

[63] M. Sushchik, N. Rulkov, L. Larson, L. Tsimring, H. Abarbanel, K. Yao, and A. Volkovskii, "Chaotic pulse position modulation: a robust method of communicating with chaos," *IEEE Communication Letters*, pp. 128–130, 2000.

[64] C. W. Wu and L. O. Chua, "A simple way to synchronize chaotic systems with applications to secure communication systems," *International Journal of Bifurcation and Chaos*, pp. 1619–1627, 1993.

[65] T. Yang and L. O. Chua, "Secure communication via chaotic parameter modulation," *IEEE Transactions on Circuits and Systems*, pp. 817 – 819, 1996.

[66] K. S. Halle, C. W. Wu, M. Itoh, and L. O. Chua, "Spread spectrum communication through modulation of chaos," *International Journal of Bifurcation and Chaos*, pp. 469–477, 1993.

[67] S. Hayes, C. Grebogi, and E. Ott, "Communicating with chaos," *Physics Review Letters*, pp. 3031–3034, 1993.

[68] G. Mazzini, G. Setti, and R. Rovatti, "Chaotic complex spreading sequences for asynchronous ds-cdma - part i: System modeling and results," *IEEE Transactions on Circuits and Systems*, pp. 515–516, 1998.

[69] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, pp. 300–304, 1960.

[70] R. G. Gallager, "Low density parity-check codes," Ph.D. dissertation, MIT press Cambridge MA, 1963.

[71] E. N. Lorenz, "Predictability: Does the flap of a butterfly's wings in brazil set off a tornado in texas?" *Washington Posts*, 1972.

[72] N. Santhi and A. Vardy, "Analog codes on graphs." [Online]. Available: arXiv:cs/060808vc1

[73] H. C. Papadopoulos and G. W. Wornell, "Maximum likelihood estimation of a class of chaotic signals," *IEEE Transactions on Information Theory*, vol. 41, pp. 312–317, 1995.

[74] K. Xie, W. Wang, and J. Li, "On the analysis and design of good algebraic interleavers," in *Proceeding of 4th International Symposium on Turbo Codes and Related Topics*, April 2006.

[75] J. Hokfelt, "On the design of turbo codes," Ph.D. dissertation, Lund University, 2000.

[76] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding," in *Proceedings of IEEE International Conference on Communications*, 1993, pp. 1064–1070.

[77] D. J. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 32, pp. 1645–1646, 1996.

[78] R. G. Gallager, "Low-density parity-check codes," Ph.D. dissertation, Cambridge MA: MIT Press, 1963.

[79] H. Jin, A. Khandekar, A. Kh, and R. J. McEliece, "Irregular repeat accumulate codes," in *Proceedings of 2nd International Symposium on Turbo Codes and Related Topics*, 2000.

[80] J. Li, K. R. Narayanan, and C. N. Georghiades, "Product accumulate codes: A class of codes with near-capacity performance and low decoding complexity," *IEEE Transactions on Information Theory*, vol. 50, pp. 31–46, 2004.

[81] R. Koetter and A. Vardy, "Algebraic soft-decision decoding of reed-solomon codes," *IEEE Transactions on Information Theory*, vol. 49, pp. 2809–2825, 2001.

[82] K. Xie and J. Li, "On accuracy of gaussian approximation in ldpc analysis," in *Proceeding of IEEE International Symposium on Information Theory*, July 2006.

Vita

Kai Xie received a B.S. degree and M.S. degree in Electrical Engineering from Tianjin University, Tianjin, China. He was admitted to Lehigh University in Fall 2005, and joined Prof. Tiffany Jing Li's group.

Kai's research interests cover diversified topics in wireless communications and signal processing. Specifically, he researched emerging topics in the theory and practice of advanced error correction coding (ECC) technologies, including analysis and design of turbo and low density parity check (LDPC) codes, interleaver design of turbo codes. Kai also contributed to novel topics and explores capabilities of ECC beyond their conventional applications, ranging from equalization and coding in powerline communications, to chaotic analog coding, and to rateless codes for network coding. Kai has published many articles in international journals and conference preceedings, and he co-authored a book chapter for powerline communications. He is also a recipient of the Rossin Doctoral Fellowship. He also cooperated with industrial projects, he proposed algorithms for base-station cooperation in LTE-advance systems during his internship at Infineon, and he is now researching and designing front-end signal processing algorithms in WCDMA/HSxPA systems at Alcate-Lucent.