

# **RANDOMIZED CONVOLUTIONAL AND CONCATENATED CODES FOR THE WIRETAP CHANNEL**

A THESIS SUBMITTED TO  
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE  
OF BILKENT UNIVERSITY  
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR  
THE DEGREE OF  
MASTER OF SCIENCE  
IN  
ELECTRICAL AND ELECTRONICS ENGINEERING

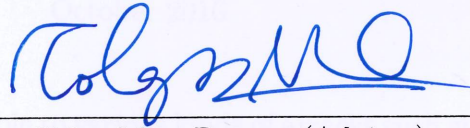
By  
Alireza Nooraiepour  
October 2016

Randomized convolutional and concatenated codes for the wiretap channel

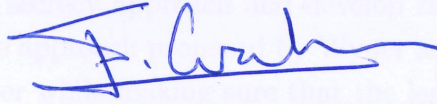
By Alireza Nooraiepour

October 2016

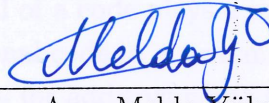
We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.



Tolga Mete Duman (Advisor)

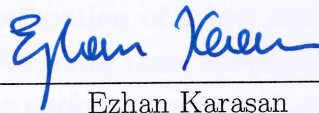


Erdal Arıkan



Ayşe Melda Yüksel Turgut

Approved for the Graduate School of Engineering and Science:



Ezhan Karahan  
Director of the Graduate School

# ABSTRACT

## RANDOMIZED CONVOLUTIONAL AND CONCATENATED CODES FOR THE WIRETAP CHANNEL

Alireza Nooraiepour

M.S. in ELECTRICAL AND ELECTRONICS ENGINEERING

Advisor: Tolga Mete Duman

October 2016

Wireless networks are vulnerable to various kinds of attacks such as eavesdropping because of their open nature. As a result, security is one of the most important challenges that needs to be addressed for such networks. To address this issue, we utilize information theoretic secrecy approach and develop randomized channel coding techniques akin to the approach proposed by Wyner as a general method for confusing the eavesdropper while making sure that the legitimate receiver is able to recover the transmitted message.

We first study the application of convolutional codes to the randomized encoding scheme. We argue how dual of a code plays a major role in this construction and obtain dual of a convolutional code in a systematic manner. We propose optimal and sub-optimal decoders for additive white Gaussian noise (AWGN) and binary symmetric channels and obtain bounds on the decoder performance extending the existing lower and upper bounds on the error rates of coded systems with maximum likelihood (ML) decoding. Furthermore, we apply list decoding to improve the performance of the sub-optimal decoders. We demonstrate via several examples that security gaps achieved by the randomized convolutional codes compete favorably with some of the existing coding methods.

In order to improve the security gap hence the system performance further, we develop concatenated coding approaches applied to the randomized encoding scheme as well. These include serial and parallel concatenated convolutional codes and serial concatenation of a low density generator matrix code with a convolutional code. For all of these solutions low-complexity iterative decoders are proposed and their performance in the wiretap channel is evaluated in terms of the security gap. Numerical examples show that for certain levels of confusion

at the eavesdropper, randomized serially concatenated convolutional codes offer the best performance.

*Keywords:* Randomized codes, wiretap channel, security gap, physical layer security, convolutional codes, turbo codes, low density generator matrix codes.

# ÖZET

## HAT DİNLEMELİ KANALLAR İÇİN RASGELELEŞTİRİLMİŞ KIVRIMLI VE UÇ UÇA EKLEMELİ KODLAR

Alireza Nooraiepour  
Elektrik ve Elektronik Mühendisliği, Yüksek Lisans  
Tez Danışmanı: Tolga Mete Duman  
Ekim 2016

Kablosuz ağlar dışı açık olan yapıları ile gizli dinleme gibi saldırılara maruz kalmaktadır. Dolayısı ile, güvenlik bu tip ağlarda üzerinde durulması gereken en zorlu işlerden biridir. Bu konuyu incelemek için, biz bilgi kuramsal yaklaşımları kullandık ve rasgeleleştirilmiş kodlama yöntemleri geliştirdik. Geliştirdiğimiz yöntemler Wyner'in yaklaşımlarına benzemektedir ve esas alıcıda mesajın doğru bir şekilde alınması koşulu ile gizli dinleyicinin kafasının karıştırılması esasına dayanmaktadır.

İlk olarak kıvrımlı kodları rasgeleleştirilmiş kodlama şeması üzerine uyguladık. Kodun ikili karşılığının bu uygulamadaki kritik önemine değindik ve kıvrımlı kodun ikili karşılığını sistematik bir biçimde elde ettik. Toplanır beyaz Gauss gürültüsü (AWGN) kanalı ve ikili simetrik kanalı için standart ve standart altı çözümler önerdik. Bu çözümlerin performans sınırlarını ise en büyük olabilirlik (ML) çözümlerinin kodlanmış sistemlerdeki mevcut alt ve üst sınırlarını genişleterek elde ettik. Ayrıca, listelemeli bir çözümler ile standart altı çözümlerinin performansını iyileştirdik. Verdiğimiz bir çok örnek ile gösterdik ki rasgeleleştirilmiş kıvrımlı kodlar sonucu giderilen güvenlik açığı mevcut kodlama yöntemleriyle kıyas edilebilecek düzeydedir.

Güvenlik açığını kapatmak yani sistem performansını daha ileriye taşımak için, uç uca eklenmiş kodlama yaklaşımını rasgeleleştirilmiş kodlama şeması üzerine uyguladık. Bu yaklaşımlar seri ve paralel uç uca eklenmiş kıvrımlı kodları ve kıvrımlı kodlu düşük yoğunluklu üretici matrisin seri olarak uç uca eklenmesini kapsamaktadır. Tüm bu çözümler için, düşük karmaşıklığa sahip yinelemeli

özücüler önerdik ve hat dinlemeli kanallardaki performanslarını güvenlik seviyeleri açısından deęerlendirdik. Elde ettięimiz nümerik sonuçlar göstermektedir ki rasgeleleřtirilmiř ve seri olarak uç uęa eklenmiř kıvrımlı kodlar en iyi performansı göstermektedir.

*Anahtar sözcükler:* Rasgeleleřtirilmiř kodlar, hat dinleme kanalı, güvenlik açığı, fiziksel katmanda güvenlik, kıvrımlı kodlar, turbo kodlar, düşük yoğunluklu üreteę matrisi kodları.

## Acknowledgement

I would like to express my deepest gratitude to my supervisor, Prof. Tolga M. Duman for his great support and guidance throughout the course of this thesis. I appreciate his ideas and comments from which I have learned a lot in the past two years. I also would like to thank Prof. Erdal Arıkan and Prof. Melda Yüksel for accepting to serve in my defense committee.

This work was supported by the Scientific and Technical Research Council of Turkey (TUBITAK) under the grant 113E223 and I gratefully acknowledge this support from TUBITAK.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Physical Layer Security . . . . .	3
1.2	Contributions of the Thesis . . . . .	8
<b>2</b>	<b>Randomized Convolutional Codes for the Wiretap Channel</b>	<b>10</b>
2.1	Introduction . . . . .	11
2.2	Channel Model . . . . .	13
2.3	Randomized Convolutional Codes – Encoding . . . . .	13
2.3.1	Randomized Encoding Method . . . . .	13
2.3.2	Dual of a Convolutional Code . . . . .	16
2.3.3	Obtaining a Subset of Convolutional Codes . . . . .	18
2.3.4	Convolutional Code Design for the Randomized Encoding Scheme . . . . .	20
2.4	Decoding Methods . . . . .	22



2.4.1	Optimal Decoder . . . . .	22
2.4.2	Sub-Optimal Decoders . . . . .	24
2.5	Performance Bounds . . . . .	28
2.5.1	Assumptions . . . . .	28
2.5.2	Performance Lower Bounds . . . . .	30
2.5.3	Performance Upper Bounds . . . . .	31
2.5.4	A Simple Example . . . . .	32
2.6	Numerical Examples . . . . .	32
2.6.1	Noiseless Main Channel . . . . .	33
2.6.2	Noisy Main Channel . . . . .	36
2.6.3	Application of List Decoding in the Randomized Encoding Scheme . . . . .	39
2.7	Chapter Summary . . . . .	41
<b>3</b>	<b>Concatenated Codes for the Wiretap Channel</b>	<b>42</b>
3.1	Review of Parallel and Serial Concatenated Convolutional Codes .	43
3.1.1	Encoding . . . . .	43
3.1.2	Decoding . . . . .	45
3.2	Concatenation of LDGM and Convolutional Codes . . . . .	50
3.2.1	LDGM-RSC Concatenated Codes . . . . .	53

3.3	Code Concatenation for the Randomized Encoding Scheme . . . .	54
3.3.1	Randomized PCCCs . . . . .	55
3.3.2	Randomized SCCCs . . . . .	60
3.3.3	Randomized LDGM-RSC codes . . . . .	64
3.4	Chapter Summary . . . . .	70
<b>4</b>	<b>Conclusions and Future Work</b>	<b>71</b>

# List of Figures

1.1	Illustration of cryptographic encryption and decryption with channel coding . . . . .	2
1.2	Wire-tap channel model. . . . .	4
1.3	Achievable region $\mathcal{R}$ . . . . .	6
2.1	Illustration of the randomized encoding scheme where $\mathbf{c}_{ij}$ denotes $j$ th codeword in $i$ th coset and $\mathbf{s}_i$ 's correspond to the all possible non-zero messages of length $k$ . $\mathbf{H}$ is a matrix whose rows are $\mathbf{h}_1$ through $\mathbf{h}_k$ introduced in (2.2). . . . .	14
2.2	Performance of the optimal decoder introduced in (2.17) over an AWGN channel using a Reed-Muller code of length 16 to encode the messages. The number of cosets or messages is $2^5$ each of which containing $2^{11}$ codewords ( $n = 16, r = 11, k = 5$ ). The lower and upper bounds are developed in Section 2.5. . . . .	23
2.3	Performance of the optimal decoder introduced in (2.19) over BSC channel using a Reed-Muller code of length 16 to encode the messages. The number of cosets or messages is $2^5$ each of which contains $2^{11}$ codewords ( $n = 16, r = 11, k = 5$ ). The lower and upper bounds are discussed in Section 2.5. . . . .	24

2.4	The overall encoder for the randomized encoding scheme when a $[7\ 5]$ convolutional code encodes random bits and a $[5\ 7]$ convolutional code (which is the dual of $[7\ 5]$ ) encodes data bits. . . . .	27
2.5	Performance of the sub-optimal decoders introduced in Section 2.4.2 and the bounds in Section 2.5 when a $[7\ 5]$ convolutional code with its dual $[5\ 7]$ has been used. The length of the codewords is 204. There are $2^{100}$ cosets each of which containing $2^{100}$ codewords. . . . .	34
2.6	Performance of the minimum Hamming distance decoder (based on trellis) and the bounds introduced in Section 2.5 over BSC when a $[7\ 5]$ convolutional code with its dual $[5\ 7]$ has been used. The length of the codewords is 104. There are $2^{50}$ cosets each of which containing $2^{50}$ codewords. . . . .	35
2.7	Effect of increasing memory size on the performance of the sub-optimal decoders using convolutional codes $[117\ 155]$ and $[133\ 171]$ with memory size $m = 6$ . The length of the codewords is 212 and there are $2^{100}$ cosets each of which represents a unique message and contains $2^{100}$ codewords. . . . .	36
2.8	Bit error probability for 3 convolutional codes with different memory sizes ( $m$ ) and an LDGM code. The number of cosets is $2^{100}$ each contains $2^{100}$ codewords ( $k = 100, r = 100$ ). Length of the LDGM code is 200 and length of convolutional codes are $200 + 2m$ . . . . .	37
2.9	Bit error probability of the eavesdropper versus the security gap (at $P_{main}^{max} \approx 10^{-5}$ ) when convolutional code $[657\ 435]$ encodes data bits for 3 different codeword lengths and two different random bit encoders in (2.31) and (2.33). Numbers of data and random bits for each curve are provided in Table 2.1. . . . .	38

2.10	Performance of randomized encoding scheme along with scrambling. Perfect scrambling which is defined in [10] has been used here where $n = 256$ , $k = 120$ , $r = 27$ . . . . .	39
2.11	Performance of List Viterbi decoding Algorithm for the randomized encoding scheme over a binary symmetric channel. There are $2^{96}$ cosets each of which consisting of $2^{28}$ codewords of length 204. . . . .	40
3.1	The encoder for a parallel-concatenated convolutional code (PCCC). . . . .	44
3.2	The encoder for a serial-concatenated convolutional code (SCCC). . . . .	44
3.3	Iterative decoding for a PCCC. . . . .	49
3.4	Iterative decoding for a SCCC. . . . .	49
3.5	Performance of PCCC and SCCC in the AWGN channel where $n$ and $k$ denote length of the codewords and data bits, respectively. . . . .	50
3.6	The encoder for the serial concatenation of a LDGM code with an RSC code. . . . .	53
3.7	The iterative decoder for the serial concatenation of a LDGM code with an RSC code. . . . .	54
3.8	Performance of LDGM-RSC scheme in the AWGN channel where $n$ and $k$ denote length of the codewords and data bits, respectively. . . . .	55
3.9	The encoder for parallel-concatenated convolutional code (PCCC). . . . .	56
3.10	The encoder for dual of a PCCC. . . . .	57
3.11	Iterative decoder for the randomized encoding scheme where one of the encoders in Figures 3.9 and 3.10 encodes random bits and the other encodes data bits. . . . .	59

3.12 Performance of the randomized PCCC in the AWGN channel where $n$ denotes the length of the codewords and transmission rate is about $1/4$ . . . . .	60
3.13 Performance of the randomized PCCC in the binary symmetric channel where $n$ denotes the length of the codewords and transmission rate is about $1/4$ . . . . .	61
3.14 The encoder for serial-concatenated convolutional code (SCCC). . . . .	61
3.15 The encoder for dual of the SCCC in Figure 3.14. . . . .	62
3.16 Iterative decoder for the randomized encoding scheme where one of the encoders in Figures 3.14 and 3.15 encodes random bits and the other encodes data bits. . . . .	63
3.17 Performance of the randomized SCCC in the AWGN channel where $n$ denotes the length of the codewords and transmission rate is about $1/4$ . . . . .	64
3.18 Performance of the randomized SCCC in the binary symmetric channel where $n$ and $k$ denote the length of the codewords and data bits, respectively, which makes transmission rate about $1/4$ . . . . .	65
3.19 Achievable security gaps for PCCC and SCCC randomized schemes for different values of $P_{eve}^{min}$ where $P_{main}^{max} \approx 10^{-5}$ . . . . .	66
3.20 The encoder for the serial concatenation of a LDGM code with an RSC code. . . . .	66
3.21 The encoder for dual of the code in the Figure 3.20. . . . .	67
3.22 Iterative decoder for the randomized encoding scheme where one of the encoders in Figures 3.20 and 3.21 encodes random bits and the other encodes data bits. . . . .	67

3.23	Joint BP decoder on the factor graph of the 2-user MAC channel .	68
3.24	Performance of the randomized LDGM-RSC in the AWGN channel where $n$ denotes the length of the codewords. . . . .	69

# List of Tables

2.1	Important results of Figure 2.9 when $P_{main}^{max} \approx 10^{-5}$ and data bits encoder is $[657\ 435]$ . $k$ and $r$ denotes number of data and random bits, respectively, and $n$ is length of the codewords. . . . .	38
3.1	Three instances of QC-LDPC code with the structure given in (3.16).	53



# Chapter 1

## Introduction

Wireless communications has become an indispensable part of the modern life through its ubiquitous applications. Wireless networks have a broadcast nature which makes them vulnerable to the potential attackers because anybody within the coverage range of a transmitter can receive its signal. Therefore, providing secure communications is one of the most important problems for today's wireless networks.

Security issues arising in communication networks can be classified into four main areas: confidentiality, integrity, authentication, and non-repudiation. Confidentiality guarantees that legitimate recipients successfully obtain their intended messages while they are protected against eavesdropping. Integrity provides communicating parties with the assurance that a message is not modified during its transmission. Authentication ensures that a recipient of information is able to identify the sender. Non-repudiation ensures that parties involved in communication cannot deny their roles, i.e., transmitting a signal or receiving it.

There are two types of attackers in wireless networks in general: passive attackers and active attackers. An active attacker intentionally disrupts the system while a passive attacker tries to interpret the signal he/she receives without any effort to modify the source of the signal, i.e., the attacker listens to the signal

but does not modify it. In this work, we mainly focus on techniques which are designed to combat passive attackers.

Figure 1.1 depicts a cryptographic encryption scheme [1] which is the basic method for conventional techniques for achieving confidentiality in communication networks. The transmitter (Alice) uses a key to encrypt the information (which is referred to as plaintext) to convert it into ciphertext. The legitimate receiver (Bob) can extract the original *plaintext* from the *ciphertext* by the corresponding key. Assume that an eavesdropper (Eve) has access to the ciphertext without any knowledge about the corresponding decryption key. Then, in practice, where Eve has limited time and can't test all possible keys, it cannot obtain the source information. Figure 1.1 illustrates this process along with the encoding and decoding steps which are aimed to combat the channel transmission errors.

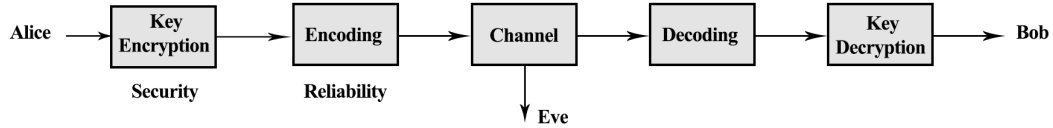


Figure 1.1: Illustration of cryptographic encryption and decryption with channel coding

There are two types of algorithms for encryption: *secret-key encryption* and *public-key encryption*. In secret-key encryption, the transmitter encrypts the plaintext and the legitimate receiver decrypts the ciphertext using the same key. On the other hand, in public-key encryption, the transmitter and receiver use different keys for encryption and decryption. Specifically, the transmitter encrypts the information with a public key which is known to all the receivers and eavesdroppers. Then, a legitimate receiver uses a private key corresponding to that public key for decryption. It is practically impossible for the eavesdroppers who do not have a private key to obtain the plaintext.

There are advantages and disadvantages associated with each of the algorithms mentioned above. Public-key algorithms make key management simple but require extensive computational resources and they are not completely secure against all kinds of attacks. Secret-key algorithms are computationally efficient

although key management is a major challenge. Hence, in practice hybrid cryptosystems [2] are employed which enables distribution of secret keys by public-key algorithms. However, the lack of infrastructure in networks makes key distribution difficult and dynamic topology of networks makes the key management expensive.

Although these methods provide a reliable way for achieving security, they lack the theoretical justification for achieving perfectly secure communications. The notion of perfect secrecy introduced in physical layer security which has emerged as a promising way to address security issues in wireless communications and other applications.

## 1.1 Physical Layer Security

Physical Layer Security is a promising way to provide secure communications without the aforementioned issues associated with the cryptography. This approach was initiated by Wyner [3] and by Csiszar and Korner [4] who proved that confidential messages can be transmitted securely without the need for using an encryption key. Cryptographic methods rely on practical mathematical difficulties in decryption to achieve security, however, information theoretic approach puts “perfect” secrecy as the ultimate goal, where the attacker will not be able to extract any information from what it receives, not because the plaintext is very powerfully encrypted, but because the received message is too noisy to understand. Furthermore, information theoretic approach does not give any prior information to the parties involved in the communication and merely relies on the differences between the channels of the legitimate receivers and attackers (i.e., randomness of communication channels).

Shannon [5] proved that a plaintext message  $M$  can be sent with perfect secrecy by transmitting a ciphertext  $c = M \oplus K$  where  $K$  is a random key and  $\oplus$  denotes the mod-2 addition. He defined the notion of *perfect secrecy* to mean that  $c$  gives no additional information about the original message  $M$ , i.e.,  $H(M) = H(M|c)$

or  $I(M; c) = 0$ , where  $H(M)$  is the entropy of the plaintext and  $H(M|c)$  is the conditional entropy of the plain-text given the cipher-text  $c$ , and  $I(M; c)$  is the mutual information between the plaintext and ciphertext. Shannon referred to such keys as pads and showed that in order to keep the message secure,  $H(K)$  must be bigger than  $H(M)$  and the key must be used only once (one-time pad).

In [3], Wyner introduced the famous wire-tap channel shown in Figure 1.2 where there is a transmitter, a legitimate receiver and an eavesdropper who tries to obtain information about the message signal.

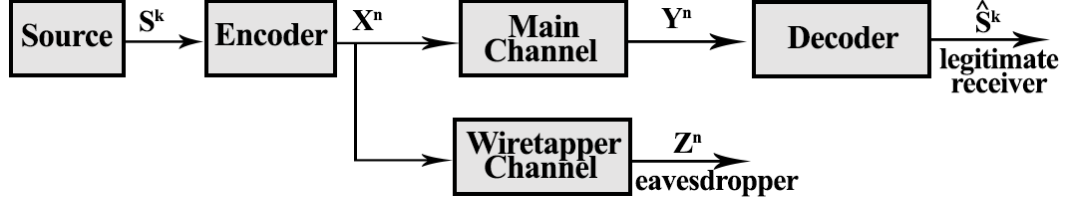


Figure 1.2: Wire-tap channel model.

Wyner defined a quantity called equivocation (denoted by  $\Delta$ ) in order to measure secrecy of the transmission as

$$\Delta = \frac{1}{K} H(\mathbf{S}^K | \mathbf{Z}^n) \quad (1.1)$$

Large values of  $\Delta$  are desirable since this implies that there is more confusion at the eavesdropper. Wyner defined the notion of perfect secrecy as  $\Delta = \frac{1}{K} H(\mathbf{S}^K)$  and proved that for discrete memoryless channels (DMCs), it can be achieved if the wiretapper channel is degraded with respect to the main channel. We note that the perfect secrecy condition introduced by Wyner is known as weak secrecy stated as  $\frac{1}{K} I(M, Z) \rightarrow 0$  in the recent literature. It is pointed out in [3] that a rate-equivocation pair  $(R, d)$  is achievable if there exists an encoder and a decoder which satisfy

$$\begin{aligned} \frac{H_S K}{N} &\geq R - \epsilon \\ \Delta &\geq d - \epsilon \\ P_e &\leq \epsilon \end{aligned} \quad (1.2)$$

where  $H_S$  is entropy of the source,  $R$  is transmission rate and  $P_e$  denotes probability of error at the legitimate receiver computed as

$$P_e = \frac{1}{K} \sum_{k=1}^K \Pr\{\mathbf{S}^K \neq \hat{\mathbf{S}}^K\} \quad (1.3)$$

where  $\hat{\mathbf{S}}^K$  denotes the decoded signal. Wyner characterized the set  $\mathcal{R}$  of achievable  $(R, d)$  pairs for the case where wiretapper channel is degraded with respect to the main channel in the following way

$$\begin{aligned} 0 &\leq R \leq C_M \\ 0 &\leq d \leq H_S \\ Rd &\leq H_S \Gamma(R) \end{aligned} \quad (1.4)$$

where  $C_M = \sup_{p(x)} I(X; Y)$  is the main channel capacity and  $\Gamma(R)$  is defined as (where  $X \rightarrow Y \rightarrow Z$  form a Markov chain)

$$\Gamma(R) = C_M - C_{Wiretapper} \quad (1.5)$$

which measures the maximum information that can be shared between transmitter and the legitimate receiver without leaking any information to Eve at any given rate  $R$  where capacity of the wiretapper channel is denoted by  $C_{Wiretapper}$ . Using (1.4), the achievable region  $\mathcal{R}$  for the pairs  $(R, d)$  is illustrated in Figure 1.3. Wyner called the  $C_S$  in this figure, *secrecy capacity* which is the maximum transmission rate that satisfies perfect secrecy condition ( $\Delta = H_S$ ). Furthermore, for the case when wiretapper's channel is degraded with respect to the main channel, he proved that there exists  $C_S$  such that

$$0 \leq C_M - C_{wiretapper} \leq C_S \leq C_M \quad (1.6)$$

where  $C_M$  and  $C_{Wiretapper}$  denote the main and wiretapper channels capacities, respectively.

Extensive research has been carried out to generalize Wyner's results. Gaussian wire-tap channel was studied in [3] and Csiszar and Korner in [4] introduced broadcast channels with confidential messages generalizing the wire-tap channel model. They consider a sender who wants to transmit common information to

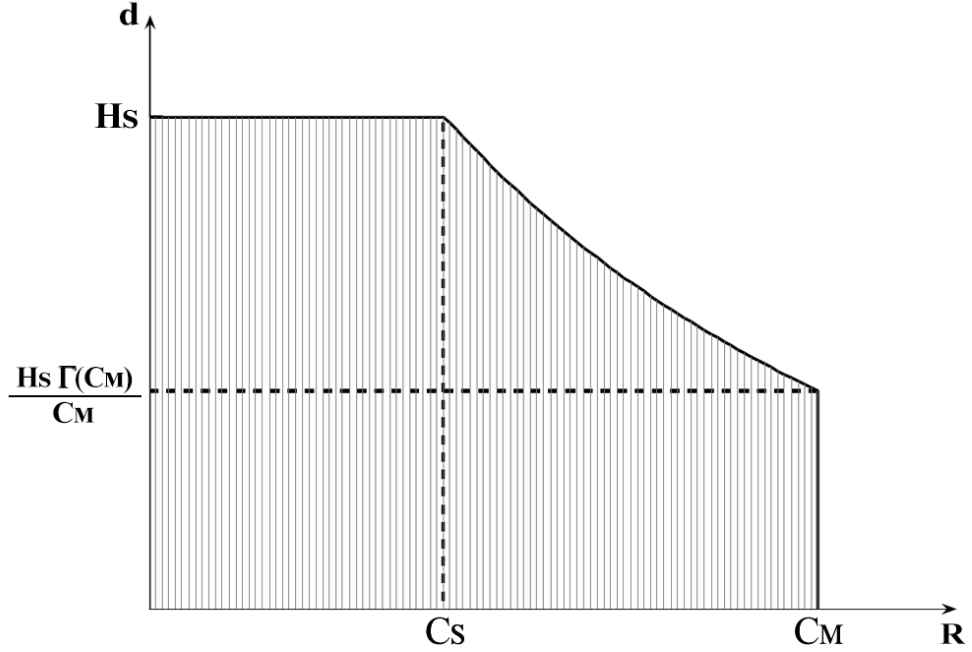


Figure 1.3: Achievable region  $\mathcal{R}$

both the legitimate receiver and the eavesdropper while a confidential message is also being transmitted to the legitimate receiver. Moreover, they provide a mathematical expression for calculating the secrecy capacity as follows

$$Cs = \max_{V \rightarrow X \rightarrow (Y, Z)} [I(V; Y) - I(V; Z)] \quad (1.7)$$

where  $V$  is an arbitrary random variable such that  $V \rightarrow X \rightarrow (Y, Z)$  is a Markov chain.

Vast majority of the research in this field focuses on information theoretic approaches to obtain achievable rate regions. In contrast, there are also a few works which propose constructive coding schemes for the wiretap channel. In fact, coding schemes exist for only a few special cases based on Wyner's basic approach in [3] which utilizes a randomized encoding scheme to achieve the secrecy capacity. This method is known as *coset-coding* where each secret message is mapped to a coset of code in a random fashion. Inspired by this method, application of low density parity check (LDPC) codes to the wiretap channel is studied in [6]. The authors prove that using capacity approaching codes for each secret message over the wiretapper channel can achieve the secrecy capacity asymptotically. More practically, when the main channel is noiseless and the wiretapper channel is a

binary erasure channel, they point out that using the dual of an LDPC code and its cosets can satisfy the security condition without the need for using capacity approaching codes. Application of lattice codes in the context of physical layer security is studied in [8] where the authors define a secrecy gain metric which was related to the theta series of lattices and show the amount of confusion at the eavesdropper. Without introducing a decoding method, they evaluate the performance of different lattices based on the secrecy gain. The confusion at the eavesdropper in [8] is the result of using a random lattice in addition to the lattice which is responsible for transmitting the original message. Application of polar codes to the randomized coding scheme is studied in [7] where the channel polarization phenomenon of polar codes enables the proposal of a practical coding scheme which achieves secrecy capacity when both main and wiretapper channels are binary symmetric.

It is also worth mentioning that randomized encoding scheme is not the only way of confusing the eavesdropper, for instance, in [9], the authors propose the use of punctured LDPC codes over the Gaussian wiretap channel where the secret messages are transmitted over punctured bits to hide data from eavesdroppers. As another transmission scheme, the authors in [10] propose to implement non-systematic coded transmission by scrambling the information bits, and characterize the bit error rate of scrambled transmissions through theoretical arguments and numerical simulations. In [11] a concatenated coding scheme based on polar codes and LDPC codes is proposed for the additive white Gaussian noise (AWGN) wiretap channel where the bit error rate (BER) performance is analyzed through density evolution. The common tread of these works is that they propose coding schemes which achieve given BER targets at Eve and Bob, while keeping the required quality difference between the main and the eavesdropper's channels (termed as security gap [9]) as small as possible.

In [12], authors link the classical information theoretic measure (equivocation) with the error rate based secrecy measures, where the main goal is to propose a secret key sharing scheme for the wiretap channel. Presence of an error-free public channel between the source and destination is considered to help the secret sharing process. The authors in [13] use an approximate version of equivocation rate as

the measure of secrecy at the eavesdropper and propose a code optimization algorithm which allows to design practical irregular LDPC codes which are able to approach the secure performance limits at moderate codeword lengths, i.e., in the order of  $10^4$  bits.

## 1.2 Contributions of the Thesis

We develop approaches of implementing Wyner's randomized encoding method with the common element in all of them being the presence of a convolutional code. First, we study how convolutional codes can be applied to the randomized encoding scheme. We develop the required mathematical background and argue that the concept of dual of a convolutional code plays a crucial role in both encoding and decoding schemes in this set-up. Using convolutional codes in the randomized encoding scheme enables us to propose effective and computationally inexpensive decoders whose objective is to choose the right coset. Moreover, we evaluate the performance of finite length (terminated) randomized convolutional codes over the Gaussian and binary symmetric wiretap channels. We note that achievable security gaps in our scheme using suitably designed generators for random and data bits, compete favorably with other techniques, e.g., the LDPC with puncturing.

We also consider the use of several concatenated codes in the randomized encoding scheme, including serial and parallel concatenated convolutional codes (turbo codes) and serial concatenation of a low density generator matrix (LDGM) with a recursive systematic convolutional (RSC) code. The iterative decoder for each scheme is proposed based on the turbo principle and the BCJR algorithm. Turbo codes have a very sharp slope in their bit error rate performance which makes them desirable for the wiretap channel in order to achieve very small security gaps. In fact, as it will be demonstrated with several examples, for certain levels of confusion at the eavesdropper, serially concatenated convolutional codes outperform the existing coding schemes proposed for the wiretap channel.



The thesis is organized as follows. In Chapter 2, we propose the use of convolutional codes in the randomized coding scheme and provide upper and lower bounds on their error rate performance. In Chapter 3, we discuss how three different concatenated codes can be applied to the present setup, and compare their performance with that of convolutional codes and other existing coding methods for the wiretap channel in the literature. Finally, we conclude the thesis in Chapter 4.

## Chapter 2

# Randomized Convolutional Codes for the Wiretap Channel

In this chapter, we study application of convolutional codes to the randomized encoding scheme introduced by Wyner [3] as a way of confusing the eavesdropper over the wiretap channel. We describe optimal and practical sub-optimal decoders for the main and wiretapper channels, and estimate the security gap which is used as the main measure of physical layer security. The sub-optimal decoder works based on the trellis of the code generated by a convolutional code and its dual where one encodes data bits and the other encodes the random bits. By developing a code design metric, we describe how these two generators should be selected for optimal performance over a Gaussian wiretap channel. We also propose application of list Viterbi decoding algorithm to this setup so as to improve the performance of sub-optimal decoders. Furthermore, we provide an analytical characterization of the system performance by extending existing lower and upper bounds for coded systems to the current randomized convolutional coding setup. We illustrate our findings via extensive simulations and numerical examples.

The chapter is organized as follows. An introduction to the problem being solved is provided in Section 2.1. The channel model is introduced in Section 2.2.

The encoding scheme and convolutional code design for the randomized coding scheme are given in Section 2.3. The optimal and several sub-optimal decoders are presented in Section 2.4. Lower and upper bounds on the error rate performance of the proposed system are developed in Section 2.5. Extensive numerical examples are provided in Section 2.6, and finally, the chapter is summarized in Section 2.7.

## 2.1 Introduction

Wiretap channel introduced by Wyner [3] is a basic model for studying secure communications and was described in chapter 1. In his original work, Wyner introduces a metric called equivocation indicating how much information can be extracted by the eavesdropper about the original message as a measure of its confusion and points out that a system designer wants to make the probability of decoding error over the main channel minimum (reliability constraint) while maximizing the equivocation (security constraint). Wyner defines the notion of secrecy capacity  $C_s$  as the maximum achievable transmission rate that satisfies the security condition. He also proves that one can achieve the secrecy capacity using a randomized encoding scheme at the transmitter which is the main source of confusion for the eavesdropper [3]. This encoding method is often referred as *coset-coding* and is studied further in the subsequent literature, e.g. in [14].

From an information theoretic point of view, the equivocation, which is defined as the conditional entropy of the secret message given the eavesdropper's observation, is a valuable metric in order to measure the level of secrecy. On the other hand, it is difficult to work with for designing practical coding schemes. Therefore, bit error rate (BER) becomes an important metric with the motivation that if the BER at the eavesdropper is close to  $1/2$ , we expect that the eavesdropper cannot extract much information about the original message from what it receives [9], [10]. In this work, we follow the same approach and use the BERs  $P_{main}$  and  $P_{eve}$  calculated through the main and eavesdropper channels, respectively, as the

measure of secrecy. Denoting the desired maximum BER through the main channel with  $P_{main}^{max}$  and the desired minimum BER through the eavesdropper channel with  $P_{eve}^{min}$ , reliability and security constraints are stated as  $P_{main} \leq P_{main}^{max}$  ( $\approx 0$ ) and  $P_{eve} \geq P_{eve}^{min}$  ( $\approx 0.5$ ), respectively. We consider  $\text{SNR}_{main}$  as the lowest SNR which satisfies the reliability constraint and  $\text{SNR}_{eve}$  as the largest SNR which satisfies the security constraint. Difference between  $\text{SNR}_{main}$  and  $\text{SNR}_{eve}$  is defined as the *security gap*. Clearly, codes with small *security gaps* are desirable.

In this chapter, we describe how convolutional codes can be applied to Wyner's randomized encoding method, evaluate the performance of finite length (terminated) randomized convolutional codes over the Gaussian and binary symmetric wiretap channels, and provide practical decoders for use at the receivers. We argue that the concept of dual of a convolutional code plays a crucial role in both encoding and decoding schemes in this setup. In the randomized encoding scheme, there are multiple codewords (i.e., members of a coset) which represent a message whereas in conventional encoding, each message is mapped to one codeword. To transmit a message, one first chooses the corresponding coset and then selects one of the codewords within that coset uniformly randomly.

Using convolutional codes in the randomized encoding scheme enables us to propose effective and computationally inexpensive decoders whose objective is to choose the right coset. Optimal decoder needs to run through all the codewords in all the cosets which makes it impractical for medium to large length codes which motivates the development of sub-optimal approaches. Furthermore, using existing algorithms [15] to compute the distance spectrum of convolutional codes, we provide lower and upper bounds on the performance of the randomized convolutional codes in terms of message error probability. The upper bound employed is based on an application of the tangential sphere bound (TSB) which is a tight bound on the maximum likelihood (ML) decoder performance [16], while the lower bound is an approximate version of Seguin's bound adapted from [17].

## 2.2 Channel Model

The wiretap channel consists of one transmitter and two receivers. For the Gaussian wiretap channel, We assume that both the main and wiretapper channels are additive white Gaussian noise (AWGN) channels and express the input-output relationship as

$$\mathbf{y} = \mathbf{x}_i + \mathbf{N} \quad (2.1)$$

where  $\mathbf{x}_i = (-1)^{c_i}$  is the Binary phase-shift keying (BPSK) modulated version of the transmitted codeword of length  $n$ .  $\mathbf{N}$  is a length  $n$  Gaussian noise vector with independent and identically distributed (i.i.d.) components with zero mean and variance  $N_0/2$ . Note that for unit energy per dimension ( $E = 1$ ),  $E_b = 1/R$  where  $E_b$  is energy per bit and  $R$  is the transmission rate. We emphasize that the model in (2.1) is used for both the main and wiretapper channels (with different noise power levels).

## 2.3 Randomized Convolutional Codes – Encoding

### 2.3.1 Randomized Encoding Method

To construct a randomized encoding scheme which aims to confuse the eavesdropper, we assign one coset to each message being transmitted as in [6]. To transmit a  $k$ -bit message we need  $2^k$  many cosets. Suppose that there are  $2^r$  codewords in each coset. Then, we need a linear code of length  $n$  and dimension at least  $k + r$  (assuming  $k + r \leq n$ ) which we call the *big code* to cover all the codewords in this setup. In this manner, each coset consists of a unique set of codewords and no  $n$ -tuple can be found which belongs to more than one coset. We choose a terminated convolutional code  $\mathcal{C}(n, r)$  (with length  $n$  and dimension  $r$ ) as the first coset which we call *small code* with generators  $\mathbf{g}_1, \mathbf{g}_2, \dots, \mathbf{g}_r$  where  $\mathbf{g}_i$ 's are  $1 \times n$  vectors. To generate the remaining  $2^k - 1$  cosets with unique codewords, we identify linearly independent  $n$ -tuples outside  $\mathcal{C}$  which we denote by  $\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_k$ .

A message denoted by *data bits*  $\mathbf{s} = [s_1, s_2, \dots, s_k]$  is mapped to the coset obtained by  $s_1\mathbf{h}_1 + s_2\mathbf{h}_2 + \dots + s_k\mathbf{h}_k + \mathcal{C}$  which makes the transmission rate  $R = k/n$ . Finally, the transmitted codeword  $\mathbf{c}$  of length  $n$  is determined by choosing a random codeword in  $\mathcal{C}$  which is done using a random vector denoted by  $\mathbf{v} = [v_1, v_2, \dots, v_r]$  (where  $v_i$ 's are i.i.d. 0's and 1's each with probability  $1/2$ ) in the following way [6]

$$\mathbf{c} = s_1\mathbf{h}_1 + s_2\mathbf{h}_2 + \dots + s_k\mathbf{h}_k + v_1\mathbf{g}_1 + v_2\mathbf{g}_2 + \dots + v_r\mathbf{g}_r. \quad (2.2)$$

Figure 2.1 illustrates the randomized encoding scheme and shows how every secret message maps to a unique coset. This method requires two sets of generators to encode the message: one for random bits ( $v_i$ 's) and one for data bits ( $s_i$ 's). It is desirable to select  $\mathbf{h}_i$ 's and  $\mathbf{g}_i$ 's such that bit error probability of  $\mathbf{s}$  through the main channel goes to 0 (reliability constraint) while it goes to  $1/2$  in the wiretapper channel (security constraint).

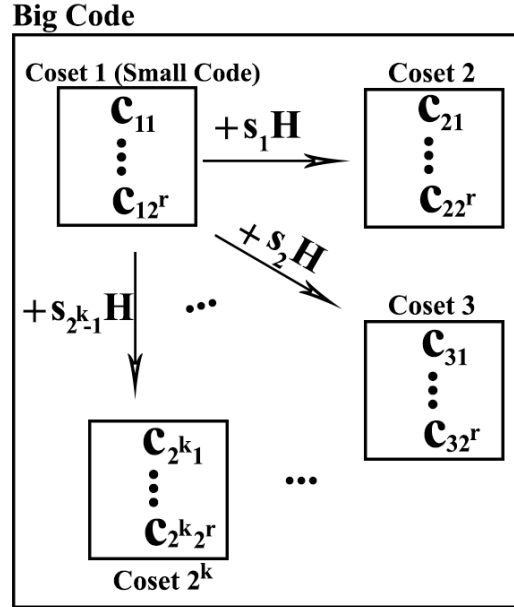


Figure 2.1: Illustration of the randomized encoding scheme where  $\mathbf{c}_{ij}$  denotes  $j$ th codeword in  $i$ th coset and  $\mathbf{s}_i$ 's correspond to the all possible non-zero messages of length  $k$ .  $\mathbf{H}$  is a matrix whose rows are  $\mathbf{h}_1$  through  $\mathbf{h}_k$  introduced in (2.2).

We note that one of the main motivations for using convolutional codes (rather than LDPC codes) is that the big code formed by two convolutional codes ( $\mathcal{C}$  and

$\mathcal{C}^\perp$ ) is another convolutional code as will be discussed in Section 2.4.2.2. Hence, its trellis structure enables us to propose efficient sub-optimal decoders which are necessary in practice. Furthermore, as will be shown in Section 2.6, achievable security gaps using suitably designed generators for random and data bits, compete favorably with other techniques, e.g., the LDPC puncturing method [9]. Finally, by utilizing the distance spectra of convolutional codes [15], we can obtain lower and upper bounds on the codeword error rates in the randomized encoding setup (see Section 2.5) which are important for a theoretical characterization of the performance at the eavesdropper and the main user, respectively.

Given the generators of  $\mathcal{C}$  ( $\mathbf{g}_i$ 's), obtaining  $\mathbf{h}_i$ 's requires an exhaustive search which is not practical for medium to large length codes. Here, we introduce a practical way to attack this problem by first defining what we refer to as *pseudo-self-dual* codes.

**Definition 1** *A linear code  $\mathcal{C}(n, r)$  with generator matrix  $\mathbf{G}$  is called pseudo-self-dual if  $\mathbf{G}\mathbf{G}^T = \mathbf{0}$ .*

**Theorem 1** *Suppose  $\mathcal{C}^\perp(n, n-r)$  is the dual of linear code  $\mathcal{C}(n, r)$ . The non-zero codewords of  $\mathcal{C}^\perp$  and  $\mathcal{C}$  are different if  $\mathcal{C}^\perp$  is not pseudo-self-dual.*

**Proof 1** *Let us denote the generator matrices of  $\mathcal{C}$  and  $\mathcal{C}^\perp$  with  $\mathbf{G}$  and  $\mathbf{G}^\perp$ , respectively. Assume that there is one non-zero codeword belongs to both of these codes, so there should be non-zero vectors  $\mathbf{u}$  and  $\mathbf{v}$  such that  $\mathbf{u}\mathbf{G} = \mathbf{v}\mathbf{G}^\perp$ . Multiplying both sides with  $(\mathbf{G}^\perp)^T$  from right side, we obtain  $\mathbf{u}\mathbf{G}(\mathbf{G}^\perp)^T = \mathbf{v}\mathbf{G}^\perp(\mathbf{G}^\perp)^T$  which results in  $\mathbf{v}\mathbf{G}^\perp(\mathbf{G}^\perp)^T = \mathbf{0}$  since  $\mathcal{C}$  and  $\mathcal{C}^\perp$  are dual of each other. But the last equality is in contradiction with the assumption that  $\mathcal{C}^\perp$  is not pseudo-self-dual. Hence, there cannot be a non-zero  $n$ -tuple which is a codeword generated by both  $\mathbf{G}$  and  $\mathbf{G}^\perp$ .*

We recall that two conditions need to be satisfied for  $\mathbf{h}_i$ 's: 1) they should not be a codeword in the small code  $\mathcal{C}$ ; 2) they should be linearly independent. Using

Theorem 1, by choosing generators of  $\mathcal{C}^\perp$  as  $h_i$ 's, the first condition is satisfied if  $\mathcal{C}$  is not pseudo-self-dual, and the second condition is satisfied since they are generators of a linear code ( $\mathcal{C}^\perp$ ).

Theorem 1 implies that it is not always possible to use generators of the  $\mathcal{C}^\perp$  to construct the cosets of  $\mathcal{C}$ . As an example, let us consider the small code  $\mathcal{C}$  to be a single parity check (SPC) code ( $n = 8, k = 7, d_{\min} = 2$ ).  $\mathcal{C}^\perp$  is then the repetition code ( $n = 8, k = 1, d_{\min} = 8$ ) which only has one generator:  $G^\perp = [1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1]$ . But this generator is a codeword in  $\mathcal{C}$  which means using it in (2.2) only reproduce the small code  $\mathcal{C}$  and will not result in a new coset. In this example, we note that  $G^\perp(G^\perp)^T = 0$  which means  $\mathcal{C}^\perp$  is pseudo-self-dual.

### 2.3.2 Dual of a Convolutional Code

Based on Theorem 1, we use the dual of a convolutional code for the randomized encoding scheme if it is not pseudo-self-dual. In this subsection, we describe how the dual of a convolutional code can be obtained in a systematic way.

For a binary convolutional encoder of rate  $a/b$  and memory  $m$ , the information sequence  $\mathbf{u} = \mathbf{u}_0\mathbf{u}_1\mathbf{u}_2\ldots$  ( $\mathbf{u}_i$ 's are  $1 \times a$ ) and the encoded sequence  $\mathbf{v} = \mathbf{v}_0\mathbf{v}_1\mathbf{v}_2\ldots$  ( $\mathbf{v}_i$ 's are  $1 \times b$ ) satisfy

$$\mathbf{v}_t = \mathbf{u}_t\mathbf{G}_0 + \mathbf{u}_{t-1}\mathbf{G}_1 + \ldots + \mathbf{u}_{t-m}\mathbf{G}_m \quad (2.3)$$

where  $\mathbf{G}_i$  is an  $a \times b$  binary matrix. That is, one can write  $\mathbf{v} = \mathbf{u}\mathbf{G}$  with

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_m \\ & \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_m \\ & & \ddots & \ddots & \ddots \end{bmatrix}. \quad (2.4)$$

The generator matrix of the dual code which is of rate  $(b-a)/a$  can be written as

$$\mathbf{G}^\perp = \begin{bmatrix} \mathbf{G}_0^\perp & \mathbf{G}_1^\perp & \cdots & \mathbf{G}_{m^\perp}^\perp \\ & \mathbf{G}_0^\perp & \mathbf{G}_1^\perp & \cdots & \mathbf{G}_{m^\perp}^\perp \\ & & \ddots & \ddots & \ddots \end{bmatrix} \quad (2.5)$$

with  $\mathbf{G}(\mathbf{G}^\perp)^T = \mathbf{0}$ . We now restate a result from [18].



**Definition 2** Reverse of a convolutional code  $\mathcal{C}$  with polynomial generator  $\mathbf{G}(D) = \mathbf{G}_0 + \mathbf{G}_1D + \dots + \mathbf{G}_mD^m$  is defined as the convolutional code  $\tilde{\mathcal{C}}$  with polynomial generator  $\tilde{\mathbf{G}}(D) = \mathbf{G}_m + \mathbf{G}_{m-1}D + \dots + \mathbf{G}_0D^m$ .

**Theorem 2** (Taken from [18]) Dual of a convolutional code  $\mathcal{C}$  with polynomial generator  $\mathbf{G}(D)$  has a polynomial generator of the form  $\tilde{\mathbf{H}}(D)$  where  $\mathbf{G}(D)(\tilde{\mathbf{H}}(D))^T = \mathbf{0}$ .

**Proof 2** For completeness, we provide a brief proof of this result. let  $\mathbf{G}(D) = \mathbf{G}_0 + \mathbf{G}_1D + \dots + \mathbf{G}_mD^m$  and denote the polynomial generator of its dual  $\mathcal{C}^\perp$  by  $\mathbf{G}^\perp(D) = \mathbf{G}_0^\perp + \mathbf{G}_1^\perp D + \dots + \mathbf{G}_{m^\perp}^\perp D^{m^\perp}$ . The reverse of  $\mathcal{C}^\perp$  is determined as  $\tilde{\mathbf{G}}^\perp(D) = \mathbf{G}_{m^\perp}^\perp + \mathbf{G}_{m^\perp-1}^\perp D + \dots + \mathbf{G}_0^\perp D^{m^\perp}$ . Consider

$$\mathbf{G}(D)(\tilde{\mathbf{G}}^\perp(D))^T = \mathbf{G}_0(\mathbf{G}_{m^\perp}^\perp)^T + (\mathbf{G}_0(\mathbf{G}_{m^\perp-1}^\perp)^T + \mathbf{G}_1(\mathbf{G}_{m^\perp}^\perp)^T)D + \dots + \mathbf{G}_m(\mathbf{G}_0^\perp)^T D^{m+m^\perp} \quad (2.6)$$

One can see that the coefficients of  $D^i$  (for all  $i$ ) in (2.6) are elements of the matrix  $\mathbf{G}(\mathbf{G}^\perp)^T$  which are equal to zero since  $\mathbf{G}$  and  $\mathbf{G}^\perp$  are dual of each other, i.e.,  $\mathbf{G}(D)(\tilde{\mathbf{G}}^\perp(D))^T = \mathbf{0}$  which results in  $\mathbf{H}(D) = \tilde{\mathbf{G}}^\perp(D)$  or equivalently,  $\mathbf{G}^\perp(D) = \tilde{\mathbf{H}}(D)$  concluding the proof.

To use Theorem 2, we need to compute  $\mathbf{H}(D)$  based on  $\mathbf{G}(D)$  such that  $\mathbf{G}(D)(\mathbf{H}(D))^T = \mathbf{0}$ . A straightforward way is to convert  $\mathbf{G}(D)$  to its systematic form by row operations. Having  $\mathbf{G}_{sys}(D) = [\mathbf{I}_k | \mathbf{P}(D)]$  one can write  $\mathbf{H}_{sys}(D) = [\mathbf{P}^T(D) | \mathbf{I}_{n-k}]$  where  $\mathbf{I}$  is the identity matrix and some elements of  $\mathbf{H}_{sys}(D)$  are rational functions of  $D$ . Multiplying  $\mathbf{H}_{sys}(D)$  by a suitable polynomial will remove the denominators and will result in  $\mathbf{H}(D)$ .

As a simple example, if  $\mathbf{G}(D) = [1 + D + D^2 \quad 1 + D^2]$  then  $\mathbf{H}(D) = [1 + D^2 \quad 1 + D + D^2]$ . Using Theorem 2 we get  $\mathbf{G}^\perp(D) = \tilde{\mathbf{H}}(D) = [1 + D^2 \quad 1 + D + D^2]$ . Hence, the dual of a  $[7 \ 5]^1$  (in octal notation) convolutional code with memory 2 is the  $[5 \ 7]$  convolutional code. Similarly, dual of a  $[117 \ 155]$  with memory 6 is  $[133 \ 171]$ . For these two cases, one can also verify that  $\mathbf{G}^\perp$  is not *pseudo-self-dual*

---

<sup>1</sup>Throughout this work, we denote convolutional codes with octal notation.

which makes them suitable for the proposed encoding scheme over the wiretap channel.

### 2.3.3 Obtaining a Subset of Convolutional Codes

As discussed in Section 2.3.1, the codewords in each coset represent a single message and are aimed at confusing the eavesdropper. If the main channel is noiseless, we are not concerned with the decoding process at the legitimate receiver, and we only want to confuse the eavesdropper. In this case, it is desirable to use as many codewords as possible in each coset. If the main channel is also noisy, then one should consider reducing the number of codewords in each coset in order to increase error correction capabilities at the legitimate receiver. As discussed in Section 2.3.1, the number of codewords in each coset is governed by the small code  $\mathcal{C}(n, r)$  introduced in Section 2.3.1 and equals  $2^r$  assuming that the random bits are being encoded by generators of the small code.

Let  $\mathcal{C}$  be a convolutional code of rate  $a/b$  with the generator matrix  $\mathbf{G}(D)$  with  $a$  rows. After finding the equivalent generator matrix  $\mathbf{G}^{[k]}(D)$  to  $\mathbf{G}(D)$  with rate  $ka/kb$  for  $k = 2, 3, \dots$ , one can obtain a subset of  $\mathcal{C}$  by choosing different rows from  $ka$  available rows of the  $\mathbf{G}^{[k]}(D)$ . Clearly, the resulting convolutional code has a smaller rate than  $\mathcal{C}$  and improved error correction capabilities.

We now explain how one can obtain an equivalent generator matrix  $\mathbf{G}^{[k]}(D)$  with rate  $k/bk$ ,  $k = 2, 3, \dots$  for a convolutional code with generator matrix  $\mathbf{G}(D)$  of rate  $1/b$ . The extension of the method to the general case (for a rate  $a/b$  code) is quite straightforward.  $\mathbf{G}^{[k]}(D)$  accepts  $k$  input bits in each time slot, so the input bits  $u_i$ 's are fed to the encoders in the following manner

$$\begin{array}{ccccccc}
\ldots & u_{i+3k-1} & u_{i+2k-1} & u_{i+k-1} & \rightarrow & g_1 \\
\ldots & u_{i+3k-2} & u_{i+2k-2} & u_{i+k-2} & \rightarrow & g_2 \\
& \vdots & \vdots & \vdots & & \vdots \\
\ldots & u_{i+2k+1} & u_{i+k+1} & u_{i+1} & \rightarrow & g_{k-1} \\
\ldots & u_{i+2k} & u_{i+k} & u_i & \rightarrow & g_k \\
\ldots & D^2 & D & 1 & & 
\end{array} \tag{2.7}$$

where “ $\rightarrow g_i$ ” means that the bits are being fed to a specific generator  $g_i$  (a row of  $\mathbf{G}^{[k]}(D)$ ) and the last row denotes the delay associated with the input bits in each column. We denote the output sequence of  $\mathbf{G}(D)$  to the input bit  $u_{i+f}$  with  $\mathbf{v}_f$  whose elements are  $v_{f,j}$  where  $0 \leq f \leq k-1$  and  $1 \leq j \leq b$ . Furthermore, we consider the corresponding output of  $\mathbf{G}^{[k]}(D)$  to the input vector  $[u_i \ u_{i+1} \ \dots \ u_{i+k-1}]$  as  $[\mathbf{o}_0 \ \mathbf{o}_1 \ \dots \ \mathbf{o}_{k-1}]$  where each  $\mathbf{o}_f$  is a vector consisting of  $b$  sequences, and each sequence is the sum of the delayed  $u_i$ 's produced through the  $k$  generators within the structure in (2.7).  $\mathbf{G}^{[k]}(D)$  and  $\mathbf{G}(D)$  are equivalent if

$$\mathbf{v}_f = \mathbf{o}_{k-f-1}, \quad 0 \leq f \leq k-1 \quad (2.8)$$

where  $\mathbf{v}_f = u_{i+f} \mathbf{G}(D)$  which is known since  $\mathbf{G}(D)$  is given. We note that each element of  $\mathbf{o}_i$  is produced by a column of  $\mathbf{G}^{[k]}(D)$ . Hence, each of the  $bk$  equations in (2.8) determines the suitable  $k$  generators,  $g_i$ 's,  $1 \leq i \leq k$  needed for the corresponding column of  $\mathbf{G}^{[k]}(D)$ .

**Example 1** Consider the [561 753] convolutional code of memory  $m = 8$  and rate  $1/2$ , i.e.,

$$\mathbf{G}(D) = [1 + D^2 + D^3 + D^4 + D^8 \quad 1 + D + D^2 + D^3 + D^5 + D^7 + D^8]. \quad (2.9)$$

Following the same steps described above, we can obtain the equivalent generator matrix of  $\mathbf{G}(D)$  with rate  $4/8$ :

$$\mathbf{G}^{[4]}(D) = \begin{bmatrix} p(D) & 1+D^2 & 0 & 1+D & 1 & 1 & 1 & 1+D \\ D & D+D^2 & p(D) & 1+D^2 & 0 & 1+D & 1 & 1 \\ D & D & D & D+D^2 & p(D) & 1+D^2 & 0 & 1+D \\ 0 & D+D^2 & D & D & D & D+D^2 & p(D) & 1+D^2 \end{bmatrix} \quad (2.10)$$

where  $p(D) = 1 + D + D^2$ . To obtain a subset of  $\mathcal{C}$ , one can use any subset of the rows of  $\mathbf{G}^{[4]}(D)$  as the generator matrix. We note that the resulting subset has a smaller rate than the original code  $\mathcal{C}$ . For example, if we choose only one of the rows of  $\mathbf{G}^{[4]}(D)$  as the generator matrix, the resulting code has a rate of  $1/8$ . ■

### 2.3.4 Convolutional Code Design for the Randomized Encoding Scheme

Earlier in this section, we discussed how a small code and its dual can be used to form the big code. Since both the small code and its dual are assumed to be convolutional codes, the big code is also a convolutional code. Clearly, the minimum pairwise distance among the codewords in each coset with respect to a specific codeword is larger than (or equal to) the minimum distance of the big code with respect to the same codeword. So, the codewords at minimum distance in the big code belong to different cosets and assuming that a minimum distance decoder is being used, they are important sources of decoding errors. Hence, a design metric becomes the minimum pairwise distance among the codewords of the big code which controls the error correcting capability of the minimum distance decoder. In practice, one should choose this distance in a way that results in the smallest security gap.

If one uses a convolutional code  $\mathcal{C}(n, r)$  (small code) to encode random bits and its dual  $\mathcal{C}^\perp(n, n-r)$  to encode the data bits, the big code would consist of all the  $2^n$   $n$ -tuples (ignoring trellis termination to zero state for the time being); a fact that results in the lowest possible minimum distance (one) for the big code. In this case, performance of the minimum distance decoder is poor from legitimate receiver's point of view. Alternatively, one can use the approach described in the previous subsection to obtain a subset of  $\mathcal{C}(n, r)$  denoted by  $\mathcal{C}'(n, r')$  where  $r' < r$ . Now, using generators of  $\mathcal{C}'$  and  $\mathcal{C}^\perp$  to encode random and data bits, respectively, the big code will have  $r' + n - r$  many generators which is less than  $n$ ; hence, the resulting big code can achieve a larger minimum distance. We note that in either case transmission rate is  $(n-r)/n$  since the data bits' encoder is the same.

Consider the small code  $\mathcal{C}$  to be a convolutional code of rate  $R = b/c$  with *minimal-basic* generator matrix  $\mathbf{G}(D)$  [18]. Equivalent generator matrices to  $\mathbf{G}(D)$  which reproduce  $\mathcal{C}$  are obtained by

$$\mathbf{G}_{2nd}(D) = \mathbf{T}(D)\mathbf{G}(D) \quad (2.11)$$

where  $\mathbf{T}(D)$  is a  $b \times b$  full rank matrix. Then, instead of working with  $\mathbf{G}(D)$ , one

may use  $\mathbf{G}_{2nd}(D)$  in Section 2.3.3 to obtain new subsets of  $\mathcal{C}$  and consequently new generators for random bits. Hence, different choices for  $\mathbf{T}(D)$  result in different generators for random bits. It is clear that different generators for random bits, result in different sets of codewords in each coset and consequently possible different minimum distances for the big code. In the next example, given the encoder for data bits, we search for an encoder for random bits which results in a big code with a large minimum distance.

**Example 2** *Let us choose the small code  $\mathcal{C}$  as the convolutional code [561 753] which is the same code given earlier in (2.9). Its dual  $\mathcal{C}^\perp$  is the optimal convolutional code of memory 8 and rate 1/2 with the generator [657 435]. If one uses generators of  $\mathcal{C}^\perp$  and the entire  $\mathcal{C}$  to encode data and random bits, respectively, the resulting big code will have a minimum distance of 2 (they do not cover all the  $n$ -tuples because of the trellis termination to zero state). However, if one uses generators of  $\mathcal{C}^\perp$  for data bits and  $[D \ D \ D \ D + D^2 \ p(D) \ 1 + D^2 \ 0 \ 1 + D]$  for random bits which is a subset of  $\mathcal{C}$  as we derived in Example 1, the big code will attain a minimum distance of 6.*

*We can improve the minimum distance even more by using (2.11)*

$$\mathbf{G}_{2nd}^{[4]}(D) = \mathbf{T}(D)\mathbf{G}^{[4]}(D) \quad (2.12)$$

*where  $\mathbf{G}^{[4]}(D)$  is the same as (2.10) and the  $4 \times 4$  matrix  $\mathbf{T}(D)$  is given by its polynomial inverse*

$$\mathbf{T}^{-1}(D) = \begin{bmatrix} 1+D & D & D & 1+D \\ D & D^2+1 & 1 & D \\ D & D & 1+D & D \\ 1+D & 1 & D & D \end{bmatrix}. \quad (2.13)$$

*After some straightforward algebra, one can calculate  $\mathbf{G}_{2nd}^{[4]}(D)$  (which is  $4 \times 8$ )*

and obtain one of its rows as

$$\begin{bmatrix} D^5 + D^4 + D^3 & D^5 + D^3 + D^2 & D^4 + D^3 & D^5 + D & D^5 + D^4 + D^3 + D^2 + D + 1 \\ D^5 + D^3 + D^2 + D + 1 & D^3 + D^2 & D^3 + D^2 + 1 \end{bmatrix} \quad (2.14)$$

Using  $\mathcal{C}^\perp$  and (2.14), we obtain a big code with minimum distance 10. Here, it is clear that data bits are encoded with rate  $1/2$  while random bits' encoding rate is  $1/8$ . We note that the code  $\mathcal{C}^\perp$  has a minimum distance of 12 which is an upper bound on the minimum distance of the big code. ■

## 2.4 Decoding Methods

### 2.4.1 Optimal Decoder

Given a received noisy vector  $\mathbf{y}$ , the optimal decoder would pick a coset index which maximizes the probability  $p(C^i|\mathbf{y})$  where  $C^i$  denotes the  $i$ th coset. Assuming there are  $M$  cosets which represent  $M$  messages and in each of them there are  $N$  codewords, the output of the optimal MAP decoder is

$$\hat{i} = \underset{i=1,2,\dots,M}{\operatorname{argmax}} p(C^i|\mathbf{y}) \quad (2.15)$$

Using Bayes' rule and the total probability theorem (assuming that the codewords in each coset have equal probabilities to be transmitted through the channel), we can write

$$\begin{aligned} p(C^i|\mathbf{y}) &= \frac{p(\mathbf{y}|C^i)p(C^i)}{p(\mathbf{y})}, \\ p(\mathbf{y}|C^i) &= \frac{1}{N} \sum_{j=1}^N p(\mathbf{y}|\mathbf{c}_{ji}), \end{aligned} \quad (2.16)$$

where  $\mathbf{c}_{ji}$  denote the  $j$ th codeword in the  $i$ th coset. Finally, for an AWGN channel and equiprobable cosets, the optimal decoder has the form

$$\hat{i} = \underset{i=1,2,\dots,M}{\operatorname{argmax}} \sum_{j=1}^N e^{-\frac{\|\mathbf{y}-\mathbf{c}_{ji}\|^2}{2\sigma^2}}, \quad (2.17)$$

where  $\sigma^2 = N_0/2$ . Note that for the main and wiretapper channel the noise variances are different, hence the resulting optimal decoding rules are different.

For the case of a binary symmetric channel with cross over probability  $p$

$$p(\mathbf{y}|\mathbf{c}_{ji}) = (1-p)^n \left( \frac{p}{1-p} \right)^{d_H(\mathbf{y}, \mathbf{c}_{ji})} \quad (2.18)$$

where  $d_H(\mathbf{y}, \mathbf{c}_{ji})$  is the Hamming distance between received vector  $\mathbf{y}$  and codeword  $\mathbf{c}_{ji}$ . In this case, the optimal decoding rule for BSC can be obtained from (2.16) as

$$\hat{i} = \underset{i=1,2,\dots,M}{\operatorname{argmax}} \sum_{j=1}^N \left( \frac{p}{1-p} \right)^{d_H(\mathbf{y}, \mathbf{c}_{ji})}. \quad (2.19)$$

We note that for optimal decoding, one goes through all the codewords in all

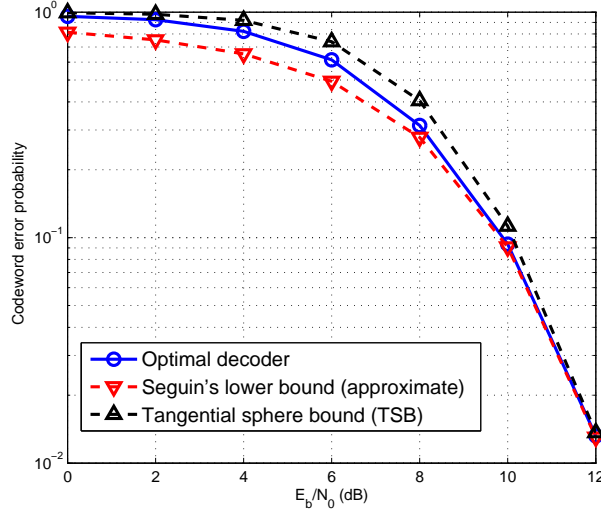


Figure 2.2: Performance of the optimal decoder introduced in (2.17) over an AWGN channel using a Reed-Muller code of length 16 to encode the messages. The number of cosets or messages is  $2^5$  each of which containing  $2^{11}$  codewords ( $n = 16, r = 11, k = 5$ ). The lower and upper bounds are developed in Section 2.5.

the cosets making the algorithm prohibitively complex to implement in practice. However, this process can be used for toy examples with small length codes. For instance, the performance of the optimal decoder is shown for a Reed-Muller code of length 16 in Figures 2.2 and 2.3 for AWGN and BSC channels, respectively (along with the corresponding performance bounds which will be introduced in

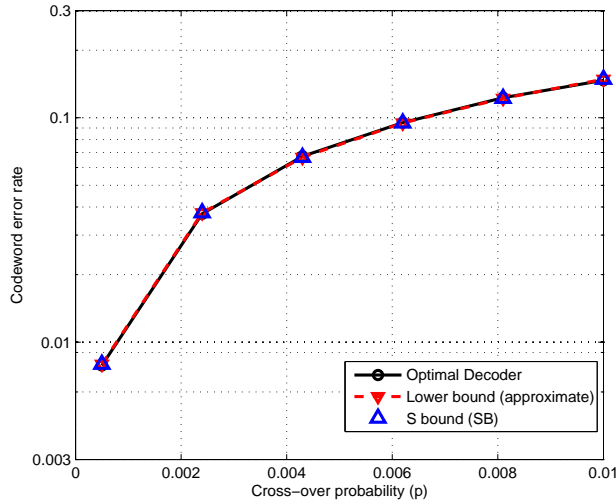


Figure 2.3: Performance of the optimal decoder introduced in (2.19) over BSC channel using a Reed-Muller code of length 16 to encode the messages. The number of cosets or messages is  $2^5$  each of which contains  $2^{11}$  codewords ( $n = 16, r = 11, k = 5$ ). The lower and upper bounds are discussed in Section 2.5.

Section 2.5). We emphasize that this is introduced as a toy example only, and the code has a poor performance in terms of the resulting security gap. We will provide examples of good codes with low security gaps in Section 2.6. The upper bound (which is a true bound) shows the worst case analysis for the main channel. On the other hand, the lower bound (which is approximate) represents the best that can be done by the eavesdropper (in terms of message error probability). Figure 2.2 and 2.3 also demonstrate that the upper and lower bounds are tight especially for high SNRs.

### 2.4.2 Sub-Optimal Decoders

The optimal decoding procedure for the randomized encoding scheme is too complex for practical implementations, hence here we consider several sub-optimal decoding alternatives.



### 2.4.2.1 Binary Gaussian Elimination

The encoding scheme in Section 2.3.1 can be written in matrix form. Suppose  $\mathbf{G}$  is the generator matrix of the small code  $\mathcal{C}(n, r)$ . We form a matrix  $\mathbf{H}$  whose rows are  $k$  linearly independent  $n$ -tuples  $\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_k$  outside  $\mathcal{C}$ . Therefore, as in [6] one can write the transmitted codeword as follows

$$\mathbf{x} = [\mathbf{s} \ \mathbf{v}] \mathbf{G}_B, \quad \mathbf{G}_B = \begin{bmatrix} \mathbf{H} \\ \mathbf{G} \end{bmatrix}. \quad (2.20)$$

Motivated by this, a rough decoding approach could be to perform hard decisions on the received vector from the channel to obtain a binary vector which will be denoted by  $\hat{\mathbf{x}}$ . Then, one can form  $[\mathbf{G}_B | \hat{\mathbf{x}}^T]$ , and through binary Gaussian elimination obtain  $[\mathbf{I} | \mathbf{x}_d^T]$  where  $\mathbf{I}$  is the identity matrix. The first  $k$  bits of  $\mathbf{x}_d$  are the decoded versions of the transmitted message  $\mathbf{s}$ .

This decoding method ignores the available soft information and may not result in a good performance, however it is a general method, i.e., given the generator matrices for random and data bits ( $\mathbf{G}$  and  $\mathbf{H}$ ), it can be applied to any kind of codes. Specifically, low density generator matrix (LDGM) codes introduced in [19] are systematic codes with generator matrices of the form  $\mathbf{G} = [\mathbf{I}_{k \times k} | \mathbf{P}_{k \times (n-k)}]$  where  $\mathbf{P}$  is a sparse matrix. Hence, given one of  $\mathbf{G}$  or  $\mathbf{H}$ , the other can be obtained, and the binary Gaussian elimination can be used for this setup with ease.

### 2.4.2.2 Trellis Based Decoding

When the Euclidean distances among the codewords in each coset are relatively large or when the SNR is sufficiently high, the summation in the optimal decoder expressions in (2.17) or (2.19) is dominated by terms which correspond to codewords at the minimum Euclidean distance to the received vector  $\mathbf{y}$ . Therefore, as an approximate decoding approach, one can find the codeword at the minimum Euclidean (or, Hamming) distance to the given received noisy vector (referred as

the minimum distance decoder). Since at high SNRs, most errors will be due to closeby codewords, we expect that the performance of this decoder will be close to that of the optimal decoder in this regime.

Following with the development in Section 2.3, we recall that the encoding process needs two convolutional codes whose trellises can be combined to form a trellis for the big code governing codewords obtained by (2.2), i.e., the codewords that are being sent through the channel. This “big” trellis enables us to find the minimum distance codeword to the output of the channel  $\mathbf{y}$  by applying the Viterbi algorithm.

The overall encoder for the big code can be implemented using generators for random and data bits in parallel. As an example, when  $\mathbf{G}(D) = [1+D+D^2 \ 1+D^2]$  encodes random bits and  $\mathbf{G}^\perp(D) = [1+D^2 \ 1+D+D^2]$  encodes data bits, the overall encoder is shown in Figure 2.4.

We note that development and analysis of this decoding approach in the randomized coding scheme is important in other possible schemes as well, e.g., for turbo codes which are basically parallel (or serial) concatenation of convolutional codes.

#### 2.4.2.3 List Decoding for Randomized Convolutional Codes

One way to improve the performance of the minimum distance decoder is to incorporate more terms in the sums of (2.17) or (2.19). Observing that the terms which correspond to the codewords at low distances to the received noisy vector make the highest contributions to the results of the sums, we propose the use of *List Viterbi Decoding Algorithm (LVA)* [22] for this purpose. Namely, LVA can be used to nominate the top  $L$  codewords which are closest to the received vector, and then one can apply the decision rule in (2.17) or (2.19) among these  $L$  most probable codewords.

We can also devise a bit-wise decoder when LVA is used in the randomized

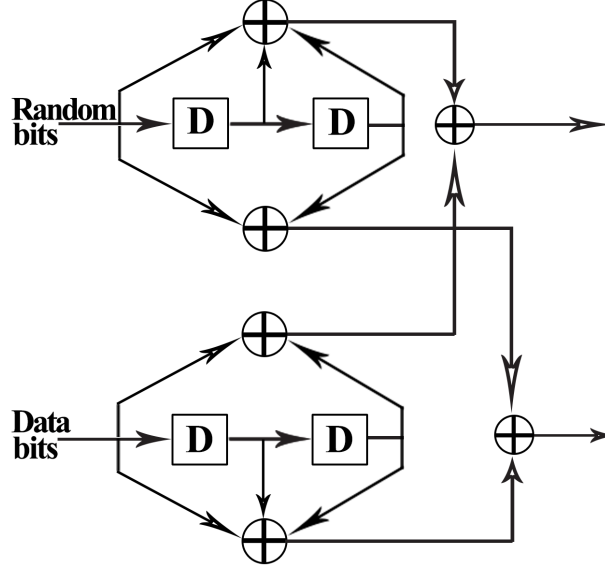


Figure 2.4: The overall encoder for the randomized encoding scheme when a  $[7\ 5]$  convolutional code encodes random bits and a  $[5\ 7]$  convolutional code (which is the dual of  $[7\ 5]$ ) encodes data bits.

encoding scheme. In this case, assuming an AWGN channel, we assign a probability to each nominated codeword based on its Euclidean distance to the received vector  $\mathbf{y}$ , which is denoted by  $p_i$  for the  $i$ th nominated codeword with

$$p_i = \frac{e^{-\frac{\|\mathbf{y}-\mathbf{c}_i\|^2}{2\sigma^2}}}{\sum_{k=1}^L e^{-\frac{\|\mathbf{y}-\mathbf{c}_k\|^2}{2\sigma^2}}}, \quad i = 1, \dots, L. \quad (2.21)$$

For the case of BSC, we assign probabilities based on Hamming distance which results in

$$p_i = \frac{\left(\frac{p}{1-p}\right)^{d_H(\mathbf{y}, \mathbf{c}_i)}}{\sum_{k=1}^L \left(\frac{p}{1-p}\right)^{d_H(\mathbf{y}, \mathbf{c}_k)}}, \quad i = 1, \dots, L. \quad (2.22)$$

where  $p$  is the cross-over probability of BSC. Then log-likelihood ratio for each bit is calculated as

$$LLR(j) = \log \frac{Pr(X_j = 1)}{Pr(X_j = 0)} = \log \frac{\sum_{i:c_i(j)=1} p_i}{\sum_{i:c_i(j)=0} p_i}, \quad j = 1, \dots, n, \quad (2.23)$$

where  $c_i(j)$  denote the  $j$ th bit of the  $i$ th nominated codeword. If  $LLR > 0$  the corresponding bit is decoded as 1 and is decoded as 0 otherwise.

## 2.5 Performance Bounds

In order to provide a theoretical assessment of the decoder performance in the randomized encoding scheme, we provide bounds on the resulting error rates. Specifically, we obtain lower and upper bounds on the error rates which indicate the best performance of the eavesdropper and the worst performance of the legitimate receiver, respectively, which are important from a design and analysis point of view.

### 2.5.1 Assumptions

As mentioned in Section 2.3.1, the adopted randomized encoding scheme maps each message to a coset of codewords. Hence, in contrast to conventional encoding, decision region for each message is not just a simple Voronoi region around the transmitted codeword. This fact results in further complications in calculating the corresponding ML decoding bounds. To proceed, we define the notion of *favorable* codewords.

**Definition 3** Suppose  $\mathbf{c}_{ij}$  which is the  $i$ th codeword in the  $j$ th coset is sent through the channel. We call all the other codewords in  $j$ th coset (i.e.,  $\mathbf{c}_{kj}$  such that  $k = 1, 2, \dots, N$ , and  $k \neq i$ ) favorable to  $\mathbf{c}_{ij}$ .

Known bounds on the ML decoding performance of linear codes can be applied to the randomized encoding scheme by making the following assumption: considering transmission of  $\mathbf{c}_{ij}$ , we ignore all the favorable codewords to  $\mathbf{c}_{ij}$ , i.e., neglect part of correct decision region, and compute lower and upper bounds on the performance of decoders in the randomized encoding scheme accordingly.

The following theorem proves the geometric uniformity of the big code after ignoring the favorable codewords.

**Theorem 3** Let  $\mathbf{c}_{ij}$  be the  $i$ th codeword in the  $j$ th coset and denote the distance spectrum of the code  $\mathcal{C}$  with respect to the codeword  $\mathbf{c}_{ij}$  by  $DS^{\mathcal{C}}\{\mathbf{c}_{ij}\}$ , and the

distance spectrum of the big code (bc) after ignoring favorable codewords with respect to  $\mathbf{c}_{ij}$  by  $DS\{\mathbf{c}_{ij}\}$ . Then  $DS\{\mathbf{c}_{ij}\} = DS\{\mathbf{c}_{lk}\}$ ,  $i \neq l$  and  $j \neq k$ , if the big code (bc) and the small code (sc) are both linear.

**Proof 3** The distance spectrum of the big code with respect to  $\mathbf{c}_{ij}$  after ignoring favorable codewords can be written as  $DS\{\mathbf{c}_{ij}\} = DS^{bc}\{\mathbf{c}_{ij}\} - DS^{coset\ j}\{\mathbf{c}_{ij}\}$  which means for each distance  $d \geq 1$  subtract the numbers of codewords with distance  $d$  in  $DS^{bc}\{\mathbf{c}_{ij}\}$  and  $DS^{coset\ j}\{\mathbf{c}_{ij}\}$  from each other. Since the big code is linear  $DS^{bc}\{\mathbf{c}_{ij}\} = DS^{bc}\{\mathbf{c}_{lk}\}$ . Linearity of small code results in  $DS^{sc}\{\mathbf{c}_{11}\} = DS^{sc}\{\mathbf{c}_{i1}\}$ . Furthermore, coset  $j$  obtained by adding a unique codeword to the small code which does not have any effect on the distance spectrum, namely,  $DS^{coset\ j}\{\mathbf{c}_{ij}\} = DS^{coset\ k}\{\mathbf{c}_{lk}\} = DS^{sc}\{\mathbf{c}_{11}\}$ , hence  $DS\{\mathbf{c}_{ij}\} = DS\{\mathbf{c}_{lk}\}$  concluding the proof.

By using Theorem 3, it is possible to compute the distance spectrum of the big code after ignoring the favorable codewords by considering only all-zero codeword as transmitted codeword, via the distance spectra of the small and big codes. Once the distance spectrum is computed, we utilize it with the existing bounds on ML decoding performance to obtain performance bounds for the randomized encoding scheme. If both the small and big codes are convolutional, their distance spectra can be obtained through efficient algorithms (e.g., [15]) which work based on their state transition matrix computed using the trellis representations of the convolutional codes employed.

We recall that the trellis of the big code is formed by trellises of the small code and its dual. If the generator for the random bits which produces the small code has memory  $m$  and the one for the data bits has memory  $n$ , then the state transition matrices of the small code and big code are  $2^m \times 2^m$  and  $(2^m \times 2^n) \times (2^m \times 2^n)$ , respectively. Specifically, for the generators selected in Example 2 (which result in the big code of minimum distance 10), the state transition matrix is  $(2^8 \times 2^5) \times (2^8 \times 2^5)$  for the resulting big code.

We further note that the derived bounds are applicable to other randomized coding setups as well (once the appropriate weight distributions are known). For

instance, one can easily obtain lower bounds on the error rates of LDPC coded systems (e.g., as in [6]) in a straightforward manner as only a subset of codewords with small weights are needed in the computation.

## 2.5.2 Performance Lower Bounds

We first note that the assumption made in Section 2.5.1, namely, ignoring part of the correct decision region, results in approximate lower bounds. Particularly, this assumption makes lower bounds overestimate the error probability. On the other hand, since distance of a codeword to its favorable codewords is much larger than its distance to the other codewords (see Section 2.3.4), we expect ignoring favorable codewords to not have a great impact on the final result.

We use Seguin's bound [17] to provide a lower bound on the decoder performance which states that the probability of error given that the signal  $\mathbf{s}_u$  is transmitted through an AWGN channel with variance  $N_0/2$ , denoted by  $P(\varepsilon|\mathbf{s}_u)$ , is lower bounded as

$$P(\varepsilon|\mathbf{s}_u) \geq \sum_{i \neq u} \frac{Q^2(\sqrt{2D_{ui}E_s/N_0})}{\sum_{j \neq u} \Psi(\rho_{ij}, \sqrt{2D_{ui}E_s/N_0}, \sqrt{2D_{uj}E_s/N_0})} \quad (2.24)$$

where  $D_{ui}$  is the Hamming distance between codewords  $i$  and  $j$ ,  $E_s/N_0$  is the SNR,  $Q$  is the usual  $Q$ -function (right tail probability of standard Gaussian distribution), and

$$\Psi(\rho, p_1, p_2) = \frac{1}{2\pi\sqrt{1-\rho^2}} \int_{p_1}^{\infty} \int_{p_2}^{\infty} \exp\left(-\frac{x^2 - 2\rho xy + y^2}{2(1-\rho^2)}\right) dx dy \quad (2.25)$$

with  $\rho_{ij}$  defined as

$$\rho_{ij} = \frac{w((\mathbf{c}_i + \mathbf{c}_u)(\mathbf{c}_j + \mathbf{c}_u))}{\sqrt{w(\mathbf{c}_i + \mathbf{c}_u)w(\mathbf{c}_j + \mathbf{c}_u)}} \quad (2.26)$$

is the correlation between two codewords  $\mathbf{c}_i$  and  $\mathbf{c}_j$  given that  $\mathbf{c}_u$  was transmitted. Here,  $w$  denotes the Hamming weight of a sequence.

It is clear from (2.24) that one can obtain a lower bound by taking only a subset of codewords into account; in other word, one does not need the entire distance spectrum to obtain a lower bound. Besides, as noted in [23], considering the

codewords at the minimum distance and  $\rho_{ij}$ 's play an important role on tightness of this bound. Finally, for the case of BSC, we use the lower bound introduced by Cohen and Merhav in [24].

### 2.5.3 Performance Upper Bounds

Similar to the lower bound, we ignore the favorable codewords in obtaining an upper bound on the error rates of the randomized encoding scheme. However, the resulting bound in this case is a true bound (not an approximate result) on the performance of the maximum likelihood decoder. This is because, we ignore part of correct decision region which naturally results in a looser characterization.

There are many upper bounds on the ML decoding performance of coded systems in the literature; to name two important ones, we cite Duman-Salehi bound [25] and tangential sphere bound (TSB) [16].

TSB is essentially based on a technique developed by Gallager [6] which utilizes the following intuitive inequality

$$P(\text{error}) \leq P(\text{error}, \underline{y} \in \mathcal{R}) + P(\underline{y} \notin \mathcal{R})$$

where  $\mathcal{R}$  is a region around the transmitted codeword. Poltyrev [16] selects  $\mathcal{R}$  to be a canonical region. It should be noted that many improved upper bounds can be derived by an appropriate selection of the region  $\mathcal{R}$ , such as tangential bound(TB) [26] (which let the radius of the cone to be infinity), Divsalar bound [27] (where the region  $\mathcal{R}$  is a hyper sphere with optimized center), sphere upper bound [28] (which is the special case of Divsalar bound with a fixed center at the transmitting point) which all used for the equal energy constellations or Hughes bound [29] which can be used for non-equal signal energies also. It is worth mentioning that the union bound can be obtained via TSB method setting region  $\mathcal{R}$  to be the whole space. TSB is one of the tightest bound known on the ML decoding error probability of binary block codes in AWGN channel [30], [31], [32]. For a detailed review of performance bounds, see [33].

TSB uses a method based on 2-levels of separation of noise components (radial

and tangential components) from the rest, resulting in

$$P(\varepsilon) \leq \int_{-\infty}^{\infty} \frac{e^{-\frac{z_1^2}{2\sigma^2}}}{\sqrt{2\pi}\sigma} \left\{ \sum_{k \leq \frac{nr_0^2}{n+r_0^2}} \left\{ S_k \int_{\beta_k(z_1)}^{r_{z_1}} \frac{e^{-\frac{z_2^2}{2\sigma^2}}}{\sqrt{2\pi}\sigma} \int_0^{r_{z_1}^2 - z_2^2} f_V(v) dv dz_2 \right\} + 1 - \gamma\left(\frac{n-1}{2}, \frac{r_{z_1}^2}{2\sigma^2}\right) \right\} dz_1 \quad (2.27)$$

where  $S_k$  is the number of codewords with Hamming weight  $k$ ,  $\beta_k(z_1) = (\sqrt{n} - z_1)/(\sqrt{n/k} - 1)$ ,  $r_{z_1} = r_0(\sqrt{n} - z_1)$ ,  $r_0$  is the optimal value of  $r_{z_1}$  [16] and

$$f_V(v) = \frac{v^{\frac{n-4}{2}} e^{-\frac{v}{2\sigma^2}}}{2^{\frac{n-2}{2}} \sigma^{n-2} \Gamma(\frac{n-2}{2})}, \quad v \geq 0, \quad (2.28)$$

$$\gamma(a, x) = \frac{1}{\Gamma(a)} \int_0^x t^{a-1} e^{-t} dt, \quad a > 0, x \geq 0.$$

For the case of a BSC, we use what is called the S bound (SB) given by [16]

$$P(\varepsilon) \leq \sum_{w=d}^{2(m_0-1)} S_w \sum_{\eta=t_w}^{m_0-1} \binom{w}{\eta} p^\eta (1-p)^{w-\eta} \sum_{k=0}^{m_0-\eta-1} \binom{n-w}{k} p^k (1-p)^{n-w-k} + \sum_{l=m_0}^n \binom{n}{l} p^l (1-p)^{n-l} \quad (2.29)$$

where  $t_w = \lceil w/2 \rceil$  and  $m_0$  is the smallest integer such that

$$\sum_{w=d}^{2m} S_w \sum_{\eta=t_w}^m \binom{w}{\eta} \binom{n-w}{m-\eta} \geq \binom{n}{m}. \quad (2.30)$$

## 2.5.4 A Simple Example

As an example the performance of lower and upper bounds introduced in Sections 2.5.2 and 2.5.3, for a Reed-Muller code is shown in Figure 2.2 and 2.3 which indicate a good match between the bounds and simulated performance of the optimal decoders. We will provide further examples for more practical codes (considering both AWGN and binary symmetric channels) in Section 2.6.

## 2.6 Numerical Examples

In this section, we provide numerical examples on the performance of the sub-optimal decoders introduced in Sections 2.4.2.1 and 2.4.2.2, theoretical bounds



introduced in Section 2.5, different code designs for the cases where main channel is noiseless or noisy, and application of list Viterbi decoding in 2.4.2.3 in the randomized encoding scheme.

### 2.6.1 Noiseless Main Channel

In this case, we assume that the main channel in wiretap channel is noiseless and the wiretapper channel is an AWGN or binary symmetric channel. As discussed in Section 2.3.4, for this scenario, the only task is to confuse the eavesdropper without worrying about the decoding process at the legitimate receiver. Therefore, the big code should have a low minimum distance; which is achieved using generators of the entire small code  $\mathcal{C}(n, r)$  to encode random bits and generators of its dual  $\mathcal{C}^\perp(n, n - r)$  for data bits (see section 2.3.4).

The approximate lower bounds on the following figures show the bounds on the best decoding capability of the eavesdropper through the wiretapper channel while the upper bounds (which are true bounds) indicate the worst case scenario for the legitimate receiver through the main channel. Both the upper and lower bounds are in terms of codeword error probabilities and they are computed under the assumptions stated in Section 2.5. We also note that the Seguin's bound is obtained by considering the codewords at minimum Hamming distance and calculating the correlations among them, while the TSB uses the entire distance spectrum.

Figure 2.5 illustrates the message error rates for the randomized encoding scheme with a convolutional encoder with generator  $[7 \ 5]$  encoding the random bits and its dual  $[5 \ 7]$  (which was derived in Section 2.3.2) encoding the data bits. The length of the data and random bits are both 100. With trellis termination, the overall codeword length is 204. That is, the small code and its dual are  $\mathcal{C}(204, 100)$  and  $\mathcal{C}^\perp(204, 100)$ , respectively, and the big code parameters are  $(204, 200)$ . Comparing the two sub-optimal decoders introduced in Section 2.4.2, we observe that the performance of the trellis based decoder is always better than performance of the binary Gaussian elimination decoder. Also TSB and Seguin's bound are quite tight in this case and all the curves meet at high SNRs

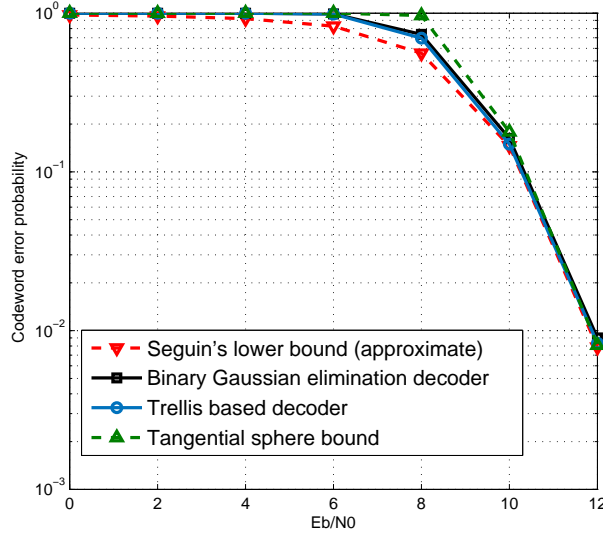


Figure 2.5: Performance of the sub-optimal decoders introduced in Section 2.4.2 and the bounds in Section 2.5 when a  $[7 \ 5]$  convolutional code with its dual  $[5 \ 7]$  has been used. The length of the codewords is 204. There are  $2^{100}$  cosets each of which containing  $2^{100}$  codewords.

as expected. Furthermore, Figure 2.6 shows the performance of the bounds and minimum distance decoder over a BSC where the lengths of the data and random bit sequences are both 50.

As a second example, we consider a convolutional encoder with polynomial generator  $[117 \ 155]$  for random bits and its dual  $[133 \ 171]$  for data bits. As shown in Figure 2.7, performance of binary Gaussian elimination is almost the same as the one in Figure 2.5, however, performance of the trellis based decoder in this case is much better than that of the binary Gaussian elimination decoder at high SNRs which is a result of using a higher memory size for convolutional encoders. For this example, lengths of the data and random bit sequences are both 100 which result in a codeword of length 212.

Seguin's bound relies on low weight codewords to provide tight lower bounds on the performance of the ML decoders [23]. We note that the lower bound in Figure 2.7 is not as tight as the one in Figure 2.5 because the minimum distance of the big code in Figure 2.7 is 2 while it is 1 in Figure 2.5. Finally, TSB is not included

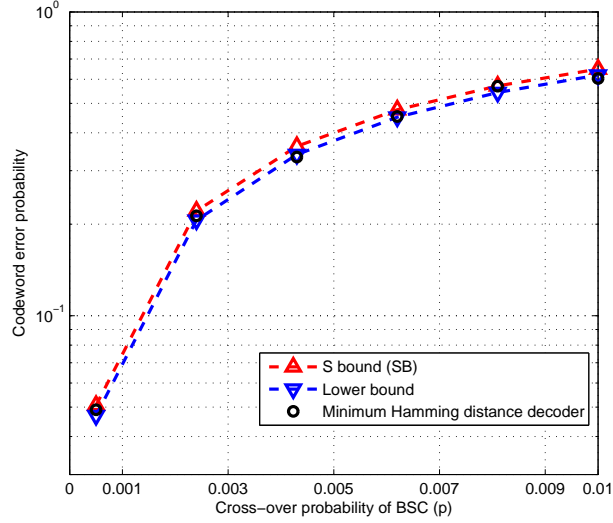


Figure 2.6: Performance of the minimum Hamming distance decoder (based on trellis) and the bounds introduced in Section 2.5 over BSC when a  $[7 \ 5]$  convolutional code with its dual  $[5 \ 7]$  has been used. The length of the codewords is 104. There are  $2^{50}$  cosets each of which containing  $2^{50}$  codewords.

in Figure 2.7 because state transition matrix of the big code is  $(64 \times 64) \times (64 \times 64)$ , and calculating the entire distance spectrum which is required for the TSB is not computationally feasible.

Figure 2.8 shows the bit error rates when trellis based decoder is used for convolutional codes while binary Gaussian elimination decoder is used for an LDGM code for data bits of length 100. The LDGM code is based on the structure given in [21] with a  $100 \times 200$  parity check matrix  $\mathbf{H}$  where degrees of check nodes are equal to 5. It has 100 variable nodes of degree 4 and the rest are of degree 1. We have  $\mathbf{H} = [\mathbf{P}_{100 \times 100} | \mathbf{I}_{100 \times 100}]$  whose generators encode random bits and  $\mathbf{G} = [\mathbf{I}_{100 \times 100} | \mathbf{P}_{100 \times 100}^T]$  whose generators encode data bits. Figure 2.8 shows that using the more complicated generators for random and data bits (higher memory sizes) improves the resulting error correction capabilities for high SNRs. Furthermore, considering existing decoding methods for the randomized encoding scheme, we obtain a better performance using convolutional codes for lower SNRs.

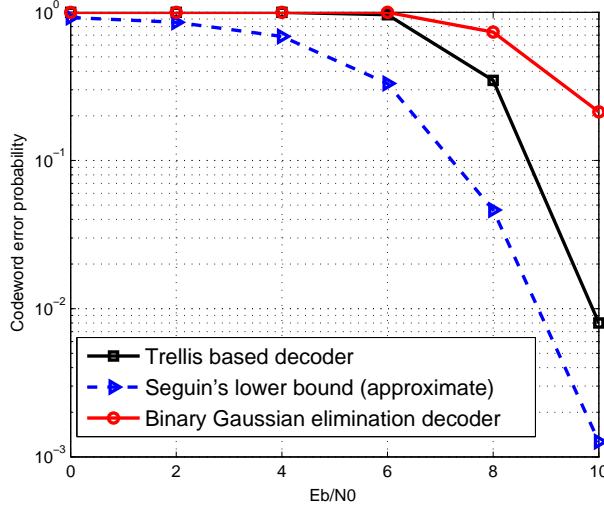


Figure 2.7: Effect of increasing memory size on the performance of the sub-optimal decoders using convolutional codes  $[117 \ 155]$  and  $[133 \ 171]$  with memory size  $m = 6$ . The length of the codewords is 212 and there are  $2^{100}$  cosets each of which represents a unique message and contains  $2^{100}$  codewords.

### 2.6.2 Noisy Main Channel

We now assume that both the main and wiretapper channels are noisy. Therefore, the generators for random and data bits should be selected in a way that result in low security gaps. We consider the channels in this part to be AWGN, and for the case of BSC we provide an example in the next subsection. In Section 2.3.4, we described how to reduce the number of codewords in each coset in randomized encoding scheme to obtain a big code with larger minimum distances to improve the decoding performance of the minimum distance decoder. To evaluate the performance of the proposed randomized convolutional coding solution, we show the BER at the eavesdropper ( $P_{eve}^{min}$ ) as a function of security gap in Figure 2.9 where convolutional code  $[657 \ 435]$  of rate  $1/2$  is used for data bits and a subset of its dual for random bits. Specifically, we use the following generator with rate  $1/8$  and memory 4 to encode the random bits

$$\begin{bmatrix} D^3 + 1 & D^4 + 1 & D^4 + D^3 + D^2 & D^4 + D^3 + D + 1 & D^3 + D^2 + D & D^3 \\ & & & & D^3 + D^2 & D^3 + D^2 + 1 \end{bmatrix} \quad (2.31)$$

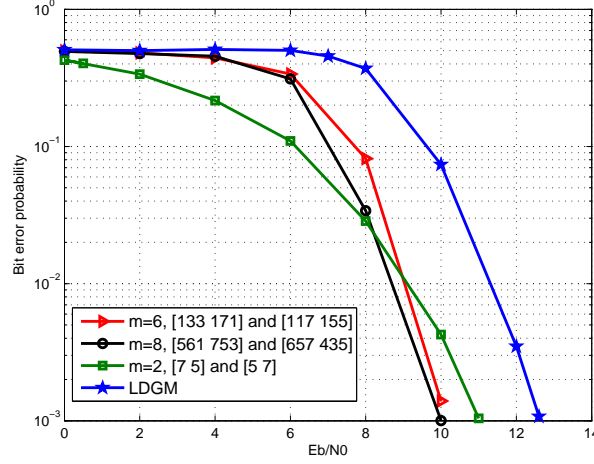


Figure 2.8: Bit error probability for 3 convolutional codes with different memory sizes ( $m$ ) and an LDGM code. The number of cosets is  $2^{100}$  each contains  $2^{100}$  codewords ( $k = 100, r = 100$ ). Length of the LDGM code is 200 and length of convolutional codes are  $200 + 2m$ .

which is obtained using section 2.3.3 and equation (2.11) with the following choice of  $\mathbf{T}^{-1}(D)$

$$\mathbf{T}^{-1}(D) = \begin{bmatrix} 1 & D^2 & 1 & 1 \\ D & 1 & 1 & 1 \\ 1 & 1 & 1 & D \\ 1 & 1 & D & 1 \end{bmatrix}. \quad (2.32)$$

We note that the resulting big code has a minimum distance of 8. One can increase this distance to 10 by using the following encoder of rate  $1/8$  and memory 5 for random bits (which is another subset of dual of [657 435])

$$\begin{bmatrix} D^4 + D^3 + 1 & D^3 + 1 & D^5 + D^4 + D^2 + D + 1 & D^4 & D^5 + D^4 + D^3 + D^2 \\ D^4 + D^2 + 1 & D^5 + D^4 + D^3 + D^2 + D + 1 & D^5 + D^4 + D^2 + D \end{bmatrix}. \quad (2.33)$$

Table 2.1 summarizes several important results from Figure 2.9 and shows that a security gap of 4 dB is achievable for  $P_{main}^{max} \approx 10^{-5}$  and  $P_{eve}^{min} \geq 0.48$  by choosing [657 435] as data bit sequence encoder and (2.33) as the random bit sequence encoder. For similar  $P_{main}^{max}$  and  $P_{eve}^{min}$  values, it is reported in [10] that alternative methods based on puncturing a (770, 385) LDPC code and scrambling a (511, 385) BCH code need a security gap of 5 and 3 dB, respectively. We

note that in order to achieve smaller security gaps in the randomized encoding scheme, it is possible to use turbo codes which are obtained via parallel or serial concatenation of two convolutional codes extending the convolutional coding setup proposed here (which is left as future work). In Figure 2.10, we use the

Table 2.1: Important results of Figure 2.9 when  $P_{main}^{max} \approx 10^{-5}$  and data bits encoder is [657 435].  $k$  and  $r$  denotes number of data and random bits, respectively, and  $n$  is length of the codewords.

	Random bits' encoder	$\text{SNR}_{\text{eve}}$	$\text{SNR}_{\text{main}}$	$P_{\text{eve}}^{\min}$
$n = 256, \quad k = 120, \quad r = 28$	Eq. (2.31)	0	5	0.41
$n = 856, \quad k = 420, \quad r = 103$	Eq. (2.31)	0	5	0.47
$n = 2056, \quad k = 1020, \quad r = 510$	Eq. (2.31)	0.5	5	0.48
$n = 2056, \quad k = 1020, \quad r = 510$	Eq. (2.33)	0.5	4.5	0.48

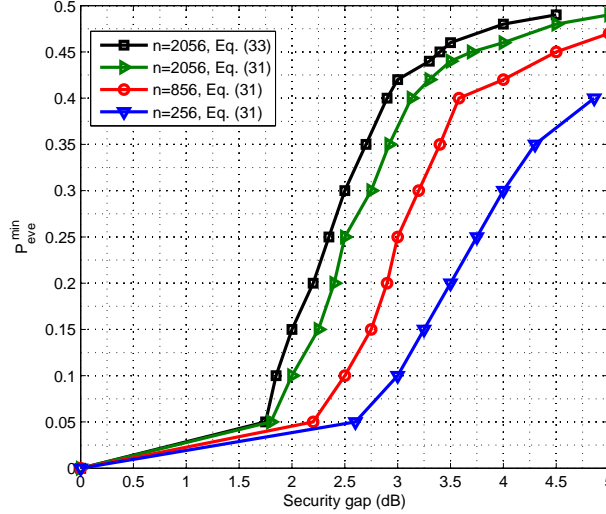


Figure 2.9: Bit error probability of the eavesdropper versus the security gap (at  $P_{main}^{max} \approx 10^{-5}$ ) when convolutional code [657 435] encodes data bits for 3 different codeword lengths and two different random bit encoders in (2.31) and (2.33). Numbers of data and random bits for each curve are provided in Table 2.1.

idea of scrambling data bits introduced in [10] along with the proposed randomized encoding scheme. The main idea in scrambling is to multiply a non-singular  $k \times k$  binary scrambling matrix  $\mathbf{S}$  with the information vector  $\mathbf{u}$  of length  $k$  before multiplying  $\mathbf{u}$  with the generator matrix  $\mathbf{G}_{k \times n}$  corresponding to a linear code in the encoding process. Authors in [10] assume that both the legitimate receiver and eavesdropper know  $\mathbf{S}$  completely and multiply the decoded sequence with

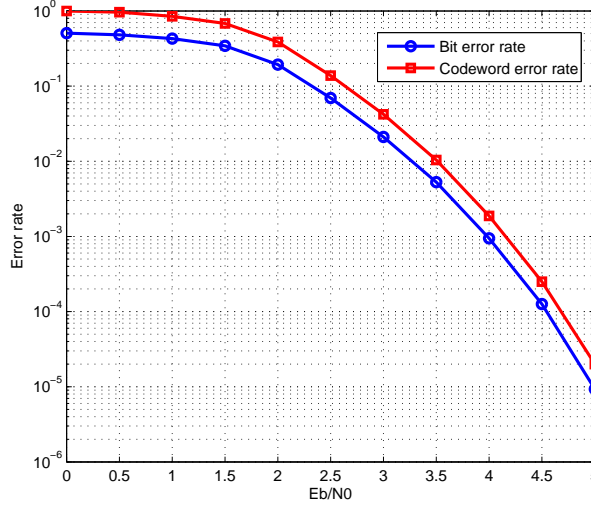


Figure 2.10: Performance of randomized encoding scheme along with scrambling. Perfect scrambling which is defined in [10] has been used here where  $n = 256$ ,  $k = 120$ ,  $r = 27$ .

$\mathbf{S}^{-1}$  to obtain the message bits. A suitable choice of  $\mathbf{S}$  makes the bit error rate close to  $1/2$  even when there is a single error bit in decoded sequence. The codes used in Figure 2.10, are the same as the ones in Example 2 which produces a big code with minimum distance 10. Figure 2.10 shows that the scrambling approach makes the bit error rate at 0 dB approximately  $1/2$  even for a small code of length 256. Hence, for  $P_{main}^{max} \approx 10^{-5}$ , we achieve the security gap of 5 dB with a code of length 256.

### 2.6.3 Application of List Decoding in the Randomized Encoding Scheme

We now investigate the performance of LVA described in Section 2.4.2.3 in the randomized coding setup with a particular list size ( $L = 40$ ). For this part, the generator for data bit sequence is  $[117 \ 155]$  of memory  $m = 6$  and rate  $R = 1/2$  while random bits gets encoded through a subset of its dual,  $[133 \ 171]$ , with

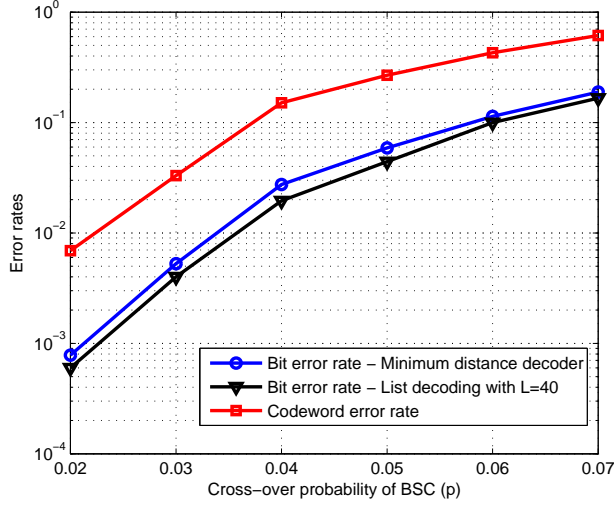


Figure 2.11: Performance of List Viterbi decoding Algorithm for the randomized encoding scheme over a binary symmetric channel. There are  $2^{96}$  cosets each of which consisting of  $2^{28}$  codewords of length 204.

$m = 5$  and rate  $R = 1/6$

$$\begin{bmatrix} 1 + D + D^2 + D^3 + D^5 & 1 + D + D^2 + D^4 + D^5 & D + D^2 + D^3 + D^4 + D^5 \\ 1 + D + D^4 & 1 + D + D^2 + D^3 + D^5 & 1 + D + D^2 + D^4 + D^5 \end{bmatrix} \quad (2.34)$$

which is obtained using section 2.3.3 and equation (2.11) with the following choice of  $\mathbf{T}^{-1}(D)$

$$\mathbf{T}^{-1}(D) = \begin{bmatrix} 1 + D^2 & D & D \\ D^2 & 1 + D^2 & 1 \\ D + D^2 & D & 1 + D \end{bmatrix}. \quad (2.35)$$

The resulting error rate performance is illustrated in Figure 2.11 where it can be seen that while LVA does not improve the codeword error probability (at least in a noticeable manner), there is an improvement in the resulting BERs. This is explained by noting that almost all the  $L$  nominated codewords in LVA belong to different cosets (as observed through extensive simulations); hence, incorporating them into (2.17) or (2.19) results in the same performance as the minimum distance decoder. However, the LVA is useful to improve the bit error rates compared to the minimum distance decoder (using (2.23)).



## 2.7 Chapter Summary

In this chapter, we have applied the convolutional codes to the randomized encoding scheme and argued that how dual of a convolutional code plays a crucial role for this purpose. Furthermore, We described how dual of a convolutional code can be obtained. By introducing a design metric, we discussed that how a convolutional code and its dual shall be chosen to result in a low security gaps which we have used a measure of security.

We proposed optimal and practical sub-optimal decoder for AWGN and BSC channels. Moreover, we provided theoretical lower and upper bounds on performance of the decoders in the wiretap channel by extending existing bounds on the ML decoding performances. Application of list decoding was also studied in this chapter as a way to improve the performance of the sub-optimal decoders.

The resulting security gaps obtained by using randomized convolutional codes outperform some existing coding methods for the wiretap channel like LDPC puncturing method [9]. However, in general scrambling coding scheme [10] provides lower security gaps in comparison to the randomized convolutional codes. In the next chapter, we will utilize concatenated codes to come up with coding schemes achieving lower security gaps.

## Chapter 3

# Concatenated Codes for the Wiretap Channel

In Chapter 2, we studied application of convolutional codes to the randomized encoding scheme for physical layer security and demonstrated that the resulting security gaps compete favorably with the ones offered by result from some other existing coding methods in the literature. A natural question that arises here is whether it is possible to propose other randomized coding schemes to achieve smaller security gaps. In this chapter, we explore several such possibilities by studying concatenated codes and applying them to the randomized encoding scheme of Wyner.

Concatenated codes are used in order to achieve better performance in terms of decoding error probabilities. Two important types of code concatenation which have been widely used in the coding theory literature are serial and parallel concatenation of convolutional codes known as turbo codes which proved to have a near Shannon limit error correcting capabilities. Technically, one can concatenate any two (or more) arbitrary codes, however, only concatenation of certain codes result in tremendous error-correction improvements. The invention of turbo codes has led to extensive research efforts on a broad area of iteratively decodable codes and iterative receiver processing. These codes have a sharp slope in their bit

error rate behaviors, and as a result, they are potential candidates for physical layer security. With this primary motivation, in this chapter, we consider the application of concatenated convolutional codes to physical layer security.

The chapter is organized as follows: we review parallel and serially concatenated convolutional codes in Section 3.1. We study serial concatenation of an LDGM code with a convolutional code in Section 3.2. Then, we study how these codes can be applied to the randomized encoding scheme in Section 3.3. And finally, we provide a short chapter summary in Section 3.4.

## 3.1 Review of Parallel and Serial Concatenated Convolutional Codes

### 3.1.1 Encoding

Berrou *et al.* discovered turbo codes in their ground-breaking paper in 1993 [34]. These codes are random-like codes and are able to achieve exceptionally good error correcting performance. In fact, for information block lengths larger than  $10^4$ , turbo codes with iterative decoding can achieve bit error rates (BERs) as low as  $10^{-5}$  within 1 dB from the Shannon's limit over an AWGN channel [35].

Parallel concatenation of two recursive systematic convolutional (RSC) codes is known as parallel concatenated convolutional codes (PCCCs) whose block diagram is depicted in Figure 3.1. Information sequence denoted by  $\mathbf{u}$  is encoded, and parity sequences  $\mathbf{p}$  and  $\mathbf{q}$  are produced. The two RSC codes are of rate  $1/2$ , separated by a  $K$ -bit interleaver. Though not essential, the RSC codes in Figure 3.1 are taken to be the same with the generator  $G(D) = [1 \ \frac{g_2(D)}{g_1(D)}]$ . We assume the information sequence is of length  $K$  so the resulting codewords are of length  $3K$  which makes the transmission rate  $1/3$ . However, puncturing mechanisms can be employed in order to increase the resulting code rate.

Benedetto *et al.* proposed serially concatenated convolutional codes (SCCC) depicted in Figure 3.2 in [36]. Here, the first RSC code (abbreviated as RSC1) encodes the information sequence, i.e.,  $u_k$ 's where  $1 \leq k \leq K$ . The resulting codeword gets permuted, and then it is fed to the second RSC code (RSC2) to generate the final codeword. So, the overall transmission rate is the multiplication of the individual rates corresponding to each constituent code. It is noted in [39] that the turbo decoding gain can still be obtained even if the outer code in this scheme is not recursive.

The role of interleaver in both PCCC and SCCC is to permute the incoming bit sequence in a pseudo-random fashion in order to assure that a sequence which results in a low weight codeword in one of the RSC codes is unable to do the same for the other one. One can use one of the well-known interleavers which offers such behavior, i.e., the  $S$ -random interleaver [37]. This interleaver separates any two input bits whose positions are within  $S$  bits of each other by an amount greater than  $S$ . For this interleaver to be effective,  $S$  should be selected as large as possible for a given value of  $K$ .

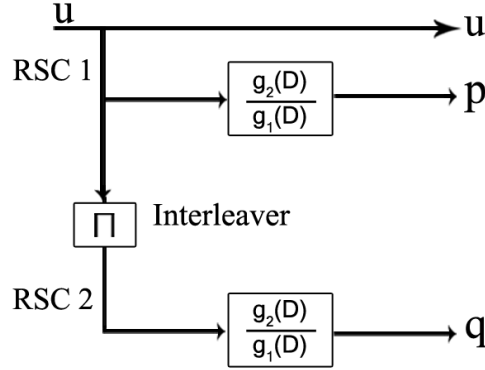


Figure 3.1: The encoder for a parallel-concatenated convolutional code (PCCC).

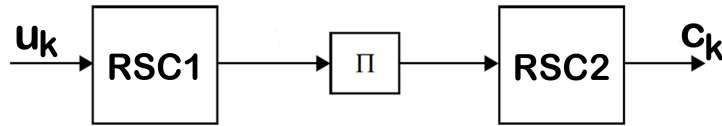


Figure 3.2: The encoder for a serially concatenated convolutional code (SCCC).

### 3.1.2 Decoding

Concatenated codes are decoded using soft-input soft-output (SISO) decoders and an iterative process. For the case of PCCC and SCCC, the SISO decoders work based on the BCJR algorithm, and iterative process involves passing soft information between these constituent decoders.

#### 3.1.2.1 Overview of the BCJR Algorithm

We now briefly discuss the main points of BCJR decoding algorithm introduced by Bahl, Cocke, Jelinek and Raviv [38] as a bit-wise maximum a posteriori probability (MAP) decoder in 1974. Denoting the posteriori probability (APP) of each information bit  $u_l$  by  $P(u_l|\mathbf{y})$ <sup>1</sup> where  $\mathbf{y}$  is the received noisy vector, the bit-wise MAP decoding criterion is given by

$$\hat{u}_l = \underset{u_l}{\operatorname{argmax}} P(u_l|\mathbf{y}) \quad (3.1)$$

We describe the logarithmic BCJR which calculates the log likelihood ratio (LLR)

$$L(u_l) = \log \left[ \frac{P(u_l = 1|\mathbf{y})}{P(u_l = 0|\mathbf{y})} \right] \quad (3.2)$$

and consequently the estimated bit is obtained by

$$\hat{u}_l = \begin{cases} 1, & L(u_l) > 0 \\ 0, & L(u_l) < 0 \end{cases} \quad (3.3)$$

Similar to the Viterbi algorithm, the BCJR algorithm works based on branch and path metrics. Every received bit cause the trellis to move from one state  $s'$  to another state  $s$ , where  $s, s' \in \mathcal{S}$  and  $\mathcal{S}$  denotes the set of all possible states. From this trellis point of view, using Bayes' rule and total probability theorem,

---

<sup>1</sup>Throughout this chapter,  $P$  denotes probability and  $p$  is used to refer to the probability density function.

we may write  $L(u_l)$  as

$$L(u_l) = \log \left[ \frac{\sum_{\mathcal{U}_l} p(s_{l-1} = s, s_l = s', \mathbf{y})}{\sum_{\mathcal{U}_0} p(s_{l-1} = s, s_l = s', \mathbf{y})} \right] \quad (3.4)$$

where  $\mathcal{U}_j$  is the set of pairs  $(s', s)$  for the state transitions  $(s_{l-1} = s') \rightarrow (s_l = s)$  ( $s_l$  denotes the encoder state at time  $l$ ) whose corresponding input labels are  $j$ . So the problem of computing LLRs boils down to how to compute  $p(s_{l-1} = s, s_l = s', \mathbf{y})$  terms. These terms can be factorized in the following manner

$$p(s_{l-1} = s, s_l = s', \mathbf{y}) = \alpha_{l-1}(s') \gamma_l(s', s) \beta_l(s) \quad (3.5)$$

where  $\alpha$ ,  $\beta$  and  $\gamma$  values are defined as (with the notation  $\mathbf{y}_a^b = [y_a, y_{a+1}, \dots, y_b]$ )

$$\begin{aligned} \alpha_l(s) &= p(s_l = s, \mathbf{y}_1^l), \\ \gamma_l(s', s) &= p(s_l = s, y_l | s_{l-1} = s'), \\ \beta_l(s) &= p(\mathbf{y}_{l+1}^L | s_l = s). \end{aligned} \quad (3.6)$$

Moreover, by several applications of Bayes' rule and the total probability theorem  $\alpha_l(s)$  and  $\beta_l(s)$  are computed using backward and forward recursions

$$\begin{aligned} \alpha_l(s) &= \sum_{s'} \gamma_l(s', s) \alpha_{l-1}(s'), \\ \beta_{l-1}(s') &= \sum_s \beta_l(s) \gamma_l(s', s). \end{aligned} \quad (3.7)$$

These are initialized by assuming that the trellis starts from and ends with the all-zero state (by using termination bits)

$$\alpha_0(s) = \beta_L(s) = \begin{cases} 1, & s = 0, \\ 0, & s \neq 0, \end{cases} \quad (3.8)$$

and the branch metric  $\gamma_l(s', s)$  is computed through

$$\begin{aligned} \gamma_l(s', s) &= p(s, y_l | s') \\ &= \frac{P(s, s')}{P(s')} \cdot \frac{p(s, s', y_l)}{P(s, s')} \\ &= P(s | s') p(y_l | s, s') \\ &= P(u_l) p(y_l | u_l) \end{aligned} \quad (3.9)$$

where  $u_l$  denotes the input label corresponding to the transition  $s \rightarrow s'$ .  $p(y_l|u_l)$  is determined based on the transmission channel is being used, for instance, for an AWGN channel

$$p(y_l|u_l) = \frac{1}{2\pi\sigma^2} \exp\left[-\frac{\|y_l - c_l\|^2}{2\sigma^2}\right], \quad (3.10)$$

and for a BSC, we have

$$P(y_l|u_l) = (1-p)^{(n-d_H(y_l, c_l))} p^{d_H(y_l, c_l)}, \quad (3.11)$$

where  $c_l$  denotes the output label corresponding to the transition  $s \rightarrow s'$ ,  $\sigma^2 = N_0/2$ ,  $p$  denotes the transition probability of the BSC, and  $d_H$  is the Hamming distance. We are now able to compute (3.5) and consequently  $L(u_l)$  in (3.4). However, since these numerical computations may not be stable, and may even diverge, in practice, the log version of BCJR is used with the following mappings

$$\begin{aligned} \tilde{\alpha}_l(s) &= \log(\alpha_l(s)) \\ &= \log \sum_{s'} \exp(\tilde{\alpha}_{l-1}(s') + \tilde{\gamma}_l(s', s)), \\ \tilde{\beta}_{l-1}(s') &= \log(\beta_{l-1}(s')) \\ &= \log \sum_s \exp(\tilde{\beta}_l(s) + \tilde{\gamma}_l(s', s)), \\ \tilde{\gamma}_l(s', s) &= \log \gamma_l(s', s) \end{aligned} \quad (3.12)$$

and the initialization rules in (3.8) are converted to

$$\tilde{\alpha}_0(s) = \tilde{\beta}_L(s) = \begin{cases} 0, & s = 0, \\ -\infty, & s \neq 0 \end{cases} \quad (3.13)$$

Finally,  $L(u_l)$  is computed using

$$\begin{aligned} L(u_l) &= \log \left[ \frac{\sum_{\mathcal{U}_1} \alpha_{l-1}(s') \gamma_l(s', s) \beta_l(s)}{\sum_{\mathcal{U}_0} \alpha_{l-1}(s') \gamma_l(s', s) \beta_l(s)} \right] \\ &= \log \left[ \sum_{\mathcal{U}_1} \exp(\tilde{\alpha}_{l-1}(s') + \tilde{\gamma}_l(s', s) + \tilde{\beta}_l(s)) \right] - \log \left[ \sum_{\mathcal{U}_0} \exp(\tilde{\alpha}_{l-1}(s') + \tilde{\gamma}_l(s', s) + \tilde{\beta}_l(s)) \right] \end{aligned} \quad (3.14)$$

where  $\mathcal{U}_j$ 's are defined earlier. It is clear that the constant terms in  $\tilde{\gamma}_l(s', s)$  can be ignored in this algorithm since they contribute the same value to all the branches and will not have a role in final decision.

We emphasize that (3.14) provides LLR information on the input bit at time  $l$ . Similar information can be obtained for the output bits from (3.14) as well by choosing the suitable sets for the sums to be used instead of  $\mathcal{U}_1$  and  $\mathcal{U}_0$ , for example, let us denote the second output at time  $k$  with  $q_k$ , then  $L(q_k)$  is readily obtained from (13) by using  $\mathcal{Q}_1$  and  $\mathcal{Q}_0$  instead of  $\mathcal{U}_1$  and  $\mathcal{U}_0$ , respectively, where  $\mathcal{Q}_j$  denotes the set of pairs  $(s', s)$  for the state transitions  $(s_{l-1} = s') \rightarrow (s_l = s)$  whose second output bit is  $j$ . Furthermore, initialization rules in 3.13, are based on the assumption that trellis ends with the all zero-state. For non-recursive convolutional codes, this could be simply achieved by adding enough zero bits to the end of input sequence. However, for the case of RSC codes, this approach does not necessarily force the corresponding trellis to the all-zero state and one need to obtain suitable terminating bits for this purpose based on the input sequence.

### 3.1.2.2 Iterative Decoder

We now describe iterative decoder for the PCCC and SCCC proposed in [34] and [36]. For the case of PCCC, a schematic of the decoder is presented in Figure 3.3. We denote the received vector with  $\mathbf{y} = [y_1, y_2, \dots, y_K]$  where  $y_k \triangleq [y^{u_k}, y^{p_k}, y^{q_k}]$  and  $y^{u_{\Pi,k}}$  is the interleaved version of  $y^{u_k}$ . Basically, there are two RSC codes in PCCC for each of which BCJR algorithm needs to be employed. One main difference is about the a priori information. The BCJR algorithm described in Section 3.1.2.1, has no prior information on a specific bit  $u_l$ , hence it assumes  $P(u_l) = 1/2$ . However, each BCJR decoder in Figure 3.3 provides a priori information for the other one, which is known as *extrinsic information* and is denoted by  $L_{21}^e$  and  $L_{12}^e$ . This would indeed affect the calculation of the branch metrics  $\gamma_l(s', s)$  in (3.9).

For calculation of the extrinsic information, it is important to pass only the information which is unknown at the other decoder. This is done by changing the branch metrics  $\gamma_l(s', s)$  accordingly, for instance, assume that  $p(u_l)$  denotes the extrinsic information coming from the decoder corresponding to RSC1, then  $p(u_l)$  must be dropped in  $\gamma_l(s', s)$  for computation of the extrinsic information which goes to the RSC1 decoder since it is already known at RSC1.



For the SCCC case, the iterative decoder is shown in Figure 3.4. We assume that the output codeword of RSC2 and RSC1 in Figure 3.2 are of form  $\mathbf{c}_2 = [v_1, q_1, v_2, q_2, \dots, v_K, q_K]$  and  $\mathbf{c}_1 = [u_1, p_1, u_2, p_2, \dots, u_K, p_K]$ , respectively, and the received vector after the channel is denoted by  $\mathbf{y} = [y^{v_1}, y^{q_1}, y^{v_2}, y^{q_2}, \dots, y^{v_K}, y^{q_K}]$ . In this case, the channel observations are for RSC2 only. Therefore, the BCJR algorithm on the trellis of RSC1 works merely based on the extrinsic information it receives from RSC2. In this case,  $\tilde{\gamma}_l(s', s)$  for RSC1 becomes

$$\tilde{\gamma}_l(s', s) = \log(P(u_k)) + \log(P(p_k)). \quad (3.15)$$

Moreover, RSC1 needs to send extrinsic information on  $u_k$ 's and  $p_k$ 's to RSC2 which can be done by equating  $\tilde{\gamma}_l(s', s)$  to  $\log(P(p_k))$  and  $\log(P(u_k))$ , respectively, and using the method introduced in Section 3.1.2.1.

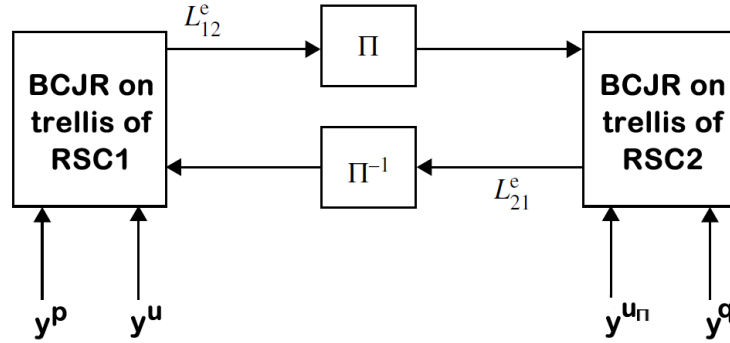


Figure 3.3: Iterative decoding for a PCCC.

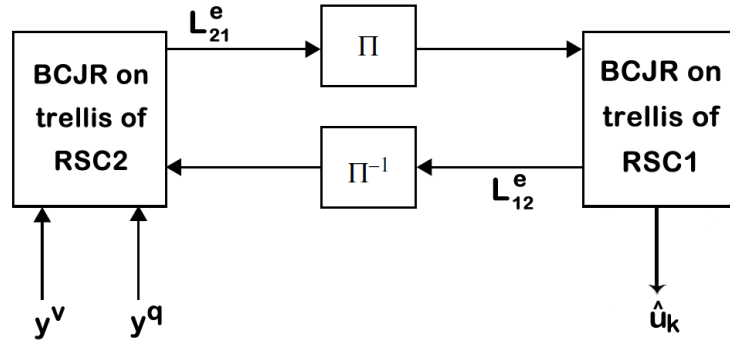


Figure 3.4: Iterative decoding for a SCCC.

Figure 3.5 shows the performance of PCCCs and SCCC's over an AWGN channel for different codeword lengths where the number of iterations is set to 8. The

constituent encoders are assumed to be  $[1\ 7/5]$  (in octal notation), and  $S$  denotes the parameter of the  $S$ -random interleaver. The trellises corresponding to the first and the outer RSC codes are forced to zero state for the PCCC and the SCCC cases, respectively.

It is clear that by increasing the codeword length, the performance of these schemes gets improved significantly. Furthermore, for similar codeword lengths, SCCCs outperform PCCCs for high SNRs while the situation is reversed in the low SNR regime.

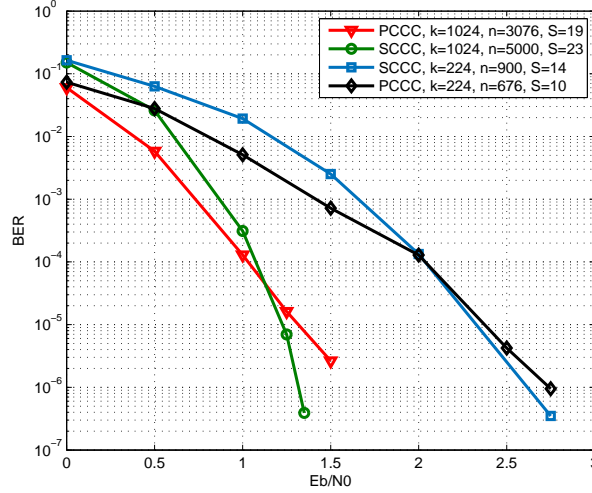


Figure 3.5: Performance of PCCC and SCCC in the AWGN channel where  $n$  and  $k$  denote length of the codewords and data bits, respectively.

## 3.2 Concatenation of LDGM and Convolutional Codes

In this section, we study another form of concatenated codes which consists of a low density generator matrix (LDGM) code and a convolutional code. Our final goal is to apply these codes to the randomized encoding scheme and evaluate their performance in terms of the resulting security gap. First, we will take a closer look at the LDGM codes.

Low density parity check (LDPC) codes invented by Gallager in [20] have a sparse parity check matrix and are considered extremely efficient in terms of their decoding algorithms which are based on the message passing method. This method exploits the sparse nature of the parity-check matrices to achieve very good performance with very low complexity. However, the generator matrix of LDPC codes is usually dense. In the context of physical layer security, this is a problem since the dual of the LDPC code needs to be considered as well. Some approaches have been proposed in the literature which try to exploit the sparse nature of the parity-check matrix also in the encoding stage. This can be achieved if, for instance, the parity check matrix has a sparse representation in lower triangular form [40]. Moreover, for the cases where this form does not hold, using only row and column permutations can result in an approximately lower triangular parity check matrices [40].

Alternatively, one can use LDGM codes [41] which have a sparse generator matrix (in addition to a sparse parity check matrix) as a very efficient way to reduce significantly the amount of calculations needed at the encoder. LDGM code performance is studied in [19] where the authors point out that since they have variable nodes with degree 1 in their Tanner graphs, they exhibit high error floors. On the other hand, they show that serial concatenation of two or more LDGM codes can reduce such floors significantly at the cost of increased complexity.

Quasi-cyclic low-density parity-check (QC-LDPC) codes have their parity-check and generator matrices formed by circulant blocks. The structure which allows for the use of simple encoding circuits, based on shift registers, that exploit the quasi-cyclic nature of the codes [43]. An interesting family of QC-LDPC codes are based on difference families [42] and have circulant blocks with row (column) weights greater than one. The structure of the parity-check matrix of these codes with length  $N$  consists of a row of sparse circulant blocks as follows:

$$\mathbf{H} = [\mathbf{H}_0 | \mathbf{H}_1 | \dots | \mathbf{H}_{N_b-1}] \quad (3.16)$$

where each  $\mathbf{H}_i$  is of size  $(N/N_b) \times (N/N_b)$ . A circulant matrix  $\mathbf{H}$  is defined over

the Galois field of order  $p$ ,  $GF(p)$ , as follows:

$$\mathbf{H} = \begin{bmatrix} h_0 & h_1 & h_2 & \dots & h_{n-1} \\ h_{n-1} & h_0 & h_1 & \dots & h_{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_1 & h_2 & h_3 & \dots & h_0 \end{bmatrix} \quad (3.17)$$

where  $h_i \in GF(p)$ ,  $i = 0, 1, \dots, n-1$ . In fact  $\mathbf{H}$  is described by one of its rows and others is obtained by cyclically shifting that row. We will denote the number of nonzero symbols in each row (or column) of  $\mathbf{H}$  with  $W[\mathbf{H}]$ . Table 3.1 provides three practical examples of such codes with the corresponding minimum distances ( $d_{min}$ ) which will be used later on.

Provided that at least one of the blocks is of full rank, the code rate is  $(N_b - 1)/N_b$ . A low density generator matrix for the codes having parity check matrix of the form (3.16) can be found easily if one of the blocks  $\mathbf{H}_i$ 's is an identity matrix. For instance, if  $\mathbf{H}_{N_b-1} = \mathbf{I}_{(N/N_b \times N/N_b)}$ , then

$$\mathbf{G} = \begin{bmatrix} & & & \mathbf{H}_0^T \\ & & & \mathbf{H}_1^T \\ & & & \vdots \\ \mathbf{I}_{(N(N_b-1)/N_b \times N(N_b-1)/N_b)} & & & \mathbf{H}_{N_b-2}^T \\ & & & \mathbf{H}_{N_b-1}^T \end{bmatrix} \quad (3.18)$$

where  $\mathbf{I}$  denotes the identity matrix and superscript  $^T$  denotes transposition. However, this would be detrimental on the error correcting capabilities of the code, since the resulting code has a smaller minimum distance.

In this work, we will use LDGM codes with the generator and parity check matrices given in (3.18) and (3.16), respectively. This would make such codes useful for the randomized encoding scheme, since  $\mathbf{G}$  and  $\mathbf{H}$  are duals of each other, and if they are not pseudo-self-dual (as defined in the previous chapter), they will provide us with two sets of generators, i.e., the rows of  $\mathbf{G}$  and the rows of  $\mathbf{H}$ , which can be used for encoding random bits and data bits, respectively.

Table 3.1: Three instances of QC-LDPC code with the structure given in (3.16).

Code $(N, K)$	$n$	$W[\mathbf{H}_i], i = 0, 1, 2, \dots, N_b - 1$	$d_{min}$
(2560, 2048)	512	$\{6, 6, 6, 6, 1\}$	7
(1880, 1504)	376	$\{5, 5, 5, 5, 1\}$	6
(1248, 936)	312	$\{5, 5, 5, 1\}$	6

### 3.2.1 LDGM-RSC Concatenated Codes

Serial concatenation of an LDGM code with an RSC code is illustrated in Figure 3.6. Data bits  $u_i$ 's are first encoded by the LDGM code and the resulting codeword is fed into a recursive systematic convolutional code. The codeword transmitted through the channel is denoted by  $\mathbf{c} = [c_1, c_2, \dots, c_K]$  where  $c_k \triangleq [v_k, q_k]$ . In this scheme, there is no need for an interleaver between the two constituent encoders since the Tanner graph for the LDGM code acts like an interleaver. The resulting transmission rate is the multiplication of the rates corresponding to each encoder. We note that the structure given in the last section usually results in high rate LDGM codes ( $R \geq 0.8$ ). If we consider an RSC code of rate  $1/2$ , the overall code is of rate would be  $R \geq 0.4$ .

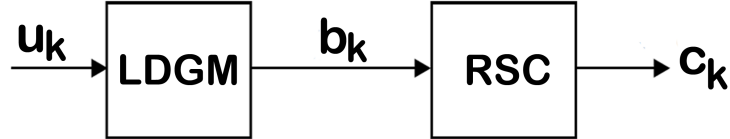


Figure 3.6: The encoder for the serial concatenation of a LDGM code with an RSC code.

By denoting the received noisy vector by  $\mathbf{y} = [y_1, y_2, \dots, y_K]$  where  $y_k \triangleq [y_k^v, y_k^q]$ , we can illustrate the corresponding iterative decoder in Figure 3.7. The channel observations are used in the BCJR algorithm applied to the trellis of the RSC code and soft information on bits  $b_k$ 's is obtained. We note that  $b_k$ 's are common between the two constituent codes in this scheme, and the extrinsic information terms are denoted by  $L_{12}^e$  and  $L_{21}^e$ .

The decoder for an LDGM code is the well-known belief propagation (BF)

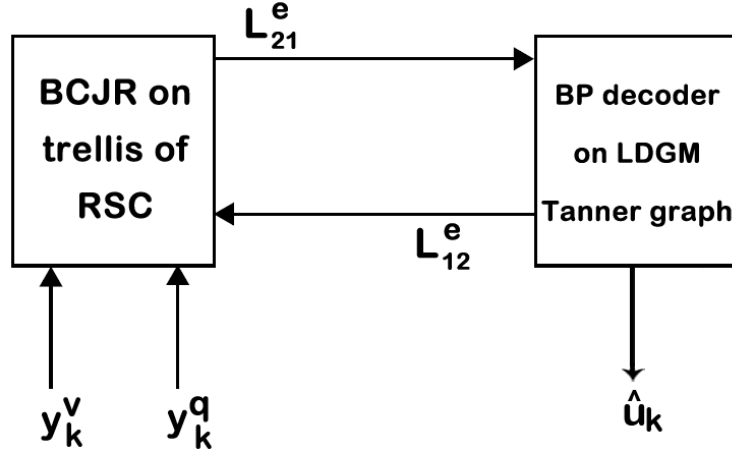


Figure 3.7: The iterative decoder for the serial concatenation of a LDGM code with an RSC code.

decoder which is initialized with the information ( $L_{21}^e$ ) provided by the BCJR algorithm on trellis of the RSC code. The LLRs are passed directly to the check nodes of the LDGM code, and then the extrinsic information comes back to each variable node. The resulting soft information is fed into the BCJR algorithm as  $L_{12}^e$  for another round of iteration. After a desired number of iterations, the final decisions on the information bits  $\hat{u}_k$ 's are made.

Figure 3.8 illustrates the performance of this coding scheme over an AWGN channel where the number of iterations is set tot 10. The RSC code  $[1\ 7/5]$  of rate  $1/2$  is used along with the LDGM codes described in Table 3.1. We note that trellis termination is enforced by adding zero bits to the end of the input sequence. We observe that the error rate results exhibit error floors which is mainly because of the structure of the LDGM codes employed here.

### 3.3 Code Concatenation for the Randomized Encoding Scheme

In this section, we apply the three concatenated codes discussed in the previous section, i.e., PCCCs, SCCCs and LDGM-RSC codes, to the randomized encoding scheme. From Chapter 2, one recalls that this scheme needs two encoders, one for

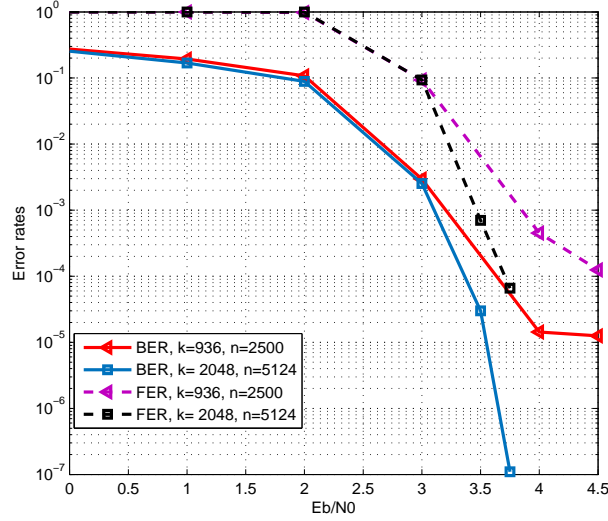


Figure 3.8: Performance of LDGM-RSC scheme in the AWGN channel where  $n$  and  $k$  denote length of the codewords and data bits, respectively.

encoding random bits and the other for data bits. Hence, based on the discussion in Chapter 2, we need to obtain the dual of the concatenated code to be employed. We also propose an iterative decoder for each case and evaluate performance of the randomized coding schemes in terms of the resulting security gaps.

### 3.3.1 Randomized PCCCs

#### 3.3.1.1 Encoding

Figure 3.9 illustrates a PCCC where the interleaved version of the input sequence is also considered in the output sequence. We note that conventionally the interleaved version of the input sequence, i.e.,  $u_{\Pi}$ , is not transmitted in PCCC, since it would be an unnecessary overhead. However, we are transmitting  $u_{\Pi}$  in this scheme, since we need to obtain dual of a PCCC, in other words, obtaining dual of a PCCC in the way we describe here (by substituting dual of each RSC code) is not feasible if one does not transmit the  $u_{\Pi}$  sequence. In order to use this code in the randomized encoding scheme, dual of a PCCC needs to be obtained. We emphasize that the PCCC consists of two RSC codes, so its dual is obtained by

substituting the dual of each RSC code, as illustrated in Figure 3.10.

To obtain the dual of an RSC code, first, one should derive the equivalent non-recursive generator by multiplying a suitable polynomial ( $g_1(D)$ ) with its encoder. We note that these two encoders result in the same set of codewords and the only difference is in the way they map messages to codewords. Then, one can use Theorem 2 from Chapter 2 to obtain the dual of the resulting non-recursive code. Finally, an equivalent systematic version of the dual can be obtained through division of the encoder by a suitable polynomial. Particularly, for the RSC codes with a polynomial generator of the form  $G(D) = [1 \quad \frac{g_2(D)}{g_1(D)}]$ , the generator for its dual is obtained as  $G^\perp(D) = [\frac{\tilde{g}_2(D)}{\tilde{g}_1(D)} \quad 1]$  where  $\tilde{g}(D)$  denotes reverse of  $g(D)$ .

We can use one of the encoders in Figures 3.9 and 3.10 to encode the random bits and the other one to encode the data bits. In this scheme, if the codeword length is  $n$ , then the number of data and random bits which equal to the number of cosets and the number of codewords in each coset, respectively, is  $n/4$ ; in other words, the transmission rate is  $1/4$ . We denote the output codeword of PCCC by  $\mathbf{c}_1 = [c_1, c_2, \dots, c_K]$  where  $c_k \triangleq [u_k, p_k, u_{\Pi,k}, q_k]$ , and the output codeword of the dual of PCCC by  $\mathbf{c}_2 = [c'_1, c'_2, \dots, c'_K]$  where  $c'_k \triangleq [p'_k, u'_k, q'_k, u'_{\Pi,k}]$ . The codeword transmitted through the channel is the modulo-2 sum of  $\mathbf{c}_1$  and  $\mathbf{c}_2$ , i.e.,  $\mathbf{c} = \mathbf{c}_1 + \mathbf{c}_2$ .

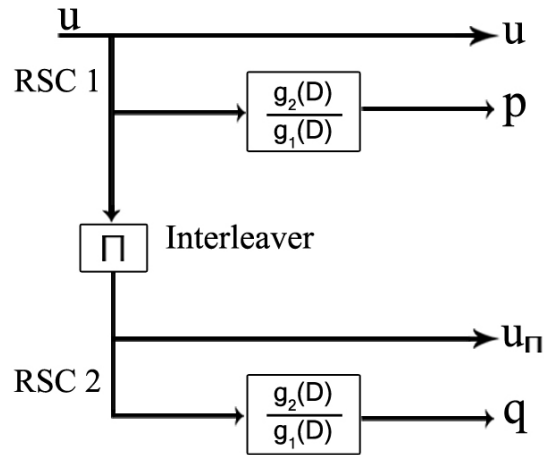


Figure 3.9: The encoder for parallel-concatenated convolutional code (PCCC).



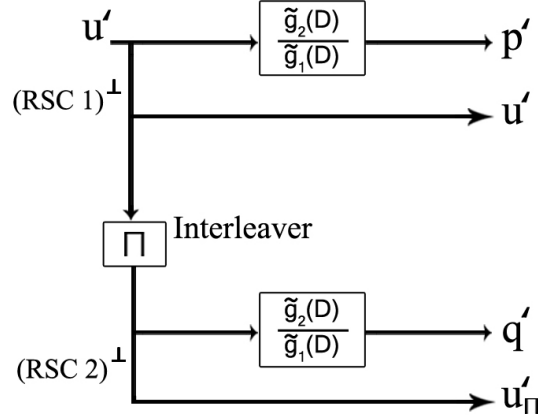


Figure 3.10: The encoder for dual of a PCCC.

### 3.3.1.2 Decoding

Earlier in this chapter, the iterative decoder for a single PCCC was described. We denote the received vector with  $\mathbf{y} = [y_1, y_2, \dots, y_K]$  where  $y_k \triangleq [y^{u_k+p'_k}, y^{p_k+u'_k}, y^{u_{\Pi,k}+q'_k}, y^{q_k+u'_{\Pi,k}}]$ . In this section, we utilize the decoder in Figure 3.3, in order to introduce an iterative decoder for the randomized encoding scheme. For this purpose, we note that the trellises of  $RSC1$  and  $RSC1^\perp$  in Figures 3.9 and 3.10 govern the bit sequence  $[u_1, p_1, u_2, p_2, \dots, u_K, p_K]$  and  $[p'_1, u'_1, p'_2, u'_2, \dots, p'_K, u'_K]$ , respectively. Hence, it is possible to form a big trellis combining them to govern the modulo-2 sum of these two bit sequences, i.e.,  $[u_1 + p'_1, p_1 + u'_1, u_2 + p'_2, p_2 + u'_2, \dots, u_K + p'_K, p_K + u'_K]$ . Having the observations  $y^{u_k+p'_k}$  and  $y^{p_k+u'_k}$ , we are able to compute the joint probabilities for the bits of the input sequences, i.e.,  $u_i$ 's and  $u'_i$ 's applying the BCJR algorithm to the big trellis. By applying the same idea for  $RSC2$  and  $RSC2^\perp$ , we propose the iterative decoder depicted in Figure 3.11 for the randomized encoding scheme. We emphasize that this decoder jointly decodes random and data bits approximating the MAP decoding rule given by

$$(\hat{u}_l, \hat{u}'_l) = \underset{(u_l, u'_l)}{\operatorname{argmax}} P((u_l, u'_l) | \mathbf{y}) \quad (3.19)$$

where  $\mathbf{y}$  is the received codeword and  $(u_l, u'_l) \in \{00, 01, 10, 11\}$ . Joint probabilities  $P((u_l, u'_l)|\mathbf{y})$  are computed using the following equation in the big trellis

$$P((u_l = k, u'_l = j)|\mathbf{y}) = \sum_{\mathcal{U}_{kj}} p(s_{l-1} = s', s_l = s, \mathbf{y}), \quad k, j \in \{0, 1\} \quad (3.20)$$

where  $\mathcal{U}_{kj}$  is the set of pairs  $(s', s)$  for the state transitions  $(s_{l-1} = s') \rightarrow (s_l = s)$  whose corresponding input labels are  $kj$ . Using the BCJR algorithm, such probabilities are computed efficiently by the factorization method described in (3.5). Specifically, using the notation introduced in (3.12)

$$\log(P((u_l = k, u'_l = j)|\mathbf{y})) = \log\left(\sum_{\mathcal{U}_{kj}} \exp(\tilde{\alpha}_{l-1}(s') + \tilde{\gamma}_l(s', s) + \tilde{\beta}_l(s))\right) \quad (3.21)$$

Here, using (3.9) and (3.11), the logarithmic branch metric  $\tilde{\gamma}_l(s, s')$  for the AWGN channel is computed as

$$\tilde{\gamma}_l(s, s') = \log(P_{u_l u'_l}^e) - \frac{\|y_l - c_l\|^2}{2\sigma^2} \quad (3.22)$$

which is used to obtain  $\tilde{\alpha}_l(s')$  and  $\tilde{\beta}_l(s)$  based on (3.12).  $P^e(u_l, u'_l)$  in (3.22) is the extrinsic information being exchanged between the two BCJR decoders in this case. Such information is denoted by  $M_{12}^e$  and  $M_{21}^e$  in Figure 3.11 which are of the form  $[\log(P_{00}^e) \log(P_{01}^e) \log(P_{10}^e) \log(P_{11}^e)]$  where  $P_{kj}^e$  denotes the extrinsic probability that  $(u_l, u'_l) = (k, j)$  satisfying  $P_{00}^e + P_{01}^e + P_{10}^e + P_{11}^e = 1$ . Furthermore, the initial value for either of  $M_{12}^e$  and  $M_{21}^e$  is set to  $[\log(0.25) \log(0.25) \log(0.25) \log(0.25)]$ . Extrinsic information  $P_{kj}^e$  is used to send only the information which is unknown to the other decoder in order to avoid positive feed-back. It is computed by using

$$\tilde{\gamma}_l(s, s') = -\frac{\|y_l - c_l\|^2}{2\sigma^2} \quad (3.23)$$

in the BCJR algorithm.

To sum up, we emphasize that the extrinsic information which is received from the decoder, say corresponding to RSC1 and RSC1<sup>+</sup>, is used to compute  $\tilde{\alpha}_l(s')$  and  $\tilde{\beta}_l(s)$  at the decoder which corresponds to RSC2 and RSC2<sup>+</sup> based on (3.22). However, the latter decoder will not use such information in computing  $\tilde{\gamma}_l(s, s')$ , in other words, it utilizes (3.23) in the BCJR algorithm. After exchanging extrinsic information between the two decoders by a preset number of iterations, the random and data bits can be jointly decoded.

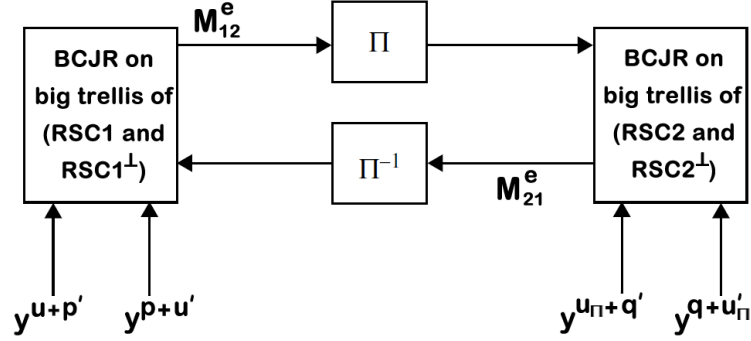


Figure 3.11: Iterative decoder for the randomized encoding scheme where one of the encoders in Figures 3.9 and 3.10 encodes random bits and the other encodes data bits.

### 3.3.1.3 Example

Figure 3.12 illustrates the performance of the iterative decoder described in Figure 3.11 over an AWGN channel.  $S$  denotes the interleaver size, the number of iterations is set to the 10 and the constituent RSC codes are  $[5/7 \ 1]$  and  $[1 \ 5/7]$ , both of rate  $1/2$  where all the trellises terminated at the all-zero state. The randomized PCCC makes the BER very high ( $\geq 0.3$ ) up to some SNR values. On the other hand, BER is very small ( $\approx 10^{-5}$ ) after another SNR value which legitimate receiver is supposed to work with. As one can see in Figure 3.8, a randomized PCCC of length  $10^4$  is able to make the difference between these two SNRs (security gap) approximately 1.8 dB.

As another example, the performance of randomized PCCC over a binary symmetric channel is shown in Figure 3.13 for three different code lengths.

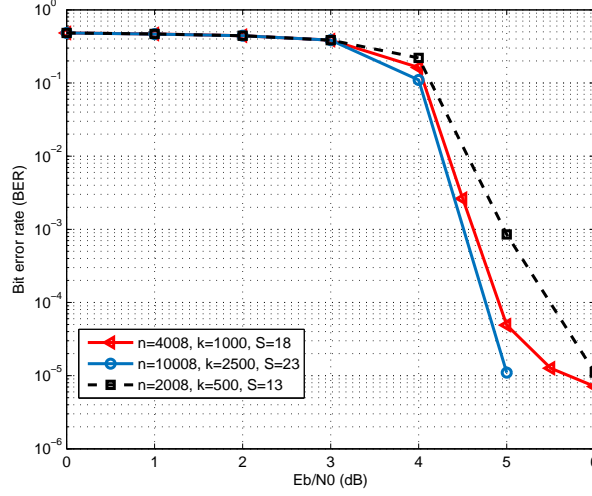


Figure 3.12: Performance of the randomized PCCC in the AWGN channel where  $n$  denotes the length of the codewords and transmission rate is about  $1/4$

### 3.3.2 Randomized SCCCs

#### 3.3.2.1 Encoding

Similar to the PCCC case, we apply SCCC to the randomized encoding scheme as well. Figure 3.14 depicts an SCCC consisting of two RSC codes with generators  $G(D) = [1 \quad \frac{g_2(D)}{g_1(D)}]$ . The dual of the SCCC is obtained by replacing each constituent RSC encoder with its dual, i.e.,  $G^\perp(D) = [\frac{\tilde{g}_2(D)}{\tilde{g}_1(D)} \quad 1]$ , as in Figure 3.15. Hence, we are able to use one of the encoders in Figures 3.14 and 3.15 to encode the random bits and the other one for the data bits. It is clear that without any puncturing, the rates of the encoders are  $1/4$ , and assuming  $\mathbf{c}_1 = [v_1, q_1, v_2, q_2, \dots, v_K, q_K]$  and  $\mathbf{c}_2 = [v'_1, q'_1, v'_2, q'_2, \dots, v'_K, q'_K]$ , the codeword transmitted through the channel is their modulo-2 sum, i.e.,  $\mathbf{c} = \mathbf{c}_1 + \mathbf{c}_2 = [v_1 + v'_1, q_1 + q'_1, v_2 + v'_2, q_2 + q'_2, \dots, v_K + v'_K, q_K + q'_K]$ .

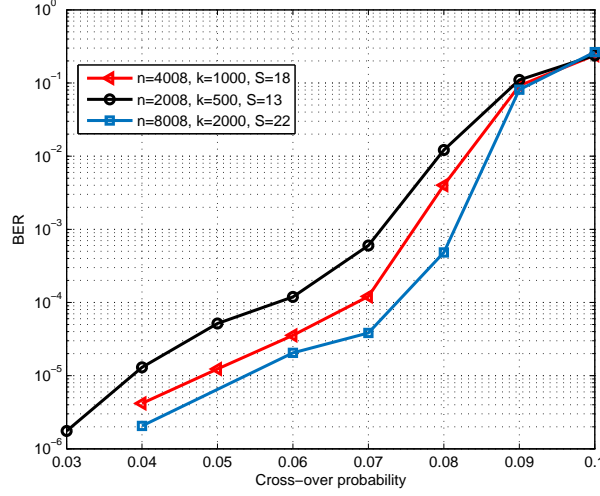


Figure 3.13: Performance of the randomized PCCC in the binary symmetric channel where  $n$  denotes the length of the codewords and transmission rate is about  $1/4$

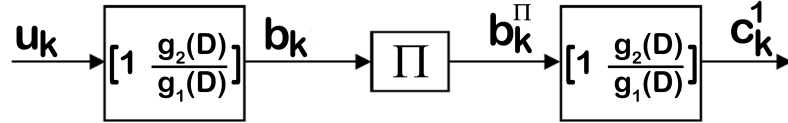


Figure 3.14: The encoder for serial-concatenated convolutional code (SCCC).

### 3.3.2.2 Decoding

We denote the received noisy vector by  $\mathbf{y} = [y_1, y_2, \dots, y_K]$  where  $y_k = [y_k^v, y_k^q] = [y^{v_k+v'_k}, y^{q_k+q'_k}]$ . By extending the decoder introduced for a single SCCC in Section 3.1.2.2, we propose an iterative joint decoder for this scenario which similar to randomized PCCC case, which estimates the best  $(u_l, u'_l)$  pair.

We note that the trellises of  $RSC2$  and  $RSC2^\perp$  in Figures 3.14 and 3.15 govern the bit sequence  $[v_1, q_1, v_2, q_2, \dots, v_K, q_K]$  and  $[v'_1, q'_1, v'_2, q'_2, \dots, v'_K, q'_K]$ , respectively. Hence, one can form a big trellis combining them which governs the modulo-2 sum of these two bit sequences, i.e.,  $[v_1 + v'_1, q_1 + q'_1, v_2 + v'_2, q_2 + q'_2, \dots, v_K + v'_K, q_K + q'_K]$ . Having the observations  $y^{v_k+v'_k}$  and  $y^{q_k+q'_k}$ , we are able to compute the joint probabilities of the input bits by applying the

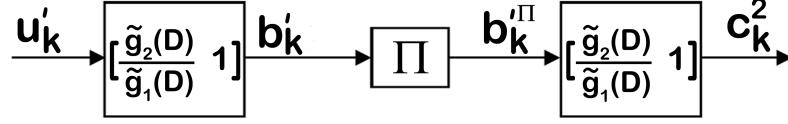


Figure 3.15: The encoder for dual of the SCCC in Figure 3.14.

BCJR algorithm to this big trellis. These bits are indeed an interleaved version of the output bits produced by the big trellis which corresponds to RSC1 and RSC1<sup>⊥</sup>, so using probabilities on such bits enables us to obtain soft information on the random and data bits pairs  $(u_l, u'_l)$  jointly. Figure 3.16 illustrates this procedure where  $\mathbf{M}_{12}^e$  and  $\mathbf{M}_{21}^e$  represent the extrinsic messages which are of the form  $[\log(P_{00}^e) \log(P_{01}^e) \log(P_{10}^e) \log(P_{11}^e)]$  with initial value set to  $[\log(0.25) \log(0.25) \log(0.25) \log(0.25)]$ .  $P_{ij}^e$  denotes the extrinsic probability that  $b_k$  equals  $i$  and  $b'_k$  equals  $j$  satisfying  $P_{00}^e + P_{01}^e + P_{10}^e + P_{11}^e = 1$ . One can see that for this case, two constituent decoders exchange information on the joint probabilities of  $b_k$ 's and  $b'_k$ 's. The decoder related to RSC2 and RSC2<sup>⊥</sup> works similar to the decoders for the PCCC case in the sense that it receives soft information from the channel and extrinsic information from the other decoder.

For an AWGN channel,  $\tilde{\alpha}_l(s')$  and  $\tilde{\beta}_l(s)$  are computed using

$$\tilde{\gamma}_l(s, s') = \log(P^e(b_l^\Pi, b_l^{\Pi})) - \frac{\|y_l - c_l\|^2}{2\sigma^2} \quad (3.24)$$

where  $c_l$  is the output label for each branch in the trellis at time  $l$ . Then the outgoing extrinsic information  $\mathbf{M}_{21}^e$  can be calculated using  $\tilde{\alpha}_l(s')$  and  $\tilde{\beta}_l(s)$  and

$$\tilde{\gamma}_l(s, s') = -\frac{\|y_l - c_l\|^2}{2\sigma^2}. \quad (3.25)$$

The decoder corresponding to RSC1 and RSC1<sup>⊥</sup> works slightly differently since there is no channel observation and the only input is the extrinsic information received from the other decoder,  $\mathbf{M}_{21}^e$ . For this decoder the logarithmic branch metric used to obtain  $\tilde{\alpha}_l(s)$  and  $\tilde{\beta}_l(s)$  is computed as

$$\tilde{\gamma}_i(s, s') = \log(P_{b_{2i-1}b'_{2i-1}}^e) + \log(P_{b_{2i}b'_{2i}}^e). \quad (3.26)$$

Then the extrinsic information on bits  $b_{2i-1}$ 's can be computed using

$$\tilde{\gamma}_i(s, s') = \log(P_{b_{2i}b'_{2i}}^e), \quad (3.27)$$

while the extrinsic information on  $b_{2i}$ 's is computed by setting

$$\tilde{\gamma}_i(s, s') = \log(P_{b_{2i-1}b'_{2i-1}}^e) \quad (3.28)$$

in the BCJR algorithm. In practice, one should set a maximum number of iterations for exchanging the extrinsic information between the two decoders. Finally, the random and data bit pair at time  $l$ ,  $(u_l, u'_l)$ , is decoded by applying the BCJR algorithm with (3.26) to the big trellis corresponding to RSC1 and RSC1<sup>⊥</sup>.

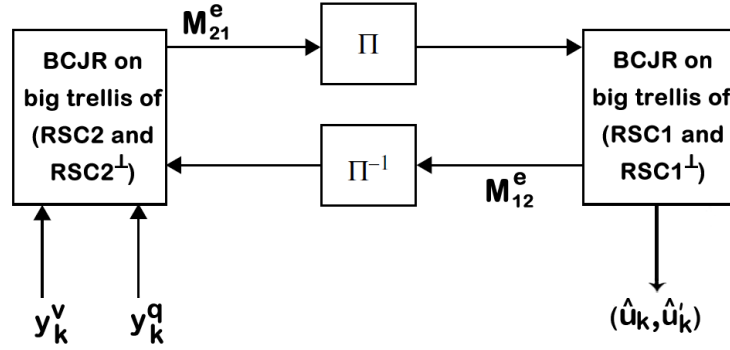


Figure 3.16: Iterative decoder for the randomized encoding scheme where one of the encoders in Figures 3.14 and 3.15 encodes random bits and the other encodes data bits.

### 3.3.2.3 Example

Figure 3.17 illustrates the performance of the iterative decoder described in Figure 3.16 over an AWGN channel.  $S$  denotes size of the interleaver, the number of iterations is set to the 10, and the constituent RSC codes are selected as  $[5/7 \ 1]$  and  $[1 \ 5/7]$ . Both are of rate  $1/2$ , and the inner trellis is terminated at the all-zero state. It is clear that the randomized SCCC works even better than the randomized PCCC in Figure 3.12 in terms of the security gap. Particularly, for  $P_{eve}^{min} \geq 0.3$  and  $P_{main}^{max} \leq 10^{-5}$ , by using a codeword of length 8004 in the randomized SCCC scheme, a security gap of less than 1 dB is achievable.

As another example, the performance of the randomized SCCC over a binary symmetric channel is shown in Figure 3.18 for two different code lengths.

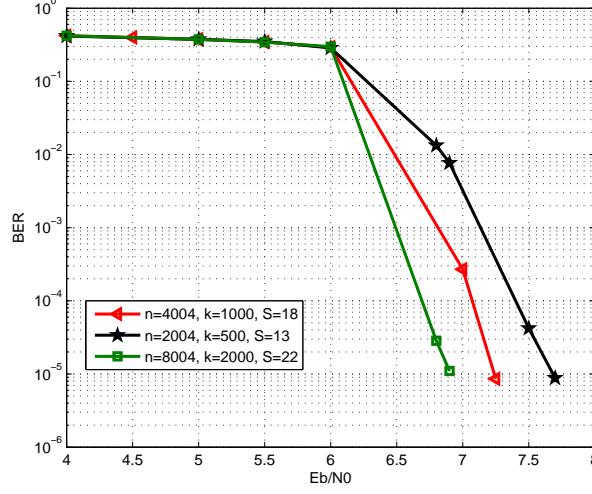


Figure 3.17: Performance of the randomized SCCC in the AWGN channel where  $n$  denotes the length of the codewords and transmission rate is about 1/4

Achievable security gaps for the randomized PCCC and SCCC schemes described in the last two sections are illustrated in Figure 3.19 for different values of  $P_{eve}^{min}$  (minimum probability of error at the eavesdropper). It can be seen that randomized SCCC can achieve smaller security gaps which makes it more suitable for the wiretap channel. We also note that the length of the code used in the randomized SCCC (8004) is smaller than that of randomized PCCC (10008). Specifically, for  $P_{eve}^{min} \geq 0.3$ , the best security gap obtained is about 0.9 dB which outperforms the existing coding schemes in the literature for these specific  $P_{eve}^{min}$  and  $P_{main}^{max}$  values.

### 3.3.3 Randomized LDGM-RSC codes

We described a coding scheme which consists of serial concatenation of an LDGM code with an RSC code in Section 3.2.1. Here, we apply such codes to the randomized encoding scheme and evaluate their performance over a wiretap channel.



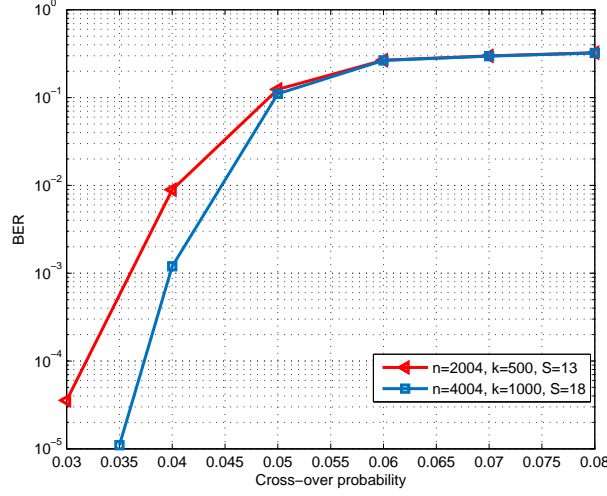


Figure 3.18: Performance of the randomized SCCC in the binary symmetric channel where  $n$  and  $k$  denote the length of the codewords and data bits, respectively, which makes transmission rate about  $1/4$

### 3.3.3.1 Encoding

Figure 3.20 illustrates an LDGM-RSC code concatenation where generator matrix of the LDGM and RSC codes are denoted by  $\mathbf{G}$  and  $[1 \frac{g_2(D)}{g_1(D)}]$ , respectively. Similar to the other randomized schemes, we need to obtain the dual of the code in Figure 3.20 by substituting each constituent encoder's dual. Hence, the dual of an LDGM-RSC code is of the form depicted in Figure 3.21. Dual of the LDGM code has the generator matrix  $\mathbf{H}$  which is its parity check matrix, i.e.,  $\mathbf{GH}^T = \mathbf{0}$ .

We use one of the encoders in Figures 3.20 and 3.21 to encode the random bits and the other one to encode data bits. We denote  $\mathbf{c} = [v_1, q_1, v_2, q_2, \dots, v_K, q_K]$ ,  $\mathbf{c}' = [v'_1, q'_1, v'_2, q'_2, \dots, v'_K, q'_K]$ . Then, the codeword transmitted through the channel is the modulo-2 sum of these two codewords, i.e.,  $\mathbf{c} + \mathbf{c}' = [v_1 + v'_1, q_1 + q'_1, v_2 + v'_2, q_2 + q'_2, \dots, v_K + v'_K, q_K + q'_K]$ .

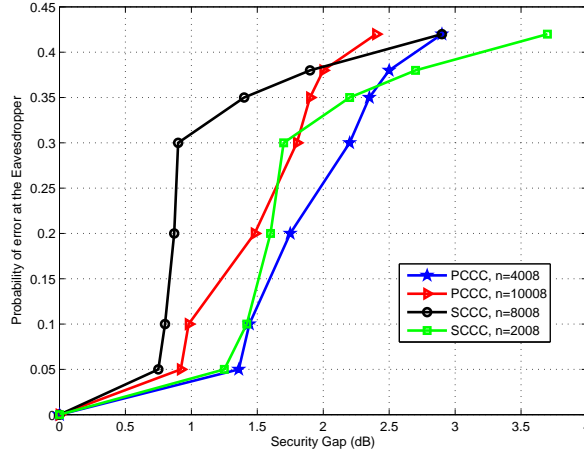


Figure 3.19: Achievable security gaps for PCCC and SCCC randomized schemes for different values of  $P_{eve}^{min}$  where  $P_{main}^{max} \approx 10^{-5}$ .

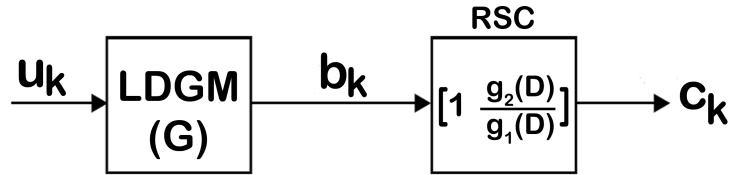


Figure 3.20: The encoder for the serial concatenation of a LDGM code with an RSC code.

### 3.3.3.2 Decoding

In this section, we provide an iterative joint decoder of the randomized LDGM-RSC coding scheme which is a generalization of the decoder introduced in Section 3.2.1 for a single LDGM-RSC code. Let us denote the received signal after the channel by  $\mathbf{y} = [y_1, y_2, \dots, y_K]$  where  $y_k = [y_k^v, y_k^q] = [y^{v_k+v'_k}, y^{q_k+q'_k}]$ .

Figure 3.22 depicts the schematic of the decoder for the randomized case. As pointed out in the previous sections, given the observation, we are able to obtain soft information on the pair of random and data bits  $(u_i, u'_i)$  applying the BCJR algorithm to the big trellis which corresponds to the RSC code and its dual. This information is sent to the other decoder as  $M_{21}^e = [P_{00}, P_{01}, P_{10}, P_{11}]$  where  $P_{ab} = P(u_i = a, u'_i = b)$ .

The other constituent decoder is the decoder for the 2-user LDPC multiple

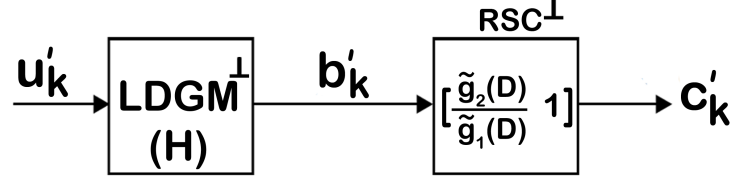


Figure 3.21: The encoder for dual of the code in the Figure 3.20.

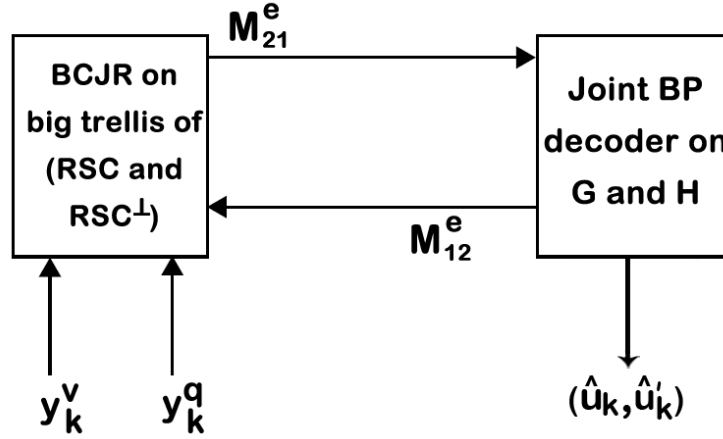


Figure 3.22: Iterative decoder for the randomized encoding scheme where one of the encoders in Figures 3.20 and 3.21 encodes random bits and the other encodes data bits.

access channel (LDPC-MAC) [44], [45] which jointly decodes the messages corresponding to the two independent senders. For the setup described here, one of the users employs the LDGM code with generator  $\mathbf{G}$  to encode the message and the other uses the LDGM with generator  $\mathbf{H}$ . Figure 3.23 illustrates the resulting Tanner graph which is composed of the 2 LDPC graphs which are connected through *state nodes*. State nodes receive information on the pair of bits  $(P_{ab})$  from the channel and exchange the information between two individual LDPC decoders. In the figure,  $m_{ab}^k$  denotes the message passed from node  $a$  to node  $b$  for user  $k$  where  $a$  and  $b$  can be  $v$  for variable node,  $c$  for check node or  $s$  for state node.

The vector  $\mathcal{P}$  received through the channel is  $[P_{00}, P_{01}, P_{10}, P_{11}]$  with  $P_{xy} = P(b_i = x, b'_i = y)$  denoting the outputs of the LDGM code and its dual with  $b_i$ 's and  $b'_i$ 's, respectively (see Figures 3.20 and 3.21).  $m_{vc}^k$ ,  $m_{cv}^k$ ,  $m_{vs}^k$  are calculated based on the regular BP decoding algorithm. The only remaining messages are

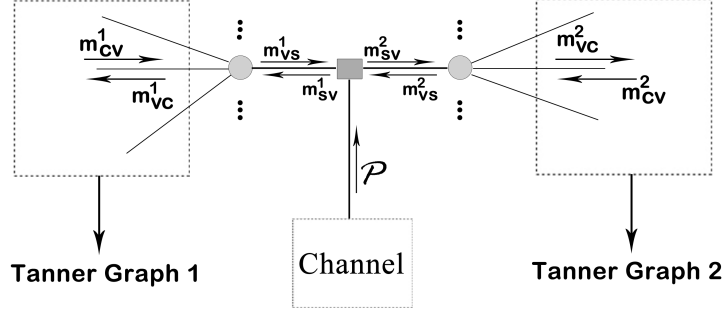


Figure 3.23: Joint BP decoder on the factor graph of the 2-user MAC channel  
 $m_{sv}^k$ 's which are passed from the state nodes to the variable nodes by computing

$$\begin{cases} m_{sv}^1 = \log \frac{P_{00}e^{m_{vs}^2+P_{01}}}{P_{10}e^{m_{vs}^2+P_{11}}}, \\ m_{sv}^2 = \log \frac{P_{00}e^{m_{vs}^1+P_{10}}}{P_{01}e^{m_{vs}^1+P_{11}}}. \end{cases} \quad (3.29)$$

For the decoder depicted in Figure 3.22, the vector  $\mathcal{P}$  equals to  $M_{21}^e$  provided by the BCJR algorithm. Then, we are able to incorporate the values in (3.29), and run the joint decoder in Figure 3.23. After one round of iteration, this decoder produces soft information on  $b_i$ 's and  $b'_i$ 's, i.e.,  $P(b_i)$  and  $P(b'_i)$ . Since  $b_i$ 's and  $b'_i$ 's are assumed to be independent of each other, it is straightforward to compute the extrinsic probabilities  $P(b_i, b'_i)$  by multiplication. Hence, we are able to send extrinsic information from the LDPC joint decoder back to the BCJR decoder, i.e.,  $M_{12}^e$  of the form  $[P_{00}, P_{01}, P_{10}, P_{11}]$  where  $P_{xy} = P(b_i = x, b'_i = y)$ .

In practice, one shall exchange extrinsic information between the two constituent decoders for a certain number of iterations. Then, the joint LDPC decoder makes the final decisions on  $b_i$ 's and  $b'_i$ 's. We note that since the LDGM codes described in Section 3.2 are systematic, a certain portion of  $b_i$ 's equals to  $u_i$ 's, in other words, information bits are obtained directly from the decoded codeword.

### 3.3.3.3 Example

Performance of the randomized LDGM-RSC codes with the decoder described in Figure 3.22 over an AWGN channel is illustrated in Figure 3.24. The number of iterations is set to 10 and the constituent RSC code is  $[1\ 7/5]$  of rate  $1/2$  which is terminated at the all-zero state. The LDGM codes are chosen from the Table 3.1. As pointed out in Section 3.2.1, such codes usually have a relatively high error floors as also depicted in Figure 3.24. However, it can be seen that for  $P_{eve}^{min} \geq 0.3$  and  $P_{main}^{max} \leq 7.5 \times 10^{-4}$ , by using codeword length of 5124 in the randomized LDGM-RSC coding scheme, a security gap of 2.5 dB is achievable.

We note that one way to improve the performance in Figure 3.24 and achieve lower security gaps with the randomized LDGM-RSC schemes, is to use more powerful LDGM codes with lower error floors. This topic has been studied in [46] where the authors point out that concatenation of two LDGM codes can reduce the error floors significantly.

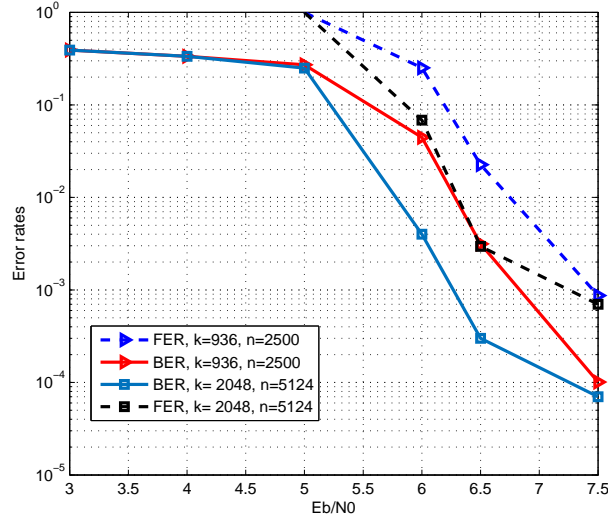


Figure 3.24: Performance of the randomized LDGM-RSC in the AWGN channel where  $n$  denotes the length of the codewords.

### 3.4 Chapter Summary

We have studied application of three concatenated codes to the randomized coding scheme for use over wiretap channels, namely, randomized PCCCs, randomized SCCCs and randomized LDGM-RSC codes. For the first two codes, we obtained the iterative decoder generalizing the existing turbo decoders in the literature. For randomized LDGM-RSC codes, we proposed the corresponding iterative decoder utilizing the joint LDPC decoder along with the BCJR decoder. Numerical results demonstrate that the randomized SCCC outperforms the other two classes of codes in terms of the security gaps by exhibiting a sharper slope in their BER behavior. Notably, we also observe that for certain selections of  $P_{max}^{main}$  and  $P_{eve}^{min}$ , the randomized SCCCs provide lower security gaps in comparison with the other coding schemes proposed for the wiretap channel in the existing literature.

## Chapter 4

# Conclusions and Future Work

We studied physical layer security in wireless communication systems using information-theoretic secrecy definitions. Specifically, we proposed coding schemes for physical layer security which are aimed at providing reliable and secure transmission to a receiver, simultaneously. These coding schemes should have a small security gap in order to achieve physical layer security even with a small degradation of the wiretapper channel with respect to the main one.

In Chapter 2, we proposed a randomized coding scheme based on convolutional codes and their duals for Gaussian and binary symmetric wiretap channels (where a code encodes data bits while its dual encodes a sequence of random bits). We described the optimal decoder and practically implementable sub-optimal alternatives. In particular, one of the decoders utilizes the trellis of the big code generated by two terminated convolutional codes which finds the codeword at the minimum (Euclidean or Hamming) distance to the received noisy vector. We also applied list Viterbi decoding to improve the decoding performance further. We devise lower and upper bounds on the error rate performance of the decoders in the proposed setup in terms of the message error probability to analytically characterize the decoder behavior at the eavesdropper and legitimate receiver, respectively. We noted that the derived bounds are applicable to other randomized coding setups as well (once the appropriate weight distributions are known).

We also developed a code design metric and utilize this metric to come up with specific convolutional encoders with small security gaps. Finally, we illustrated our findings via extensive numerical examples.

In Chapter 3, we employed the results of Chapter 2 to design other randomized coding schemes based on concatenated codes to provide improved security gaps. We applied three different concatenated coding schemes to the randomized encoding method: PCCC, SCCC and serial concatenation of an LDGM code with an RSC code. For each of these schemes, we proposed an iterative decoding method to estimate the transmitted codeword. For PCCC and SCCC randomized schemes, the decoder is a generalized version of the decoder proposed for turbo codes. For the LDGM-RSC randomized coding scheme, we utilized the LDPC joint decoder along with the BCJR algorithm employing a big trellis.

Numerical results show that using concatenated codes can improve the security gaps significantly for certain rates. Specifically, the SCCC based randomized encoding scheme achieves a security gap of 0.9 dB when  $P_{eve}^{min} \geq 0.3$ ,  $P_{main}^{max} \leq 10^{-5}$  for a transmission rate of  $R = 1/4$  and codeword length of about 8000 bits over an AWGN channel. We note that this scheme outperforms the other coding strategies in the literature for the wiretap channel for these  $P_{main}^{min}$  and  $P_{eve}^{min}$  values. For higher rates, e.g.,  $R \geq 0.4$ , LDGM-RSC randomized codes can be used which achieve a security gap of 2.5 dB for  $P_{eve}^{min} \geq 0.3$ ,  $P_{main}^{max} \leq 7.5 \times 10^{-5}$  and codeword length of 5120. We believe that the error floors associated with the LDGM codes result in such behavior, i.e., higher security gaps in comparison with the SCCC and PCCC randomized coding schemes.

As a future research direction, we note that our proposed lower bound in Chapter 2 is an approximation, so one can develop true lower bounds and tighter upper bounds on performance of decoders in the wiretap channel by taking the effects of favorable codewords into account. Moreover, one can design more powerful convolutional generators for random and data bits to come up with big codes with higher minimum distances than those of the codes we used in Chapter 2 in order to improve the security gaps. Studying the relationship between using encoders with higher memory sizes ( $m > 2$ ) in the randomized PCCC and SCCC,



and the resulting security gaps can be another research direction.

The LDGM-RSC randomized coding scheme can also be investigated in more detail in order to achieve improved results. Particularly, it is possible to use more powerful LDGM codes (e.g., those introduced in [21] consisting of  $\psi$ -unitary matrices for encoding). Such codes have larger minimum distances compared to the ones used in this work, and consequently they may result in better BER performances. Furthermore, it may be possible to design degree distributions for the LDGM codes in order to obtain the best possible security gaps.

We note that systematic LDGM codes used in this work are known to have high error floors mainly because the degree of a portion of their variable nodes is one and consequently they do not help in the decoding process. One solution to reduce the error floors in LDGM codes is to use a concatenated schemes, formed by an outer high-rate LDGM code followed by an inner low-rate LDGM code [19], [46]. One can use such codes in the randomized scheme and utilize the decoder proposed in this work by further concatenating them with an RSC code. It is clear that the resulting scheme would use an LDGM-LDGM-RSC code and its dual for encoding random and data bits.

Finally, we note that for the case where an LDGM (or LDPC) code is used along with its dual for the randomized encoding scheme, finding the corresponding decoder is still an open problem. In this case, the joint LDPC decoder does not work anymore because the channel observation fails to start the decoding process. Therefore, finding a practical decoder for this scenario is also important.

# Bibliography

- [1] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*. Boca Raton, FL, USA: CRC Press, 1996.
- [2] H. Dennis and E. Kiltz, “Secure hybrid encryption from weakened key encapsulation,” in *Proceedings of the 27th Annual International Cryptology Conference (CRYPTO)*, pp. 553–571, Santa Barbara, CA, USA, Aug. 2007.
- [3] A. Wyner, “The Wire-Tap Channel,” *Bell System Technical Journal*, vol. 54, no. 8, pp. 1355–1387, Oct. 1975.
- [4] I. Csiszár and J. Körner, “Broadcast channels with confidential messages,” *IEEE Trans. Inf. Theory*, vol. IT-24, no. 3, pp. 339–348, May 1978.
- [5] C. E. Shannon, “Communication theory of secrecy systems,” *Bell System Technical Journal*, vol. 28, pp. 656–715, Oct. 1949.
- [6] A. Thangaraj, S. Dihidar, A. Calderbank, S. McLaughlin and J. Merolla, “Applications of LDPC Codes to the Wiretap Channel,” *IEEE Trans. Inform. Theory*, vol. 53, no. 8, pp. 2933–2945, Aug. 2007.
- [7] H. MahdaviFar and A. Vardy, “Achieving the secrecy capacity of wiretap channels using polar codes,” *IEEE Trans. Inf. Theory*, vol. 57, no. 10, pp. 6428–6443, Oct. 2011.
- [8] F. Oggier, J.-C. Belfiore, P. Solé, “Lattice codes for the wiretap Gaussian channel: Construction and analysis,” *IEEE Trans. on Inf. Theory*, vol. 62, no. 10, pp. 5690–5707, Oct. 2016.

- [9] D. Kline, Jeongseok Ha, S. McLaughlin, J. Barros and Byung-Jae Kwak, "LDPC Codes for the Gaussian Wiretap Channel," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 3, pp. 532-540, Sept. 2011.
- [10] M. Baldi, F. Bambozzi, and F. Chiaraluce, "Coding with scrambling, concatenation, and HARQ for the AWGN wire-tap channel: A security gap analysis," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 3, pp. 883-894, Jun. 2012.
- [11] Y. Zhang, A. Liu, C. Gong, G. Yang, and S. Yang, "Polar-LDPC concatenated coding for the AWGN wiretap channel," *IEEE Commun. Lett.*, vol. 18, no. 10, pp. 1683-1686, Oct. 2014.
- [12] C. W. Wong, T. F. Wong, and J. M. Shea, "Secret-sharing LDPC codes for the BPSK-constrained Gaussian wiretap channel," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 3, pp. 551-564, Sept. 2011.
- [13] M. Baldi, G. Ricciutelli, N. Maturo, and F. Chiaraluce, "Performance assessment and design of finite length LDPC codes for the Gaussian wiretap channel," in *Proc. IEEE ICC Workshops*, Jun. 2015, pp. 435-440.
- [14] L. H. Ozarow and A. D. Wyner, "Wire-tap channel II," *Bell Lab. Tech. J.*, vol. 63, no. 10, pp. 2135-2157, Dec. 1984.
- [15] R. McEliece, "How to compute weight enumerators for convolutional codes," in *Communications and Coding*, M. Darnell and B. Honary, Eds. New York: Wiley, pp. 121-141, Jan. 1998.
- [16] G. Poltyrev, "Bounds on the decoding error probability of binary linear codes via their spectra," *IEEE Trans. Inform. Theory*, vol. 40, no. 4, pp. 1284-1292, July 1994.
- [17] G. Seguin, "A lower bound on the error probability for signals in white Gaussian noise," *IEEE Trans. Inform. Theory*, vol. 44, no. 7, pp. 3168-3175, Nov. 1998.
- [18] R. Johannesson and K. Zigangirov, *Fundamentals of convolutional coding*. Piscataway, NJ: IEEE Press, 1999.

- [19] J. Garcia-Frias and W. Zhong, "Approaching Shannon performance by iterative decoding of linear codes with low-density generator matrix," *IEEE Commun. Lett.*, vol. 7, no. 6, pp. 266–268, Jun. 2003.
- [20] R. G. Gallager, *Low density parity check codes*. Cambridge, MA: MIT Press, 1963.
- [21] M. Baldi, F. Bambozzi, and F. Chiaraluce, "On a family of circulant matrices for quasi-cyclic low-density generator matrix codes," *IEEE Trans. Inform. Theory*, vol. 57, no. 9, pp. 6052–6067, Sept. 2011.
- [22] N. Seshadri and C.-E. Sundberg, "List Viterbi decoding algorithm with applications," *IEEE J. Sel. Areas Commun.*, vol. 42, no. 2/3/4, pp. 313–323, Feb./Mar./Apr. 1994.
- [23] A. Ozelikkale and T. M. Duman, "Lower bounds on the error probability of turbo codes," *IEEE Int. Symp. on Inf. Theory*, pp. 3170–3174, Honolulu, HI, July 2014.
- [24] A. Cohen and N. Merhav, "Lower bounds on the error probability of block codes based on improvements on de Caen's inequality," *IEEE Trans. Inform. Theory*, vol. IT-52, no. 2, pp. 290–310, Feb. 2004.
- [25] T. M. Duman and M. Salehi, "New performance bounds for turbo codes," *IEEE Trans. Commun.*, vol. 46, pp. 717–723, June 1998.
- [26] E. R. Berlekamp, "The technology of error-correcting codes," *Proc. IEEE*, vol. 68, pp. 564–593, May 1980.
- [27] D. Divsalar, "A simple tight bound on error probability of block codes with application to turbo codes," TMO Progress Rep., NASA, JPL, Tech. Rep 42-139, Nov. 1999.
- [28] H. Herzberg and G. Poltyrev, "Techniques of bounding the probability of decoding error for block coded modulation structures," *IEEE Trans. Inform. Theory*, vol. 40, no. 3, pp. 903–911, Aug. 1994.

- [29] L. Hughes, "A simple upper bound on the error probability for orthogonal signals in white noise," *IEEE Transactions on Communications*, vol. 40, no. 4, pp. 670, Aug. 2002.
- [30] I. Sason and S. Shamai, "On improved bounds on the decoding error probability of block codes over interleaved fading channels, with applications to turbo-like codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 6, pp. 2275-2299, Sept. 2001.
- [31] S. Yousefi, "Bounds on the performance of maximum-likelihood decoded binary block codes in AWGN interference," Ph.D. dissertation, Dept. Elec. Comput. Eng., Univ. Waterloo, Waterloo, ON, Canada, 2002.
- [32] I. Sason and S. Shamai, "Improved upper bounds on the ML decoding error probability of parallel and serial concatenated turbo codes via their ensemble distance spectrum," *IEEE Trans. Inform. Theory*, vol. 46, no. 1, pp. 24-47, Jan. 2000.
- [33] I. Sason and S. Shamai (Shitz), "Performance analysis of linear codes under maximum-likelihood decoding: A Tutorial," in *Foundations and Trends in Communications and Information Theory*. Delft, The Netherlands: Now Publishers, July 2006, vol. 3, pp. 1-222.
- [34] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding," in *Proc. Int. Communications Cconf. (ICC'93)* Geneva, Switzerland, May 1993, pp. 1064-1070.
- [35] T. M. Duman, "Turbo codes and turbo coded modulation systems: analysis and performance bounds," Ph.D. dissertation, Northeastern University, Dept. of Electrical and Computer Engineering, Boston, MA, May 1998.
- [36] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design, and iterative decoding," *IEEE Trans. Inform. Theory*, vol. 44, pp. 909-926, May 1998.
- [37] D. Divsalar and F. Pollara, "Multiple turbo codes for deep-space communications," JPL TDA Progress Report, pp. 42-121, May 1995.

- [38] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, pp. 284–287, Mar. 1974.
- [39] W. Ryan and S. Lin, *Channel codes, classical and modern*, Cambridge University Press, Cambridge, UK, 2009.
- [40] T. Richardson and R. Urbanke, "The capacity of low-density paritycheck codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.
- [41] J. F. Cheng and R. J. McEliece, "Some high-rate near capacity codecs for the Gaussian channel," in *Proc. 34th Allerton Conf. Communications, Control and Computing*, Allerton, IL, Oct. 1996.
- [42] S. J. Johnson and S. R. Weller, "A family of irregular LDPC codes with low encoding complexity," *IEEE Commun. Lett.*, vol. 7, no. 2, pp. 79–81, Feb. 2003.
- [43] M. Baldi, F. Bambozzi, and F. Chiaraluce, "On a family of circulant matrices for quasi-cyclic low-density generator matrix codes," *IEEE Trans. Inform. Theory*, vol. 57, no. 9, pp. 6052–6067, Sept. 2011.
- [44] A. Roumy and D. Declercq, "Characterization and optimization of LDPC codes for the 2-user Gaussian multiple access channel," *EURASIP J. Wireless Commun. Netw.*, vol. 2007, Article ID: 074890, May 2007.
- [45] Shahrouz Sharifi, A. Korhan Tanc and Tolga M. Duman, "LDPC code design for the two-user Gaussian multiple access channel," *IEEE Trans. on Wireless Communications*, vol. 15, no. 4, pp. 2833–2844, Apr. 2016.
- [46] M. González-López, F. J. Vázquez-Araújo, L. Castedo, and J. Garcia-Frias, "Serially-concatenated low-density generator matrix (SCLDGM) codes for transmission over AWGN and Rayleigh fading channels," *IEEE Trans. Wireless Commun.*, vol. 6, no. 8, pp. 2753–2758, Aug. 2007.