Pontifícia Universidade Católica
do Rio de Janeiro

Robert Mota Oliveira

Cost-Effective Construction and Decoding of Polar Codes

Tese de Doutorado

Thesis presented to the Programa de Pós–graduação em Engenharia Elétrica, do Departamento de Engenharia Elétrica da PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Engenharia Elétrica.

Advisor: Prof. Rodrigo Caiado de Lamare

Rio de Janeiro
August 2022

Pontifícia Universidade Católica
do Rio de Janeiro

Robert Mota Oliveira

Cost-Effective Construction and Decoding of Polar Codes

Thesis presented to the Programa de Pós–graduação em Engenharia Elétrica da PUC-Rio in partial fulfillment of the requirements for the degree of Doutor em Engenharia Elétrica. Approved by the Examination Committee:

Rio de Janeiro, August the 25th, 2022

Robert Mota Oliveira

The author received the diploma degree in electrical engineering from the Espírito Santo Federal University, Vitória, Espírito Santo, Brazil, in 1999, and the M.Sc. degree in electrical engineering from the Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Rio de Janeiro, Brazil, in 2018.

Bibliographic data

Oliveira, Robert Mota

Cost-Effective Construction and Decoding of Polar Codes / Robert Mota Oliveira; advisor: Rodrigo Caiado de Lamare. – 2022.

115 f: il. color. ; 30 cm

Tese (doutorado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Elétrica, 2022.

Inclui bibliografia

1. Engenharia Elétrica – Teses. 2. Códigos Polares. 3. Polarização Não-uniforme. 4. Comprimento Arbitrário. 5. Aproximação por partes. 6. Reponderação adaptativa. I. de Lamare, Rodrigo Caiado. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Engenharia Elétrica. III. Título.

CDD: 621.3

To my family, for their support
and encouragement.

Acknowledgments

First, I would like to thank God for giving me strength and focus to successfully complete the PhD degree in Electrical Engineering at one of the most respected Brazilian universities, the Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio).

I would like to thank my mother and my father, for their support and encouragement, not only during my PhD but since I have started my undergraduate course.

I thank my wife Fabrini for her unconditional love and companionship over the years. And to my daughter Eduarda for understanding my long periods of absence.

I am grateful to have Prof. Rodrigo C. de Lamare as my advisor. During these four years, Prof. de Lamare has been always hard-working, enthusiastic, fair and sincere. Always available to help, Prof. de Lamare contributed incessantly to the construction of this thesis and my training as a researcher, for which I am immensely grateful.

I would like to thank all my colleagues and professors in the Centro de Estudos de Telecomunicações (CETUC/ PUC-Rio), but especially Prof. Lukas Landau and my friends Alireza Danaee, Dr. Roberto Di Renna, Dr. Silvio Pinto and Lucas Oliveira, who contributed to a friendly environment at CETUC.

Finally, this study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, and in part by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

Abstract

Oliveira, Robert Mota; de Lamare, Rodrigo Caiado (Advisor). **Cost-Effective Construction and Decoding of Polar Codes**. Rio de Janeiro, 2022. 115p. Tese de Doutorado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

Erdal Arıkan introduced the polar codes in 2009. This is a new class of error correction codes capable of reaching the Shannon limit. Using cyclic redundancy check concatenated list successive cancellation decoding and fast code constructs, polar codes have become an attractive, high-performance error correction code for practical use. Recently, polar codes have been adopted for the 5^{th} generation standard for cellular systems, more specifically for the uplink and downlink control information for the extended Mobile Broadband (eMBB) communication services. However, polar codes are limited to block lengths to powers of two, due to a recursive Kronecker product of the 2x2 polarizing kernel. For practical applications, it is necessary to provide flexible length polar code construction techniques. Another aspect analyzed is the development of a technique of construction of polar codes of low complexity and that has an optimum performance on additive white Gaussian noise channels, mainly for long blocks, inspired by the optimization of the Gaussian approximation construction. Another relevant aspect is the parallel decoding power of the belief propagation decoder. This is an alternative to achieve the new speed and latency criteria foreseen for the next generation standard for cellular systems. However, it needs performance improvements to become operationally viable, both for 5G and for future generations. In this thesis, three aspects of polar codes are addressed: the construction of codes with arbitrary lengths that are intended for maximizing the flexibility and efficiency of polar codes, the improvement of the construction method by Gaussian approximation and the decoding of codes using an adaptive reweighted belief propagation algorithm, as well as the analysis of trade-offs affecting error correction performance.

Keywords

Polar Codes; Non-uniform Polarization; Arbitrary-length; Piecewise approximation; Adaptive reweightin.

Resumo

Oliveira, Robert Mota; de Lamare, Rodrigo Caiado. **Construção econômica e decodificação de códigos polares**. Rio de Janeiro, 2022. 115p. Tese de Doutorado – Departamento de Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro.

Erdal Arıkan introduziu os códigos polares em 2009. Trata-se de uma nova classe de códigos de correção de erros capaz de atingir o limite de Shannon. Usando decodificação de cancelamento sucessivo em lista, concatenada por verificação de redundância cíclica e a construções rápida de código, os códigos polares tornaram-se um código de correção de erros atraente e de alto desempenho para uso prático. Recentemente, códigos polares foram adotados para o padrão de geração 5^{th} para sistemas celulares, mais especificamente para as informações de controle dos canais reverso e direto para os serviços de comunicação eMBB. No entanto, os códigos polares são limitados a comprimentos de bloco a potências de dois, devido a um produto Kronecker recursivo do kernel polarizador 2x2. Para aplicações práticas, é necessário fornecer técnicas de construção de código polar de comprimento flexível. Outro aspecto a ser analisado é o obtenção de uma técnica de construção de códigos polares de baixa complexidade e que tenha um ótimo desempenho em canal de ruído aditivo gaussiano branco, principalmente para blocos longos, inspirada na otimização da construção da aproximação gaussiana. Outro aspecto relevante é o poder de decodificação paralela do decodificador de propagação de crenças. Esta é uma alternativa para atender aos novos critérios de velocidade e latência previstos para o padrão de próxima geração para sistemas celulares. No entanto, ele precisa de melhorias de desempenho para tornar-se operacionalmente viável, tanto para 5G quanto para as gerações futuras. Nesta tese, três aspectos dos códigos polares são abordados: a construção de códigos com comprimentos arbitrários que visam maximizar a flexibilidade e eficiência dos códigos polares, o aprimoramento do método de construção por métodos gaussianos aproximação e a decodificação de códigos usando um algoritmo adaptativo de propagação de crenças reponderadas, bem como analisar quaisquer compromissos que afetem o desempenho da correção de erros.

Palavras-chave

Códigos Polares; Polarização Não-uniforme; Comprimento Arbitrário; Aproximação por partes; Reponderação adaptativa.

Table of contents

List of figures

List of tables

List of Abreviations

3GPP	The 3rd Generation Partnership Project
5G	5th Generation
ADE	Accumulative Design Error
AGA	Approximate Gaussian Approximation
APGA	Approximate Piecewise Gaussian Approximation
AR-SBP	Adaptive Reweighting Sparse Belief Propagation
AWGN	Additive White Gaussian Noise
BEC	Binary Erasure Channel
BCH	Bose–Chaudhuri–Hocquenghem
BER	Bit Error Rate
BP	Belief Propagation
BPSK	Binary Phase Shift Keying
B-DMCs	Binary Symmetric Discrete Memoryless Channels
CA-SCL	CRC-Aided Successive Cancellation List
CRC	Cyclic Redundancy Check
C	Check Nodes
CW	Column Weights
DE	Density Evolution
EGA	Exact Gaussian Approximation
eMBB	Enhanced Mobile Broadband
FER	Frame Error Rate
GA	Gaussian Approximation of Density Evolution
HARQ	Hybrid Automatic Repeat Request
IoT	Internet of Things
$\mathcal{L}$	List in SCL and CA-SCL
L	Likelihood-Ratio
LLR	Log-Likelihood-Ratio
LDPC	Low-Density Parity-Check
MC	Monte Carlo Simulations
MG	Modified Gaussian
MK	Multi-Kernel Techniques
ML	Maximum Likelihood
NDP	Number of Different Positions

NW-RBP – Node-Wise Residual Belief Propagation
NUC – Non-Uniform Channel
NUPGA – Non-Uniform Polarization based on Gaussian Approximation
PC – Polar Codes
PD – Polarization-Driven
PDF – Probability Density Function
PE - Processing Element
PO – Partial Order Theory
PGA – Piecewise Gaussian Approximation
PM – Path Metric
PW – Polarization Weight
RM – Reed-Muller
RMSE – Root Mean Square Error
RQUP – Reversal Quasi-Uniform Puncturing Scheme
SC – Successive Cancellation Decoder
SCL – Successive Cancellation List Decoder
SPGA – Simplified Piecewise Gaussian Approximation
SPA – Sum-Product Algorithm
T_{max} – Maximum Number of Iterations
V – Variable Nodes

1 Introduction

Polar codes (PC) and channel polarization theory were introduced by Arıkan [1] in 2009. Such codes constitute a powerful channel coding scheme, with a low complexity coding and decoding. In a scenario where the decoder uses Successive Cancellation (SC), the channel is Additive White Gaussian Noise (AWGN) and the codewords are long, the maximum capacity of binary symmetric discrete memoryless channels (B-DMCs) can be achieved [1].

Choosing the most reliable channels for information transmission is the basic premise for polar codes construction. The channel reliability depends on the codeword length and the signal-to-noise ratio, according to the channel polarization theorem [1]. The channel polarization theorem proves that in channel polarization, for a long enough codeword, the bit channels [1] tend to two conditions: either they become noiseless or noisy. The former have high reliability and consequently are used for the transmission of information bits. These channels are used to transmit the bits that are known to both the transmitter and receiver. These bits are called frozen bits.

The 3rd Generation Partnership Project (3GPP) Group selected polar codes for the 5^{th} generation (5G) [2], where they are used for uplink/downlink control channels. Their implementations require low complexity which motivated researchers to improve their performance. Several alternatives often appear in the technical literature which are aimed to improve the efficiency of code construction and decoding techniques.

In this thesis, three aspects of polar codes are addressed: the construction of codes with arbitrary lengths, the improvement of the construction method by Gaussian approximation and the decoding of codes using an adaptive reweighted belief propagation algorithm.

1.1 Motivation

The development of the 5^{th} generation wireless communication systems (5G) and other wireless standards has been conducted through massive investments in fundamental research and development, and has been motivated by the exponential increase in data transmission in today's networks [3]-[6].

The different use cases of future wireless services will be increasingly demanding, whether it is mobile broadband, vehicular communications or low-power sensors that build the internet of things.

In addition, 5G systems have challenging requirements regarding the [7] transmission rates, power consumption, coverage, and latency in data transmission. Among the central elements of 5G systems are the waveform used in the transmission and the channel coding technique adopted to ensure the required performance for the wireless links. Recently, polar codes have been adopted for the control channels of the enhanced Mobile Broadband (eMBB) scenario of 5G systems [8].

Polar codes have the potential to meet all these requirements if the set formed by the code construction and the decoder architecture is well designed [1]. Among the main obstacles for the implementation are an adequate code construction technique, an efficient puncturing, shortening and extension scheme, which is of fundamental importance to meet the length and the code rate of proposed structure in 5G systems, and efficient decoding architectures.

In their original construction, polar codes only allowed code lengths that are powers of two, while the message length (the number of information bits) is arbitrary. This suggests that the code length of standard polar codes will be 64, 128, 256, 512, 1024, and so on. This restriction on the code length is a major drawback of polar codes, which needs to be overcome for practical applications.

Another aspect in the construction of polar codes is the inaccuracy of the Gaussian Approximation (GA) construction method when the code length is long. The GA method is widely used for construction of polar codes. This inaccuracy is caused by the selection of the channel due to the GA calculation error, which uses a two-segment approximation function. The result is a catastrophic loss of performance for long codes.

Additionally, low decoding latency can be achieved with parallel decoding, which could greatly speed up the decoding when compared to serial decoding by successive cancellations, originally proposed for polar codes. However, these decoders lack improvements to become operationally viable. Improvements can be obtained by joint code construction with these decoders, or with direct implementation of meta-heuristics to aid the decoding, or even with methods to reduce the number of iterations for the same performance.

1.2 Objectives

The objectives of this thesis are to present, analyze and discuss a new method for polar codes construct with arbitrary length, an efficient and simplified alternative for GA construction, and a belief propagation decoding algorithm based on the concept of reweighting for polar codes. Firstly, presents the non-uniform channel polarization theory in the use of polar codes construction of arbitrary length. Next, we present a simplified piecewise approximation as an alternative to the Gaussian approximation construction method, with performance gains for medium blocks and high precision for long blocks. In this thesis, we consider short blocks those with code lengths up to 512 bits, medium blocks those with code lengths between 512 and 2048 bits and long blocks those with code lengths greater than 2048 bits. Lastly, we present an adaptive reweighting decoding algorithm based on sparse belief propagation for polar codes, which obtains fast convergence by reducing the number of iterations required for successful decoding when compared to standard BP.

1.3 Contributions

The contribution of this thesis is to propose a set of techniques to enhance the performance of polar codes by improving their construction and decoding. Thus, the main contributions can be summarized as:

1. Development of the non-uniform polarization technique in the construction of polar codes for arbitrary lengths. This technique is suitable for decoding by successive cancellation and Gaussian noise channel. The coding and decoding procedures are the same as those applied in the original polar codes. In particular, we describe the design of rate-compatible polar codes using the nonuniform polarization method. In addition, we include a detailed description of the algorithm that performs code extension, an analysis of the approach and extra simulation results for various application scenarios.

2. Development of a simplified piecewise approximation function of the Gaussian approximation construction method. Gaussian approximation is a popular method for polar codes construction. However, it is constituted by transcendental functions of hard numerical solution, mainly for long code lengths. The piecewise approximation function presents high precision and with low computational cost, being composed only by polynomial functions.

3. Development of an adaptive reweighting decoding strategy based on sparse belief propagation, which obtains a reduction in the average number of iterations without increasing the algorithmic complexity, that is, an efficient low-latency decoding.

1.4 Structure of this Thesis

The rest of this document is organized as follows:

- In Chapter 2, a review of the literature is provided, covering the fundamentals of polarization, polar codes, puncturing, shortening and extension methods and a number of key concepts which will be used throughout this work. The general coding system including channel models is introduced and the specific features of polar codes, including the properties of the encoder and the decoder. We also introduce the basic successive cancellation decoder, successive cancellation list decoder, belief propagation decoder and low-density parity-check code-like polar codes decoder.
- In Chapter 3, the channel polarization theory is revisited [1], we present the non-uniform polarization of channels, and using the induction method to compare with the uniform polarization of channels, we show that it also achieves channel capacity and present the algorithms for implementing shortening and extension techniques based on nonuniform polarization based GA.
- In Chapter 4, we present the construction of polar codes based on Piecewise Gaussian Approximation techniques. Compared to the standard GA construction method, the proposed method presents performance gains for medium blocks and improves accuracy for long blocks. The researched scenario is for successive cancellation decoding of data transmitted over an AWGN channel.
- In Chapter 5, we present the problem of belief propagation decoding, the issue of reducing the number of iterations. We introduce an adaptive reweighting decoding strategy based on sparse belief propagation that can decrease decoder latency without increasing the computational complexity.
- Chapter 6 concludes this thesis, discussing the main results, future directions and research opportunities.

1.5 List of Publications

Some of the results in this thesis have been published, or will be submitted to publications.

Journal Papers:

1. **R. M. Oliveira** and R. C. De Lamare, “Design of Rate-Compatible Polar Codes Based on Non-Uniform Channel Polarization” in *IEEE Access*, vol. 9, pp. 41902-41912, 2021, doi: 10.1109/ACCESS.2021.3065816.
2. **R. M. Oliveira** and R. C. de Lamare, “Polar Codes Based on Piecewise Gaussian Approximation: Design and Analysis,” in *IEEE Access*, vol. 10, pp. 73571-73582, 2022, doi: 10.1109/ACCESS.2022.3190393.

Conference Papers:

1. **R. M. Oliveira** and R. C. de Lamare, “Non-Uniform Channel Polarization and Design of Rate-Compatible Polar Codes,” in *International Symposium on Wireless Communication Systems (ISWCS)*, 2019, pp. 537-541, doi: 10.1109/ISWCS.2019.8877311.
2. **R. M. Oliveira** and R. C. de Lamare, “Rate-Compatible Polar Codes Construction Based on Extension and Non-Uniform Channel Polarization for Iot Applications,” in *IEEE Statistical Signal Processing Workshop (SSP)*, 2021, pp. 116-120, doi: 10.1109/SSP49050.2021.9513862.
3. **R. M. Oliveira** and R. C. De Lamare, “Construction of Polar Codes Based on Piecewise Gaussian Approximation,” in *17th International Symposium on Wireless Communication Systems (ISWCS)*, 2021, pp. 1-5, doi: 10.1109/ISWCS49558.2021.9562206.
4. L. M. de Oliveira, **R. M. Oliveira** and R. C. de Lamare, “Q-Learning-Driven BP decoding for Polar Codes,” in *XXXIX Simposio Brasileiro de Telecomunicações e Processamento de Sinais*, pp. 26–29, 2021. Selected for the SBrT 2021 Best Paper Award.

2 Fundamentals and State-of-the-Art

2.1 Chapter Overview

This chapter provides a review of polar codes and their encoding and decoding methods. It begins with a description of the channel polarization phenomenon and then describes the encoding method. The low-complexity successive cancellation decoding method is reviewed with a simplified algorithm description. We continue with the description of the successive cancellation list decoding as well as the Cyclic Redundancy Check (CRC)-aided successive cancellation list decoding, which are widely used in the literature. Then, the Belief Propagation (BP) decoding and the Low-Density Parity-Check Code (LDPC) like polar codes decoder are detailed.

2.2 System Model

Figure 2.1 shows a block diagram of the polar coding system considered in this thesis.

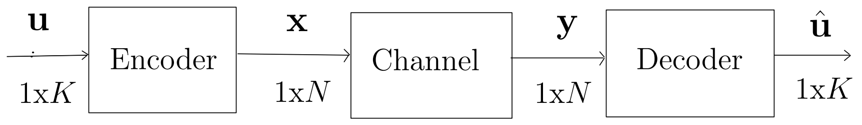


Figure 2.1: System model of the polar coding system.

.

In this system, $\mathbf{u}$ is the binary *message* vector that is transmitted with K bits, where $\mathbf{u} \in \{0,1\}^{1\text{x}K}$. It is through the generator matrix $\mathbf{G}$ that the message $\mathbf{u}$ is encoded, producing the codeword x with N bits, that is,

$$\mathbf{x} = \mathbf{u} \cdot \mathbf{G}, \tag{2-1}$$

with $\mathbf{x} \in \{0,1\}^{1\text{x}N}$. Considering a binary phase-shift keying (BPSK) modulation and an AWGN channel, the *codeword* $\mathbf{x}$ is transmitted over an AWGN channel, resulting in the received vector

$$\mathbf{y} = (\mathbf{1} - 2\mathbf{x}) + \mathbf{w}, \tag{2-2}$$

where $\mathbf{1}$ is an all-one vector, where $\mathbf{w}$ is the vector corresponding to the Gaussian noise.

In the decoding step, the decoding algorithm observes $\mathbf{y}$ in order to estimate $\mathbf{u}$. We call it an estimated message $\hat{\mathbf{u}}$, and if $\mathbf{u} = \hat{\mathbf{u}}$ we say that the message has been fully recovered.

Let $W : \mathcal{X} \rightarrow \mathcal{Y}$ denote a binary discrete memoryless channel (B-DMC), with input alphabet $\mathcal{X}$, output alphabet $\mathcal{Y}$, and the channel transition probability $W(y|x)$, $x \in \mathcal{X}$, $y \in \mathcal{Y}$, where $\mathcal{X} \in \{0, 1\}$ and $\mathcal{Y} \in \mathbb{R}$. We write $W^N : \mathcal{X}^N \rightarrow \mathcal{Y}^N$ to denote the vector channel that corresponds to N independent uses of W with input alphabet $\mathcal{X}^N$, output alphabet $\mathcal{Y}^N$ and transition probabilities $W^N(y_1^N|x_1^N) = \prod_{i=1}^N W(y_i|x_i)$ [1]. Consider the input vector $\mathbf{u} = \mathbf{u}_1^K$ and the output vector $\mathbf{x} = \mathbf{x}_1^N$ and $\mathbf{y} = \mathbf{y}_1^N$. According to the notation suggested in [1]: an $\mathbf{a}_1^N$ to denote a vector $(a_1, a_2, \ldots, a_N)$ and $|\mathbf{a}_1^N|$ is the length. Specifically $K = |\mathbf{u}_1^K|$.

A channel W is defined as memoryless if its transition probability is

$$W_{\mathcal{Y}|\mathcal{X}}(y|x) = \prod_{i}^{N} W_{\mathcal{Y}_i|\mathcal{X}_i}(y_i|x_i). \tag{2-3}$$

The mutual information of the channel with equiprobable inputs, or symmetric capacity, is defined by [1]

$$I(W) \triangleq \sum_{y \in \mathcal{Y}} \sum_{x \in \mathcal{X}} \frac{1}{2} W(y|x) \log \frac{W(y|x)}{\frac{1}{2}W(y|0) + \frac{1}{2}W(y|1)} \tag{2-4}$$

and the corresponding reliability metric, the Bhattacharyya parameter, is defined by

$$Z(W) \triangleq \sum_{y \in \mathcal{Y}} \sqrt{W(y|0)W(y|1)}. \tag{2-5}$$

Note that the Bhattacharyya parameter is the measure of reliability and it is an upper bound on the probability of Maximum Likelihood (ML) decoding [1]. All logarithms used throughout this thesis are base-2 logarithms.

2.3 Channel Polarization

The channel polarization operation creates N synthetic channels $\{W_N^{(i)} : 1 \leq i \leq n\}$ from N independent copies of the B-DMC W [1]. The polarization phenomenon decomposes the symmetric capacity, $I(W_N^{(i)})$ of these synthetic channels towards 0 or 1 such that $I(W_N^{(i)}) \simeq 0$ implies that the i-th channel is completely noisy and $I(W_N^{(i)}) \simeq 1$ implies that the i-th channel is perfectly noiseless. The capacity separation enables an encoder to send information (free) bits through the noiseless channels and redundancy (frozen) bits through the noisy channels.

Let $\mathcal{A}$ be the information set and $\mathcal{A}^c$ be the frozen set. The input vector, $\mathbf{u}_1^N$ consists of both information bits $u_{\mathcal{A}}$ and frozen bits $u_{\mathcal{A}^c}$ such that $u_{\mathcal{A}} \in \mathcal{X}^K$ and $u_{\mathcal{A}^c} \in \mathcal{X}^{N-K}$, $K = |\mathbf{u}_1^K|$.

2.3.1 Channel Combining

A B-DMC W_N is generated by combining two independent copies of $W_{N/2}$. At the 0^{th} level of the recursion, we have only one copy of W and we set $W_1 \triangleq W$. The first level of the recursion combines two copies of W_1 for obtaining the channel $W_2 : \mathcal{X}^2 \rightarrow \mathcal{Y}^2$, as shown in Figure 2.2 [1] with transition probabilities,

$$W_2(y_1, y_2|u_1, u_2) = W(y_1|u_1 \oplus u_2)W(y_2|u_2), \tag{2-6}$$

with $\oplus$ being the binary operator XOR.

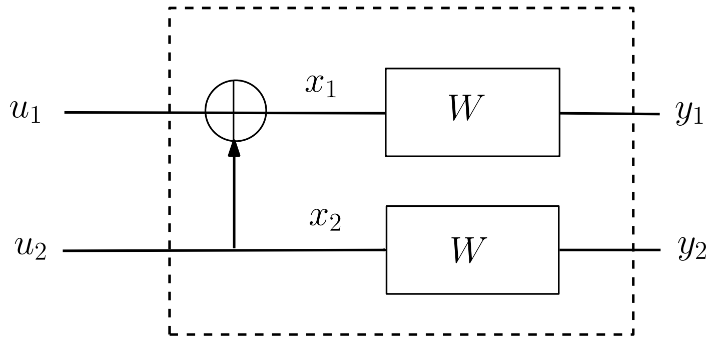


Figure 2.2: Construction of W_2 from W.

In a similar procedure, W_N is constructed recursively from $W_{N/2}$, $W_{N/4}$, ..., W_2, W in n steps, where $N = 2^n$ [1].

2.3.2 Channel Splitting

After the combination of two independent copies of W to create the vector channel $W_2 : \mathcal{X}^2 \rightarrow \mathcal{Y}^2$, we continue the channel polarization procedure by splitting back this channel into two B-DMCs, $W_1 : \mathcal{X} \rightarrow \mathcal{Y}^2$ and $W_2 : \mathcal{X} \rightarrow \mathcal{Y}^2 \times \mathcal{X}$, defined as [1]

$$W_2^{(1)}(y_1^2|u_1) \triangleq \frac{1}{2} \sum_{u_2 \in \mathcal{X}} W_2(y_1^2|u_1^2), \tag{2-7}$$

$$W_2^{(2)}(y_1^2, u_1|u_2) \triangleq \frac{1}{2} W_2(y_1^2|u_1^2), \tag{2-8}$$

for successive cancellation decoding, which will be defined in the next sections.

The transition probabilities are calculated in consecutive order from the top splitting operation to the bottom splitting operation, because the decision bit u_1 must be known before the bottom splitting operation.

Having synthesized the vector channel W_N, the next step is channel splitting. This involves splitting the vector channel W_N into N B-DMCs $W_N^{(i)} : \mathcal{X} \to \mathcal{Y}^N \times \mathcal{X}^{i-1}$, $1 \leq i \leq N$. The transition probability [1] of the bit-channel $W_N^{(i)}$ is defined as

$$W_N^{(i)}(y_1^N, u_1^{i-1}|u_i) \triangleq \sum_{u_{i+1}^N \in \mathcal{X}^{N-i}} \frac{1}{2^{N-1}} W_N(y_1^N|u_1^N). \tag{2-9}$$

Each pair of the new B-DMC channels is obtained after applying the single-step polar transform to one of the channels of the previous step:

$$(W_{N/2}^{(i)}, W_{N/2}^{(i)}) \to (W_N^{(2i-1)}, W_N^{(2i)}). \tag{2-10}$$

For the recursive channel transformation [1] we have $(W_{N/2}^{(i)}, W_{N/2}^{(i)}) \to (W_N^{(2i-1)}, W_N^{(2i)})$, $1 \leq i \leq \frac{N}{2}$. For the special case that W is a binary erasure channel (BEC) [1] with erasure probability ϵ, we have that

$$I(W_N^{(2i-1)}) = I(W_{N/2}^{(i)})^2, \tag{2-11}$$
$$I(W_N^{(2i)}) = 2I(W_{N/2}^{(i)}) - I(W_{N/2}^{(i)})^2, \tag{2-12}$$

where $I(W_1^{(1)}) = 1 - \epsilon$, and

$$Z(W_N^{(2i-1)}) = 2Z(W_{N/2}^{(i)}) - Z(W_{N/2}^{(i)})^2, \tag{2-13}$$
$$Z(W_N^{(2i)}) = Z(W_{N/2}^{(i)})^2, \tag{2-14}$$

where $Z(W_1^{(1)}) = \epsilon$.

The bit-channel $W_N^{(i)}$ is the channel that a successive cancellation decoder sees when decoding the i^{th} bit u_i with perfect knowledge of channel output $\mathbf{y}_1^N$.

2.4 Encoding of Polar Codes

Polar codes can be encoded by using a simple linear mapping. For the codeword with block length N, the generator matrix $\mathbf{G}_N$ is defined [1] as

$$\mathbf{G}_N = \mathbf{B}_n \mathbf{F}^{\otimes n}, \tag{2-15}$$

for any $N = 2^n$ as $n \geq 1$, where $\mathbf{B}_N$ is a bit-reversal matrix and $\mathbf{F}^{\otimes n}$ is the N^{th} Kronecker power of the matrix $\mathbf{F} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$. Below we show the construction of the arrays $\mathbf{F}^{\otimes 2}$ and $\mathbf{F}^{\otimes 3}$:

$$\mathbf{F}^{\otimes 2} = \begin{bmatrix} \mathbf{F} & 0 \\ \mathbf{F} & \mathbf{F} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

and

$$\mathbf{F}^{\otimes 3} = \begin{bmatrix} \mathbf{F}^{\otimes 2} & 0 \\ \mathbf{F}^{\otimes 2} & \mathbf{F}^{\otimes 2} \end{bmatrix} = \begin{bmatrix} \mathbf{F} & 0 & 0 & 0 \\ \mathbf{F} & \mathbf{F} & 0 & 0 \\ \mathbf{F} & 0 & \mathbf{F} & 0 \\ \mathbf{F} & \mathbf{F} & \mathbf{F} & \mathbf{F} \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

In [1], the input sequence $\mathbf{u}_1^N$ is permuted by a permutation matrix $\mathbf{B}_N$, i.e.,

$$\mathbf{x}_1^N = \mathbf{u}_1^N \mathbf{B}_N \mathbf{G}_N. \tag{2-16}$$

Since there are two ways of designing successive cancellation decoder graphs [1], the permutation only serves as a reordering of the indices of $(x_1, \ldots, x_N)$ and it does not affect the properties of polar codes, we skip this permutation for simplicity of presentation throughout the thesis.

A length K polar encoder has an input vector $\mathbf{u}_1^N$ and an output vector $\mathbf{x}_1^N$ with length N. The mapping of $\mathbf{u} \to \mathbf{x}$ is linear such that $\mathbf{x}_1^N = \mathbf{u}_1^N \mathbf{G}_N$. The rows of $\mathbf{G}_N$ are linearly independent and they form the basis for the code space. We exemplify below the encoding operation using the $\mathbf{G}_8$ generator matrix,

$$\begin{aligned} \mathbf{u}_1^4 &= [u_0, u_1, u_2, u_3] = [1, 1, 0, 1], \\ \mathbf{x}_1^8 &= [0, 0, 0, u_0, 0, u_1, u_2, u_3] \times \mathbf{G}_8, \\ &= [0, 0, 0, 1, 0, 1, 0, 1] \times \mathbf{G}_8, \\ \mathbf{x}_1^8 &= [0, 0, 1, 0, 0, 1, 0, 0]. \end{aligned}$$

In general, an N-bit encoder includes $\mathcal{O}(\frac{N}{2}\log_2 N)$ XOR operations.

2.5 Successive Cancellation Decoding

It has been proved in [1] that polar codes with SC decoding can achieve the capacity of B-DMCs. Since then, many other decoding algorithms have been proposed [9]-[13]; to improve the error rate performance in the finite-length regime. We review the SC decoding algorithm in this section.

The SC decoder estimates the transmitted bits $\mathbf{u}_1^N$ as $\hat{\mathbf{u}}_1^N$ by using the received vector $y_1^N \in \mathcal{Y}^N$ at the B-DMC, $W_N : \mathcal{X}^N \to \mathcal{Y}^N$. The channel likelihood information from the received vector is represented a log-likelihood ratio (LLR). The decoder performs soft-decision decoding as computing intermediate LLR values. After a sequence of LLR computations, the SC decoder computes $\hat{\mathbf{u}}_1^N$ hard decisions in a successive order from $\hat{u}_1$ to $\hat{u}_N$. In other words, $\hat{u}_i$ is decided according to $\hat{u}_1^{i-1}$ for $1 < i \leq N$.

A high-level description of the SC decoding algorithm is illustrated in Algorithm 1 [14] [15]. The algorithm takes the received vector $\mathbf{y}_1^N$, the code block length N and the information set $\mathcal{A}$ as input and calculates the estimated free bits $\hat{u}_{\mathcal{A}}$ as an output vector. There are N decision steps in the algorithm. If a hard decision belongs to the frozen set $\mathcal{A}^c$, the decision will be a frozen decision such that it is known by both encoder and decoder. Otherwise, the decoder sets its hard decision with respect to the soft decision information. After all N decisions are calculated, the output of the decoder is made by the hard decisions, which belong to the free set.

```
Algorithm 1: Successive Cancellation Decoding
Input: received codeword, $\mathbf{y}_1^N$
Input: code block length, $N$
Input: information set, $\mathcal{A}$
Input: frozen bit vector, $u_{\mathcal{A}^c}$
Output: estimated free bits, $\hat{u}_{\mathcal{A}}$
begin
    for $i = 1 : N$ do
        if $i \notin \mathcal{A}$ then
            $\hat{u}_i = 0$
        else
            if $\log\left(\frac{W_n^{(i)}(\boldsymbol{y}_1^n, \hat{u}_1^{i-1}|\hat{u}_i=0)}{W_n^{(i)}(\boldsymbol{y}_1^n, \hat{u}_1^{i-1}|\hat{u}_i=1)}\right) \geq 0$ then
                $\hat{u}_i = 0$
            else
                $\hat{u}_i = 1$
            end
        end
    end
    return $\hat{u}_{\mathcal{A}}$
end
```

Three different functions are defined to illustrate the behavior of the SC decoder. These functions are called f, g, and d. The f function is responsible for the calculation of top channel splitting operation. The f function, in likelihood ratio (L) representation, is

$$\begin{aligned} f(L_a, L_b) &= L_c, \\ &= \frac{W(y_1^2|\hat{u}_c = 0)}{W(y_1^2|\hat{u}_c = 1)}, \\ &= \frac{W(y_1|\hat{u}_a = 0)W(y_2|\hat{u}_b = 0) + W(y_1|\hat{u}_a = 1)W(y_2|\hat{u}_b = 1)}{W(y_1|\hat{u}_a = 0)W(y_2|\hat{u}_b = 1) + W(y_1|\hat{u}_a = 1)W(y_2|\hat{u}_b = 0)}, \\ &= \frac{L_a L_b + 1}{L_a + L_b}, \end{aligned} \tag{2-17}$$

where in (2-17) both numerator and denominator are divided by

$W(y_1|\hat{u}_a = 1)W(y_2|\hat{u}_b = 1)$. The f function, in LLR (γ) representation, is

$$\begin{aligned} f(\gamma_a, \gamma_b) &= \gamma_c, \\ &= 2\tanh^{-1}(\tanh(\frac{\gamma_a}{2})\tanh(\frac{\gamma_b}{2}), \\ &= \text{sign}(\gamma_a\gamma_b)\min(|\gamma_a|, |\gamma_b|), \end{aligned} \tag{2-18}$$

where sign function is signal function, the min-sum approximation is defined for BP decoding of LDPC codes [16] and this approximation was used in SC decoding of polar codes for the first time in [17].

The g function computes the bottom channel splitting operation in the SC decoder. The g function in likelihood ratio representation of soft decisions is

$$g(L_a, L_b) = \frac{W(y_1^2|\hat{u}=0)}{W(y_1^2|\hat{u}=1)}, \tag{2-19}$$

$$= \begin{cases} \frac{W(y_1|\hat{u}_a=0)W(y_2|\hat{u}_b=0)}{W(y_1|\hat{u}_a=1)W(y_2|\hat{u}_b=1)} & \text{if } \hat{u}=0, \\ \frac{W(y_1|\hat{u}_a=1)W(y_2|\hat{u}_b=0)}{W(y_1|\hat{u}_a=0)W(y_2|\hat{u}_b=1)} & \text{if } \hat{u}=1, \end{cases}$$

$$g(L_a, L_b, \hat{u}) = L_a^{(1-2\hat{u})}L_b. \tag{2-20}$$

The g function in LLR representation is

$$g(\gamma_a, \gamma_b, \hat{u}) = (-1)^{(\hat{u})}\gamma_a + \gamma_b. \tag{2-21}$$

Lastly, hard decisions are computed from soft decisions, such that

$$\hat{u}_i = \begin{cases} 0, & \text{if } \quad i \notin A, \\ 0, & \text{if } \quad i \in A \quad \text{and} \quad \frac{\mathrm{W(y,\hat{u}_1^{(i-1)}|\hat{u}_i=0}}{\mathrm{W(y,\hat{u}_1^{(i-1)}|\hat{u}_i=1}} \quad \geq \quad 1, \\ 1, & \text{otherwise.} \end{cases} \tag{2-22}$$

In (2-21), $\hat{u}$ represents the previously-decoded bit and corresponds to the propagation of decoded bits from left to right in the decoding graph [1].

2.6 Successive Cancellation List Decoding

The successive cancellation list (SCL) decoding algorithm has been introduced by Tal and Vardy [18] and generates $\mathcal{L}$ candidates for decoding or possibilities, increasing the probability that one of the results will be correct when compared to SC [19, 20]. With the use of CRC, this method has proven to be very effective for reducing the error rates of polar codes. If $\mathcal{L}$ is large enough, the performance of ML decoding is achieved because a sufficient number of decoding paths are visited [21].

Different from the selection of a particular path in each decision step in the SC decoding, the SCL extends two paths, $\hat{u}_i = 0$ and $\hat{u}_i = 1$, for each decision step. The complexity of SCL decoding which $\mathcal{O}(\mathcal{L}N \log N)$ is constrained by the size of the list. So, $\mathcal{L}$ more reliable paths are preserved at every decision step.

Let $\hat{u}_i[\tau]$ be the estimate $\hat{u}_i$ in the τth path, where $\tau \in 1, 2, ..., \mathcal{L}$. The path metric (PM) is used to measure the reliability of the paths. The PM of $\hat{u}_i[\tau]$ is approximated by

$$PM_1^{(i)} = PM_1^{(i-1)} + c_i[\tau] \cdot |\mathcal{L}_n^{(i)}[\tau]|, \tag{2-23}$$

where i is the index of the decision step, $PM_\tau^{(0)} = 0$, $c_i[\tau] = 0$ when $i \in A^c$, $|\mathcal{L}_n^{(i)}[\tau]|$ is the path branch value and $c_i[\tau] = 1$ otherwise.

Since u_i has two possible values, the $\mathcal{L}$ paths or candidates with lowest metrics are selected in the decision step. After the last bit has been decoded, the path with lowest metric is considered as the decoding result.

A high level description is shown in Algorithm 2. The SCL algorithm takes the received vector $\mathbf{y}_1^N$, the code block length N, the information set $\mathcal{A}$, the frozen bit vector $u_{\mathcal{A}^c}$ and the maximum list size $\mathcal{L}$ as input and calculates the estimated information bits $\hat{u}_{\mathcal{A}}$ as output. The current list size variable (c$\mathcal{L}$) is set to 1 at the initialization of the algorithm. If the i^{th} hard decision belongs to the frozen set $\mathcal{A}^c$, the i^{th} hard decisions of all $\mathcal{L}$ lists are updated with the frozen decision, u_i. In case of a free decision, the decoder checks whether the current list size is equal to the maximum list size. If they are not equal, the current list size doubles and the decoder can track likelihoods of both decisions. In case all lists are occupied, the decoder sorts $2\mathcal{L}$ likelihoods to continue with the best $\mathcal{L}$ decoding paths. At the end of the last decision step, the decoder outputs the free bits from the best entry in the list as $\hat{u}_{\mathcal{A}}$.

Algorithm 2: Successive Cancellation List Decoding

```
Input: received codeword, $\mathbf{y}_1^N$
Input: code block length, $N$
Input: information set, $\mathcal{A}$
Input: frozen bit vector, $u_{\mathcal{A}^c}$
Input: maximum list size, $\mathcal{L}$
Output: estimated free bits, $\hat{u}_{\mathcal{A}}$
begin
    c$\mathcal{L} = 1$, // current list size variable
    for $i = 1 : N$ do
        if $i \notin \mathcal{A}$ then
            for $\tau = 1 :$ c$\mathcal{L}$ do
                $\hat{u}_{\tau,i} = u_i$
            end
        else
            if c$\mathcal{L} \neq \mathcal{L}$ then
                for $\tau = 1 :$ c$\mathcal{L}$ do
                    $\hat{u}_{\tau,i} = 0$
                    $\hat{u}_{\tau+\mathrm{c}\mathcal{L},\mathrm{i}} = 1$
                end
                c$\mathcal{L} = 2$c$\mathcal{L}$
            else
                s = sort$(W_N^{(i)}(\mathbf{y}_1^N, \hat{u}_1^{i-1}|\hat{u}_{1,i}^{\mathcal{L}}))$
                for $\tau = 1 :$ c$\mathcal{L}$ do
                    $\hat{u}_{\tau,i} = s_\tau$
                end
            end
        end
    end
    return $\hat{u}_{\mathcal{A}}$
end
```

2.7 CRC-Aided Successive Cancellation List Decoding

The CRC-Aided Successive Cancellation List Decoding (CA-SCL) algorithm consists of SCL decoding with CRC. The aim of the algorithm is to increase the throughput of the SCL decoder [18, 19], [21]-[23].

A high level description of the algorithm is shown in Algorithm 3. Inputs of the adaptive SCL decoding algorithm are the received vector $\mathbf{y}_1^N$, the code block length N, the information set $\mathcal{A}$, the frozen bit vector $u_{\mathcal{A}^c}$ and the list size $\mathcal{L}$. The output of the algorithm is the free bit vector $\hat{u}_{\mathcal{A}}$.

At the beginning of the algorithm, the SC decoder calculates a free bit candidate vector. If the CRC of that vector is true, the algorithm terminates with the output of the SC decoder. In case of incorrect CRC vector, the algorithm calls an SCL algorithm with the list $\mathcal{L}$. At this time, the SCL algorithm calculates $\mathcal{L}$ hard decision candidate vectors. If one of them has a valid CRC, the algorithm terminates with that output. If none of them has a valid CRC, the algorithm terminates with the most probable hard decision candidate vector [24].

Algorithm 3: CRC-Aided Successive Cancellation List Decoding

```
   Input: received codeword, $\mathbf{y}_1^N$
   Input: code block length, $N$
   Input: information set, $\mathcal{A}$
   Input: frozen bit vector, $u_{\mathcal{A}^c}$
   Input: maximum list size, $\mathcal{L}$
   Output: estimated information bits, $\hat{u}_{\mathcal{A}}$
 1 begin
 2     $j = false$, // valid CRC vector of SC variable
 3     $k = false$, // valid CRC vector of SCL variable
 4     $\hat{u}_{\mathcal{A}} = Sucessive\ Cancellation\ Decoding\ (\mathbf{y}_1^N, N, \mathcal{A}, u_{\mathcal{A}^c})$
 5     $j = Cyclic\ Redundancy\ Check\ Decoding\ (\hat{u}_{\mathcal{A}})$
 6     if $j$ is true then
 7         return $\hat{u}_{\mathcal{A}}$
 8     else
 9         $\hat{u}_{l,\mathcal{A}} =$
             $Sucessive\ Cancellation\ List\ Decoding\ (\mathbf{y}_1^N, N, \mathcal{A}, u_{\mathcal{A}^c}, \mathcal{L})$
10         for $l = 1 : \mathcal{L}$ do
11             $k = Cyclic\ Redundancy\ Check\ Decoding\ (\hat{u}_{l,\mathcal{A}})$
12             if $k$ is true then
13                 $\hat{u}_{\mathcal{A}} = \hat{u}_{\tau,\mathcal{A}}$
14                 return $\hat{u}_{\mathcal{A}}$
15             end
16         end
17         $\hat{u}_{\mathcal{A}} = \hat{u}_{\tau,\mathcal{A}}$
18         return $\hat{u}_{\mathcal{A}}$
19     end
20 end
```

The CRC polynomial project [25] depends on the maximum total length N of the block to be designed considering data and CRC. According to [26], the maximum total length of the block is given by 2^{d-1}, where d is the degree of the generator polynomial. The CRC-8 is used for $N \leq 128$ and the CRC-16 is suggested for $N \leq 32768$.

2.8 Belief Propagation Decoding

The BP decoder is a message passing decoder with an iterative processing over the factor graph of any polar code [89]. The factor graph of BP decoder [89] is based on corresponding polarization matrix $\mathbf{G}^N$, composed of n stages, each one with $N/2$ processing elements (PEs), and $(n+1)N$ nodes. Two types of LLRs are transmitted over the factor graph: the left-to-right message $R_{i,j}^{(t)}$ and the right-to-left message $L_{i,j}^{(t)}$, where i and j denotes the j^{th} node at the i^{th} stage whereas t denote the t^{th} iteration.

In the LLR domain, the input LLRs for BP decoding of polar codes are initialized as:

$$L_{n+1,j}^{(0)} = \ln \frac{P_r(x_j = 0|y_j)}{P_r(x_j = 1|y_j)} = \frac{2y_j}{\sigma^2}, \tag{2-24.1}$$

$$R_{1,j}^{(0)} = \begin{cases} 0 \quad \text{if j} \in \mathcal{A}, \\ +\infty \quad \text{if j} \in \mathcal{A}^c, \end{cases} \tag{2-24.2}$$

where x_j and y_j denote the j^{th} bit of modulated and received vector, respectively.

The forward and backward propagation of the LLRs over the PEs, shown in Figure 2.3, is based on the following iterative updating rules:

$$\begin{aligned} L_{i,j}^{(t)} &= g(L_{i+1,j}^{(t-1)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}), \\ L_{i,j+N/2^i}^{(t)} &= g(L_{i+1,j}^{(t-1)}, R_{i,j}^{(t)}) + L_{i+1,j+N/2^i}^{(t-1)}, \\ R_{i+1,j}^{(t)} &= g(R_{i,j}^{(t)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}), \\ R_{i+1,j+N/2^i}^{(t)} &= g(L_{i+1,j}^{(t-1)} + R_{i,j}^{(t)}) + R_{i,j+N/2^i}^{(t)}, \end{aligned} \tag{2-25}$$

where $g(a,b)$ is referred to as the operator:

$$\begin{aligned} g(a,b) &= \ln \frac{1+e^{a+b}}{e^a + e^b}, \\ g(a,b) &= 0.9375 \cdot sign(a) \cdot sign(b) \cdot \min(|a|,|b|). \end{aligned} \tag{2-26}$$

When the maximum number of iterations (T_{max}) is reached, the information bit $\hat{u}_j$ and the transmitted codeword $\mathbf{x}_j$ are estimated based on their

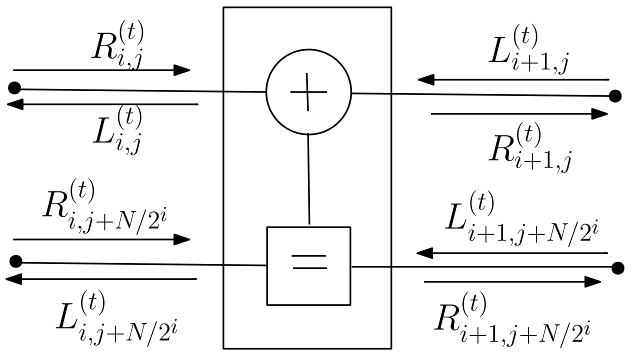


Figure 2.3: Processing element update.

LLRs using the following hard decision criteria:

$$\hat{u}_j = \begin{cases} 0, & \text{if } L_{1,j}^{\mathrm{T}_{max}} + R_{1,j}^{\mathrm{T}_{max}} > 0, \\ 1, & \text{otherwise,} \end{cases} \tag{2-27.1}$$

$$\hat{x}_j = \begin{cases} 0, & \text{if } L_{n+1,j}^{\mathrm{T}_{max}} + R_{n+1,j}^{\mathrm{T}_{max}} > 0, \\ 1, & \text{otherwise.} \end{cases} \tag{2-27.2}$$

A pseudocode of the BP decoding algorithm can be seen in Algorithm 4.

Algorithm 4: Belief Propagation Decoding Algorithm

```
Input: maximum loop, $\mathrm{T}_{max}$
Input: received codeword, $\mathbf{y}$
Input: code block length, $N$
Input: matrix generate, $\mathbf{G}_N$
Output: estimated codework, $\hat{\mathbf{x}}$
1  begin
2      $j = 1$
3      while $j < T_{max}$ do
4          update the message L
5          update the message R
6          if $\hat{\boldsymbol{x}}\boldsymbol{G}^T = 0$ then
7              return $\hat{\boldsymbol{x}}$
8          else
9              $j = j + 1$
10     end
11     if $j > T_{max}$ then
12         return $\hat{\boldsymbol{x}}$
13     end
14 end
```

2.8.1 LDPC-Like Decoder for Polar Codes

The parity check matrix $\mathbf{H}$ of polar codes can be constructed from the corresponding generator matrix $\mathbf{G}_N$ [27]. The conventional factor graph of the BP decoder is converted into the bipartite Tanner graph in a similar way to LDPC codes. Then a pruning process is applied to make the graph sparse [27]. Sparse means that the number of zeros in the parity check matrix $\mathbf{H}$ is much higher than the number of ones. The rows of $\mathbf{H}$ are called check nodes (C) and the columns of $\mathbf{H}$ are called variable nodes (V), which are usually represented graphically by a Tanner graph [28]. Figure 2.4 shows a Tanner graph and its parity check matrix $\mathbf{H}$.

$$H = \begin{array}{c} \begin{array}{ccccccc} V_1 & V_2 & V_3 & V_4 & V_5 & V_6 & V_7 \end{array} \\ \left(\begin{array}{ccccccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right) \end{array} \begin{array}{c} \\ C_1 \\ C_2 \\ C_3 \end{array}$$

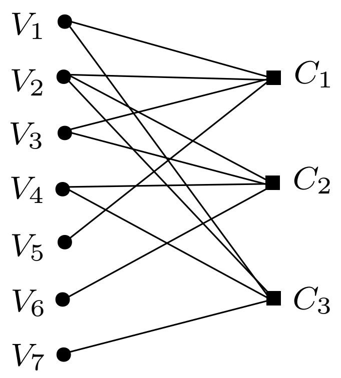


Figure 2.4: Tanner graph and parity check matrix $\mathbf{H}$.

The LDPC-like decoder for polar codes is based on the Sum-Product Algorithm (SPA), which is a message-passing decoder with iterative processing over the factor graph, in which the LLRs about the coded bits are exchanged along the edges of the Tanner graph. First, each V gets its corresponding LLR in the received vector, which we denote here the V's intrinsic LLR. Each V represents a bit in the codeword. In the beginning, Vs send their LLRs to their connected Cs. Each C processes all messages sent from its connected Vs, and finally predicts a value for each V. Each V then gets all these suggested values from its connected Cs, adds them all together and also to its intrinsic LLR, and obtains a more precise LLR. This concludes one iteration. If the stopping criterion is not met, each V computes and sends its so-far computed LLR (with little change) to each C and the iterations continue, until the stopping criterion is met. As the algorithm proceeds, the computed LLR for each V gets closer to a fixed point. This process is repeated until a codeword is found or the maximum number of iterations is reached. The key observation is that the processing at the Cs and Vs depends only on locally available information, which allows for an efficient and parallelizable decoding process.

In terms of notation, if the cth C is connected to the vth V, the message from that C to that V is shown as $\Lambda_{c\to v}$, and the message in the opposite direction is $\lambda_{v\to c}$. At the beginning of the algorithm, the messages are initialized as follows:

$$\Lambda_{c\to v} = \mathbf{0} \tag{2-28}$$

and

$$\lambda'_{v\to c} = \mathbf{y}, \tag{2-29}$$

where $\lambda'_{v\to c}$ is the initial condition. Each iteration of the algorithm executes the following three steps: C update, V update and test stopping criterion.

For the C update, all the Cs produce their messages to all their connected Vs. The message from cth C to the vth V can be computed with

$$\Lambda_{c\to v} = 2 \cdot \tanh^{-1}\left(\prod_{k=1, v\neq v'}^{\mathrm{V}} \tanh\left(\frac{\lambda_{v\to c}}{2}\right)\right). \tag{2-30}$$

The notation $v' \neq v$ in (2-30) means that to compute the message to a certain V, the message from all the other connected Vs is taken into account, except for the message that has come from that V.

For the V update, all the Vs produce their messages to all their connected Cs. The message from vth V to the cth C is computed according to

$$\lambda_{v\to c} = \lambda'_{v\to c} + \sum_{k=1}^{\mathrm{C}} \Lambda_{c\to v}. \tag{2-31}$$

Until the stopping criterion, at each iteration, a more precise estimate of the transmitted codeword is expected to be achieved. The obtained $\hat{\mathbf{x}}$ at the end of each iteration is determined as

$$\hat{\mathrm{x}} = \begin{cases} 1, & \lambda_{v\to c} \leq 0, \\ 0, & \lambda_{v\to c} > 0. \end{cases} \tag{2-32}$$

If the test stopping criterion is not reached, $\lambda'_{v\to c}$ is updated

$$\lambda'_{v\to c} = \lambda_{v\to c} \tag{2-33}$$

and we have a new C update and V update.

If the test stopping criterion is reached, but $i < \mathrm{V}$, then $i = i + 1$ and $\lambda'_{v\to c}$ is updated with a new codeword

$$\lambda'_{v\to c} = \mathbf{y} \tag{2-34}$$

and a new cycle of steps begins.

A high level description of the LDPC-like decoder can be seen in Algorithm 5.

Algorithm 5: LDPC-Like decoder of polar codes

```
Input: maximum loop, $T_{max}$
Input: received codeword, $\mathbf{y}$
Input: sparse matrix of $\mathbf{G}_N$, $\mathbf{H}$
Input: number of Cs, C
Input: number of Vs, V
Output: estimated codework, $\hat{\mathbf{x}}$
1  begin
2      T = 0; Finished = 0
3      for $i$ = 1:C do
4          for $j$ = 1:V do
5              $\lambda'(\mathrm{i,j})_{v\to c} = y(\mathrm{j}) * H(\mathrm{i,j})$
6          end
7      end
8      repeat
9          for $i$ = 1:C do
10             for $j$ = 1:V do
11                 $\Lambda(\mathrm{i,j})_{v\to c} = 2 \cdot \tanh^{-1}\left(\prod_{k=1,k\neq \mathrm{j}}^{V} tanh\left(\frac{\lambda'(\mathrm{i},k)_{v\to c}}{2}\right)\right)$
12             end
13         end
14         for $j$ = 1:V do
15             $\lambda_{v\to c} = \lambda'_{v\to c} + \sum_{n=1}^{N} \Lambda_{c\to v}$
16             $\hat{x}(\mathrm{j}) = \begin{cases} 1, & \lambda(\mathrm{i,j})_{v\to c} \leq 0 \\ 0, & \lambda(\mathrm{i,j})_{v\to c} > 0, \end{cases}$
17         end
18         for $i$ = 1:C do
19             for $j$ = 1:V do
20                 $\lambda'(\mathrm{i,j})_{v\to c} = \lambda(\mathrm{i,j})_{v\to c}$
21             end
22         end
23     until $T = T_{max}$ or $\hat{\boldsymbol{x}}\boldsymbol{H}^T = 0$;
24     return $\hat{\boldsymbol{x}}$
25 end
```

2.9 Polar Codes Construction

The aim of the polar codes construction is to determine the sets $\mathcal{A}$ and $\mathcal{A}^c$ according to the capacity of the individual channels [1] [29]. Since polar codes are channel-specific codes, the code construction may differ from channel to channel. Channel parameters such as σ for AWGN and ϵ for BEC are inputs to a code construction method. In general, the recursive equation for calculating top and bottom channels is (2-11) and (2-12).

For the BEC channel W, the construction for polar codes (PC) with $N = 8$, $K = 4$, which can be written PC(8,4), and with erasure probability ϵ = 0.3 is shown in Figure 2.5, according to Algorithm 6.

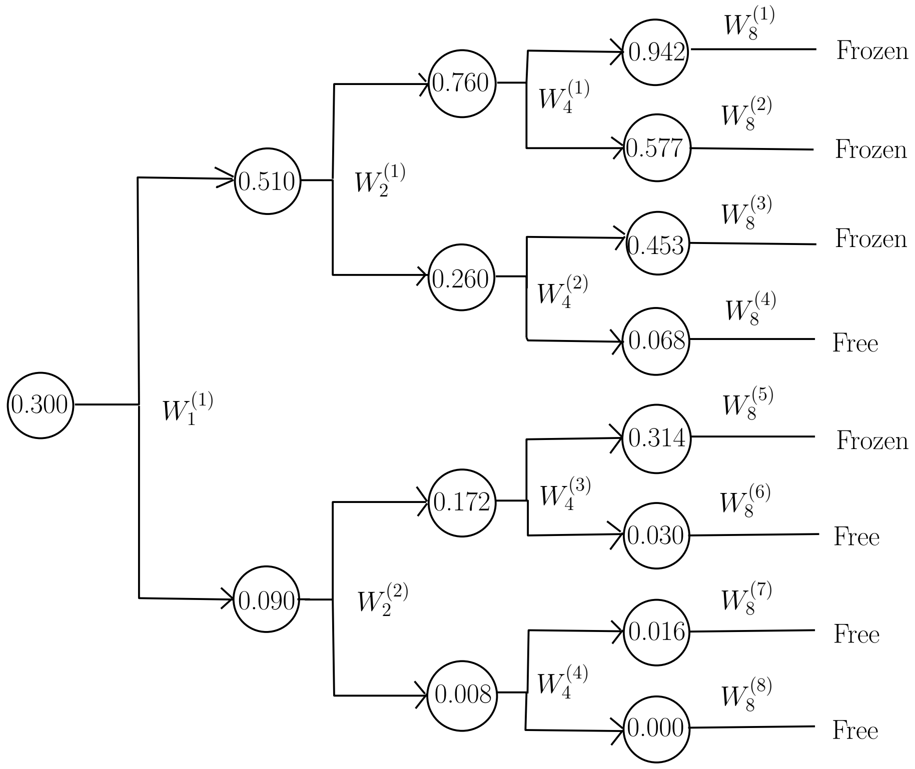


Figure 2.5: Construction of PC(8,4) for BEC with ϵ=0.3.

Initially, the reliability of the smallest channel, $W_1^{(1)}$, is set as $\epsilon = Z_0$ = 0.3. After that the reliability of the first channel, $W_2^{(1)}$, is calculated as $Z(W_2^{(1)}) = 2Z(W_1^{(1)}) - Z(W_1^{(1)})^2$, where $Z(W_n^{(i)})$ is the erasure probability of the i-th length n channel starting from top. At the same time, the second channel can be calculated as $Z(W_2^{(2)}) = Z(W_1^{(1)})^2$. The polar code is designed by selecting the $|\mathcal{A}|$ channel to transmit information bits such that $Z(W_N^{(i)}) \leq Z(W_N^{(j)})$, with $i \in \mathcal{A}$ and $j \in \mathcal{A}^c$.

At the end of stage n (in this case $n = 3$), the erasure probability of all length-n channels appears. At this point, the channels which have the lowest K erasure probabilities are set as free and the others are set as frozen. The algorithm has n stages and performs the calculations. Here is the numerical example of the polarization stages:

Algorithm 6: Polar codes Construction Algorithm for the BEC Channel

```
Input: code length, $N$
Input: information bits number, $K$
Input: erasure probability, $\epsilon$
Output: $\mathrm{F} = \{0, 1, \ldots, N-1\}$ with $|\mathrm{F}| = N$
1  begin
2      $n = log_2 N$
3      $Z \in \mathbb{R}^N, Z(0) = \epsilon$
4      for $i = 1$ to $n$ do
5          $d = 2^i$
6          for $j = 1$ to $\frac{d}{2} - 1$ do
7              $Z(d/2 + \mathrm{j}) = 2Z(\mathrm{j}-1) - Z(\mathrm{j}-1)^2$
8              $Z(\mathrm{j}) = Z(\mathrm{j}-1)^2$
9          end
10     end
11     return $F$ = Sorts $Z$ indices in ascending order
12 end
```

- stage 1:
  $Z(W_2^{(1)}) = 2(Z_0) - (Z_0)^2 = 2(0.3) - (0.3)^2 = 0.51$ and
  $Z(W_2^{(2)}) = (Z_0)^2 = (0.5)^2 = 0.09$.
- stage 2:
  $Z(W_4^{(1)}) = 2Z(W_2^{(1)}) - Z(W_2^{(1)})^2 = 0.76$,
  $Z(W_4^{(2)}) = 2Z(W_2^{(2)}) - Z(W_2^{(2)})^2 = 0.172$,
  $Z(W_4^{(3)}) = Z(W_2^{(1)})^2 = 0.265$ and
  $Z(W_4^{(4)}) = Z(W_2^{(2)})^2 = 0.008$.
- stage 3, by induction:
  $Z(W_4^{(1)}) = 0.942$,
  $Z(W_4^{(2)}) = 0.577$,
  $Z(W_4^{(3)}) = 0.453$,
  $Z(W_4^{(4)}) = 0.068$,
  $Z(W_4^{(5)}) = 0.314$,
  $Z(W_4^{(6)}) = 0.03$,
  $Z(W_4^{(7)}) = 0.016$ and
  $\mathrm{Z}(\mathrm{W}_4^{(8)}) \approx 0$.

The channels $W_4^{(1)}, W_4^{(2)}, W_4^{(3)}, W_4^{(4)}, W_4^{(5)}, W_4^{(6)}, W_4^{(7)}, W_4^{(8)})$ can be written with $(W_0, W_1, W_2, W_3, W_4, W_5, W_6, W_7)$. We define the polarization vector as

$$\mathbf{b} \triangleq \Big[\ Z(W_0); Z(W_1); \ldots; Z(W_{N-1})\ \Big]^T. \tag{2-35}$$

An example for stage 3 we have $\mathbf{b} = [0.942, 0.577, 0.453, 0.068, 0.314, 0.03, 0.016, 0.00001]^T$. For K=4, $\mathcal{A} = [0, 1, 2, 4]$ and $\mathcal{A}^c = [3, 5, 6, 7]$.

The factor graph representation of an 8-bit encoder is shown in Figure 2.6. Note that the channel quality determines to which channels will be allocated data and which ones will be frozen. Figure 2.6 is a graphic representation of the operation performed by equation (2-16), with **u**=[1,1,0,1] and **x**=[1,1,0,0,0,0,1,1].

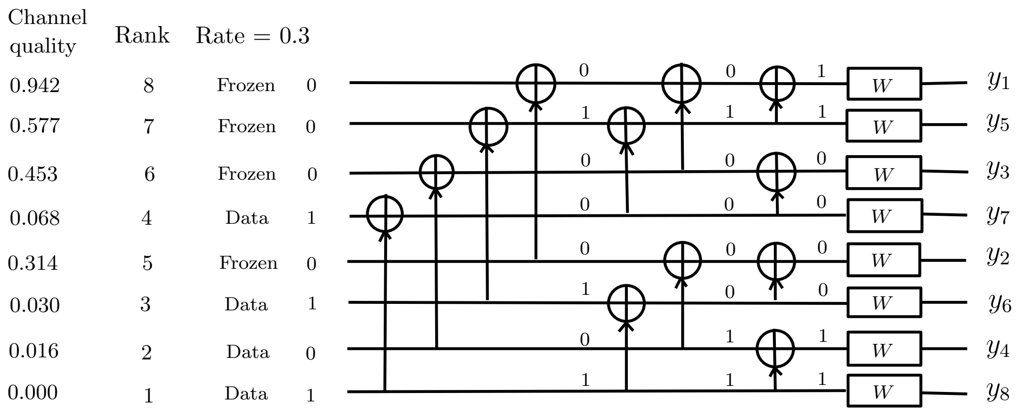


Figure 2.6: Encoding procedure for PC(8,4) for BEC with $\epsilon = 0.3$.

The following equations are obtained for codeword **x**:

$$\begin{aligned}
x_1 &= u_1 \oplus u_2 \oplus u_3 \oplus u_4 \oplus u_5 \oplus u_6 \oplus u_7 \oplus u_8,\\
x_2 &= u_5 \oplus u_6 \oplus u_7 \oplus u_8,\\
x_3 &= u_3 \oplus u_4 \oplus u_7 \oplus u_8,\\
x_4 &= u_7 \oplus u_8,\\
x_5 &= u_2 \oplus u_4 \oplus u_6 \oplus u_8,\\
x_6 &= u_6 \oplus u_8,\\
x_7 &= u_4 \oplus u_8,\\
x_8 &= u_8. \qquad (2\text{-}36)
\end{aligned}$$

The information bits can be observed at the output of a polar encoder using non-systematic encoding of polar codes [1].

2.10 State-of-the-Art

Polar codes are the latest breakthrough in coding theory, as they are the first capacity achieving channel codes with low-complexity encoding and decoding algorithms. Recent research has shown the possibility of applying polar codes and the polarization phenomenon in various signal processing and communication problems [3]. Although the aforementioned aspects of polar codes are very attractive, the issue of code design and the low computational complexity of decoding need improvement.

Some artificial intelligence techniques are currently being investigated for code design. The usage of genetic algorithms and reinforcement learning in the polar codes construction allows a more efficient design, as they take into account the joint optimization of the decoder and channel types [99]. The results obtained for the BP decoder and the AWGN channel are superior to previous techniques. These optimization techniques are flexible and can be used in a wide range of scenarios by matching decoder types and channel types.

As for the computational complexity of decoding, the SC decoder can obtain good performance with low complexity $\mathcal{O}(N \log N)$, but requires a larger block length N, while SCL decoding performs well with higher complexity $\mathcal{O}(\mathcal{L}N \log N)$. Most researchers focus on providing better error correction performance via using all possible computational resources, such as high frequency multi-core central processing unit (CPU) and graphics processing unit (GPU) [24]. With the proposed improvements in BP [92] decoding, its performance was comparable to that of the SCL decoder, with the advantage of allowing parallel decoding and with the complexity of $\mathcal{O}(N)$. Recently, with the possibility of using SPA for decoding polar codes [27], new possibilities are open to research, among them the joint decoding of LDPC and polar codes.

2.11 Chapter Summary

In this chapter the fundamental concepts of polar codes have been described. A system model has been introduced and some key theoretical aspects of channel polarization have been demonstrated. We have also covered the encoding procedure of polar codes, the successive cancellation decoding scheme and its variants, and the belief propagation decoding scheme and its variants. Then, the construction of polar codes has been introduced and the state-of-the-art in polar codes has been briefly reviewed.

3
Rate-Compatible Polar Codes Based on Non-Uniform Channel Polarization

3.1
Chapter Overview

In this chapter we introduce an new non-uniform channel polarization (NUC) theory in the construction of polar codes of arbitrary length [84]. We present a generalization of the polarization theory for non-uniform channels and we present the technique of construction by Non-Uniform Polarization based on the Gaussian Approximation (NUPGA). In this scenario, the main aspects of the channel polarization theory are maintained, namely, the conservation of the associated channel capacity and the induction to the polarized channel [1].

3.2
Introduction and Literature Review

In [1] it has been observed that the polar codes construction, in its standard form, has the limitation in code length be a power of two, i.e., $N = 2^n$. However, code length flexibility is required for practical applications. There are several construction techniques applied in the standard polar codes model proposed by Arıkan [1] considering the SC decoder. Among the main polar codes construction techniques are: the Bhattacharyya parameter [1], density evolution (DE) [14] and [29, 30], GA [15] and the polarization weight (PW) [31].

To estimate the channel reliability, the study in [1] proposed two methods, the evolution of the Bhattacharyya parameter and Monte Carlo (MC) simulations. In the MC method, the estimate of the channel reliability is obtained by simulations that consider the channel type and the decoder type. For each simulation loop an encoding and decoding step is necessary, which increases the complexity of the method. In the DE method, we need to compute the probability density function (PDF) of the log-likelihood-ratio of each channel first and then choose the channels that are most likely to be correct. The polar codes construction by the DE method is complex due to the convolution functions, and its precision depends on the degree of complexity of the implementation of the convolution functions. Similar to the MC method, the DE method has high computational cost.

The work in [29] proposes upper and lower bounds to simplify the DE method for estimating the channel reliability using two approximation methods. For AWGN channels, the study in [15] proposed the GA method, which is much less complex than DE and has good precision for short code lengths. With lower complexity and with performance similar to GA, PW [31] is a construction method that implements the partial order theory (PO) [33]. Remark that in polar codes construction, the estimate of the channel reliability depends on the channel type [1]. PO is based on the observation that the channel orderings are degraded according to the binary expansion of their indexes. It is worth to note that PO is universal and independent of the channel model. PO provides quality information for all channels. The use of PO simplifies the polar codes construction. Both in [34, 35] show a comparative study of the polar codes construction, in term of algorithm complexity and performance. The scenario used is the AWGN channel and SC and SCL decoders for various rates and codewords. A performance study on the AWGN channel of the GA method in polar codes construction can be found at [36, 37]. In [34], good design of polar codes was verified with several construction methods, with SC decoding and for various scenarios varying both the code length and the code rate. Polar codes can also be constructed and adapted to a specific decoder, for example, construction of polar codes for SCL decoding [38] and BP decoding [39, 40]. In [41] the authors propose a genetic algorithm framework that jointly optimizes the polar codes construction and rate with a specific decoder.

Polar codes are limited to code lengths given by $N = 2^n$, with $n > 1$. We can group the polar codes construction of variable length in several techniques, the main ones being: arbitrary kernels techniques, multi-kernel techniques (MK), puncturing techniques, shortening techniques and extension techniques are chosen to obtain a length of $2^{n-1} < M < 2^n$. Polarization matrices of various sizes, for example 3x3, 5x5 and 7x7, have been used to construct polar codes of any length. Bose–Chaudhuri–Hocquenghem (BCH) kernel matrices proposed in [42] and the code decompositions proposed in [43] both have restrictions on the size of the kernel matrices. Square polarizing kernels larger than two have been proposed in [44]-[46], while a polar codes construction with mixed kernel sizes has been proposed in [47, 48]. By considering different polarization kernels of alternate dimensions, MK improves block length flexibility. Although the general coding and decoding structure follows the same structure of standard polar codes, there is an increased complexity with the generalization. The polar codes construction using the Reed-Muller (RM) rule [49]-[51] can improve the performance in terms of the error rate.

The main shortening and puncturing techniques can be found in [32], [52]-[62]. Generally, puncturing or shortening degrade the code performance because when the number of bits punctured or shortened increases, the code length decreases. In [52] a study on the main puncturing and shortening techniques is carried out, including the column weights (CW) and the reversal quasi-uniform puncturing scheme (RQUP). Puncturing techniques are applied in scenarios where the decoding is BP. We find in [53, 54] the main studies on punctured polar codes. Among the techniques used we can mention the reduced generator matrix, exponent connection, minimum distance, stop tree drilling and schemes applied to hybrid automatic repeat request (HARQ). Several of these can be found at [55, 56]. In [32, 57] we have a performance analysis of puncturing codes based on the DE construction technique. The shortening techniques are applied to the construction of polar codes with SC and SCL decoding. As in the scheme we have adopted, the decoder is SC or SCL, it is a fact that the shortened bits are known. So, in these shortened bit positions, an LLR is defined as infinite. An efficient shortening method is reported in [58], where the shortened bits set is optimized simultaneously with the frozen bits set. In [59] a shortening method is presented and presents good results. The technique is based on using CW to reduce the size of the generating matrix. The main technique for reducing the generating matrix used in the shortened polar codes was proposed by [60] and is known as RQUP. The polarization-driven (PD) shortening technique has been presented in [61] based on the reduction of the generator matrix along with a strategy for the choice of bits shortened according to the channel polarization indices associated with the line index of the generator matrix. Recently, the PW algorithm has been used in a puncturing and shortening technique as reported in [62].

Moreover, PC has the potential for Internet of Things (IoT) applications in HARQ schemes [63]-[65]. In the HARQ mechanism [66], the original message is retransmitted by sending some specifically constructed codewords. In many cases the message is extended with bits of additional redundancy, maintaining the original sequence of the information bits. In terms of complexity, it is similar to the standard polar codes, both in the encoder and in the decoder. Nevertheless, there is a significant increase in complexity when designing flexible-length polar codes by concatenated codes [67], [68] and by asymmetric kernel construction [69]. In both cases the polar codes construction is specific to each kernel dimension without generalization gains. In [70] a chained polar sub-code technique is presented for effective polar codes construction. The author in [66] proposes an optimized polarizing matrix extension method. In [72] the author proposed that an arbitrary number of incremental coded bits, generated by extending the polarization matrix such that multiple re-transmissions are aggregated to produce a longer polar codes with extra coding gain. The author in [71] proposes to extend the polar codes in order to maximize the throughput of the system through the choice of a finite set of linear combinations of message bits. In [73] the author proposes three alternatives for polar codes extension, based on a matrix union arrangement and re-polarization by the method of Tal-Vardy [29], with gains in terms of coding and decoding complexity.

In the works of [103] and [104] methods are proposed to generalize the channel polarization in scenarios of parallel transmission or when the channel parameter is unknown. In [103], a construction technique for multichannel polar codes has been reported, including a scheme for modulation and bit interleaving, resulting in rate-matching compatibility. The scheme proposed in [104] deals with scenarios with parallel channels and random channel parameters. In uniform polarization, the same value of the Bhattacharyya parameter is given for all channels according to the Arıkan construction [1]. In terms of construction, it is equivalent to the use of the same LLR defined for all channels. In the proposed NUPGA technique, we prove that even if we assign different LLR values to the channel, which characterizes its non-uniformity, the validity of the channel polarization principle is guaranteed. Therefore, it is possible to construct polar codes of arbitrary length while maintaining their rate-compatibility.

Then, we present an algorithm for PC shortening and also an algorithm for PC extension, both based on the NUPGA technique. The NUPGA-based algorithms jointly implement the technique for shortened or extended channels with the re-polarization of the channels. We also present a generalization of the construction algorithm, which is used for both polarization of the initial channel and re-polarization of the shortened PC, and a simplified construction technique for extended polar codes. The existing techniques are compared with the proposed NUPGA technique in various simulations exploring different combinations of code length and code rate. A key feature of the proposed designs is that the encoder and decoder structures are the same as that of the original polar codes [1], thus having the same complexity.

3.3 Fundamentals of Polar Codes Construction

The key idea behind the construction of polar codes is to find the best channels for transmitting bits of information. Based on the channel polarization phenomenon, we observed that after implementing a linear transformation on the W channel inputs, the effective channels seen by some of the bits are better than the original W channel and others get worse. In other words, these channels are polarized as the code length N increases and tends towards a perfect channel (capacity 1) or completely noisy (capacity 0). These polarized channels are well conditioned for transmission: the perfect channels to send the information bits and we set to zero the bits sent over the other channels.

For the construction of arbitrary-length polar codes, a generalization of channel polarization is necessary for the definition of non-uniform polarization, maintaining the primary results of the channel polarization theory.

We generalize the channels and prove that the total capacity is maintained, that is, that the channel polarization theory is valid for non-uniform channels.

3.3.1 Channel Capacity

In Figure 3.1 we show a B-DMC channel, designated by the symbol W, with input U and output Y.

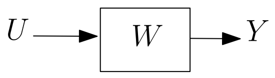


Figure 3.1: The channel W.

The input symbol U on B-DMC channel is considered a discrete random variable. Similarly, the symbol at the output of the channel is modeled by another discrete random variable Y. Then, a set of B-DMC channels from Figure 3.1 can be shown as in Figure 3.2a. The Bhattacharyya parameter is $Z(W)$ for all B-DMC channels. Consider the transmission of N different symbols $[\mathrm{U}_1\mathrm{U}_2 \cdots \mathrm{U}_N]$ through the channel in a serial manner. These symbols that are transmitted, in our modeling are considered independent and identically distributed (i.i.d.) random variables. Without loss of generality, we consider that the transmission of each symbol is through each channel separately, as in Figure 3.2a, where $\mathbf{U} = [U_1\ U_2\ \cdots\ U_N]$ and $\mathbf{Y} = [Y_1\ Y_2\ \cdots\ Y_N]$. Therefore, the deduction of the system capacity will be the same as if we use other channels, that is, non-uniform channels, as suggested in Figure 3.2(b).

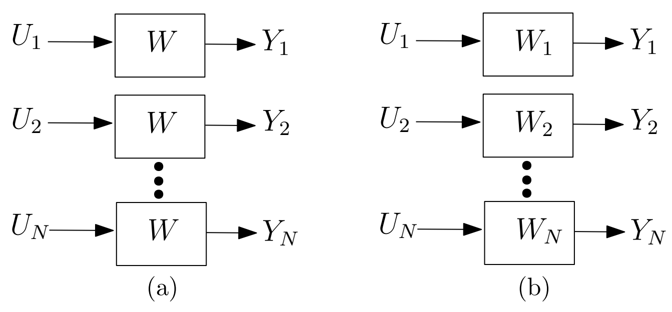


Figure 3.2: (a) Uniform B-DMC and (b) non-uniform B-DMC.

Now, the Bhattacharyya parameter is different for all B-DMC channels. The mutual information for Figure 3.2a and Figure 3.2b are shown below.

The mutual information of the uniform channel is given by

$$\begin{aligned} I(\mathbf{U};\mathbf{Y}) = I(U_1;\mathbf{Y}) + I(U_2;\mathbf{Y}|U_1) + I(U_3;\mathbf{Y}|U_1,U_2) \\ + \cdots + I(U_N;\mathbf{Y}|U_1,U_2,\cdots,U_{N-1}). \end{aligned} \tag{3-1}$$

We consider U_1 and Y_2, Y_3, $\cdots$, Y_N independent from each other, so we have:

$$\begin{aligned} I(U_1;\mathbf{Y}) &= I(U_1;Y_1), \\ I(U_2;\mathbf{Y}|U_1) &= I(U_2;Y_2), \\ I(U_3;\mathbf{Y}|U_1,U_2) &= I(U_3;Y_3), \\ I(U_N;\mathbf{Y}|U_1,U_2,\cdots,U_{N-1}) &= I(U_N;Y_N). \end{aligned}$$

Then, (3-1) can be written as

$$I(\mathbf{U};\mathbf{Y}) = I(U_1;Y_1) + I(U_2;Y_2) + I(U_3;Y_3) + \cdots + I(U_N;Y_N). \tag{3-2}$$

Let the capacity be $\mathcal{C} = \max I(\mathbf{U};\mathbf{Y})$, then we have

$$\begin{aligned} \max I(\mathbf{U};\mathbf{Y}) &= \max I(U_1;Y_1) + \max I(U_2;Y_2) \\ &\quad + \cdots + \max I(U_N;Y_N), \\ &= N\mathcal{C}. \end{aligned} \tag{3-3}$$

That is, (3-2) cannot be simplified as in (3-3). In addition, it is necessary to ensure that

$$\begin{aligned} I(U_1;\mathbf{Y}) < I(U_3;\mathbf{Y}|U_1,U_2) \leq I(U_2;\mathbf{Y}|U_1) \\ < \cdots < I(U_N;\mathbf{Y}|U_1,U_2,\cdots,U_{N-1}), \end{aligned} \tag{3-4}$$

which means that the individual capacities increase in an orderly manner but the total capacity remains constant, i.e., the total capacity of the channels is maintained, regardless of whether the channels are equal or not, that is, uniform or non-uniform with different Bhattacharyya parameters. Then, for uniform channels, according to (3-3), and for non-uniform channels, with the inequality of (3-4), the capacity of the channels is conserved, and we show that non-uniform polarization schemes achieve symmetric capacity:

$$\max I(\mathbf{U};\mathbf{Y}) = \sum_{i=1}^{N} \mathcal{C}_i = N\mathcal{C}. \tag{3-5}$$

Therefore, we can proposed methods to construct polar codes that take into account different Bhattacharyya parameters. Then we verify the convergence of the polarization theory [1] for NUC. The channel capacity, channel polarization and polarization convergence will be further studied in the next sections.

3.3.2 Uniform Construction

The channel polarization can be graphically represented by a construction tree [1], as show in Figure 3.3. We can see that a unique value of W is used in the construction, which is actually a simplification for the parameter $Z(W)$.

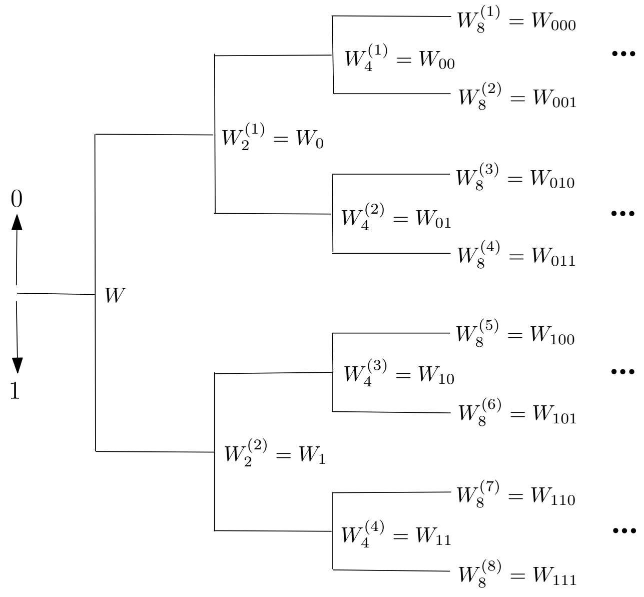


Figure 3.3: The channel construction tree.

On the BEC channel, for example, being $(W_N^{(i)}, W_N^{(i)}) \rightarrow (W_{2N}^{(2i-1)}, W_{2N}^{(2i)})$ produces two B-DMC:

$$\begin{aligned}(W_N^{(i)}, W_N^{(i)}) &\rightarrow (W_{2N}^{(2i-1)}, W_{2N}^{(2i)}),\\ Z(W_{2N}^{(2i)}) &\leq 2Z(W_N^{(i)}) - Z(W_N^{(i)})^2,\\ Z(W_{2N}^{(2i-1)}) &\leq Z(W_N^{(i)})^2,\\ Z(W_{2N}^{(2i-1)}) &\leq Z(W_{2N}^{(2i)}),\end{aligned} \tag{3-6}$$

according to [1], we have

$$\begin{aligned}I(W_{2N}^{(2i)}) &= I(U_1; Y_1, Y_2),\\ I(W_{2N}^{(2i-1)}) &= I(U_2; Y_1, Y_2|U_1),\end{aligned} \tag{3-7}$$

where U_1 and U_2 are i.i.d. From the chain rule, it follows that

$$\begin{aligned}I(W_{2N}^{(2i)}) + I(W_{2N}^{(2i-1)}) &= I(U_1; Y_1, Y_2) + I(U_2; Y_1, Y_2|U_1),\\ &= 2I(W_N^{(i)}),\end{aligned} \tag{3-8}$$

and

$$I(W_{2N}^{(2i)}) \geq I(W_N^{(i)}), \tag{3-9}$$

which results in

$$I(W_{2N}^{(2i)}) \geq I(W_{2N}^{(2i-1)}). \tag{3-10}$$

For the case of the BEC channel [1]

$$Z(W_{2N}^{(2i-1)}) = 2Z(W_N^{(i)}) - Z(W_N^{(i)})^2, \tag{3-11}$$

$$Z(W_{2N}^{(2i)}) = Z(W_N^{(i)})^2, \tag{3-12}$$

where reliability and cumulative rate must satisfy [1]

$$\sum_{i=1}^{N} I(W_N^{(i)}) = NI(W), \tag{3-13}$$

$$\sum_{i=1}^{N} Z(W_N^{(i)}) \leq NZ(W). \tag{3-14}$$

In uniform channel polarization there is always a set of B-DMCs that reaches capacity when $N \rightarrow \infty$ and arbitrarily small δ such that $\lim_{N\rightarrow\infty} \delta \approx 0$

$$\begin{aligned}\lim_{N\rightarrow\infty} \frac{\sum_{i=1}^{N} I(W_N^{(i)})}{N} &= I \in (1-\delta, 1] \text{ or}\\ \lim_{N\rightarrow\infty} \frac{\sum_{i=1}^{N} I(W_N^{(i)})}{N} &= (1-I) \in [0, \delta),\end{aligned} \tag{3-15}$$

where the values of I converge to 0 or to 1.

3.4 Non-Uniform Construction

In this section, we show a generalization of the equations presented in Section 3.3, we will see that channel polarization can also be applied to NUC.

We now have two channels W, which are independent, and we will consider them to be non-uniform, so $W_{(i)} : \mathcal{X}_{(\mathrm{i})} \rightarrow \mathcal{Y}_{(\mathrm{i})}$, as shown in Figure 3.4, where we rewrite (2-3) as

$$W(y_1^N|x_1^N) = \prod_{i=1}^{N} W_{(i)}(y_i|x_i), \tag{3-16}$$

with $W_{(i)}(y|x) \neq W_{(j)}(y|x)$ if $i \neq j$.

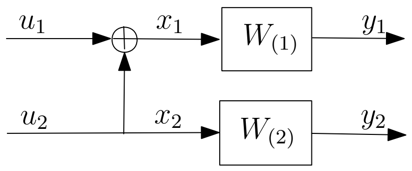


Figure 3.4: The NUC channel W_2.

The symmetric capacity (2-4) and the $Z(W)$ parameter (2-5) [1] for any $W_{(i)}$, are rewritten as

$$I(W_{(i)}) = \sum_{y\in Y}\sum_{x\in X}\frac{1}{2}W_{(i)}(y|x)\log\frac{W_{(i)}(y|x)}{\frac{1}{2}W_{(i)}(y|0)+\frac{1}{2}W_{(i)}(y|1)}, \tag{3-17}$$

$$Z(W_{(i)}) = \sum_{y_i\in Y}\sqrt{W_{(i)}(y_i|0)W_{(i)}(y_i|1)}, \tag{3-18}$$

and

$$\log\frac{2}{1+Z(W_{(i)})} \leq I(W_{(i)}) \leq \sqrt{1+Z(W_{(i)})^2}. \tag{3-19}$$

For W_2 we rewrite (2-7) and (2-8) as

$$W_2^{(1)}(y_1^2|u_1) = \sum_{u_2}\frac{1}{2}W_{(1)}(y_1|u_1\oplus u_2)W_{(2)}(y_2|u_2), \tag{3-20}$$

$$W_2^{(2)}(y_1^2,u_1|u_2) = \frac{1}{2}W_{(1)}(y_1|u_1\oplus u_2)W_{(2)}(y_2|u_2). \tag{3-21}$$

Using the BEC channel again as an example, for a comparison with Section 3.3.2, Figure 3.5 shows an alternative polarization tree and, for the case of the parameter Z, we have:

$$
\begin{aligned}
Z(W_2^{(2)}) &= \sum_{y_1^2,u_1} \sqrt{W_2^{(2)}(y_1^2,u_1|u_2=0)W_2^{(2)}(y_1^2,u_1|u_2=1)}, \\
&= \sum_{y_1^2,u_1} \frac{1}{2}\sqrt{W_{(1)}(y_1|u_1)W_{(2)}(y_2|0)} \\
&\quad \cdot \sqrt{W_{(1)}(y_1|u_1)W_{(2)}(y_2|1)}, \\
&= \sum_{y_2,u_1} \sqrt{W_{(2)}(y_2|0)W_{(2)}(y_2|1)} \\
&\quad \cdot \sum_{y_1,u_1} \frac{1}{2}\sqrt{W_{(1)}(y_1|u_1)W_{(1)}(y_1|u_1)}, \\
&= Z(W_{(2)})Z(W_{(1)}).
\end{aligned} \tag{3-22}
$$

From (3-11) and (3-12):

$$
Z(W_2^{(1)}) \le Z(W_{(1)}) + Z(W_{(2)}) - Z(W_{(1)})Z(W_{(2)}), \tag{3-23}
$$

$$
Z(W_2^{(2)}) \le Z(W_{(1)})Z(W_{(2)}), \tag{3-24}
$$

$$
Z(W_2^{(2)}) \le Z(W_2^{(1)}). \tag{3-25}
$$

So (3-13) and (3-14) are equivalent to

$$
\sum_{i=1}^{N} I(W_N^{(i)}) = \sum_{i=1}^{N} I(W_{(i)}), \tag{3-26}
$$

$$
\sum_{i=1}^{N} Z(W_N^{(i)}) \le \sum_{i=1}^{N} Z(W_{(i)}). \tag{3-27}
$$

We can show that (3-8) can be obtained by performing the following operations:

$$
\begin{aligned}
I(W_2^{(1)}) &= I(Y_1;Y_2|U_1), \\
I(W_2^{(2)}) &= I(Y_1;Y_2|U_1,U_2), \\
I(W_2^{(1)}) + I(W_2^{(2)}) &= I(Y_1;Y_2|U_1) + I(Y_1;Y_2|U_1,U_2), \\
&= I(W_1) + I(W_2).
\end{aligned}
$$

So, for any set of B-DMC, we can rewrite (3-15) for the non-uniform polarization channel:

$$
\begin{aligned}
&\lim_{N\to\infty} \frac{\sum_1^N I(W_{(i)})}{N} = I \in (1-\delta, 1] \text{ or} \\
&\lim_{N\to\infty} \frac{\sum_1^N I(W_{(i)})}{N} = (1-I) \in [0, \delta).
\end{aligned} \tag{3-28}
$$

where the values of I converge to 0 or to 1, which is a novel result related to the established result for uniform channel polarization in (3-15).

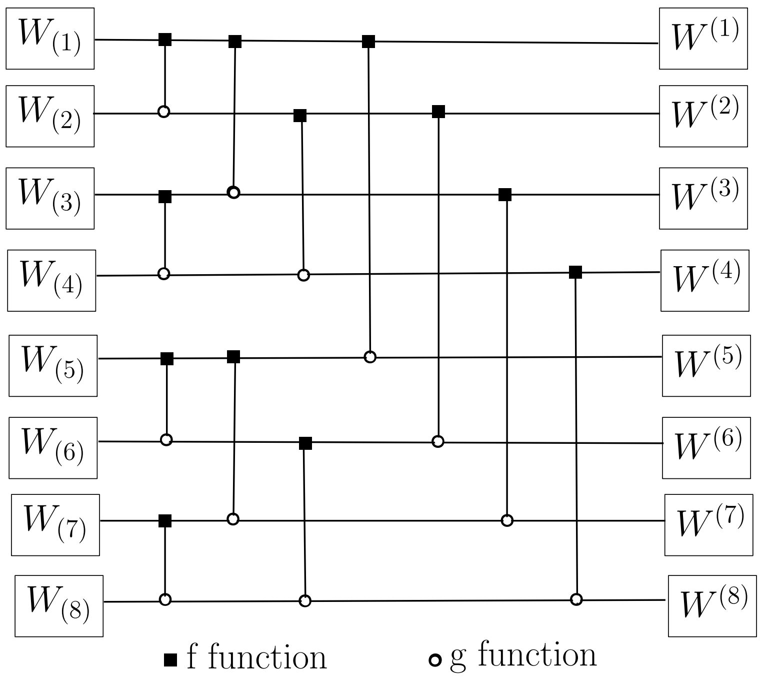


Figure 3.5: Alternative polarization tree.

3.5 NUPGA Design Algorithms

In this section, we detail the NUPGA method and the implementation of the non-uniform construction algorithms. First, consider the block diagram of the polar coding system shown in Figure 3.6. Unlike the system originally shown in Figure 2.1, we include in Figure 3.6 the code shortening or extension (arbitrary length) step.

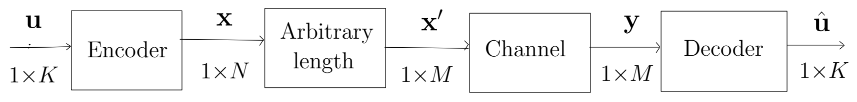


Figure 3.6: System model with shortening or extension step.

3.5.1 NUPGA-Based Shortening

The PD [61] is a starting point to define the channels that will initially be shortened. Shortening techniques reduce the length of the codeword from N to M, that is, $2^{n-1} < M < 2^n$. The number of bits of information is represented by K. The constants N and M represent, respectively, the code lengths of the standard polar codes and the shortened polar codes. Note that $K < M < N$. The indexes set of the shortened bits is represented by the symbol P, also called the shortening pattern. The cardinality of the shortened bits is represented by $|P| = N - M$. Thus, for shortened polar codes, the code rate is represented by $R = K/M$. Note that the decoder knows the shortened bits P. When decoding, the corresponding LLRs are set to infinity.

Consider that the vector P contains the channels obtained by the PD. In the first step, the codeword is generated by setting the set P set to zero. In the next step, the length of the codeword is reduced by P. With code shortening, we have changes in the bit channels reliability, which deteriorates performance when compared to the original code. In this regard, the study in [61] indicates that the order of channel polarization does not change after shortening.

In AWGN channels, the LLRs of each sub-channel, namely $L_N^{(i)}$, the channel polarization can be estimated with the recursive GA algorithm proposed by [15]

$$\begin{aligned} E(L_N^{(2i-1)}) &= \phi^{-1}(1-(1-\phi(E(L_{N/2}^{(i)})))^2), \\ E(L_N^{(2i)}) &= 2E(L_{N/2}^{(i)}), \end{aligned} \tag{3-29}$$

with $E[\cdot]$ is the expected value operator, and ϕ is defined as

$$\phi(x) = \begin{cases} \exp(-0.4527x^{(0.86)} + 0.0218) & \text{if } 0 < x \leq 10, \\ \sqrt{\frac{\pi}{x}}(1-\frac{10}{7x})\exp(-\frac{x}{4}) & \text{if } x > 10. \end{cases} \tag{3-30}$$

In NUPGA, the GA equation in (3-29) is generalized, making it possible to treat arbitrary code lengths. This results in the following proposed recursions:

$$\begin{aligned} E(L_N^{(2i-1)}) &= \phi^{-1}(1-(1-\phi(E(L_{(1)}^{(i)})))(1-\phi(E(L_{(2)}^{(i)})))), \\ E(L_N^{(2i)}) &= E(L_{(1)}^{(i)})E(L_{(2)}^{(i)}). \end{aligned} \tag{3-31}$$

In Algorithm 7 we have the description of the NUPGA shortening algorithm.

3.5.2 NUPGA-Based Extension

In this section, we present an additional contribution: the polar code extension using the proposed NUPGA technique. A simple polar codes extension scheme can be implemented as suggested in Figure 3.7 [73]. The key idea is to extend the codeword without changing the sequence of the information bits. The extend is performed in channel vector P_1^M and the information bits u_1^K. The connection between the additional channels P_1^M and channels $u_{K-|P|}^K$ is carried out by the sequence of u_1^K. On the other hand, the extension should improve the performance of the resulting code. In the extension scheme, the

Algorithm 7: Proposed NUPGA Shortening Algorithm

```
Input: code block length, $N$
Input: info block length, $K$
Input: shortened set, $P$
Input: design-SNR, $E_{dB} = (RE_b/N_o)$ in dB
Output: result vector, $F$
begin
    $S = 10^{E_{dB}/10}$
    $n = \log_2 N$
    $L \in R^N$ initialize $[E(L_1^{(i)})]_1^N = 4S$
    Upgrade with shortening vector $[E(L_1^{(i)})]_1^N$ with $P$
    for $i = 1 : n+1$ do
        $d = 2^{(i-2)}$
        for $k = 1 : 2^{(i-1)} : n$ do
            for $b = 0 : d-1$ do
                if $E(L_{k+b}^{(i-1)}) = 0$ or $E(L_{k+b+d}^{(i-1)}) = 0$ then
                    $E(L_{k+b}^{(i)}) = E(L_{k+b}^{(i-1)})$
                    $E(L_{k+b+d}^{(i)}) = E(L_{k+b+d}^{(i-1)})$
                end
                $E(L_{k+b}^{(i)}) =$
                 $\phi^{-1}(1-(1-\phi(E(L_{k+b}^{(i-1)})))(1-\phi(E(L_{k+b+d}^{(i-1)}))))$
                $E(L_{k+b+d}^{(i)}) = E(L_{k+b}^{(i-1)})E(L_{k+b+d}^{(i-1)})$
            end
        end
    end
    $F =$ Find indices of smallest elements $(E[L], K)$
    return $F$
end
```

complexity is $\mathcal{O}((N+M)\log(N+M))$, for $N = 2^n$, is smaller than the puncturing and shortening scheme, which is $\mathcal{O}((2^{n+1})\log(2^{n+1}))$. Extension scheme is efficient for kernels with low dimension ($N < 512$) and for extension of $|P| < 50\%$ of N. Using the proposed NUPGA technique, we can consider all additional bit channels P_1^M frozen and as bits output from polarized channels $u_{K-|P|}^K$, uniform and limited to the length of the extension. For the encoder we have the same definition, that is, P_1^M frozen. In the decoder, we have

$$\hat{u}_1^K = f_{\text{ext}}(\text{LLR}((u_1^K) + \text{LLR}(u_{K-|P|}^K)). \tag{3-32}$$

Note that according to (3-26), we have

$$\sum_{i=1}^{N} I(W_N^{(i)}) + \sum_{i=1}^{M} I(W_M^{(i)}) = \sum_{i=1}^{N} I(W_{(i)}) + \sum_{i=1}^{M} I(W_{(i)}),$$

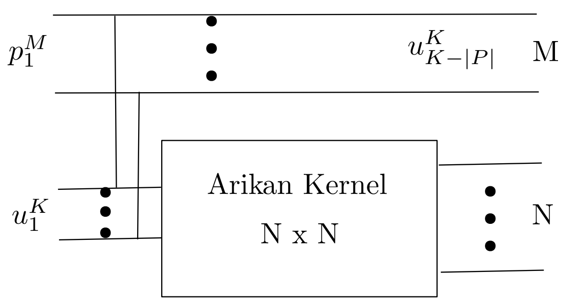


Figure 3.7: Polar codes extension.

which only occurs in the following two cases: either $W_N^{(i)}$ and $W_M^{(i)}$ are noisy channels and both of the $I(\cdot)$ are equal to 0, or $W_N^{(i)}$ and $W_M^{(i)}$ are both perfect channel such that both $I(\cdot)$ are equal to 1. Using NUPGA, when $W_N^{(i)}$ is perfect and $W_M^{(i)}$ is useless. In other words, if the extended bit channel $W_M^{(i)}$ is a noisy channel and excluding the case that both $W_N^{(i)}$ and $W_M^{(i)}$ are perfect channels, the re-polarization improves the reliability of the shortened channels.

Regarding to (3-27), we have

$$\sum_{i=1}^{N} Z(W_N^{(i)}) + \sum_{i=1}^{M} Z(W_M^{(i)}) \leq \sum_{i=1}^{N} Z(W_{(i)}) + \sum_{i=1}^{M} Z(W_{(i)}),$$

and with the use of NUPGA, we have

$$\sum_{i=1}^{M} Z(W_M^{(i)}) \leq \sum_{i=1}^{N} Z(W_N^{(i)})$$

and

$$\sum_{i=1}^{M} Z(W_{(i)}) \leq \sum_{i=1}^{N} Z(W_{(i)}),$$

which ensures that all extended channels will be frozen channels. With the proposed NUPGA extension scheme increases the message through the addition of bits, maintains the positions of the information bits.

The NUPGA Extension Algorithm is based on the implementation of the equation (2-28). This method is similar to the extension of the polarization matrix proposed in [72] and [73]. Note that the construction method for polar codes extension allows us to maintain the same encoder and decoder for standard polar code [1]. In the proposed NUPGA extension algorithm, the original codeword of the channels initially designed is first extended by adding new bits. Thus, it is possible to increase the code length by gradually adding new bits, making it possible to build codewords of any length. The information bit channels are polarized according to the reliability of the bit channel calculated from the new extended channels.

Extension is a useful technique in HARQ transmissions that is used to improve the performance in noisy channels, allowing detection adjustments and synchronization. For this, the extension must allow the increment of bits in the message maintaining the original position of the information bits. In practice, there is an increase in redundancy bits. The performance gain in relation to the original message length is low. However, in addition to its low complexity, it has better performance than the shortening technique IoT scenarios [63]-[65].

The main idea of the proposed NUPGA extension algorithm is to generate the new bit channels as frozen bits and make the associated information bits more reliable than before. The extension length is $N' = (N + M)$ and the new rate is K/N', which can still be decoded efficiently. Therefore, extended channels are obtained with the proposed NUPGA extension algorithm. The details of the NUPGA extension algorithm are shown in Algorithm 8.

Algorithm 8: Proposed NUPGA Extension Algorithm

```
Input: code block length, $N$
Input: info block length, $K$
Input: shortened set, $P$
Input: extension length, $\Delta M$
Input: design-SNR, $E_{dB} = (RE_b/N_o)$ in dB
Output: resulting vector, $F \in \{0, 1, \ldots, N + \Delta M - 1\}$
1  begin
2      $S = 10^{E_{dB}/10}$
3      $n = \log_2 N$
4      Initialize $[E(L_1^{(i)})]_1^N = 4S$ and $[E(L_1^{(i)})]_N^{\Delta M} = 0$
5      for $i = 1 : n + 1$ do
6          $d = 2^{(i-2)}$
7          for $k = 1 : 2^{(i-1)} : n$ do
8              for $b = 0 : d - 1$ do
9                  if $E(L_{k+b}^{(i-1)}) = 0$ or $E(L_{k+b+d}^{(i-1)}) = 0$ then
10                     $E(L_{k+b}^{(i)}) = E(L_{k+b}^{(i-1)})$
11                     $E(L_{k+b+d}^{(i)}) = E(L_{k+b+d}^{(i-1)})$
12                 end
13                 $E(L_{k+b}^{(i)}) =$
                     $\phi^{-1}(1 - (1 - \phi(E(L_{k+b}^{(i-1)})))(1 - \phi(E(L_{k+b+d}^{(i-1)}))))$
14                 $E(L_{k+b+d}^{(i)}) = E(L_{k+b}^{(i-1)})E(L_{k+b+d}^{(i-1)})$
15             end
16         end
17     end
18     $F =$ Find indices of smallest elements $(E[L], K)$
19     return $F$
20 end
```

3.6 Numerical Results

The performance of the proposed NUPGA-based shortening and extension algorithms is assessed in this section against competing approaches such as CW [59], RQUP [60] and PD [61]. For performance analysis we compared the Bit Error Rate (BER) and Frame Error Rate (FER). We adopt Binary Phase shift keying (BPSK) signaling over the AWGN channel for the evaluation. In the simulations, SC and SCL decoders were considered with randomly generated codewords and different code rates. For all simulations in this thesis we used the Monte Carlo method with 500000 iterations [18].

In Figure 3.8 we compare NUPGA for shortening with [34] using an SC decoder, where we show that the performance of the shortened code is inferior to that of the standard code. Being E_b/N_0 the energy per bit to noise power spectral density ratio, is a normalized signal-to-noise ratio (SNR) measure, E_b is the signal energy associated with each user data bit and N_0 is the noise spectral density. A similar pattern can be seen in Figure 3.9 with the SCL decoder [18] and list size $L = 16$. Figure 3.10 shows the performance over AWGN of NUPGA for shortening with CA-SLC [18] decoding aided by CRC codes with size 24 and $L = 16$. In Figure 3.12 we compare the performance of the proposed NUPGA extension algorithm with the PD and NUPGA shortening algorithms with SCL and $L = 16$.

In the first example, in Figure 3.8, we show the standard polar codes [34] called the mother code (MC). In the simulation we use MC with $N = 512$ and rate $R = 1/2$; $M = 320$ with $R = 1/2$. We compare them with CW [59], RQUP [60] and PD [61] in addition to the proposed NUPGA shortening techniques. In Figure 3.9, we show the performance for $M = 400$ with $R = 1/2$, with CA-SCL decoding with $L = 16$. In Figure 3.10, the performance for: $M = 400$, $R = 1/4$, AWGN, CA-SCL, $L = 16$ and CRC with length 24. In Figure 3.11, we compare the performance of the shortened polar codes $M = 280$ and $K = 128$, the proposed NUPGA extension technique with three other curves: with MC [34] of length $N = 256$, the PD [61] algorithm and the NUPGA shortening under CA-SCL with $L = 16$ and CRC with length 24. The results show that the NUPAG extension technique outperforms the NUPGA shortening and the PD algorithms using list decoding with CRC.

We notice in all simulations that the proposed NUPGA technique has gains in performance in the scenarios studied. As shown in Figure 3.10, we can see that for low rates under list decoding, the gain tends to be greater. In Figure 3.8 we observe that the performance gain of NUPGA is 0.5 dB (BER) as compared to the approach of CW [59].

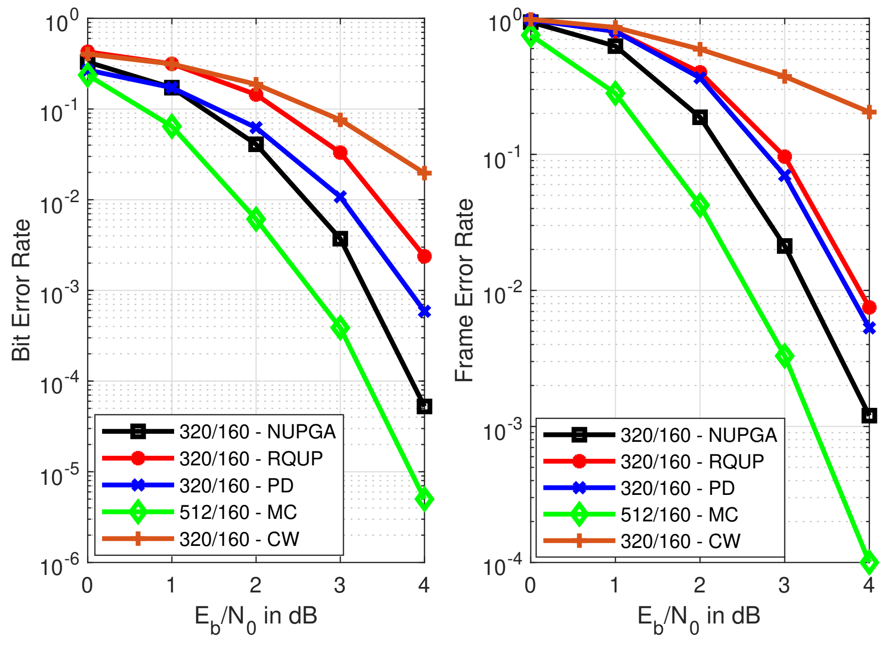


Figure 3.8: Comparative BER and FER performance of PC, with $N = 512$ and $R = 1/2$, and $M = 320$ with $R = 1/2$, of NUPGA, RQUP [60], PD [61], MC [34] and CW [59] PC construction methods.

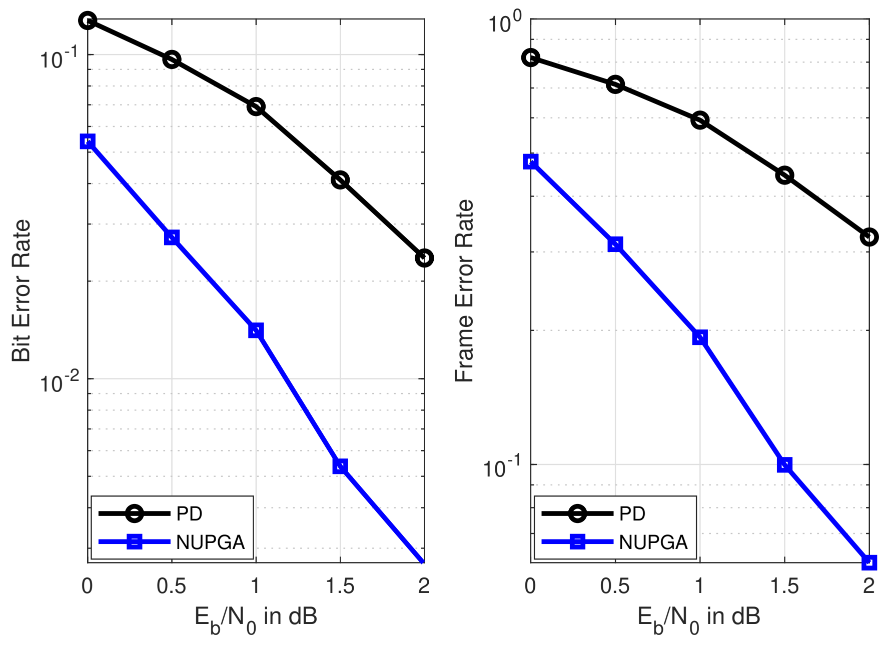


Figure 3.10: Comparative BER and FER performance of PD [61] and NUPGA shortening algorithms with $M = 400$ and $K = 50$ using CA-SCL with $L = 16$ and $CRC = 24$.

In Figure 3.9, we verify that the gain obtained by NUPGA is less than

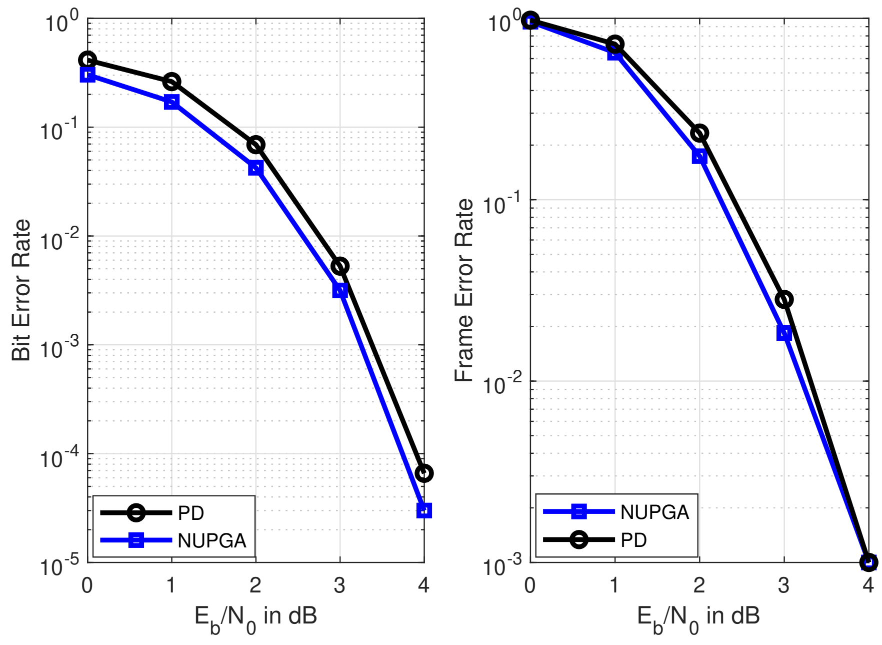


Figure 3.9: BER and FER performance of PD [61] and NUPGA design algorithms: $M = 400$, $K = 200$ and CA-SCL with $L = 16$.

0.1 dB. In Figure 3.10 we notice that the gain is up to 1.2 dB, which indicates that the NUPGA is advantageous for low rates. In Figure 3.11, we observe that the NUPGA extension algorithm achieves a performance improvement over that obtained by the NUPGA shortening algorithm, which is around 0.1 dB. We can see that according to the curves a modest incremental extension has good BER performance, maintaining the same FER performance as the original code [34].

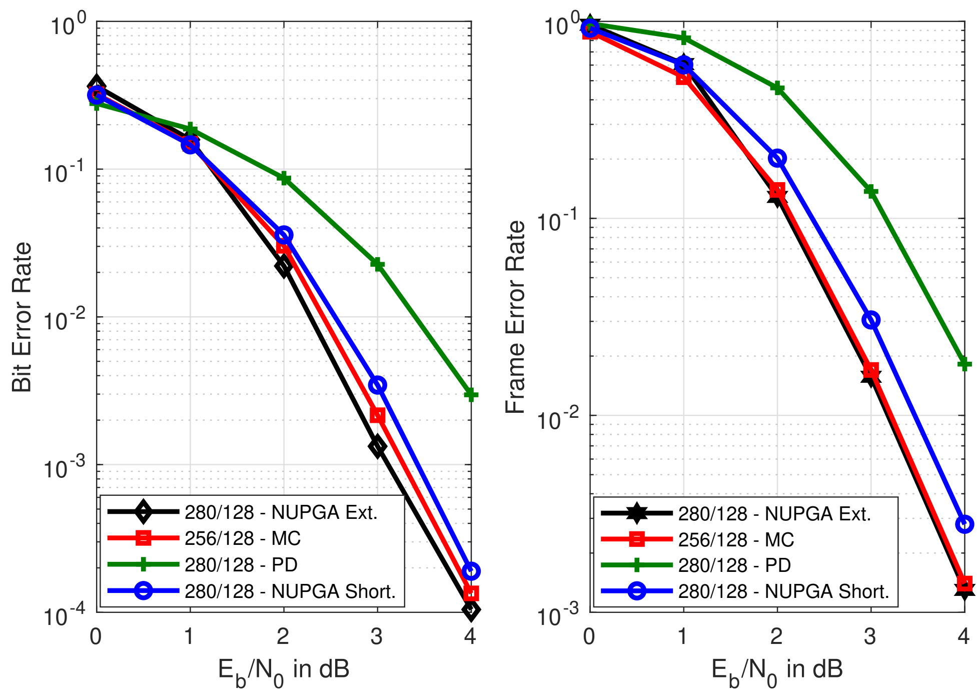


Figure 3.11: Performance of NUPGA extension with PD and NUPGA shortening, decoder CA-SCL with $L = 16$ and CRC with length 24.

In Figure 3.12, we can observe application of the NUPGA extension for $N = 64$ with $K = 32$, and $N = 128$ with $K = 64$ [63, 64]. Note that K is kept constant and N is extended. For the first case, we have the 4-bit extension for $N = 64$, that is, $M = (4, 8, 12)$. In the second case, we have the 8-bit extension for $N = 128$, that is, $M = (8, 16, 32)$. We can see that for both cases we have small gains (BER).

3.7 Chapter Summary

This chapter described a technique for polar codes construction for arbitrary code length, called NUPGA. NUPGA is designed for a scenario where the transmission is through an AWGN channel and under successive cancellation decoding. As by default, the construction of the polar codes is limited to the length of the code proportional to the power of two. We propose a generalization of the polarization theory [1] to define non-uniform channels and with this approach we can polar codes construction with arbitrary length. With NUPGA, we repolarized the projected synthetic channels, choosing more efficiently the positions of the information bits. In addition, we present a

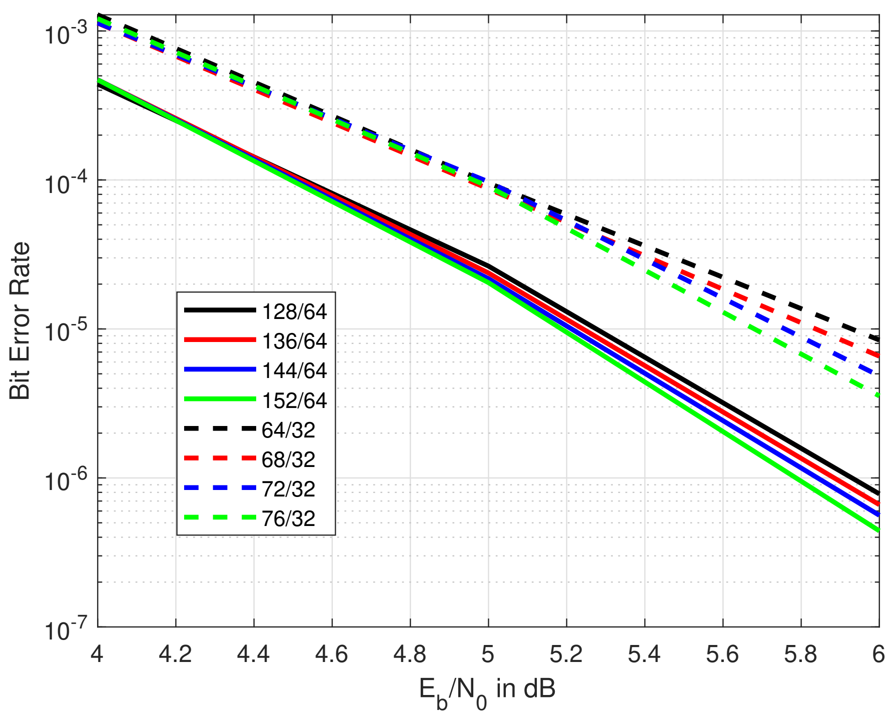


Figure 3.12: Performance of NUPGA extension for different N and K values [63, 65].

generalization of the GA for the polarization and repolarization processes and an extension technique for polar codes. The construction of the polar codes based on NUPGA presents better performance than the existing techniques, as demonstrated in the simulation results is this chapter.

4 Polar Codes Based on Piecewise Gaussian Approximation

4.1 Chapter Overview

In this chapter, we present the construction of polar codes based on Piecewise Gaussian Approximation (PGA) techniques. The PGA approach is first optimized and then compared to the Gaussian approximation construction method, showing performance gains for medium blocks and high precision for long blocks, in scenarios with successive cancellation decoding and AWGN channel. Based on the PGA, we develop two approximations based on multi-segmented polynomials that are easy to implement. We present the Approximate PGA (APGA) that is optimized for medium blocks and provides a performance improvement without increasing complexity. Furthermore, we develop the Simplified PGA (SPGA) as an alternative to the GA construction method, which is optimized for long blocks and achieves high construction accuracy.

4.2 Introduction and Literature Review

The work of Chung [74] introduced GA construction, in the context of LDPC code construction and was firstly in [15] for the construction of polar codes. Gaussian approximation construction method was originally described in integral form, known as Exact GA (EGA). Due to the complex integration, EGA has a high computational cost associated with the numerical solution, which increases exponentially with the code length, and, consequently, with the polarization levels. As an alternative, Chung [74] proposed the Approximate GA (AGA), which is an approximation composed of a two-segment function. We note that this alternative is implemented by transcendental functions, which leads to a high computational complexity. The author in [75] also proposed an alternative to numerical integration, approximating GA by a three-segment function. Trifonov also proposed in [76] a multi-segment polynomial approximation, without the use of transcendental and inverse functions. The AGA performance for short and medium blocks for the AWGN channel is similar to the Tal and Vardy method, but it fails for long blocks due to the approximation error around zero.

For long blocks, polar codes construction with EGA is expensive due to numerical integration, in addition to the errors associated with the numerical integration solution, and AGA construction [74] is imprecise due to the approximation error around zero. Furthermore, both require inverse and transcendental functions, and are composed of complex recursive functions. Some attempts at improved approximations with better performance than AGA were proposed in [77]-[79]. The work of Fang [77] analyzed the first and second derivatives of EGA and developed a simplified multi-segment polynomial approximation without using transcendental functions and without the need to calculate an inverse function. Dai [78] introduced the concepts of polarization violation set, the polarization reversal set and a new metric named cumulative-logarithmic error, which results in an algorithm that uses transcendental and inverse functions. Ochiai et al [79] analyzed the behavior of EGA in the logarithmic domain and proposed another approximation based on a logarithmic function (transcendental function) and that employs an algebraic expression for the inverse function.

Then, we develop in this thesis two improved approximations for GA based on Piecewise Gaussian Approximation, PGA. In particular, we develop high-precision approximations using only multi-segment polynomial functions, which replace the need for numerical integration, function inversion and transcendental functions. In [80] we reported the preliminary results. Here, we have expanded the work in [80] by including the application of PGA and extended design techniques to medium and large blocks, with theoretical analysis and extra simulation results of various application scenarios. In particular, we devise a novel strategy for a piecewise approximation method for polar codes construction, resulting in improved performance for medium block lengths. Similar to the original GA function, PGA is used in integral form. Then, we propose an approximation called Approximate PGA, APGA, through a new criterion of the approximation inspired by a detailed analysis of the behavior of the PGA function. APGA is a simplified alternative multi-segment polynomial approximation, which is computationally more convenient for construction of polar codes with medium blocks. By drawing inspiration from the analysis, we devise an approximation for long blocks, called Simplified PGA, SPGA, also in the form of a multi-segment polynomial function. The proposed method can be generalized to extremely long or extremely short lengths and is able to adapt to any channel condition. Moreover, we show that the difference in accuracy between the approximation methods can be obtained by the Number of Different Positions (NDP), initially introduced by Kern [81], and we derive an index that measures the general quality of the proposed approximation, called Accumulative Design Error (ADE). The rate-compatible polar codes design that uses GA in its construction, as reported in [61], [82]-[84], can have a significant impact on its performance when consider the APGA and SPGA constructions.

4.3 Gaussian Approximation

The set $\mathcal{A}^{\mathrm{c}}$ is obtained by polar codes construction. The construction depends on several parameters, the main ones being: the codeword length N, the number of the information bits K, the type/model channel will be used for transmission, the target SNR, or design-SNR, and the decoding approach. All construction methods covered in thesis will consider the AWGN channel and the SC decoder, mainly due to the large number of articles with results of the polar codes construction with GA and SC decoding for this channel, which allows for the comparison of polar codes construction methods.

In the GA construction, the LLR, namely $L_N^{(i)}$, is used as a Gaussian distribution function with a mean equal to half of the variance [34]. Therefore, the mean of the LLRs is a sufficient statistic for their iterative update. Rewriting (3-30) of GA [74], we have

$$E\left(L_N^{(2i-1)}\right) = \phi^{-1}\left(1 - \left(1 - \phi\left(E\left(L_{N/2}^{(i)}\right)\right)\right)^2\right), \tag{4-1}$$

$$E\left(L_N^{(2i)}\right) = 2E\left(L_{N/2}^{(i)}\right), \tag{4-2}$$

with

$$L_1^{(0)} = \frac{2}{\sigma^2}. \tag{4-3}$$

The quantity $L_N^{(i)}$ denotes the LLR of the channel $W_N^{(i)}$, σ^2 and $E(\cdot)$ are the variance and the mean, respectively. In practice, in order to construct polar codes, we have $E(L_N^{(i)}) = L_N^{(i)}$. The function $\phi(x)$ is defined as:

$$\phi(x) = \begin{cases} 1 - \frac{1}{\sqrt{4\pi x}} \int\limits_{\mathbb{R}} \tanh\left(\frac{u}{2}\right) e^{\frac{-(x-u)^2}{4x}} \mathrm{d}u, & \text{if } x > 0, \\ 1, & \text{if } x = 0, \end{cases} \tag{4-4}$$

is an exact function being known as EGA [77].

However, we have a complex integral function. The computational complexity will inevitably increase as the code length and polarization level increases. That is, the GA approach above has a numerical computation problem. The function $\phi(x)$ can arbitrarily approach zero as x becomes very large. For example [79], for x around 1000, a possible value in long code length construction, $\phi(x)$ can assume values lower than 10^{-100}. We can solve the function $\phi(x)$ and $\phi^{-1}(x)$ with the bisection method [80]. However, as x becomes large, $\phi(x)$ becomes very smaller, which generates numerical inaccuracy, and consequently, generates an error in the code construction.

The author in [74] also proposed a simplification of $\phi(x)$ by a two-segment approximation function described by

$$\phi(x)_{AGA} \approx \begin{cases} e^{-0.4527x^{(0.86)}+0.0218}, & \text{if } 0 < x \leq 10, \\ \sqrt{\frac{\pi}{x}}\left(1-\frac{10}{7x}\right)e^{-\frac{x}{4}}, & \text{if } x > 10, \end{cases} \tag{4-5}$$

which is the so-called AGA [77]. For codes with long block lengths, AGA induces performance losses due to the approximation error caused by the difference between $\phi(x)$ and $\phi(x)_{AGA}$ for $x = 0$, that is, the AGA approach above has a numerical computation problem.

$$\phi(0)_{AGA} = e^{0.0218} > \phi(0) = 1. \tag{4-6}$$

The zero approximation error of $\phi(0)_{AGA}$ show in (4-6).

A detailed analysis of the approximation error of (4-5) and its effects on large block lengths can be found in [78, 79]. Additionally, the AGA algorithm implements the calculation of transcendental, inverse and complex recursive functions, which can be avoided.

The computational complexity of the EGA construction is given by $\mathcal{O}(Nm)$, where N is the length of the code and m is the number of iterations to calculate the numerical solution of the integration and the function inversion, both from (4-1). The larger the value of m, the more accurate the numerical solution of integration and function inversion will be. The proposed approximations reduce the computational complexity of constructing the polar codes to $\mathcal{O}(N)$, similar to the computational complexity in [1]. Note that the cost is associated to the design phase of the polar codes. Since the codes are designed the operation cost is the same for all designs.

4.4 Piecewise Gaussian Approximation

It is known that EGA was originally proposed to design LDPC codes [74] and when applied in the construction of polar codes [15] it generates codes with good performance. However, it is not known if EGA (4-4) can be improved for the construction of polar codes. From (4-4), for the purpose of analysis we define the function:

$$\psi(x,u) = \tanh\left(\frac{u}{2}\right)\frac{1}{\sqrt{4\pi x}}e^{\frac{-(u-x)^2}{4x}}, \tag{4-7}$$

where we notice that the compound function is the product of a Gaussian function

$$g(x,u) = \frac{1}{\sqrt{4\pi x}} e^{\frac{-(u-x)^2}{4x}},$$

with a hyperbolic tangent function

$$\tanh\left(\frac{u}{2}\right),$$

which we will call the Modified Gaussian (MG) function. An example of $\tanh(\frac{u}{2})$ and $g(x,u)$ is shown in Figure 4.1, for $x \approx 0.08$.

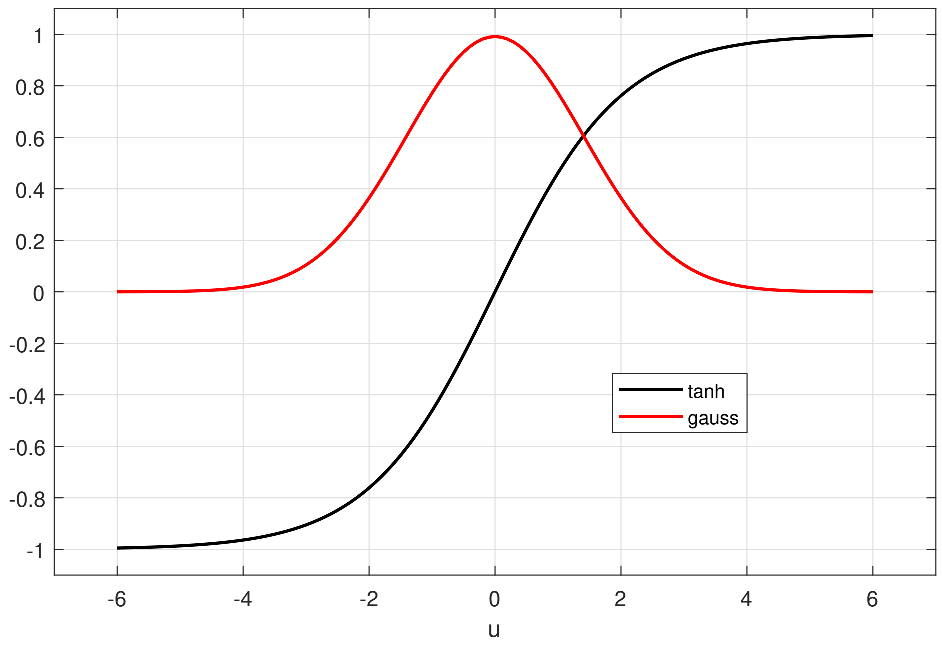


Figure 4.1: Example of tanh and Gaussian function.

The $\tanh(\frac{u}{2})$ is an odd, i.e., zero-centered function, $\tanh(\frac{u}{2}) \in [-1,1]$, with $(\tanh\frac{u}{2})$ approaches -1 for the interval $\frac{u}{2} \in (-\infty,-4)$ and $(\tanh\frac{u}{2})$ approaches +1 for the interval $\frac{u}{2} \in (+4,\infty)$. In Figure 4.2 we can notice the compound function $\psi(x,u)$ (4-7), and its behavior for several values of x (mean), varying u. Each curve represents a $g(x,u)$ function with x ranging from 0 to 14. Note that $\tanh(\frac{u}{2})$ for $u < 1$ has greater importance than $g(x,u)$, for small values of x and for x approaching 0. Moreover, with the increase in x, the behavior tends to be of a $g(x,u)$ function, i.e, the Gaussian function becomes dominant.

As explained in [74], to maintain the accuracy of EGA, it is important to preserve the symmetry condition [85], expressed as $F(x) = F(-x)e^{-x}$, where $F(x)$ is the density of an LLR message. For $g(x,u)$ function this condition can only be met by the mean. Observing $\psi(x,u)$, we have that the symmetry condition is preserved independently of the $\tanh(\frac{u}{2})$ function.

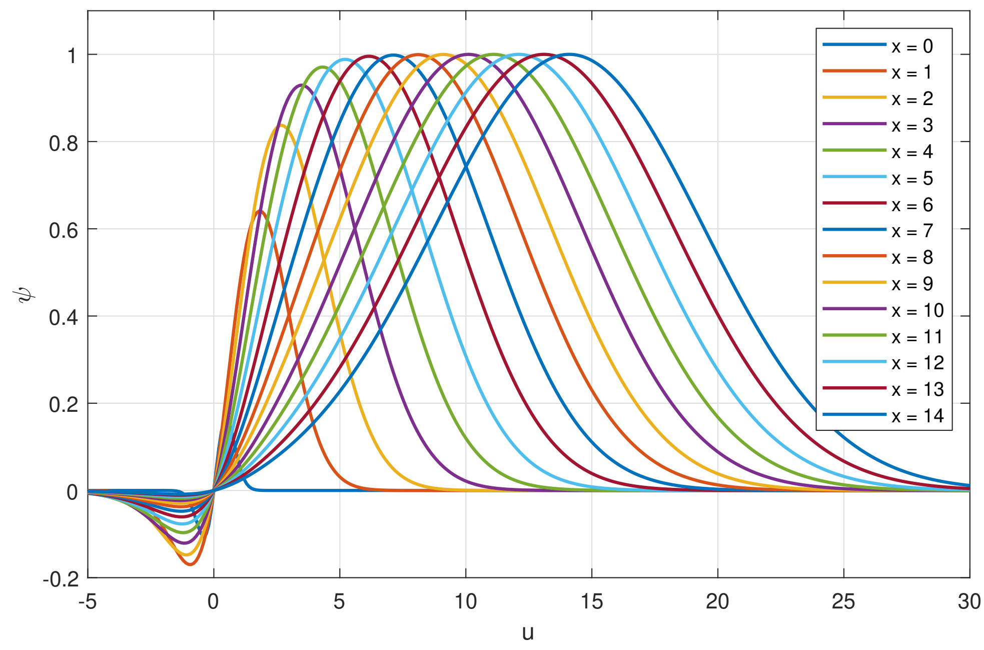


Figure 4.2: Function ψ in (4-7), with increment of x (mean) tends to the behavior of $g(x,u)$, i.e., Gaussian.

Thus, to improve the performance of the GA construction for polar codes, we present a piecewise approximation for the function $\phi(x)$ in (4-4) and a new function is proposed to replace the function $\tanh(\frac{u}{2})$. In this approximation, we use an exponential function with the following terms: $ae^{bx} + ce^{dx}$.

The proposed piecewise function optimized for polar codes is

$$\phi_p(x) = \begin{cases} 1 - \frac{1}{\sqrt{4\pi x}} \int\limits_{\mathbb{R}} f\left(\frac{u}{2}\right) e^{\frac{-(u-x)^2}{4x}} \mathrm{d}u, & x > 0, \\ 1, & x = 0, \end{cases} \tag{4-8}$$

with

$$f(x) = \begin{cases} ae^{bx} + ce^{dx}, & -3.1 \leq \ x \leq 3.1, \\ 1, & x > 3.1, \\ -1, & x < -3.1. \end{cases} \tag{4-9}$$

The parameters $a, b, c,\ d$ and cutoff at $x = \pm 3.1$ was obtained by exhaustive search for minimum FER in a PC(512,128), PC(1024,512) and PC(2048,1024) constructed with (4-8) and (4-9), with design-SNR $= (0, 1, 2, 3)$, $E_b/N_0 = (0, 1, 2, 3, 4)$ and $5 \cdot 10^5$ iterations. Initially, the odd function $f(x)$ is defined as $f(x) \in [-1, 1]$, $x \in \mathbb{R}$, $\lim_{x\to-\infty} f(x) = -1$, $\lim_{x\to+\infty} f(x) = +1$ and $f(0) = 0$, according to

$$\min_{f(x)}(\text{FER}). \tag{4-10}$$

Generalization is possible because the function obtained in (4-10) has the same initial format as tanh, which is the starting point for the optimization.

The parameters a, b, c and d are continuously adjusted until a performance superior to the EGA is obtained. During the exhaustive search, the function $f(x)$ was tested with various formats, some of these formats are known functions. Some of these functions are represented in Figure 4.3.

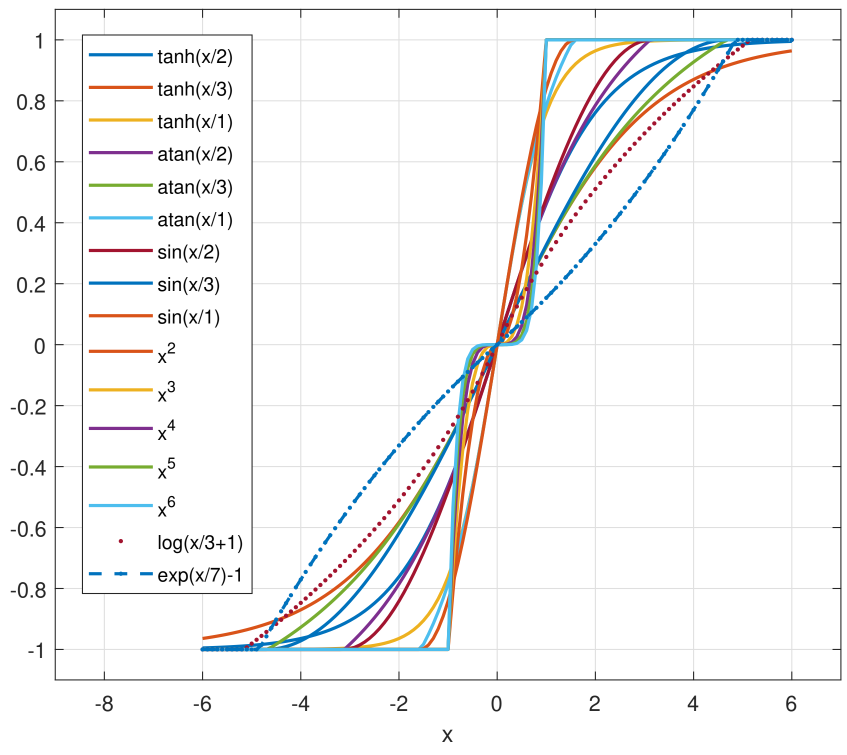


Figure 4.3: Some $f(x)$ functions tested while exhaustive search in (4-10). For $x < 0$, consider $-x^2$, $-x^4$, $-x^6$, $-\log(-x/3 + 1)$ and $\exp(-x/7) - 1$.

The optimized parameters are $a = 1.9e^7$, $b = 8.4e^{-9}$, $c = -1.8e^7$ and $d = -8.5e^{-9}$.

In order to represent the PGA construction, we have updated equation (4-1) as

$$E\left(L_N^{(2i-1)}\right) = \phi_p^{-1}\left(1 - \left(1 - \phi_p\left(E\left(L_{N/2}^{(i)}\right)\right)\right)^2\right). \tag{4-11}$$

Using the same format and limits proposed by [75], using Root Mean Square Error (RMSE), we develop an approximation to the proposed function $\phi_p(x)$ in (4-8) given by

$$\phi_p(x) \approx \begin{cases} e^{-0.0484x^2 - 0.3258x}, & 0 \le x < 0.867861, \\ e^{-0.4777x^{0.8512} + 0.1094}, & 0.867861 \le x < 10, \\ \sqrt{\frac{\pi}{x}}\left(1 - \frac{1.509}{x}\right)e^{-\frac{x}{3.936}}, & x \ge 10. \end{cases} \tag{4-12}$$

The values of x are chosen to improve the approximation given by (4-5) around $x = 0$, where $\phi(0)_{AGA} > 1$, (4-6). Thus, this approximation improves accuracy, but it is still constituted by transcendental functions.

In Algorithm 9 we have the description of the proposed PGA construction algorithm with ϕ_p in (4-1). For the calculation of the function ϕ_p^{-1}, inverse function of (4-4), by the bisection method. An optimized approach is presented in Algorithm 10.

Algorithm 9: PGA construction

```
    Input: code length, $N$
    Input: information bits, $K$
    Input: design-SNR $E_{dB} = (RE_b/N_0)$ in dB
    Output: $F \in \{0, 1, \ldots, N-1\}$ with $|F| = N$
 1  begin
 2      $S = 10^{E_{dB}/10}$;
 3      $n = \log_2 N$;
 4      $W \in R^N, W(0) = 4S$;
 5      for $i$ = 1 to $n$ do
 6          $d = 2^i$ ;
 7          for $j$ = 1 to $\frac{d}{2} - 1$ do
 8              $W(j) = \phi_p^{-1}(1 - (1 - \phi_p(W(j-1)))^2)$;
 9              $W(d/2 + j) = 2W(j-1)$;
10          end
11      end
12      $F$ = Sorts W indices in ascending order;
13  end
```

Algorithm 10: ϕ_p^{-1}function

```
Input: value input, $y$
Output: value output, $x$
begin
    if $y = 0$ then
        x = 0;
        return;
    end
    aux = 1; base = $\phi_p$(aux); iteration-times = 20;
    if $y \leq base$ then
        while $y < base$ do
            aux = aux×0.1; $base$ =$\phi_p$(aux);
        end
        anchor1 = aux;
        anchor2 = aux×10 for $i$ = 1 to iteration-times do
            x = (anchor1+anchor2)/2; aux = $\phi_p$(x);
            if $y <= aux$ then
                anchor2 = x;
            else
                anchor1 = x;
            end
        end
    else
        while $y >= base$ do
            aux = aux + 10; base = $\phi_p$(aux);
        end
        anchor1 = aux - 10;
        anchor2 = aux;
        for $i$ = 1 to iteration-times do
            x = (anchor1+anchor2)/2; aux = $\phi_p$(x);
            if $y <= aux$ then
                anchor2 = x;
            else
                anchor1 = x;
            end
        end
    end
end
```

In Figure 4.4, we can see the functions in (4-1) with the $\phi(x)$ for EGA in (4-4), and the the functions in (4-11) with the $\phi_p(x)$ for PGA in (4-8). In the next section we will see the performance improvement due to this difference between the EGA and PGA functions.

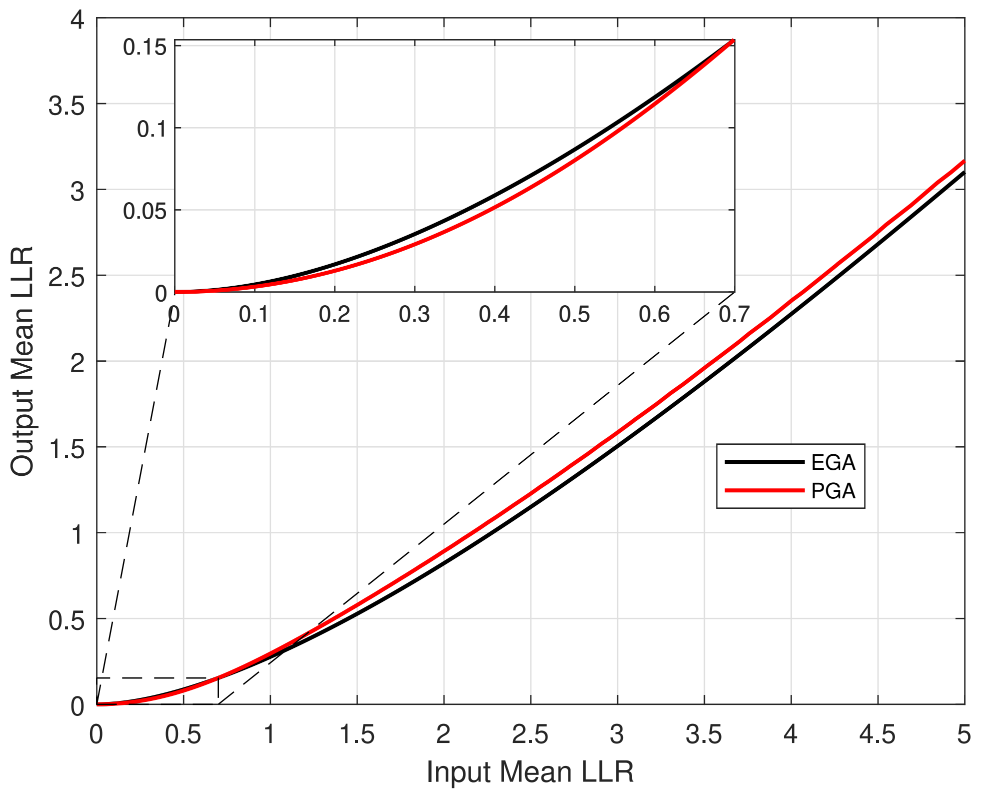


Figure 4.4: Comparison of EGA and PGA with input mean LLR.

4.5 Proposed PGA and Design Techniques

In this section, we present a detailed study of PGA by investigating the behavior of $\psi(x, u)$ in (4-7). We identify the key points and the statistical distribution of the results. Based on the statistics and in order to approximate PGA, we eliminate the transcendental functions and the inverse function. The approximation is generated with polynomial functions and we propose APGA for polar codes construction. Using a similar strategy, we propose an approximation for long blocks, called SPGA. We remark that APGA is an approximation of PGA, which uses the function $\phi_p(x)$ in (4-8) and is optimized for medium blocks, whereas SPGA is an approximation of EGA, which uses the functions $\phi(x)$ in (4-4) and has been optimized for long blocks.

4.5.1
Modified Gaussian Analysis

In a more detailed analysis of the $\psi(x,u)$ function in (4-7), we can notice in Figure 4.5b that from $x > 20$, that is, for the mean greater than 20, the behavior of $\psi(x,u)$ is Gaussian, i.e.,

$$\{x > 20 \text{ and } u \in (-\infty, +\infty) \mid \psi(x,u) \approx g(x,u)\}.$$

It means that from that point on, $\psi(x,u)$ can be well approximated by a polynomial function of degree 1, i.e.,

$$\{x > 20 \mid \int_{\mathbb{R}} \psi(x,u)du \approx 1\}.$$

Next, we observe that in $\psi(x,u)$ from $x \in [6,20]$, the maximum point is identical to that of $g(x,u)$, as reproduced in Figure 4.5a, i.e, $\{x \in [6,20] \text{ and } u \in (-\infty,+\infty) \mid \max(\psi(x,u)) = \max(g(x,u))\}$. Here we have one more key point at 6 and an important interval of study for the mean between 6 and 20.

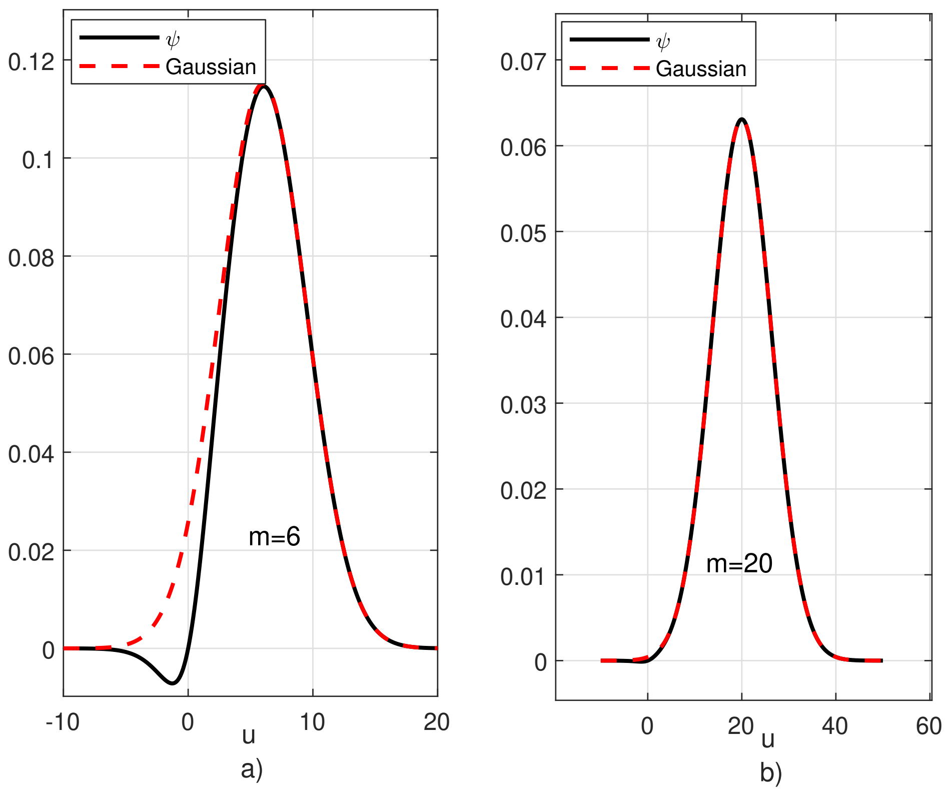


Figure 4.5: We can observe that from m=6 the $\phi(m,u)$ and $g(x,u)$ already have the same maximum value, as observed in a), and are completely equal in b), for m=20.

As observed in Figure 4.2, for $x < 1$ the $\tanh(\frac{u}{2})$ is more dominant than $g(x, u)$. This behavior can be better seen in Figure 4.6, where for each figure the scale was reduced by 10^{-5}.

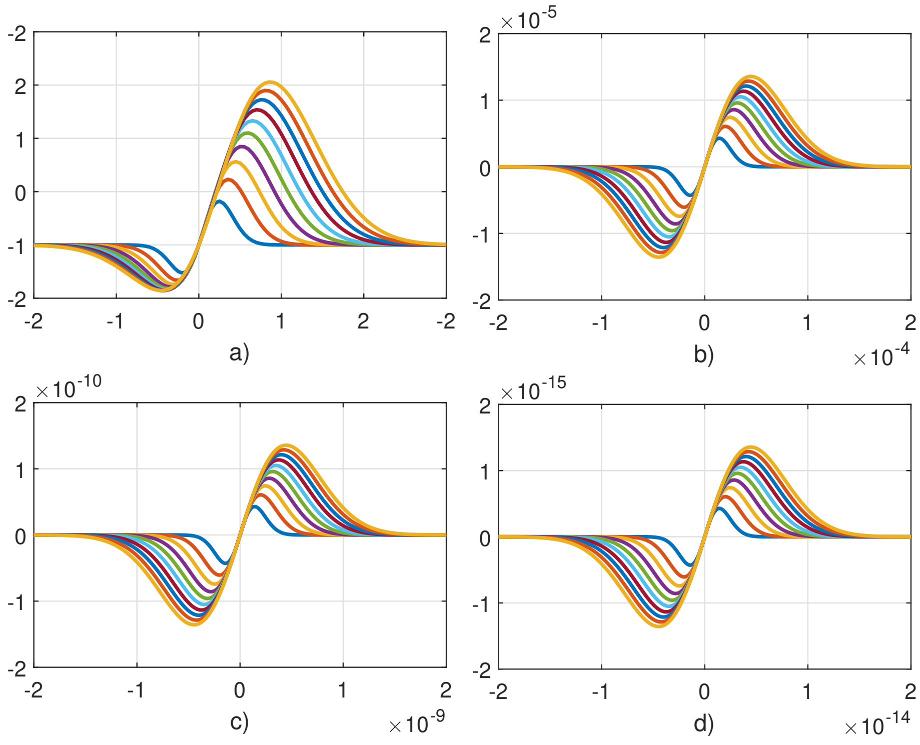


Figure 4.6: $\psi(x, u)$ function (4-7) behavior around zero, for each sub-figure the scale was reduced by 10^{-5}.

According to our analysis, we observe that points 1, 6 and 20, respectively from Figure 4.6 and Figure 4.5, are of are of fundamental importance for understanding the behavior of the $\phi(x)$ function, for they mark the points at which $\phi(x) \approx g(x, u)$.

4.5.2
Statistical Analysis of the PGA Function.

Since the key points in the previous section are obtained, we need to perform a statistical analysis of PGA, investigating the distribution of the LLRs obtained by (4-11). For the design of the simplified approximation, we must make sure that we will have a good accuracy in the regions with the highest concentration of LLRs.

Let us then examine the statistical concentration of LLRs, considering intervals that include points 1, 6 and 20 as limits. See in Figure 4.7 all LLRs generated by PGA for polar codes with lengths $N = 256, 512, 1024, 2048$, total of 30720 LLR values. Note that there is a concentration in the range $[0, 1]$.

Looking in more detail the concentration of LLRs in the interval $[0,1]$, it is observed that there is always a greater concentration around zero for LLRs $\to 0$. This is because for LLR < 1, the PGA generates new LLRs closer and closer to zero because it uses a squared term in (4-11). This implies a greater resolution of the approximation in this interval.

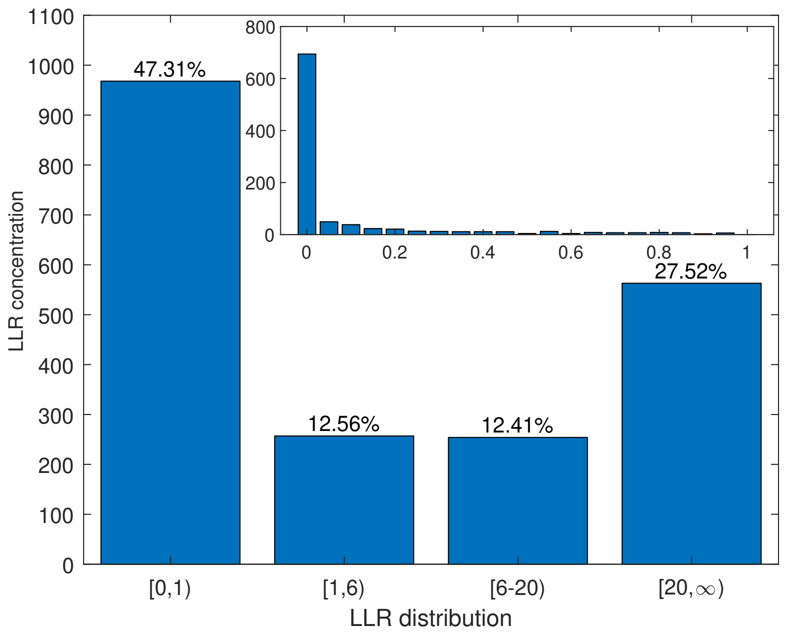


Figure 4.7: LLR histogram for PGA construction.

As a result of this analysis, we suggest one more key point, with a LLR value of 0.2, to be used in the intervals for the simplified approximation.

4.5.3 Approximate PGA for Medium Blocks

With the parameter obtained from the analysis of the behavior of the function ψ and the histogram of the LLRs of PGA, we propose to approximate the PGA (4-11) with a piecewise polynomial form given by

$$E[L_N^{(2i-1)}] = A(E[L_{N/2}^{(i)}]), \tag{4-13}$$

$$E[L_N^{(2i)}] = 2E[L_{N/2}^{(i)}], \tag{4-14}$$

with

$$A(x) = \begin{cases} 0.323x^2, & x \leqslant 0.2, \\ -0.1x^3 + 0.43x^2 - 0.039x - 0.005, & 0.2 < x \leqslant 1, \\ -0.003x^3 + 0.063x^2 + 0.432x - 0.2, & 1 < x \leqslant 6, \\ -0.0002x^3 + 0.012x^2 + 0.777x - 1.023, & 6 < x \leqslant 20, \\ 0.9803x - 2.109, & x > 20, \end{cases} \tag{4-15}$$

which is denoted as APGA. This approximation was obtained by minimum squared error curve-fitting. This operation involves only summations and multiplications, and avoids any transcendental functions. In Figure 4.8a we can notice the accuracy of APGA in relation to PGA.

4.5.4 Simplified PGA for Long Blocks

Using the same approximation strategy and analysis as in the previous subsection, we propose the SPGA design technique for polar codes with long blocks. Similar to the approximations obtained previously, our objective is to obtain a simplified polynomial multi-segment function, using for this the same limits of (4-15). The proposed function has been designed by minimum squared error curve-fitting, with only addition and multiplication operations, without any transcendental functions. The simplified approximation is given by

$$E[L_N^{(2i-1)}] = S(E[L_{N/2}^{(i)}]), \tag{4-16}$$

$$E[L_N^{(2i)}] = 2E[L_{N/2}^{(i)}], \tag{4-17}$$

with

$$S(x) = \begin{cases} -0.256x^3 + 0.461x^2 + 0.002x, & x \leqslant 0.2, \\ -0.064x^3 + 0.294x^2 + 0.05x - 0.004, & 0.2 < x \leqslant 1, \\ -0.005x^3 + 0.092x^2 + 0.316x - 0.133, & 1 < x \leqslant 6, \\ 0.002x^2 + 0.908x - 1.588, & 6 < x \leqslant 20, \\ 0.995x - 2.459, & x > 20, \end{cases} \tag{4-18}$$

which is denoted as SPGA.

The EGA and PGA functions are distinct functions, as shown in Figure 4.4, and two approximation functions, APGA for PGA and SPGA for EGA, are shown in Figure 4.8a and Figure 4.8b, respectively.

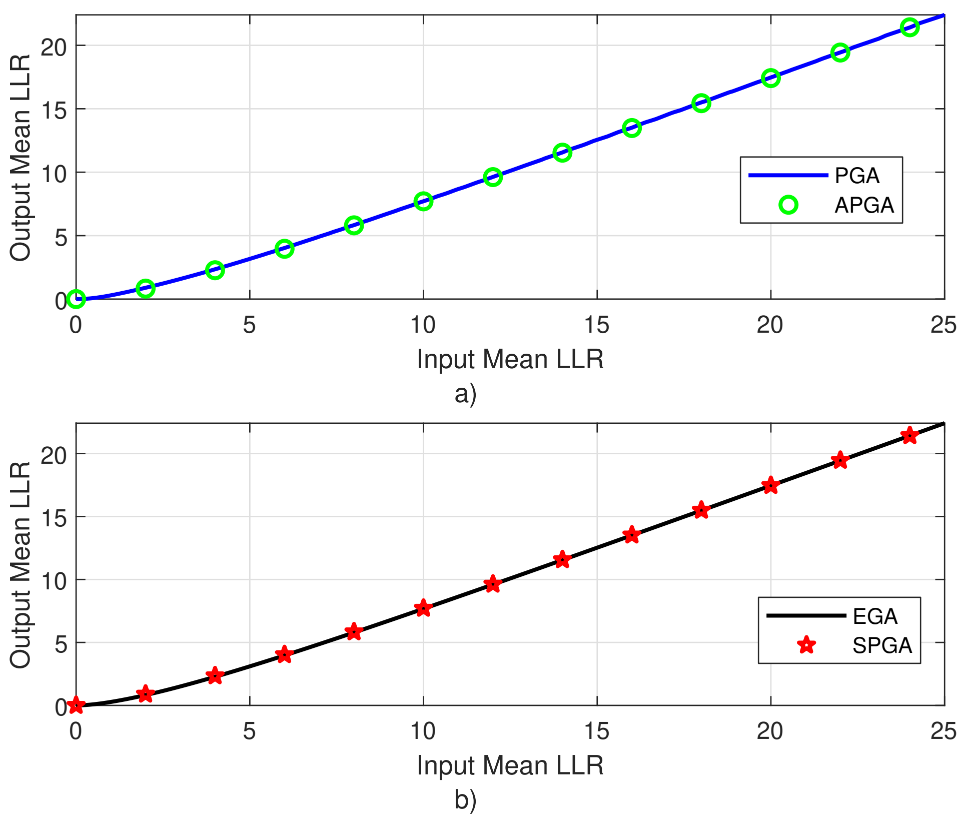


Figure 4.8: a) Accuracy of the APGA approximation to the PGA, and b) Accuracy of the SPGA approximation to the EGA.

4.5.5 General Algorithm for APGA and SPGA

Here, we present a general construction algorithm that can be used for both APGA and SPGA. In particular, note that lines 8 and 9 of Algorithm 11 represent, respectively, the use of (4-15) and (4-18). We can observe the mathematical simplification obtained when compared with (4-1), (4-5) and (4-12), that is, without the need to calculate an inverse function and without transcendental functions.

Recall that the objective of the proposed APGA and SPGA methods is to recursively calculate the reliability of each channel and all of them can be implemented with just a few calculations, which may involve non-linear functions. The overall complexity is proportional to the codeword length N.

4.5.6 Accumulative Design Error

The number of different positions between $\mathcal{A}_c$ and $\mathcal{A}_c^{ref}$, NDP [81], is a measure of the dispersion in polar codes construction. We used NDP to evaluate of approximation accuracy when comparing APGA and PGA; and

Algorithm 11: General algorithm for APGA and SPGA

```
Input: code length, $N$
Input: information bits, $K$
Input: design-SNR, $E_{dB} = (RE_b/N_0)$ in dB
Output: F $= \{0, 1, \ldots, N-1\}$ with $|\mathrm{F}| = N$
1  begin
2      $S = 10^{EdB/10}$
3      $n = log_2 N$
4      $W \in \mathbb{R}^N, W(0) = 4S$
5      for $i$ = 1 to $n$ do
6          $d = 2^i$
7          for $j$ = 1 to $\frac{d}{2} - 1$ do
8              $W(d/2 + \mathrm{j}) = A(W(\mathrm{j}-1))$; for APGA (4-15) or
9              $W(d/2 + \mathrm{j}) = S(W(\mathrm{j}-1))$; for SPGA (4-18)
10             $W(\mathrm{j}) = 2W(\mathrm{j}-1)$
11         end
12     end
13     return $F$ = Sorts $W$ indices in ascending order
14 end
```

also to evaluate the accuracy when comparing SPGA to EGA. According to [81], we define as reference the set of frozen bits of PGA and EGA, called the reference set of frozen bits ($\mathcal{A}_c^{ref}$). We use it as a reference to compare the sets of frozen bits ($\mathcal{A}_c$) the all construction methods. The number of different positions between $\mathcal{A}_c$ and $\mathcal{A}_c^{ref}$ is defined by

$$|\mathcal{A}_c \setminus \mathcal{A}_c^{ref}| \triangleq |\{x \in \mathcal{A}_c : x \notin \mathcal{A}_c^{ref}\}|. \qquad (4\text{-}19)$$

We can use NDP as an indication of the quality of the approximation for a given n. The smaller the NDP measure, the smaller the number of different frozen positions between $\mathcal{A}_c$ and $\mathcal{A}_c^{ref}$, which can lead to better FER for $\mathcal{A}_c$, closer to the FER with the ideal positions $\mathcal{A}_c^{ref}$. This approach is more effective for measuring the quality of approximations because we effectively compare the polar codes construction designs.

We then define a mathematical expression for the channel difference with a unified index to compare the approximation methods for long blocks, which we call Accumulative Design Error (ADE), and whose main property is to account for the NDP. We define ADE as

$$\mathrm{D}(n) = \sum_{i=1}^{n} X^i, \qquad (4\text{-}20)$$

where $n = \log_2 N$ and X is the NDP for n. We have that the set $\mathcal{A}_c$ has different values for each approximation method. We can say that there is an optimal polynomial multi-segment approximation of (4-1), such that

$$\lim_{n\to\infty} \mathrm{D}(n) = 0,$$

that is, the approximation function is identical to the original function ($\mathcal{A}_c = \mathcal{A}_c^{ref}$), but the computational cost is prohibitive.

The following demonstrates a way to define how much more accurate one approximation is than another in terms of NDP. We can consider that there is a two sub-optimal polynomial approximation, which is feasible and has low computational cost, called p_a and p_b, where we get $\mathcal{A}_c^a$ and $\mathcal{A}_c^b$, respectively, such that

$$\sum_{i=1}^{n} |\mathcal{A}_c^a \setminus \mathcal{A}_c^{ref}|(i) < \sum_{i=1}^{n} |\mathcal{A}_c^b \setminus \mathcal{A}_c^{ref}|(i) \text{ or}$$
$$\sum_{i=1}^{n} X_a^i < \sum_{i=1}^{n} X_b^i. \tag{4-21}$$

The proof is given in Appendix A.

4.6 Numerical Results

In this section, we evaluate the proposed APGA and SPGA construction techniques and compare them against existing approaches for several scenarios. In particular, we assess the performance of the proposed construction techniques for medium and long blocks. We follow the terminology in terms of block lengths adopted by related work on polar codes for the code construction scenario, as noted in [77]-[79]. In subsection A, "Designs for medium blocks", the comparison is made for block lengths up to 2048 bits, as can be seen in Table 4.1 and in Figures 4.9, 4.10 and 4.11. In subsection B, "Designs for long blocks", we compare the methods for blocks with lengths greater than 4096 bits. In Tables 4.2 and 4.3, and NPD comparisons are made for blocks from 2048 bits to 131072 bits. Additionally, in Figure 4.12 we compare the FER performance for blocks for $n = 12$, $n = 14$ and $n = 16$, which is equivalent to long blocks with length of 4096 bits, 16384 bits and 65536 bits, respectively.

In the following, we illustrate the results of MC simulations, with the AFF3CT toolbox [86]. We simulated for Binary Phase Shift Keying (BPSK) modulation, AWGN channel, SC decoder and design-SNR = 1 dB. The simulation loops adopt as stopping criterion the counting of 200 frame errors. The exact GA is calculated with extremely high accuracy through careful numerical integration.

4.6.1 Designs for Medium Blocks

In Table 4.1 we have the NDP between APGA and PGA, for $R = (1/2, 1/3, 2/3)$. The observed values are due to the application of APGA in the construction of polar codes. Note that NDP has an increasing trend as N increases.

Table 4.1: NDP between APGA and PGA.

R	128	256	512	1024	2048
1/2	0	0	0	2	8
1/3	0	0	0	2	6
2/3	0	0	2	2	6

In Figure 4.9 we have the performance between PGA and EGA. For $N \geq 128$, the FER graph shows an increasing PGA gain with respect to EGA with increasing N, being 0.25 dB for $N = 2048$.

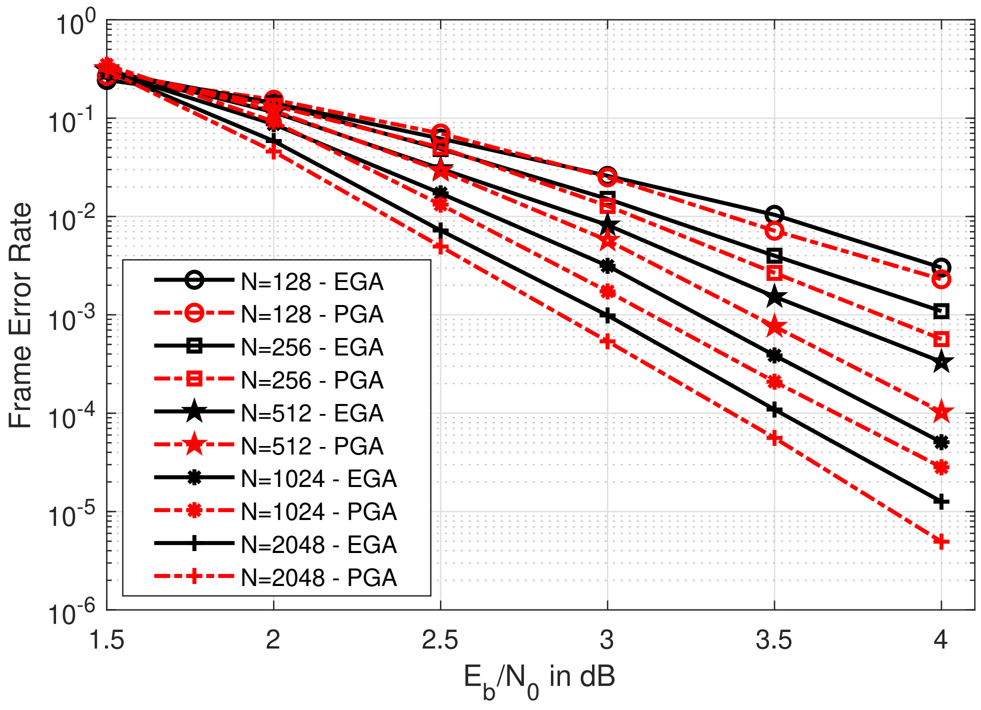


Figure 4.9: FER performance of EGA and PGA, $N = 2^n$, $n = (7, 8, 9, 10, 11)$ and $R = 1/2$.

The performance of PGA and EGA is shown in Figure 4.10 for various block lengths. We remark the FER gain for $N \geq 256$, reaching 0.15 dB for

$N = 2048$. Note also that $N = 128$ there is no difference between PGA and EGA performances.

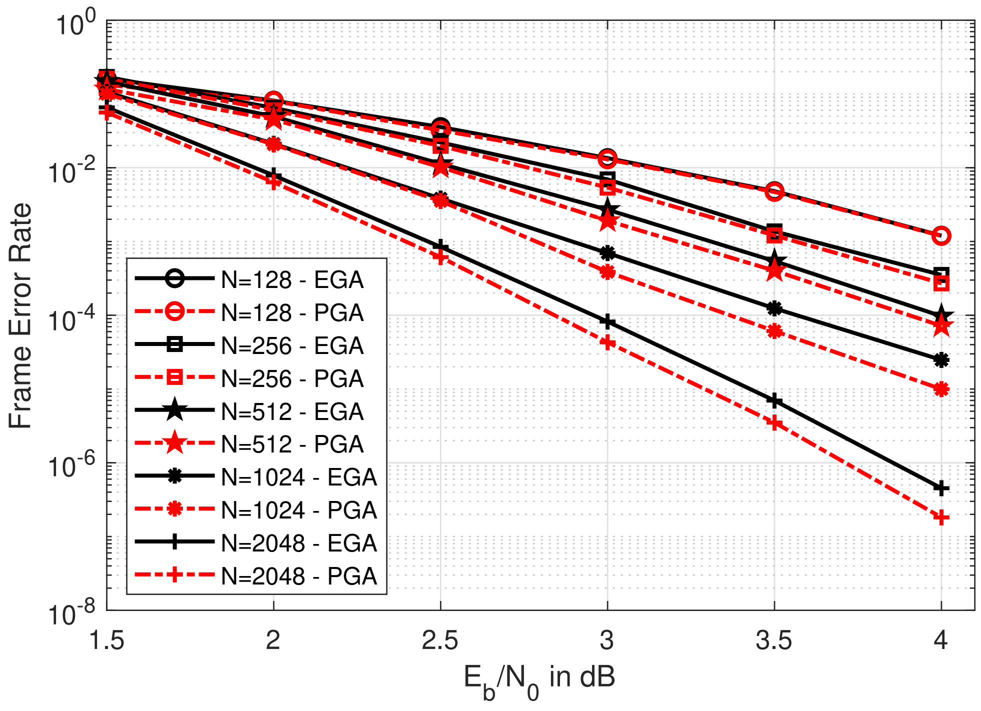


Figure 4.10: FER performance of EGA and PGA, $N = 2^n$, $n = (7, 8, 9, 10, 11)$ and $R = 1/3$.

Now, we compare PGA and EGA, with $R = 2/3$ and various block lengths in Figure 4.11. Note an FER gain of $N \geq 512$ from PGA, reaching 0.25 dB for $N = 2048$. According to the Table 4.1, the ADE between PGA and EGA for $N = 128$ is zero, that is, the code construction is the identically. The same can be observed for $N = 256$.

Note that in Figure 4.10 the FER for PGA and EGA at $N = 128$ is the same, that is, for polar codes with $N = 128$ and $R = 1/3$ the set $\mathcal{A}$ obtained by the PGA method is the same set $\mathcal{A}$ obtained by the EGA method. The same can be seen in Figure 4.11 for $N = 128$ and for $N = 256$, both with $R = 2/3$. The PGA for $N = 128$ is the same as EGA for $N = 128$ and the PGA for $N = 256$ is the same as EGA for $N = 256$.

We can note that in the scenario of medium blocks, with PGA we observe a continuous improvement in the FER when compared to EGA.

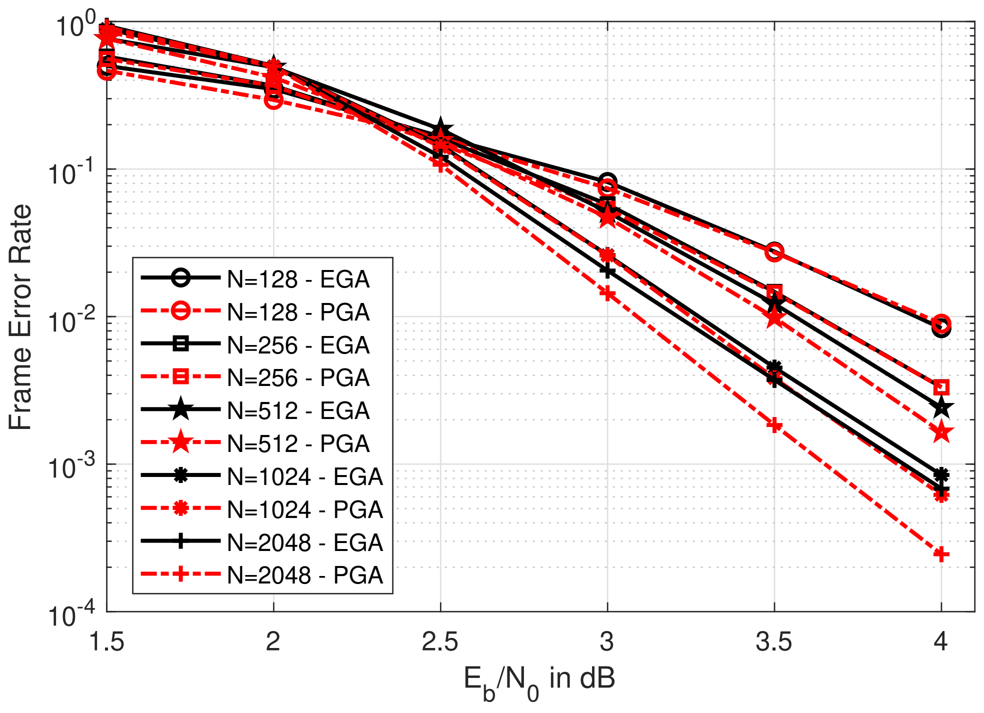


Figure 4.11: FER performance of EGA and PGA, $N = 2^n$, $n = (7, 8, 9, 10, 11)$ and $R = 2/3$.

4.6.2
Designs for Long Blocks

Performance comparison by NDP can be used as a construction quality parameter. However, we observe that small NDP, that is, very small percentage differences of construction, are hardly observable in terms of FER curves, despite representing a better approximation. Another important aspect in comparing the approximation methods is the RMSE. It can be used effectively when the intervals used for approximation are equal, which is not the case in our analysis.

We present in the tables the NDP difference between the construction of the SPGA and the constructions proposed by Trifonov (RGA) [76], Fang (SGA) [77], Dai (AGA-4) [78] and Ochiai (LGA) [79]; for $R = (1/2, 1/3, 2/3)$. We included in the comparison the approximation RGA due to its polynomial format, which meets the simplification requirements objective of this study. The approximation SGA follows the same strategy. This difference in channels, i.e., NDP, represents the quality of constructions in relation to the EGA. The most accurate construction achieves the smallest NDP, i.e, the difference for the reference set of frozen bits ($\mathcal{A}_c^{ref}$).

Initially, in Table 4.2, we have NDP for $R = 1/2$. Observe that SPGA has lower NDP with the increase of N. This means that the SPGA approximation is more accurate than the others approximation.

Table 4.2: NDP for EGA with $R = 1/2$.

N	RGA	SGA	AGA-4	LGA	SPGA
2048	2	4	2	2	0
4096	6	10	2	2	0
8192	16	18	2	2	4
16384	30	46	10	8	10
32768	68	90	12	16	16
65536	158	176	42	42	34
131072	342	332	78	84	60

In Table 4.3, for $R = 2/3$, the comparison is between SPGA, RGA and SGA, which are polynomial approximations. Among them, the SPGA remains with a lower NDP, that is, more accurate than the others. In turn, the design techniques AGA-4 and LGA have the smallest NDP, but it should be remarked that they are approximations of greater complexity that include transcendental functions.

Table 4.3: NDP for EGA with $R = 2/3$.

N	RGA	SGA	AGA-4	LGA	SPGA
2048	6	0	0	0	0
4096	24	6	4	4	2
8192	54	18	4	4	6
16384	132	32	8	8	10
32768	386	96	22	24	28
65536	766	396	34	32	106
131072	1840	428	78	80	336

For $R = 1/3$, as can be seen in Table 4.4, SPGA remains with the lowest NDP, which suggests that it is the most accurate approximation. We can see that the SPGA has the lowest NDP in the three scenarios, that is, for $R = (1/2, 1/3, 2/3)$, and this characteristic is maintained with the increase of N. We can conclude that SPGA is the most accurate approximation of EGA.

In fact, we can demonstrate that the proposed approximations are better as evidenced by the ADE, which also indicates better performance. For example, for EGA in Table 4.4 we have NPD for EGA of the methods

RGA, SGA, AGA-4, LGA and SPGA. According to (4-20), the ADE account for the NDP for each n. In other words, comparing with EGA, for each method, the difference of channels in the code project is calculated by n, and finally we total differences. In this way, we can compare all construction methods in relation to EGA, and the smaller the ADE, the better the approximation for EGA. So, we have

$$\begin{aligned}\mathrm{ADE}_{\mathrm{SPGA}}(17) &= \sum_{n=1}^{17} X^n,\\ &= X^{11} + X^{12} + X^{13} + X^{14}\\ &+ X^{15} + X^{16} + X^{17},\\ &= 0 + 4 + 0 + 10 + 14 + 32 + 44,\\ &= 104.\end{aligned}$$

It can be noted that $\mathrm{ADE}_{\mathrm{SPGA}}(17)$ has the lowest value among the methods in all analyzed scenarios. Hence,

$$\mathrm{ADE}_{\mathrm{SPGA}}(17) < \mathrm{ADE}_{\mathrm{SGA}}(17) < \mathrm{ADE}_{\mathrm{RGA}}(17). \tag{4-22}$$

Table 4.4: NDP for EGA with $R = 1/3$.

N	RGA	SGA	AGA-4	LGA	SPGA
2048	6	4	2	2	0
4096	4	2	2	4	4
8192	10	10	10	8	0
16384	26	14	8	10	10
32768	48	26	18	12	14
65536	96	46	34	32	32
131072	192	118	72	66	44

In Figure 4.12 we have the FER performance of the code construction alternatives observed in Table 4.4, for $R = 1/3$. Although the AGA-4 and LGA approximations have performances comparable to EGA, they are more complex approximations due to transcendental functions.

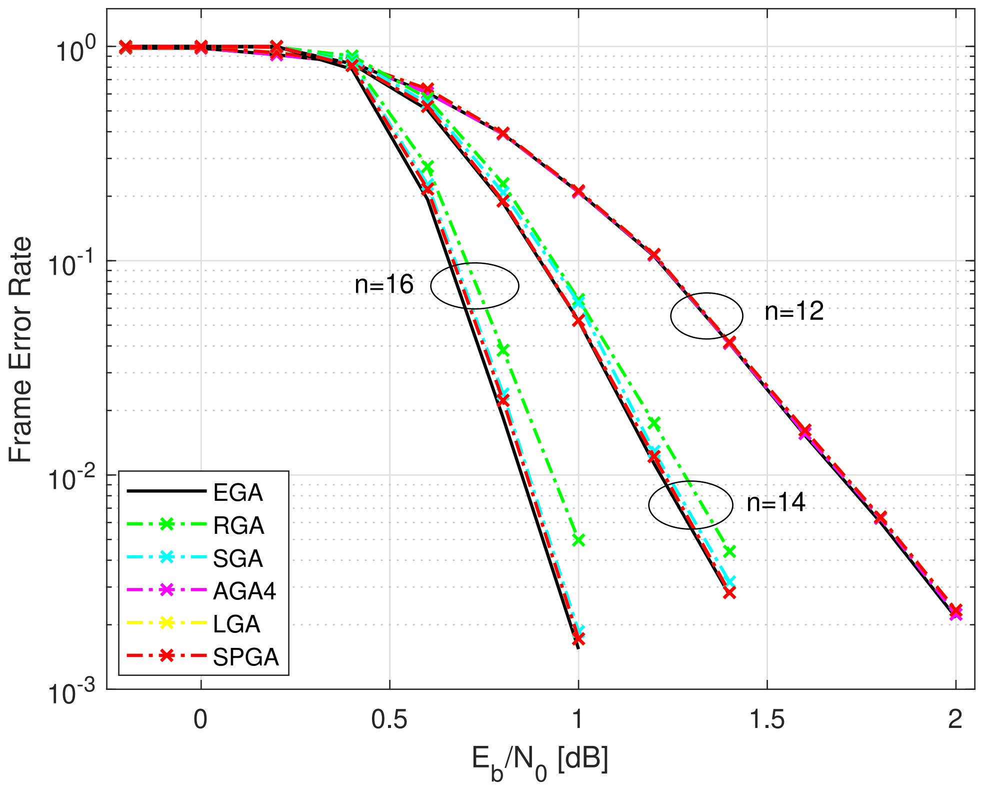


Figure 4.12: FER comparison between EGA, RCA [70], SGA [77], AGA-4 [78], LGA [79] and SPGA, $N = 2^n$, $n = (12, 14, 16)$ and $R = 1/3$, with SC decoder and AWGN channel.

That the proposed SPGA method has better performance as compared to other code construction methods. This is because the SPGA method was optimized with the aid of the analysis of ψ in (4-7). All other methods work almost equally well, with very close approximations to EGA. We can observe the performance similarity among the approximations, despite the SPGA being the best approximation in the scenario considered.

In Figure 4.13 we compare the proposed PGA, GA and the methods for long blocks proposed by Fang [77], Dai [78] and Ochiai [79]. The results show that these methods have the same FER performance as GA.

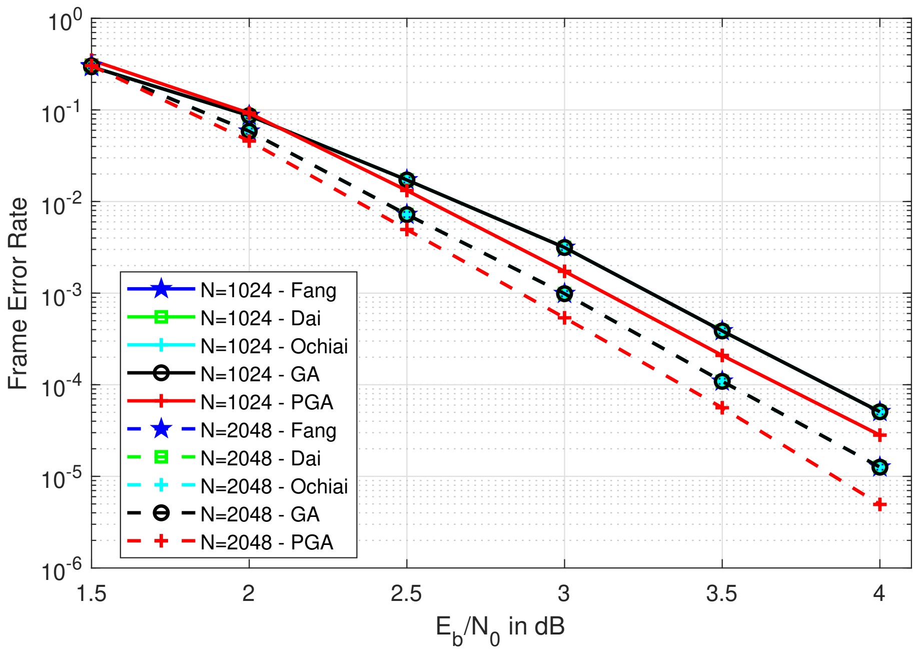


Figure 4.13: FER performance between GA, PGA, Fang [77], Dai [78] and Ochiai [79] for $R = 1/2$, SC decoder and AWGN channel.

4.7 Chapter Summary

In this chapter we have presented a novel method for polar codes construction, called PGA, which is suitable for medium and long blocks. We have also presented APGA based on an analysis of the behavior of the PGA function, the identification of its key points and the analysis of the statistical distribution of their LLRs. Using the same analysis criterion, we have developed SPGA for long blocks, in the form of a multi-segment function. Moreover, we have introduced ADE as a figure of merit for comparison of designs because it effectively measures the difference in polar codes construction methods. The effectiveness of the APGA and SPGA approaches has been investigated by comparing them with other design techniques with approximation the same complexity through simulations and analytical arguments, using the ADE of the analyzed approximation techniques.

5
Adaptive Reweighted Sparse Belief Propagation Decoding

5.1
Chapter Overview

In this chapter, we present an adaptive reweighted sparse belief propagation (AR-SBP) decoder for polar codes. The technique is inspired by decoders that employ the sum-product algorithm for low-density parity-check codes. In particular, the adaptive decoding strategy introduces reweighting of the exchanged LLRs in order to refine the message passing, improving the performance of the decoder and reducing the number of required iterations. An analysis of the convergence is carried out along with a study of their complexity. Numerical examples show that the proposed AR-SBP decoding technique outperforms existing decoding algorithms for a reduced number of iterations, enabling low latency applications.

5.2
Introduction and Literature Review

Increasing polar codes decoding speed is an important area of research motivated by the performance requirements of 5G wireless networks [2]. The first decoder proposed for polar codes was the SC decoder [1], which can achieve good error-correcting capability with low complexity. However, the SC decoder is characterized by serial decoding, which is an error-prone decoding strategy. Because of this, these SC decoders often have low performance for applications that require high-speed real-time decoding with low latency. The SCL decoding [18] was proposed to improve the error-correction performance of SC, since it stores the most likely codewords in a list, reducing error probability and improving the performance. Moreover, SCL can be further enhanced by concatenating a cyclic redundancy check code [18]. As reported in the study in [9], [87, 88]; CRC-aided successive cancellation list decoding attains promising error-correction performance. One major disadvantage of SC and SCL decoding is a long decoding latency due to their serial decoding nature. Furthermore, these decoding techniques also do not provide soft-in/soft-out information and, thus, they are not suitable for iterative decoding and detection applications

[105]-[109].

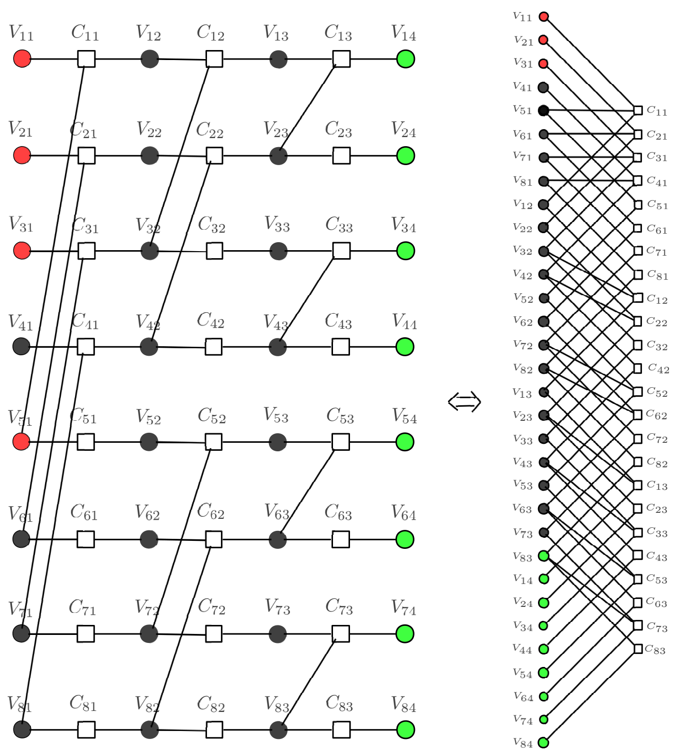
The BP decoder is an alternative to the aforementioned problems. The BP decoder was introduced for polar codes in [89], due to its particular advantage with respect to parallelism, high throughput and low latency. Nevertheless, BP decoding requires a large number of iterations to achieve good performance. A way to improve the performance is to employ BP list decoding [90], which operates when the factor graph of the standard polar codes fails to produce the correct decoding result and the permuted version of the standard graph may yield improved estimates. To improve the convergence of the BP decoder, a reweighting technique based on Euclidean distance was presented in [91], in addition to developing a Q-Learning algorithm to ensure an optimized BP decoding performance. The author in [92] proposed a polar codes design rule that optimizes the convergence of BP decoding and employs puncturing and extension techniques.

BP decoding is performed over a factor graph that corresponds to the generator matrix $\mathbf{G}_N$ [89], which is dense with many short cycles [93]. Cammerer [27] introduced an alternative called LDPC-like polar codes decoder, which is an application of SPA for the decoding of polar codes. It employs a pruning technique that transforms the dense BP decoding into a sparse BP decoding, and uses systematic coding to maximize the coding performance. However, its performance is still worse than the standard BP and gets even worse for long blocks. Ebada [94] proposed a performance improvement in terms of BER and FER with the polar codes construction optimized for LDPC-like decoding by genetic algorithms. The corresponding performance is comparable to SCL but has even better performance for long blocks. Chen [95] proposed a polar decoding method based on both the Layered BP and the modified Node-Wise Residual BP (NW-RBP) scheduling strategies. The performance obtained is comparable to SCL even for long blocks.

In this chapter, we present the adaptive reweighting sparse BP decoder for polar codes. The AR-SBP decoder is simple to implement and is based on the LDPC-like decoding. The proposed reweighting can reduce the number of iterations without increasing the complexity of the decoder. Therefore, the proposed reweighting algorithm allows the reduction of the average number of iterations. Other decoders require more complex implementations and do not have a fast convergence. For example, the NW-RBP decoder needs a low number of iterations but it is necessary to implement a step of searching and classification of the residues [95].

5.3 Dense G Matrix

The original BP decoder employs factor graph representations of the generator matrix **G**, where subsequent iterations are conducted on the polar codes encoding factor graph [89]. The original polar BP decoder was covered in Section 2.8. The original polar BP decoder factor graph of a PC(8,4), shown in Figure 5.1 and described in [94, Fig. 1], consists of n stages, where each stage contains two types of nodes, $V_{i,j}$ and $C_{i,j}$.



V_{11} C_{11} V_{12} C_{12} V_{13} C_{13} V_{14}
V_{21} C_{21} V_{22} C_{22} V_{23} C_{23} V_{24}
V_{31} C_{31} V_{32} C_{32} V_{33} C_{33} V_{34}
V_{41} C_{41} V_{42} C_{42} V_{43} C_{43} V_{44}
V_{51} C_{51} V_{52} C_{52} V_{53} C_{53} V_{54}
V_{61} C_{61} V_{62} C_{62} V_{63} C_{63} V_{64}
V_{71} C_{71} V_{72} C_{72} V_{73} C_{73} V_{74}
V_{81} C_{81} V_{82} C_{82} V_{83} C_{83} V_{84}
$\Leftrightarrow$
v_{11} v_{21} v_{31} v_{41} v_{51} v_{61} v_{71} v_{81} v_{12} v_{22} v_{32} v_{42} v_{52} v_{62} v_{72} v_{82} v_{13} v_{23} v_{33} v_{43} v_{53} v_{63} v_{73} v_{83} v_{14} v_{24} v_{34} v_{44} v_{54} v_{64} v_{74} v_{84}
c_{11} c_{21} c_{31} c_{41} c_{51} c_{61} c_{71} c_{81} c_{12} c_{22} c_{32} c_{42} c_{52} c_{62} c_{72} c_{82} c_{13} c_{23} c_{33} c_{43} c_{53} c_{63} c_{73} c_{83}

Figure 5.1: Original BP decoder factor graph for a PC(8,4) with $\mathcal{A}^c = \{4, 6, 7, 8\}$ and corresponding dense factor graph non-sparse **H**.

Instead of a fully parallel message update, the nodes are updated stage-by-stage. Thus, a single full decoding iteration describes the consecutive activation of stage 1 to n (forward) and from n to 1 (backward). Traditionally, BP decoding is terminated when reaching a pre-defined maximum number of iterations or achieving a specific pre-defined stopping criterion.

The parity-check matrix **H** of a polar codes with generator matrix **G** can be constructed from the columns of **G** with indices in $\mathcal{A}^c$, where $\mathcal{A}^c$ denotes the set of frozen indices. The resulting matrix **H** is a highly dense parity-check matrix with a maximum check node degree of N. Consequently, this leads to a decoding failure if sum-product decoding algorithms are performed on this dense **H** [27].

5.4 Transforming a Dense G Matrix in a Sparse H Matrix

The author in [27] proposes that polar codes can be interpreted as LDPC codes with an underlying sparse Tanner graph. This is illustrated in Figure 5.2, BP original factor graph on the left, dense **G**, and its corresponding bipartite factor graph, i.e., sparse **H**, on the right. With the pruning techniques proposed in [27], the size of the bipartite graph can be significantly reduced and made practical to use. This means that any polar codes can be decoded using conventional LDPC decoders based on the SPA. This has the great advantage of reusing the existing hardware implementations of LDPC decoders, in addition to making use of the available systematic LDPC analysis and design tools.

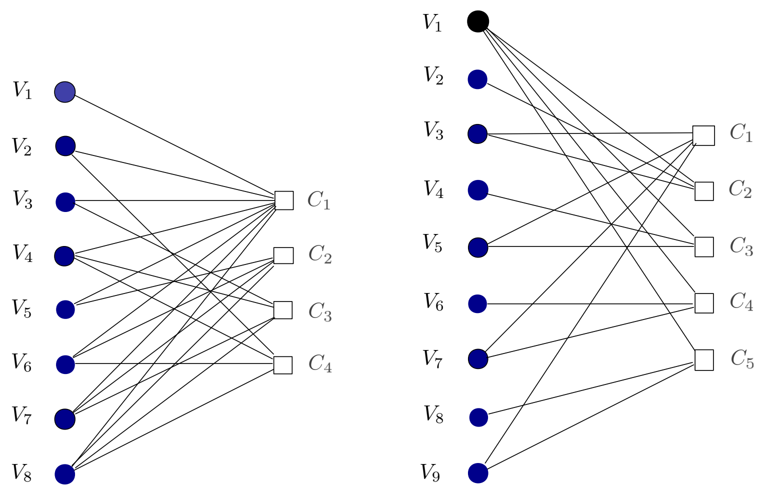


Figure 5.2: Dense vesus sparse Tanner graphs for a PC(8,4).

As depicted in [27] and [94], the $\log_2(N)+1$ sets of variable nodes V and $\log_2(N)$ sets of check nodes C in BP original factor graph can be re-grouped into a bipartite graph consisting of only two sets: Vs and Cs. Applying some basic pruning, the graph can be significantly reduced with an approximate reduction factor of 80% [27]. For more details on the pruning of the originally large bipartite graph, refer to [27]. A graphic example of the pruning process can be seen in Figure 5.3, and described in [27, Fig. 4]. Note that the pruning method is very efficient in reducing the graph.

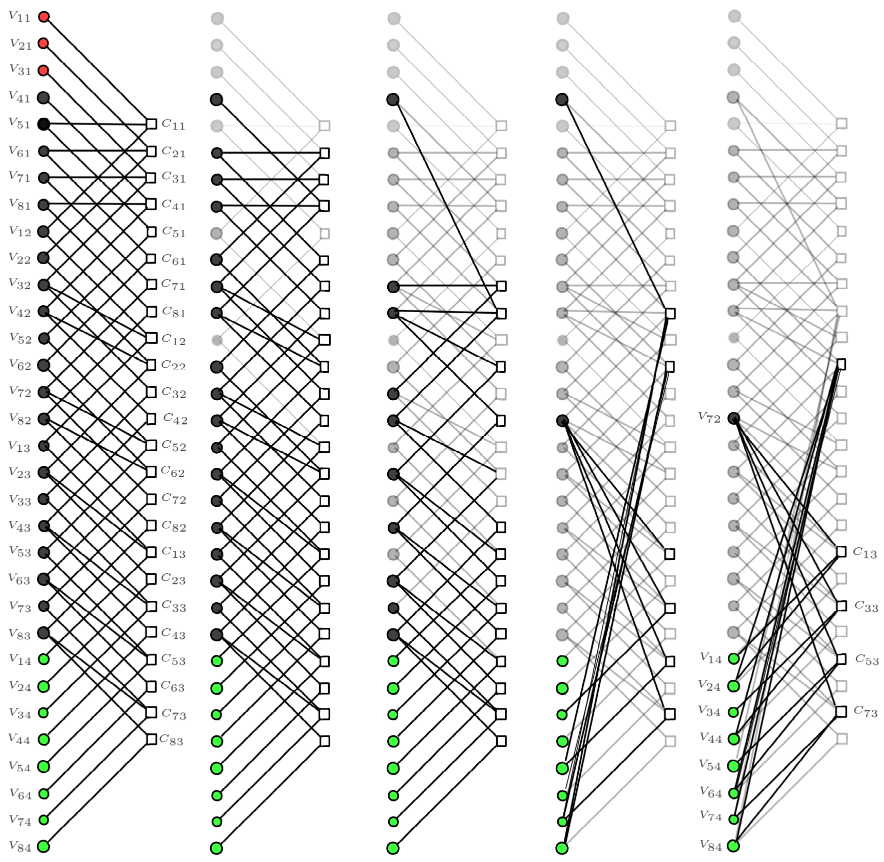


Figure 5.3: Example pruning of the PC(8, 4) graph with $\mathcal{A}^c = 4, 6, 7, 8$.

5.5 Proposed Adaptive Reweighted Sparse BP Algorithm

In this section, we present an adaptive reweighting technique for BP decoders. The adaptive reweighting method is based on the distance between the $\lambda_{v\to c}$ at the iteration T and whether their signals have changed over each iteration. Note that in Figure 5.4, the reweighting factor, $\rho \in (0, 1]$, is applied in V and must guarantee the convergence of the decoding algorithm to a fixed point $\lambda_{v\to c}$ [96]. When $\rho = 1$, we have the message passing BP decoding algorithm for polar codes. As we can see in (2-31), the LLR update is applied to V nodes. Therefore, the reweighting process consists of assessing how LLRs evolve in terms of signal and modulus over time in the V updates of messages.

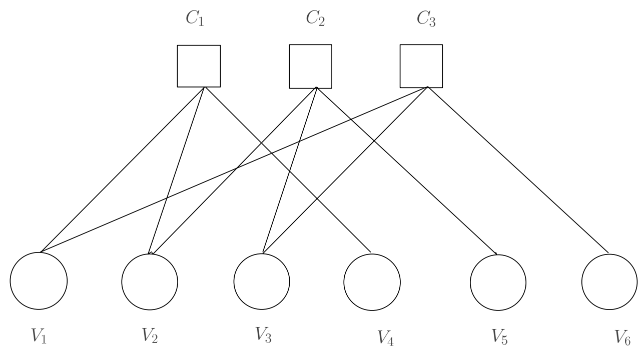


Figure 5.4: Example of a Tanner graph.

As the node updates depend on the signal and modulus operators, it is undeniable the importance of these operators on the reweighting process. Thus, based on (2-31), we introduce three reweighting factors, ρ, β, and Δ; which will adaptively modify how the LLRs are updated, as can be in equations (5-1), (5-2) and (5-3).

Rewriting (2-31) and including a reweighting implementation, we have

$$\lambda_{v\to c} = \rho \cdot \left(\lambda'_{v\to c} + \sum_{n=1}^{N} \Lambda_{c\to v} \right), \tag{5-1}$$

where ρ is the associated edge weight, and

$$\rho = 1 - \beta \cdot \left[\frac{||\lambda_{v\to c}| - |\sum_{n=1}^{N} \Lambda_{c\to v}||}{(|\lambda_{v\to c}| + |\sum_{n=1}^{N} \Lambda_{c\to v}|)} \right] \cdot \Delta, \tag{5-2}$$

where Δ is correction factor of the adjustment direction, or increment or decrement, with

$$\Delta = \operatorname{sign}(\lambda_{v\to c} + \sum_{n=1, v\neq v'}^{N} \Lambda_{c\to v}). \tag{5-3}$$

The β parameter suggests the general correction factor. We initially consider β equal to 1.

The V updates can be summarized in signal and modulus successive operations, the reweighting is based on the distance between the LLRs of $|\lambda_{v\to c}|$ and $|\lambda'_{v\to c}|$, and whether the signals have changed over the iterations. Note that when $|\lambda_{v\to c}|$ and $|\lambda'_{v\to c}|$ are close to each other, ρ is approximately equal to 1. Consequently, the V update is similar to (2-30). Besides that, as these values deviate, the greater the reweighting and the signal deviations are considered by Δ.

Furthermore, it is worth noting that, in this implementation, β is a general factor for all processing elements, whose simulations have shown that it must belong to the range (0,1]. Thus, an open problem is how to set up the best β for a specific input. In a similar reweighting scenario for standard BP

[89], the authors in [91] configure the best β with the Q-learning algorithm development.

A high-level description of the AR-SBP algorithm is illustrated in Algorithm 12. The algorithm takes the received $\mathbf{y}$, the code block length N, the maximum number of iterations T_{max}, and calculates the estimated codeword $\hat{\mathbf{x}}$ as an output vector.

Algorithm 12: Adaptative Reweighted sparse BP decoder

```
Input: maximum loop, $\mathrm{T}_{max}$
Input: sparse matrix of $\mathbf{G}_N$, $\mathbf{H}$
Input: number of cs, C
Input: number of vs, V
Input: vector LLR output, $\mathbf{y}$
Output: estimated codework, $\hat{\mathbf{x}}$
begin
    $\mathrm{T} = 0$;
    for $i$ = 1:$C$ do
        for $j$ = 1:$V$ do
            $\lambda'(\mathrm{i,j})_{v\to c} = y(\mathrm{j}) * H(\mathrm{i,j})$
        end
    end
    repeat
        $\mathrm{T} = \mathrm{T} + 1$
        for $i$ = 1:$C$ do
            for $j$ = 1:$V$ do
                $\Lambda(\mathrm{i,j})_{v\to n} = 2 \cdot \tanh^{-1}\left(\prod_{k=1,k\neq \mathrm{j}}^{V} tanh\left(\frac{\lambda'(\mathrm{i},k)_{v\to c}}{2}\right)\right)$
            end
        end
        for $j$ = 1:$V$ do
            for $i$ = 1:$C$ do
                $\Delta(\mathrm{i,j}) = \mathrm{sign}\left(\lambda(\mathrm{i,j})_{v\to c} + \sum_{k=1}^{C} \Lambda(k,\mathrm{j})_{c\to v}\right)$
                $\rho(\mathrm{i,j}) = 1 - \beta\left[\frac{||\lambda'(\mathrm{i,j})_{v\to c}| - |\sum_{n=1}^{N}\Lambda(\mathrm{i,j})_{c\to v}||}{(|\lambda'(\mathrm{i,j})_{v\to c}| + |\sum_{n=1}^{N}\Lambda(\mathrm{i,j})_{c\to v}|)}\right]\Delta(\mathrm{i,j})$
            end
            $\lambda_{v\to c} = \rho * \left(\lambda'_{v\to c} + \sum_{n=1}^{N}\Lambda_{c\to v}\right)$
            $\hat{x}(\mathrm{j}) = \begin{cases} 1, & \lambda(\mathrm{i,j})_{v\to c} \leq 0 \\ 0, & \lambda(\mathrm{i,j})_{v\to c} > 0, \end{cases}$
        end
        for $i$ = 1:$C$ do
            for $j$ = 1:$V$ do
                $\lambda'(\mathrm{i,j})_{v\to c} = \rho(\mathrm{i,j}) \cdot \lambda(\mathrm{i,j})_{v\to c}$
            end
        end
    until $T = T_{max}$ or $\hat{\boldsymbol{x}}\boldsymbol{H}^T = 0$;
    return $\hat{\boldsymbol{x}}$
end
```

5.6 Convergence Analysis

The proposed AR-SBP algorithm is an iterative algorithm in which graph nodes exchange statistical information through a sequence of message passing updates. For the graph model, updates can be derived as a form of parallel dynamic programming and are guaranteed to converge and calculate the correct marginal distributions at each node. In the standard BP algorithm, messages are adjusted by weights based on edges determined by the graph structure, in which all weights are unitary. For adequate choices of these weights, it is known that the proposed AR-SBP algorithm converge [96]. Moreover, it has the added benefit of being convex and, consequently, that message passing updates tend to be more stable [97].

According to [96], the convergence of the AR-SBP algorithm can be guaranteed by any of the following explicit conditions: row sum condition and column sum condition, among others. That is, there is always a choice of edge weights for which the associated AR-SBP algorithm converges.

For the row sum condition, we have

$$\max_{(v\to c)} \left(\sum_{u\in V\setminus c} \rho_u + (1-\rho_V) \right) \lambda^n_{v\to c} < 1. \tag{5-4}$$

The author in [96] considers all $\lambda_{v\to c}$ normalized ($\lambda^n_{v\to c}$), i.e. $\sum^V \lambda^n_{v\to c} = 1$. Soon, all $\lambda^n_{v\to c} < 1$. The term $\rho_V = \sum^V \rho$, so we have

$$1 - \sum_{u\in V\setminus c} \rho_u - \sum^V \rho = 1 - \rho_c < 1,$$

and since $\rho_c < 1$, and eq. (5-4) is answered.

For the column sum condition, we have

$$\max_{(v\to c)} \left\{ \rho_C \left(\sum_{u\in C} \Lambda_{u\to c} \right) + (1-\rho_C)\Lambda_{v\to c} \right\} < 1, \tag{5-5}$$

where ρ_C are reweighting parameters associated with check nodes that are set to $\rho_C = 1$. Let us now consider all $\lambda_{v\to c}$ normalized ($\lambda^n_{v\to c}$), i.e., $\sum^V \lambda^n_{v\to c} = 1$. According to eq. (2-30) then all $\Lambda_{v\to c} < 1$ and eq. (5-5) holds.

In turn, the ρ reweighting parameter has an adaptive characteristic depending on the absolute values of the analyzed LLR. We notice that its value tends to γ with the evolution of the iterations since $(||\lambda_{n\to m}| - |\sum_{n=1,n\neq n'}^{N} \Lambda_{m\to n}||)$ tends to zero. Therefore, we have

$$\begin{aligned}\lim_{T\to\infty}\rho &= \lim_{T\to\infty}\left(\gamma - \left[\frac{||\lambda_{v\to c}| - |\sum_{n=1,v\neq v'}^{N}\Lambda_{c\to nv}||}{(|\lambda_{v\to c}| + |\sum_{n=1,v\neq v'}^{N}\Lambda_{c\to nv}|)}\right]\cdot\Delta\right), \\ &= \gamma,\end{aligned} \tag{5-6}$$

with γ set to 1. The ρ factor tends to accelerate convergence with its value decreasing with each iteration. Note further that ρ does not increase the complexity of the BP decoder, which remains $\mathcal{O}(2\mathrm{T}_{max})$ [98].

5.7 Numerical Results

In this section, we present simulation results to evaluate the performance of the AR-SBP decoding algorithm introduced in Section 5.5. The proposed AR-SBP decoder is applied to polar codes to obtain faster convergence, which we show with a detailed analysis of the average iterations. In addition, we will also show a comparative analysis of the proposed AR-SBP decoder with other competing decoders.

In Table 5.1 we have the average number of iterations for PC(128,64), PC(256,128), PC(512,256); for the BP, NW-RBP and the proposed AR-SBP decoders. We can notice that the AR-SBP decoder presents lower average number of iterations than the sparse BP decoder, and close to the NW-RBP decoding algorithm. The AR-SBP decoding algorithm has a convergence close to that of the NW-RBP decoder whereas the AR-SBP decoder is much more efficient from a computational viewpoint than the NW-RBP decoder, which requires the implementation of a step of searching and classification.

Table 5.1: Average number of iteration, T_{max}=20.

N	E_b/N_0(dB)	1	1.5	2	2.5	3	3.5	4
128	BP	20.00	20.00	20.00	20.00	18.00	17.05	15.23
	AR-SBP	19.11	14.22	12.05	9.97	8.71	7.83	7.31
	NW-RBP	19.09	12.18	10.59	9.37	8.41	7.71	7.25
256	BP	20.00	20.00	20.00	19.00	17.13	16.25	13.46
	AR-SBP	19.32	15.35	12.15	10.11	9.13	8.15	7.33
	NW-RBP	19.21	12.85	11.05	9.83	8.75	8.1	7.29
512	BP	20.00	20.00	20.00	18.31	16.74	15.81	12.92
	AR-SBP	19.48	15.53	12.01	10.11	9.23	8.15	7.25
	NW-RBP	19.47	13.21	11.32	10.13	8.55	8.12	7.35

In Figure 5.5 we can see the percentage of reduction of the number of iterations from T_{max}=20. A reduction in the order of 60% is observed for E_b/N_0 $>$ 3dB.

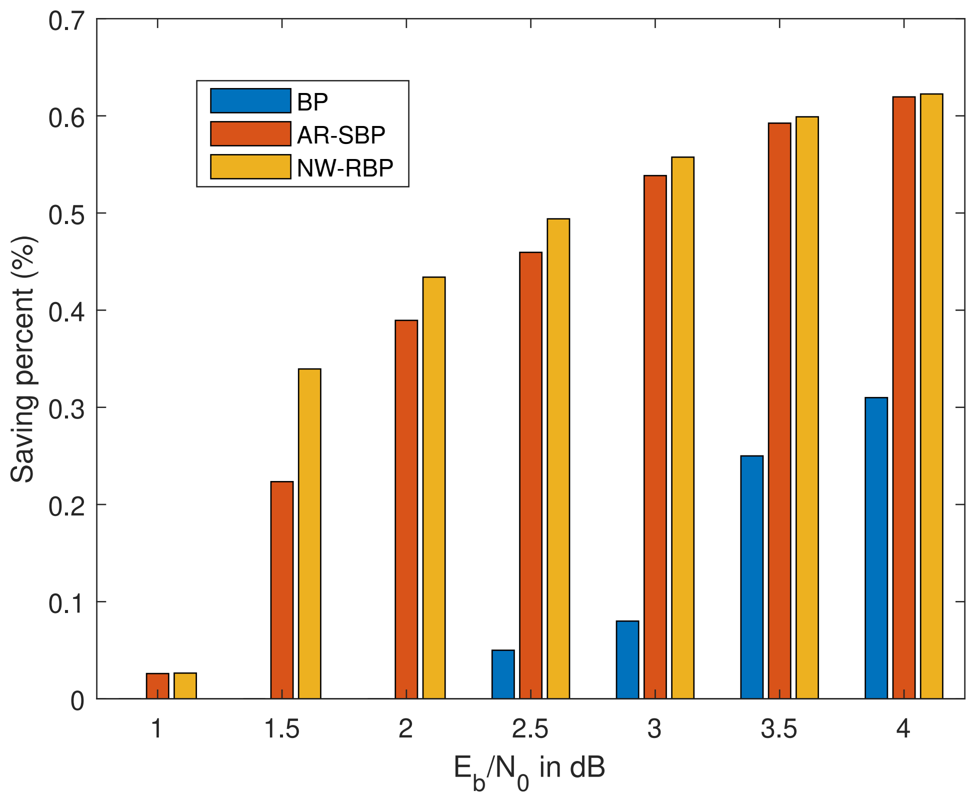


Figure 5.5: The percent of iteration reduction, T_{max}=20.

In Table 5.2 we can see the influence of the rate R on the average number of iterations. We notice that the average number of iterations decreases with the reduction of the rate R, due to the increase in coding redundancy.

Table 5.2: Influence of the rate R on the average number of iterations, T_{max}=20.

$E_b/N_0(dB)$	1	1.5	2	2.5	3	3.5	4
R=1/4	19.17	14.89	11.47	9.53	8.68	7.75	7.02
R=1/2	19.32	15.35	12.15	10.11	9.13	8.15	8.15
R=3/4	19.50	16.09	13.09	10.82	9.60	8.50	8.50

Moreover, the influence on the average number of iterations of the limit imposed by the maximum number of iterations, which can be seen in Table 5.3. We note that the average number of iterations for the two curves converges from $E_b/N_0 = 2$ dB. For $E_b/N_0 < 2$ dB, the BER performance for T_{max} =15 has a significant deterioration.

Table 5.3: Influence on the average number of iterations on the limit imposed by the maximum number of iterations.

$E_b/N_0(dB)$	0.5	1	1.5	2	2.5	3	3.5	4
T$_{max}$=15	15	15	14.89	12.15	10.11	9.13	8.15	8.15
T$_{max}$=20	20	19.32	15.35	12.15	10.11	9.13	8.15	8.15

We illustrate this convergence in Figure 5.6. Note that T$_{max}$=15 is a performance limiter for low E_b/N_0.

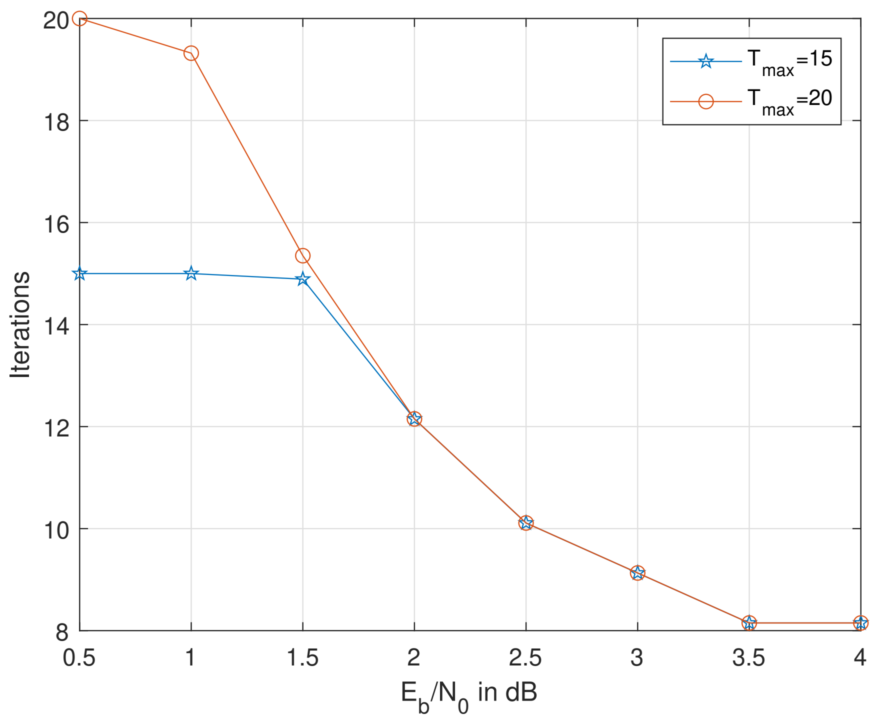


Figure 5.6: Comparison of iterations between limited T$_{max}$=15 and T$_{max}$=20.

In Figure 5.7 we can see the PC(256,128) BER performance with $E_b/N_0 = 2$ dB, for BP, NW-RBP and AR-SBP decoders. We can see that AR-SBP is superior to BP and very close to the performance of NW-RBP. In this scenario, from the sixth iteration, the decoders AR-SBP and NW-RBP are equivalent.

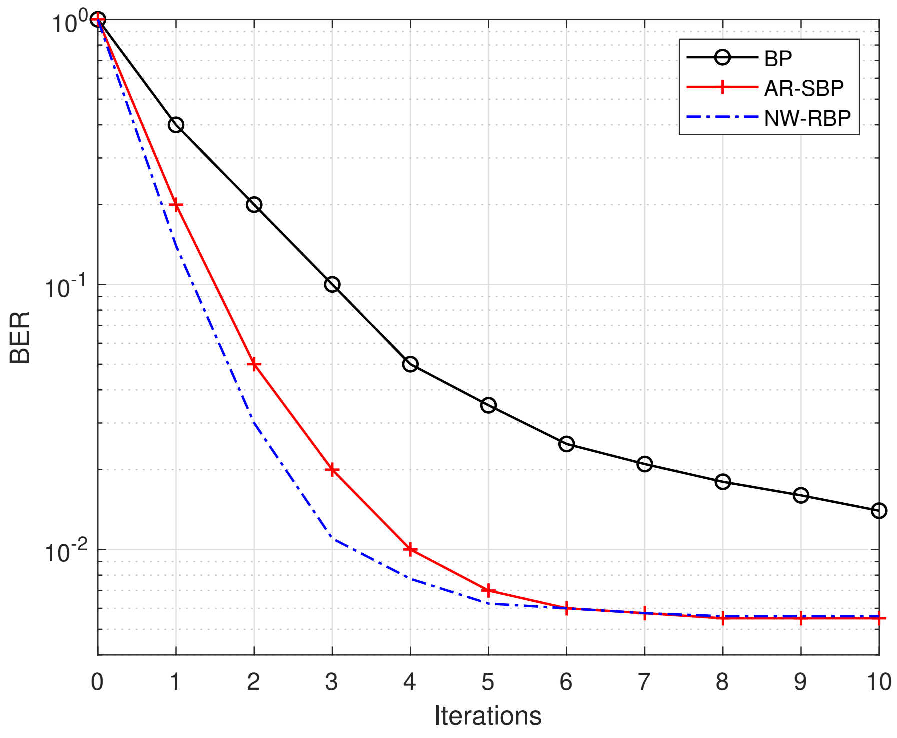


Figure 5.7: Comparison of iterations between sparse BP, AR-SBP and NW-RBP.

In Figure 5.8 we can compare the performance of the BP decoder [89] with the AR-SBP decoder for $R = 1/2$, considering three T_{max} options for BP: 20, 40 and 60. For the AR-SBP we consider T_{max}=20. In this simulation, the channel is AWGN and the polar codes was constructed with the GA algorithm [74] and the design-SNR set to 1 dB. We observed that the performance of the AR-SBP decoder is superior to the BP decoder for a lower T_{max}. The low performance of the BP is due to the excess of short cycles. Moreover, we observed that the AR-SBP decoder has a high success rate in decoding with a low number of iterations.

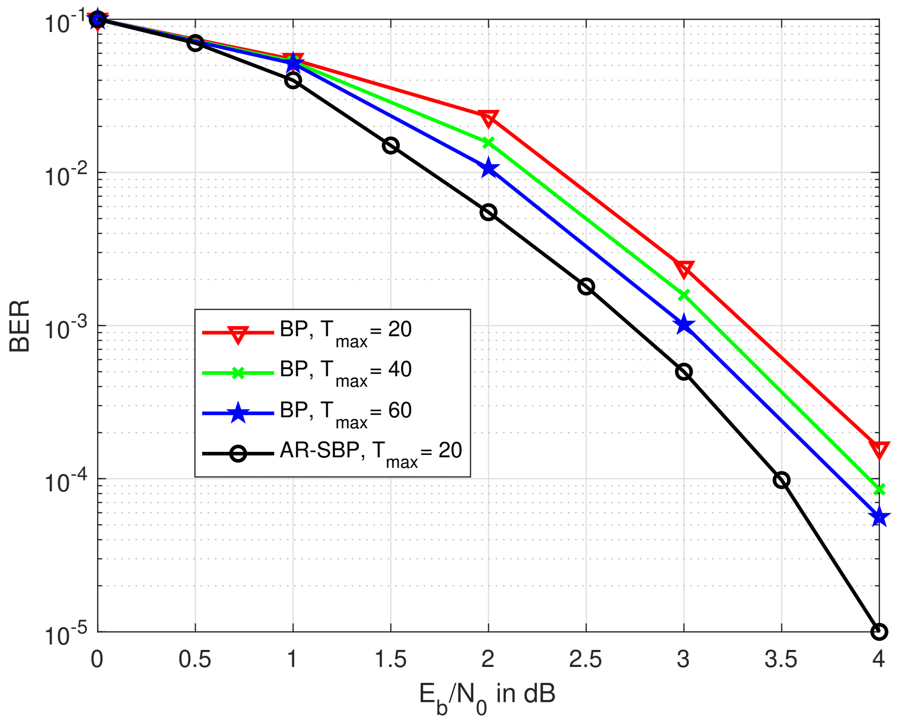


Figure 5.8: Comparison of performance by T_{max} between BP and AR-SBP.

In the next figures we have a comparative analysis of the performance of the polar codes $N = 256$ for several rates R and for the BP, SCL and AR-SBP decoders. For the BP [89] decoder we use T_{max} =60. For SCL decoder [29] we use the list size $\mathcal{L}$=128, this value of $\mathcal{L}$ being very close to the maximum likelihood ($\mathcal{L}$=256 for $N = 256$). In addition, for AR-SBP decoder we use T_{max} =20. The channel is AWGN, the design-SNR is set to 1dB and the polar codes construction method is GA [74]. Note that for N =256 the polar codes construction optimization for the BP decoder by genetic algorithm generates the same $\mathcal{A}^c$ as the GA polar codes construction method [41]. As for the complexity of the decoding algorithm [98] we have for the BP the complexity is $\mathcal{O}(2T_{max} \log N)$, for the SCL the complexity is $\mathcal{O}(\mathcal{L}N \log N)$ and for the AR-SBP the complexity is $\mathcal{O}(2T_{max})$, being the AR-SBP the least complex.

Figure 5.9 shows the performance for $R = 2/3$. In this simulation, we observed that the performance of the AR-SBP decoder is better than the BP decoder, but inferior to the SCL. Compared to BP the performance of AR-SBP is better due to the elimination of short cycles. For high data transmission rates, because of the excess of short cycles we have the low performance in the BP decoders [27]. On the other hand, the SCL decoder presented the best performance at high data transmission rates. Even with a reduced $\mathcal{A}^c$ set, that is, fewer bits for error correction, the size of the list $\mathcal{L}$ allows one to increase the percentage of successful decoding.

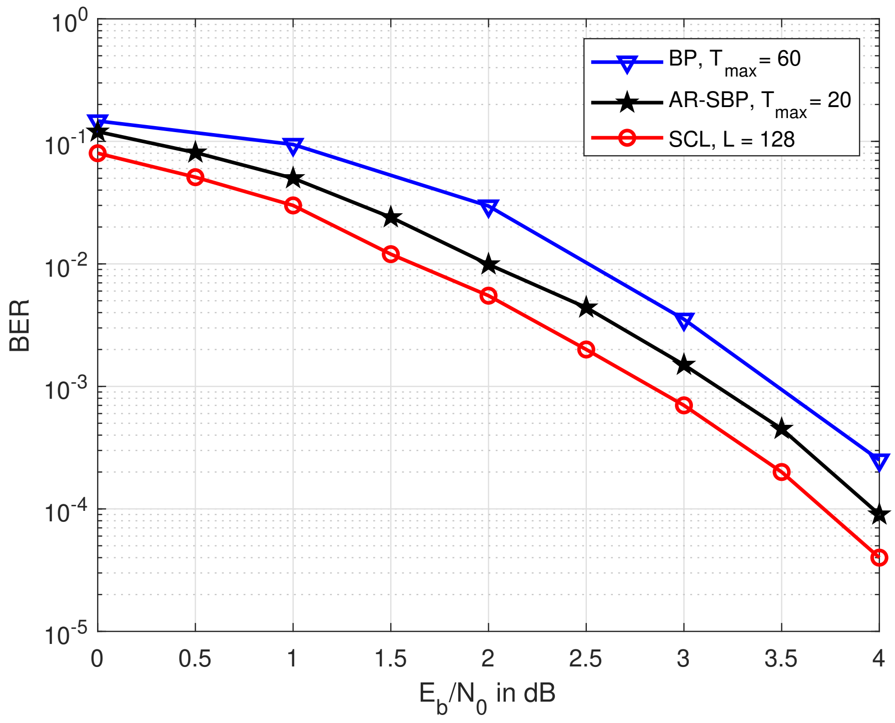


Figure 5.9: Polar codes performance for $N = 256$ and $R = 2/3$, between BP, AR-SBP and SCL, for T$_{max}$=60, T$_{max}$=20 and $\mathcal{L}$=128, respectively.

Figure 5.10 shows the performance for $R = 1/2$. In this simulation we noticed that the performance of the AR-SBP decoder is better than that of the BP decoder and comparable to that of the SCL decoder. The BP decoder presents low performance due to a large number of short cycles when compared to the AR-SBP decoder. As for the SCL decoder, even considering the size of the list, its performance is similar to AR-SBP. At this rate the set $\mathcal{A}^c$ is large enough to guarantee a high rate of successful AR-SBP decoded codewords.

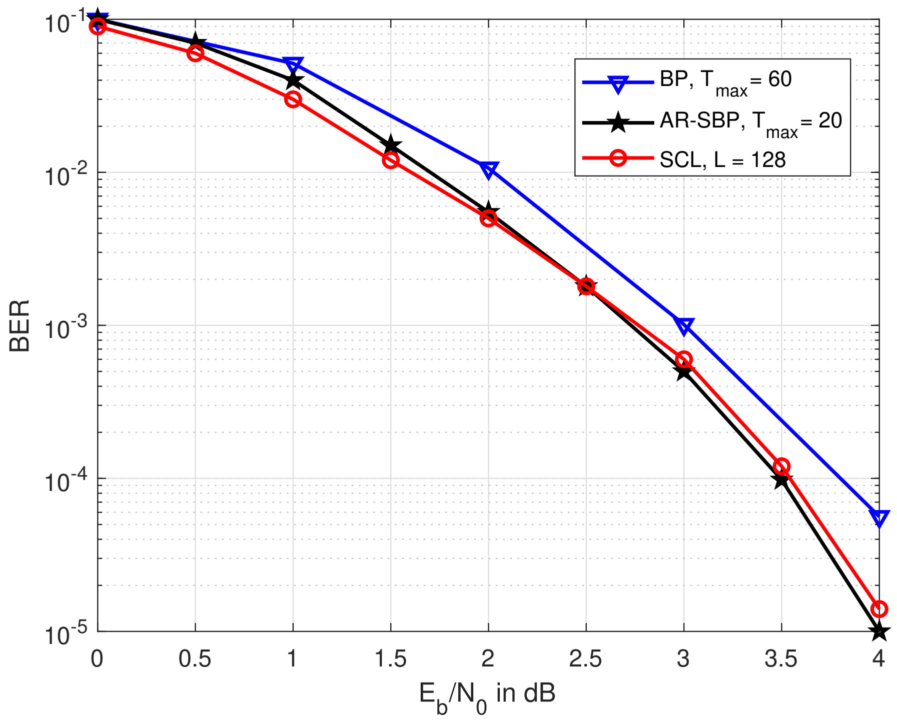


Figure 5.10: Polar codes performance for $N = 256$ and $R = 1/2$, between BP, AR-SBP and SCL, for T$_{max}$=60, T$_{max}$=20 and $\mathcal{L}$=128, respectively.

Figure 5.11 shows the performance for $R = 1/3$. At this rate, the $\mathcal{A}^c$ set is larger, that is, more bits for decoding error correction. In this simulation we noticed that the performance of the AR-SBP decoder is better than the BP decoder and slightly better comparable to the SCL decoder. At low data transmission rates the success of AR-SBP decoding increases considerably. This fact is not observed in the BP decoder, which continues to be penalized by its short cycles when compared to the AR-SBP decoder. As for the SCL decoder, adding the low data transmission rates to the list size, which is the best scenario for high decoding performance, we observe that it is only slightly superior to AR-SBP. This fact implies that the AR-SBP is an excellent alternative for scenarios with codes of low rates if we consider its low computational complexity.

In general, when we compare the performance of polar codes for different rates, we observe that BP decoding always has the worst performance. The proposed AR-SBP decoder, in low and medium R rate, has similar performance to the SCL decoder whereas it is much less complex than the SCL decoder.

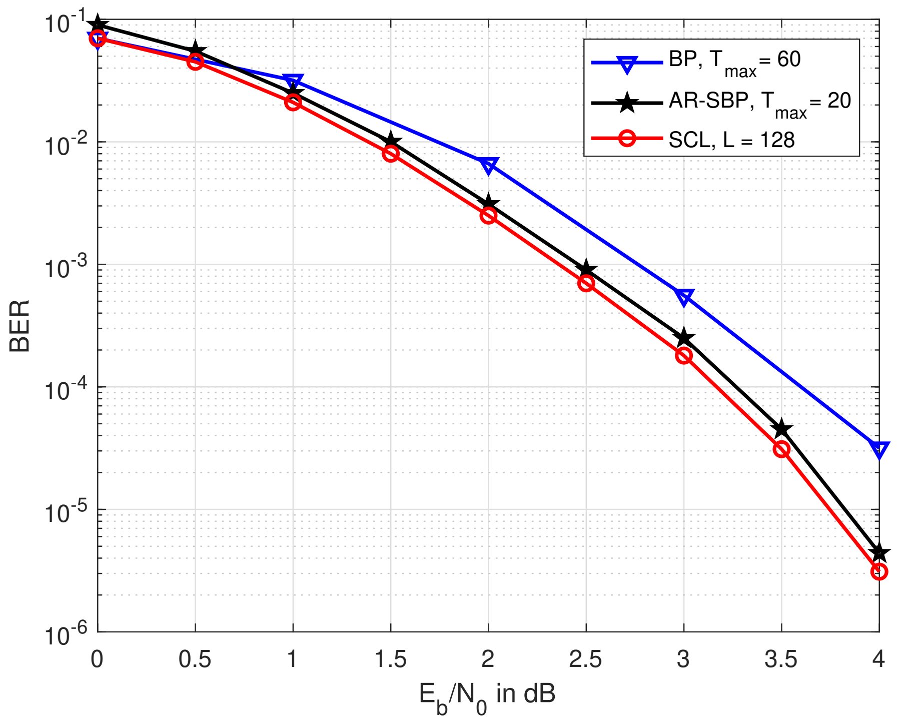


Figure 5.11: Polar codes performance for $N = 256$ and $R = 1/3$, between BP, AR-SBP and SCL, for T$_{max}$=60, T$_{max}$=20 and $\mathcal{L}$=128, respectively.

5.8 Chapter Summary

In this chapter we have presented the AR-SBP decoder for polar codes that seeks to converge with a reduced number of iterations. More specifically, from our experience, adaptive reweighting is ideal for minimizing the convergence time of successful decoding. The proposed AR-SBP algorithm meets the convergence criterion and the numerical results illustrate the performance of the proposed AR-SBP technique, which are promising in relation to existing approaches, without increasing complexity. Therefore, the AR-SBP algorithm, based on message passing with adaptive reweighting provides reduced decoding latency and can handle schemes that perform iterative decoding and detection procedures. Finally, the simulation results have shown that the performance of the proposed AR-SBP decoder for polar codes is better than the BP decoder, and closely approaches the SCL decoder, in some cases it is better (Fig. 5.10). Numerical results illustrate the performance of the proposed AR-SBP technique against existing approaches.

6 Conclusions and Future Work

In this chapter, we draw the concluding remarks of this thesis and discuss potential future works.

6.1 Conclusions

Within the technological scope proposed for 5G systems and for future systems, there is a clear need for flexibility in code lengths to meet all the potential scenarios. Therefore, several techniques for the construction of polar codes with arbitrary length have been studied, which have good trade-off between performance and implementation complexity. However, other aspects of polar coded systems still need to be improved. One focus of attention is the search for more efficient techniques for polar codes construction with long blocks, where some improvements in the GA technique have been proposed. Another research focus is on improving the performance of BP decoders. In fact, due to its parallel processing feature, the performance gain and computational advantages of BP-type decoders as compared to the SC decoder are evident. Thus, this thesis has presented and discussed novel polar code construction for arbitrary length, efficient approximation of GA for medium and long blocks, and improvements in message passing decoding using BP-type algorithms.

In Chapter 2, the technical background of this thesis is provided. Firstly, fundamental concepts of polar codes are presented, which includes the polarization phenomenon, encoding and decoding methods. It then describes in further detail the encoding method, the SC decoding method, the SCL method, the CRC-aided SCL decoding and the BP decoding methods for polar codes.

Chapter 3 addresses the design and implementation of a polar codes construction technique named NUPGA for arbitrary code lengths, which is specific for systems with the SC decoder and for AWGN channels. NUPGA consists of the consideration that the capacity of synthetic channels in the polarization process is non-uniform. The NUPGA technique makes use of the non-uniform polarization obtained by shortening and re-polarizing the synthetic channels to obtain an improved code construction. NUPGA is also useful for designing extended polar codes. It has been also shown that non-uniform polarization achieves the capacity of B-DMC channels. Furthermore, from the results of simulations, it can be seen that the NUPGA designs demonstrate better BER/FER performance for several code construction scenarios. The design of extended polar codes also demonstrates better performance than shortened polar codes in the proposed scenarios.

In Chapter 4 we have presented the construction of polar codes based on PGA techniques. Specifically, we developed two approaches with the advantage of being based on multi-segmented polynomials and of easy implementation. We designed APGA which is optimized for medium blocks and SPGA as an alternative to GA which is optimized for long blocks. Simulation results have shown that the APGA and SPGA construction techniques are very efficient approximations of GA, with notable performance improvement for medium blocks, in addition to less complexity than competing approaches. The effectiveness of the APGA and SPGA approaches was investigated by comparing them with other design techniques with the same order of complexity.

In Chapter 5, we have introduced the AR-SBP decoder for polar codes. The proposed adaptive decoding strategy of AR-SBP introduces the reweighting of swapped LLRs to refine message passing, improving decoder performance and reducing the number of iterations required. More specifically, in our investigation, we have verified that adaptive weighting is ideal for minimizing the convergence time of successful decoding. The proposed AR-SBP algorithm meets the convergence criterion and the numerical results illustrate that the performance of AR-SBP is promising in relation to existing approaches, without increasing complexity.

6.2 Future Work

1. As discussed in Chapters 3 and 4, the starting point is the construction technique based on the GA, both to generate flexibility in code construction with arbitrary length, and in the efficient search for approximations for medium and long lengths. However, recent works investigate more efficient code constructions using search algorithms, exhaustively or optimized by genetic algorithms, with promising preliminary results. A framework that considers polar code construction together with the decoding strategy and the channel would be interesting.

2. Another approach for the design of polar codes that would be promising for future research is the design of polar codes for block fading channels,

which should consider the outage probability rather than the capacity as reported for LDPC codes [101], [102].

3. As discussed in Chapter 5, we showed that the LDPC-like decoders for polar codes can be optimized. Furthermore, the entire legacy of LDPC research should be considered as a source of inspiration of decoding techniques for polar codes in search of performance improvement. Among the important implications is the reuse of hardware. Furthermore, a new topic of investigation is suggested: universal decoding, and additional topics like new extension, puncturing and shortening techniques should be investigated, because they are heavily dependent on the type of decoder.

Bibliography

[1] ARIKAN, E.. **Channel Polarization: A Method for Constructing Capacity Achieving Codes for Symmetric Binary-Input Memoryless Channels**. IEEE Transactions on Information Theory, 55:3051–3073, 2009.

[2] MCC SUPPORT. **Technical Specification Group Radio Access Network**. 3GPP 3GPP, 2018. Acesso em: Novembro de 2018.

[3] OSSEIRAN, A.; BOCCARDI, F.; BRAUN, V.; KUSUME, K.; MARSCH, P.; MATERNUA, M.; QUESETH, O.; SCHELLMANN, M.; SCHOTTEN, H.; TAOKA, H.; TULLBERG, H.; UUSITALO, M. A.; TIMUS, B.; FALLGREN, M.. **Scenarios for 5G Mobile and Wireless Communications: The Vision of the METIS Project**. IEEE Communications Magazine, 52:26–35, 2014.

[6] DEMESTICHAS, P.; GEORGAKOPOULOS, A.; KARVOUNAS, D.; TSAGKARIS, K.; STAVROULAKI, V.; LU, J.; XIONG, C.; YAO, J.. **5G on the Horizon: Key Challenges for the Radio-Access Network**. IEEE Vehicular Technology Magazine, 8:47–53, 2013.

[7] CHEN, S.; ZHAO, J.. **The Requirements, Challenges, and Technologies for 5G of Terrestrial Mobile Telecommunication**. IEEE Communications Magazine, 52:36–43, 2014.

[8] MCC SUPPORT. **Final Report of 3GPP TSG RAN WG1 87**. 3GPP 3GPP, 2017. Acesso em: Novembro de 2017.

[9] BALATSOUKAS-STIMMING, A.; BASTANI PARIZI, M.; BURG, A.. **Llr-based successive cancellation list decoding of polar codes**. IEEE Transactions on Signal Processing, 63:5165–5179, 2015.

[13] KAHRAMAN, S.; VITERBO, E.; ÇELEBI, M.. **Folded successive cancellation decoding of polar codes**. Communications Theory Workshop, 5:57–61, 2014.

[14] MORI, R.; TANAKA, T.. **Performance and construction of polar codes on symmetric binary-input memoryless channels**. International Symposium on Information Theory (ISIT), 1:1496–1500, 2009.

[15] TRIFONOV, P.. **Efficient design and decoding of polar codes**. IEEE Transactions on Communications, 60:3221–3227, 2012.

[16] FOSSORIER, M. P. C.; MIHALJEVIC, M.; IMAI, H.. **Reduced complexity iterative decoding of low-density parity check codes based on belief propagation**. IEEE Transactions on Communications, 47:673–680, 1999.

[17] LEROUX, C.; TAL, I.; VARDY, A.; GROSS, W. J.. **Hardware architectures for successive cancellation decoding of polar codes**. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 1:1665–1668, 2011.

[18] TAL, I.; VARDY, A.. **List decoding of polar codes**. IEEE Transactions on Information Theory, 61:2213–2226, 2015.

[19] CAO, C.; FEI, Z.; YUAN, J.; KUANG, J.. **Low complexity list successive cancellation decoding of polar codes**. IET Communications, 17:3145–3149, 2014.

[20] LIN, B.; SHEN, H.; TSE, D.; TONG, W.. **Low-latency polar codes via hybrid decoding**. 8th international Symposium on turbo Codes and Iterative Information Processing (ISTC), 1:223—227, 2014.

[21] KAI NIU; KAI CHEN. **CRC-Aided Decoding of Polar Codes**. IEEE Communications Letters, 16:1668–1671, 2012.

[23] MURATA, T.; OCHIAI, H.. **On design of CRC codes for polar codes with successive cancellation list decoding**. IEEE International Symposium on Information Theory (ISIT), 1:1868–1872, 2017.

[24] SARKIS, G.; GIARD, P.; VARDY, A.; THIBEAULT, C.; GROSS, W. J.. **Increasing the speed of polar list decoders**. IEEE Workshop on Signal Processing Systems (SiPS), 16:1–6, 2014.

[25] WASSERMAN, D. R.. **Designing polar codes to minimize the BER of CRC-aided list decoding**. Information Theory and Applications Workshop (ITA), 1:1–3, 2016.

[26] LIN, S.; COSTELLO, D.. **Error Control Coding**. Pearson, New York, 2nd edition, 2004.

[27] CAMMERER, S.; EBADA, M.; ELKELESH, A.; BRINK, S. T.. **Sparse graphs for belief propagation decoding of polar codes**. 2018 IEEE International Symposium on Information Theory (ISIT), 1:1465–1469, 2018.

[28] FOSSORIER, M.. **Polar codes: Graph representation and duality**. IEEE Communications Letters, 19:1484–1487, 2015.

[29] TAL, I.; VARDY, A.. **How to construct polar codes**. IEEE Transactions on Information Theory, 59:6562–6582, 2013.

[30] MORI, R.; TANAKA, T.. **Performance of polar codes with the construction using density evolution**. IEEE Communications Letters, 13:519–521, 2009.

[31] HE,G.; BELFIORE, J.; LAND, I.; YANG, G.; LIU, X.; CHEN, Y.; LI, R.; WANG, J.; GE, Y.; ZHANG, R.; TONG, W.. **B-expansion: A theoretical framework for fast and recursive construction of polar codes**. IEEE Global Communication Conference (GLOBECOM 2017), 0:1–6, 2017.

[32] ZHANG, L.; ZHAN, Z.; WANG, X.; YU, Q.; CHEN, Y.. **On the Puncturing Patterns for Punctured Polar Codes**. IEEE International Symposium on Information Theory, 1:121–125, 2014.

[33] SCHURCH, C.. **A partial order for the synthesized channels of a polar code**. IEEE International Symposium on Information Theory (ISIT), 0:220–224, 2016.

[34] VANGALA, H.; VITERBO, E.; HONG, Y.. **A comparative study of polar code constructions for the awgn channel**. 3GPP 3GPP, 2015.

[35] LI, J.; HU, M.; CHENG, Z.. **Research on polar code construction algorithms under gaussian channel**. 2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN), 1:515–518, 2018.

[36] WU, D.; LI, Y.; SUN, Y.. **Construction and block error rate analysis of polar codes over awgn channel based on gaussian approximation**. IEEE Communication Letters, 18:1099–1102, 2014.

[37] WU, D.; LI, Y.; SUN, Y.. **A practical construction method for polar codes in awgn channels**. IEEE 2013 Tencon - Spring, 1:223–226, 2013.

[38] YUAN, P.; PRINZ, T.; BOCHERER, G.. **Polar code construction for list decoding**. 3GPP 3GPP, 2017. Acesso em: Novembro de 2020.

[39] QIN, M.; GUO, J.; BHATIA, A.; I FABREGAS, A. G.; SIEGEL, P.. **Polar code constructions based on llr evolution**. IEEE Communications Letters, 21:1221–1224, 2017.

[40] SUN, S.; ZHANG, Z.. **Designing practical polar codes using simulation-based bit selection**. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 7:594–603, 2017.

[41] A. ELKELESH, M. EBADA, S. CAMMERER AND S. TEN BRINK. **Decoder-tailored polar code design using the genetic algorithm**. IEEE Transactions on Communications, 67:4521–4534, 2019.

[42] KORADA, S.B.; SASOGLU, E.; URBANKE, R.. **Polar codes: Characterization of exponent, bounds, and constructions**. IEEE Transactions on Information Theory, 56:6253–6264, 2010.

[43] PRESMAN, N.; SHAPIRA, O.; LITSYN; S.. **Binary polar code kernels from code decompositions**. IEEE International Symposium on Information Theory Proceedings, 1:179–183, 2011.

[44] ZHANG, L.; ZHANG, Z.; WANG, X.. **Polar code with block-length $n = 3^n$**. Wireless Communications and Signal Processing (WCSP), International Conference, 1:1–6, 2012.

[46] ZHILIANG HUANG AT ALL. **On the successive cancellation decoding of polar codes with arbitrary linear binary kernel**. 3GPP 3GPP, 2017. Acesso em: Novembro de 2021.

[47] GABRY, F.; BIOGLIO, V.; LAND, I.; BELFIORE, J.-C.. **Multi-kernel construction of polar codes**. IEEE International Conference on Communications Workshops (ICC Workshops), 1:761–765, 2017.

[48] BENAMMAR,M.; GABRY, F.; BIOGLIO, V.; LAND, I.. **Multi-kernel polar codes: Proof of polarization and error exponents**. IEEE Information Theory Workshop (ITW), 1:101–105, 2017.

[49] N. HUSSAMI, S. B. KORADA, AND R. URBANKE. **Performance of polar codes for channel and source coding**. IEEE International Symposium on Information Theory, 1:1488–1492, 2009.

[51] MONDELLI, M.; HASSANI, S. H.; URBANKE, R. L.. **From polar to reedmuller codes: A technique to improve the finite-length performance**. IEEE Transactions on Communications, 62:3084–3091, 2014.

[52] BIOGLIO, V.; GABRY, F.; LAND, I.. **Low-complexity puncturing and shortening of polar codes**. IEEE Wireless Communications and Networking Conference Workshops (WCNCW), 1:1–6, 2017.

[53] HONG, S. N.; HUI, D.; MARIC, I.. **Capacity-achieving rate-compatible polar codes**. IEEE Transactions on Information Theory, 1:7620–7632, 2017.

[54] CHEN, K.; NIU, K.; LIN, J.. **A hybrid ARQ scheme based on polar codes**. IEEE Communications Letters, 17:1996–1999, 2013.

[55] NIU, K.; CHEN, K.; LIN, J.. **Beyond turbo codes: Rate-compatible punctured polar codes**. IEEE International Conference on Communications (ICC), 17:3423–3427, 2013.

[56] ESLAMI, A.; PISHRO-NIK, H.. **On finite-length performance of polar codes: Stopping sets, error floor, and concatenated design**. IEEE Transactions on Communications, 61:919–929, 2013.

[57] KIM, J.; KIM, J.H.; KIM, S.H.. **An Efficient Search on Puncturing Patterns for Short Polar Codes**. International Conference on Information and Communication Technology Convergence (ICTC), 1:182–184, 2015.

[58] MILOSLAVSKAYA, V.. **Shortened polar codes**. IEEE Transactions on Information Theory, 61:4852–4865, 2015.

[59] WANG, R.; LIU, R.. **A novel puncturing scheme for polar codes**. IEEE Communications Letters, 18:2081–2084, 2014.

[60] NIU, K.; DAI, J.; CHEN, K.; LIN, J. ZHANG, Q. T.; VASILAKOS, A. V.. **Rate-Compatible Punctured Polar Codes: Optimal Construction Based on Polar Spectra**. CORNELL UNIVERSITY LIBRARY. CORNELL UNIVERSITY, 2017. https://arxiv.org/pdf/1612.01352.pdf, acesso em: dezembro of 2017.

[61] OLIVEIRA, R. M.; LAMARE, R. C. DE. **Rate-compatible polar codes based on polarization-driven shortening**. IEEE Communications Letters, 22:1984–1987, 2018.

[62] JANG, M.; AHN, S.; JEONG, H.; KIM, K.; MYUNG, S.; KIM, S.; YANG, K.. **Rate matching for polar codes based on binary domination**. IEEE Transactions on Communications, 63:6668–6681, 2019.

[63] MOHAMMADI, M. S.; COLLINGS, I. B.; ZHANG, Q.. **Simple hybrid arq schemes based on systematic polar codes for iot applications**. IEEE Communications Letters, 21:975–978, 2017.

[64] LIANG, H.; LIU, A.; ZHANG, Q.; ZHANG, Y.. **Design of systematic polar coded selective-ir hybrid arq transmission for iot**. IEEE/CIC International Conference on Communications in China (ICCC), 1:1–5, 2017.

[65] LIANG, H.; LIU, A.; ZHANG, Q.; ZHANG, Y.. **A systematic multi-packet harq scheme using polar codes for iot**. IEEE 19th International Conference on Communication Technology (ICCT), 1:307–311, 2019.

[66] GAO, J.; FAN, P.; LI, L.. **Optimized polarizing matrix extension based harq scheme for short packet transmission**. IEEE Communications Letters, 24:951–955, 2020.

[67] TRIFONOV, P.; SEMENOV, P.. **Generalized concatenated codes based on polar codes**. 8th International Symposium on Wireless Communication Systems, 1:442–446, 2011.

[68] MAHDAVIFAR, H.; EL-KHAMY, M.; LEE, J.; KANG, I.. **On the construction and decoding of concatenated polar codes**. International Symposium on Information Theory (ISIT), 1:952–956, 2013.

[69] A. CAVATASSI AT ALL. **Asymmetric construction of low-latency and length-flexible polar codes**. IEEE International Conference on Communications (ICC), 1:1–6, 2019.

[70] TRIFONOV, P.. **Chained polar subcodes**. 11th International ITG Conference on Systems, Communications and Coding (SCC), 1:1–6, 2017.

[71] SABER, H.; MARSLAND, I.. **An incremental redundancy Hybrid ARQ scheme via puncturing and extending of polar codes**. IEEE Transactions on Communications, 63:3964–3973, 2015.

[72] ZHAO, M.; ZHANG, G.; XU, C.; ZHANG, H.; LI, R. J.. **An adaptive ir-harq scheme for polar codes by polarizing matrix extension**. IEEE Communications Letters, 22:1306–1309, 2018.

[73] HUANG, Y.; CHANG, H.; LI H.. **Re-polarization processing in extended polar codes**. IEICE Transactions on Communications, E100-B:1765–1777, 2017.

[74] CHUNG,S.-Y.; RICHARDSON, T. J.; URBANKE, R. L.. **Analysis of sum-product decoding of low-density parity-check codes using a gaussian approximation**. IEEE Transactions on Information Theory, 47:657–670, 2001.

[75] HA, JEONGSEOK; KIM, JAEHONG; MCLAUGHLIN; S. W.. **Rate-compatible puncturing of low-density parity-check codes**. IEEE Transactions on Information Theory, 50:2824–2836, 2004.

[76] TRIFONOV, P.. **Randomized chained polar subcodes**. IEEE Wireless Communications and Networking Conference Workshops (WCNCW), 1:25–30, 2018.

[77] FANG, Z.; GAO, J.; LIU, R.. **A simplified gaussian approximation algorithm for polar codes**. 3rd IEEE International Conference on Computer and Communications (ICCC), 1:2429–2433, 2017.

[78] DAI, J.; NIU, K.; SI, Z.; DONG, C.; LIN, J.. **Does gaussian approximation work well for the long-length polar code construction?** IEEE Access, 5:7950–7963, 2017.

[79] OCHIAI, H.; MITRAN P.; POOR, H. V.. **Capacity-approaching polar codes with long codewords and successive cancellation decoding based on improved gaussian approximation**. IEEE Transactions on Communications, 1:31–43, 2021.

[80] OLIVEIRA, R. M.; LAMARE, R. C. DE. **Construction of polar codes based on piecewise gaussian approximation**. 17th International Symposium on Wireless Communication Systems (ISWCS), 1:1–5, 2021.

[81] KERN, D.; VORKOPER, S.; KUHN, V.. **A new code construction for polar codes using min-sum density**. 8th International Symposium on Turbo Codes and Iterative Information Processing (ISTC), 5:228–232, 2014.

[82] OLIVEIRA, R. M.; LAMARE, R. C. DE. **Non-uniform channel polarization and design of rate-compatible polar codes**. 16th International Symposium on Wireless Communication Systems (ISWCS), 1:537–541, 2019.

[84] OLIVEIRA, R. M.; LAMARE, R. C. DE. **Design of rate-compatible polar codes based on non-uniform channel polarization**. IEEE Access, 9:41902–41912, 2021.

[85] RICHARDSON, T. J.; SHOKROLLAHI, A.; URBANKE, R.. **Design of capacity-approaching low-density parity-check codes**. IEEE Transactions on Information Theory, 47:619–637, 2001.

[86] CASSAGNE, A.; HARTMANN, O.; LEONARDON, M.; HE, K.; LEROUX, C.; TAJAN, R.; AUMAGE, O.; BARTHOU, D.; TONNELLIER, T.; PIGNOLY, V.; LE GAL, B.; JEGO, C.. **Aff3ct: A fast forward error correction toolbox!** Elsevier SoftwareX, 10:100–345, 2019.

[87] LIN, J.; XIONG, C.; YAN, Z.. **A high throughput list decoder architecture for polar codes**. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 24:2378–2391, 2016.

[88] KIM, D.; PARK, I.-C.. **A fast successive cancellation list decoder for polar codes with an early stopping criterion**. IEEE Transactions on Signal Processing, 66:4971–4979, 2018.

[89] ARIKAN, E.. **Polar codes: A pipelined implementation**. Proc. 4th ISBC (2010), 1:11–14, 2010.

[90] ELKELESH, A.; EBADA, M.; CAMMERER, S.; BRINK, S. T.. **Belief propagation decoding of polar codes on permuted factor graphs**. IEEE Wireless Communications and Networking Conference (WCNC), 1:1–6, 2018.

[91] DE OLIVEIRA, L. M.; OLIVEIRA, R. M.; DE LAMARE, R. C.. **Q-learning-driven bp decoding for polar codes**. XXXIX Simposio Brasileiro de Telecomunicações e Processamento de Sinais, 1:26–29, 2021.

[92] HAN, S.; HA, J.. **Polar codes for fast converging belief-propagation decoding**. 2021 International Conference on Information and Communication Technology Convergence (ICTC), 1:773–775, 2021.

[93] GOELA, N.; KORADA, S. B.; GASTPAR, M.. **On lp decoding of polar codes**. Proc. IEEE Information Theory Workshop, 1:1–5, 2010.

[94] EBADA, M.; ELKELESH, A.; BRINK, S. T.. **Optimizing polar codes compatible with off-the-shelf ldpc decoders**. 2019 IEEE Information Theory Workshop (ITW), 1:1–5, 2019.

[95] CHEN,Y. -M.; YANG, Y. -J.; HUANG, B. -L.; LEE, H. -C.; LI, C. -P.. **Polar decoding with schedule diversity based on ldpc-like sparse graphs**. 2020 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), 1:1–5, 2020.

[96] ROOSTA, T. G.; WAINWRIGHT M. J.; SASTRY, S. S.. **Convergence analysis of reweighted sum-product algorithms**. IEEE Transactions on Signal Processing, 56:4293–4305, 2008.

[97] WAINWRIGHT, M. J.; JAAKKOLA, T. S.; WILLSKY, A. S.. **A new class of upper bounds on the log partition function**. IEEE Transactions Information Theory, 51:2313–2335, 2005.

[98] WANG, Y.; YU, H.; WANG, W.. **A general decoder architecture for ldpc and polar codes on sparse bipartite graphs**. IEEE 5th International Conference on Computer and Communications (ICCC), 1:1585–1591, 2019.

[99] EBADA, M.; CAMMERER, S.; ELKELESH, A.; TEN BRINK, S.. **Deep learning-based polar code design**. 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton), 67:177–183, 2019.

[100] GIARDINA, C. R.; CHIRLIAN, P. M.. **Proof of weierstrass approximation theorem using band-limited functions**. Proceedings of the IEEE, 61:512–512, 1973.

[101] UCHOA, A. G. D.; HEALY, C.; DE LAMARE, R. C. ; SOUZA, R. D.. **Design of ldpc codes based on progressive edge growth techniques for block fading channels**. IEEE Communications Letters, 15(11):1221–1223, 2011.

[102] HEALY, C. T.; DE LAMARE, R. C.. **Design of ldpc codes based on multipath emd strategies for progressive edge growth**. IEEE Transactions on Communications, 64(8):3208–3219, 2016.

[103] MAHDAVIFAR, H.; EL-KHAMY, M.; LEE, J.; KANG, I.. **Compound polar codes**. 2013 Information Theory and Applications Workshop (ITA), 1:1–6, 2013.

[104] KIM, J.; LEE, J.. **Polar codes for non-identically distributed channels**. EURASIP Journal on Wireless Communications and Networking, 1:1–6, 2016.

[105] WANG, X.; POOR, H.. **Iterative (turbo) soft interference cancellation and decoding for coded cdma**. IEEE Transactions on Communications, 47(7):1046–1061, 1999.

[109] UCHOA, A. G. D.; HEALY, C. T. ; DE LAMARE, R. C.. **Iterative detection and decoding algorithms for mimo systems in block-fading channels using ldpc codes**. IEEE Transactions on Vehicular Technology, 65(4):2735–2741, 2016.

A
Proof of equation (4-21)

According to the Weierstrass Approximation Theorem [100], suppose that $f : [a, b] \to \mathbb{R}$ is a continuous real-valued function defined on the real interval [a,b]. For every $\varepsilon > 0$, there exists a polynomial p such that for all $x \in [a, b]$, we have

$$|f(x) - p(x)| < \varepsilon.$$

Given the continuous and increasing function $f(x)$, being that $x_i < x_{i+1}$ implies $f(x_i) < f(x_{i+1})$, by simplifying (4-1), we have

$$f(x) = \phi^{-1}\left(1 - (1 - \phi(x))^2\right). \tag{A-1}$$

Now, consider two polynomial approximations $p_a(x)$ and $p_b(x)$, such that

$$|f(x_i) - p_a(x_i)| < \varepsilon_a, \tag{A-2}$$

and

$$|f(x_i) - p_b(x_i)| < \varepsilon_b, \tag{A-3}$$

with $x_i \in [a, b]$ and approximation errors ε_a and ε_b. Additionally, for the inequality below

$$|f(x_i) - p_a(x_n)| > \varepsilon_a, \tag{A-4}$$

for $\forall x_n \in [a, b]$, $\forall x_i \in [a, b]$ and $x_i \neq x_n$; and

$$|f(x_i) - p_b(x_n)| > \varepsilon_a, \tag{A-5}$$

for $\forall x_n \in [a, b]$ and $\forall x_i \in [a, b]$, including $x_i = x_n$, we have that $f(x)$ is strictly increasing, so we have the guarantee that

$$\varepsilon_a < \varepsilon_b. \tag{A-6}$$

This result can be generalized as $\forall p_j(x_n)$ and approximation errors ε_j, with $j \in \mathbb{N}$, so

$$|f(x_i) - p_j(x_n)| < \varepsilon_j, \tag{A-7}$$

and

$$|f(x_i) - p_j(x_n)| > \varepsilon_a, \text{ for } \forall j, \tag{A-8}$$

then

$$\varepsilon_a < \varepsilon_j, \text{ for } \forall j, \tag{A-9}$$

and if

$$|f(x_i) - p_j(x_n)| > \varepsilon_{j+1}, \text{ for } \forall j, \tag{A-10}$$

then

$$\varepsilon_a < \varepsilon_j < \varepsilon_{j+1} < \cdots < \varepsilon_{j+n}. \tag{A-11}$$

So, we have to give the sets

$$\mathrm{F} = \{f(x_1), f(x_2), \ldots, f(x_n)\},$$
$$\mathrm{P}_a = \{p_a(x_1), p_a(x_2), \ldots, p_a(x_n)\},$$
$$\mathrm{P}_b = \{p_b(x_1), p_b(x_2), \ldots, p_b(x_n)\},$$

with $x_i \in [a,b]$, $i \in [1,...,n]$ and conditions given in (A-7), (A-8), (A-9) and (A-10), then we have

$$|f(x_i) - p_a(x_i)| < \varepsilon_a < |f(x_i) - p_b(x_i)|,$$
$$|f(x_i) - p_a(x_i)| < \varepsilon_a < \varepsilon_b.$$

Using these results, we have for the approximations RGA, SGA, SPGA:

$$|f_{\mathrm{EGA}}(x_i) - p_{\mathrm{SPGA}}(x_n)| < \varepsilon_1,$$
$$|f_{\mathrm{EGA}}(x_i) - p_{\mathrm{RGA}}(x_n)| < \varepsilon_2,$$
$$|f_{\mathrm{EGA}}(x_i) - p_{\mathrm{SGA}}(x_n)| < \varepsilon_3.$$

and condition in (A-10)

$$|f_{\mathrm{EGA}}(x_i) - p_{\mathrm{SPGA}}(x_n)| > \varepsilon_1,$$
$$|f_{\mathrm{EGA}}(x_i) - p_{\mathrm{RGA}}(x_n)| > \varepsilon_1,$$
$$|f_{\mathrm{EGA}}(x_i) - p_{\mathrm{SGA}}(x_n)| > \varepsilon_1.$$
$$|f_{\mathrm{EGA}}(x_i) - p_{\mathrm{RGA}}(x_n)| > \varepsilon_2,$$
$$|f_{\mathrm{EGA}}(x_i) - p_{\mathrm{SGA}}(x_n)| > \varepsilon_2.$$

then, (A-11),

$$\varepsilon_1 < \varepsilon_2 < \varepsilon_3. \tag{A-12}$$

Equation (A-12) determines the improved approximation of p_{SPGA} over p_{RGA} and p_{SGA}. We know that $\mathcal{A}_c^{ref}$ is obtained from f_{EGA}, the $\mathcal{A}_c^{a}$ is obtained from p_{SPGA}, the $\mathcal{A}_c^{b}$ is obtained from p_{RGA} and the $\mathcal{A}_c^{c}$ is obtained from p_{SGA}. According to (4-19) and the result in (A-12) we can consider that

$$|\mathcal{A}_c^{a} \setminus \mathcal{A}_c^{ref}| < \tau, \tag{A-13}$$

$$|\mathcal{A}_c^{b} \setminus \mathcal{A}_c^{ref}| < \tau, \tag{A-14}$$

and

$$|\mathcal{A}_c^{c} \setminus \mathcal{A}_c^{ref}| < \tau, \tag{A-15}$$

with $\tau > 0$ and $\tau \in \mathbb{N}$ for a given n. Subtracting (A-13) from (A-14) and

(A-15), we have

$$|\mathcal{A}_c^a \setminus \mathcal{A}_c^{ref}| < |\mathcal{A}_c^b \setminus \mathcal{A}_c^{ref}| < |\mathcal{A}_c^c \setminus \mathcal{A}_c^{ref}|, \tag{A-16}$$

and for every set n, we have

$$\sum_{i=1}^{n} |\mathcal{A}_c^a \setminus \mathcal{A}_c^{ref}| < \sum_{i=1}^{n} |\mathcal{A}_c^b \setminus \mathcal{A}_c^{ref}| < \sum_{i=1}^{n} |\mathcal{A}_c^c \setminus \mathcal{A}_c^{ref}|. \tag{A-17}$$

Therefore, the above expression can be rewritten as equation (4-21), i.e.,

$$\sum_{i=1}^{n} X_a^i < \sum_{i=1}^{n} X_b^i < \sum_{i=1}^{n} X_c^i. \tag{A-18}$$

In fact, the inequality in (A-18) will always hold under the conditions established in (A-12). Therefore, we can verify in Table A.1 the RMSE among the polynomial approximation alternatives for the interval $x \in [0,20]$.

Table A.1: RMSE between RGA, SGA, SPGA and EGA.

RMSE	RGA	SGA	SPGA
EGA	0.036	0.0338	0.0215

We can observe that SPGA has the smallest RMSE which implies the best ADE and, therefore, the best performance.