TECHNISCHE UNIVERSITÄT MÜNCHEN

Fakultät für Elektrotechnik und Informationstechnik

Code Analysis and Design for Successive Cancellation List Decoders

Mustafa Cemil Coşkun

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

genehmigten Dissertation.

Vorsitzender:	Prof. Dr.-Ing. Wolfgang Kellerer
Prüfer der Dissertation:	1. Prof. Dr. sc. techn. Gerhard Kramer
	2. Prof. Dr. Henry Pfister

Die Dissertation wurde am 16.11.2021 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 01.04.2022 angenommen.

Acknowledgements

It may seem a trivial task to compile a (possibly incomplete) list of people to acknowledge. However, when it comes to actually doing that... I realized that it requires a severe compression! Still, I feel very lucky to have such a difficulty, which stems from a great number of greater people who made this somewhat long journey much fun.

I was one of the few lucky persons, who had the privilege to be advised by Gerhard Kramer. I have always been impressed by how prompt he grasps an idea and provides immediate feedback, which did not push for quick outputs; instead, motivated me to dedicate enough time for relevant problems. He made me keep in mind the practical aspects of the problems while backing the solutions with a theoretical analysis. In addition, he did not only provide freedom but also encouraged for my decisions, which enabled me to grow as an independent researcher. Last but not least, he encouraged me to teach an entire Bachelor course in Singapore for two consecutive years, which improved my teaching skills.

My mentor Gianluigi Liva is the person I owe the most in my research experience. He is an absolute genius in coding, but I will focus more on his positive attitude. I have met him at a summer school, where I was attracted a lot by his passion for research and coding. This motivated me to follow a similar path; hence, I ended up with this thesis! If you were lucky enough to meet him, you would say all the positive words to describe him. Among all, he is the best (and most dedicated) teacher I have ever had, who can explain any sophisticated concept with a much simpler example. All in all, I concluded that the people you work with are at least as important as the problem itself thanks to him.

I thank Henry Pfister for a great semester at Duke, which motivated me for decisions shaping my future. I was particularly impressed by very elegant directions from him to tackle problems we faced with. Every meeting with him resulted in many interesting ideas, where one can easily write another thesis with them. The only part which disappointed me was the epidemic preventing me to spend more time on other projects that we have planned. Last but not least, I thank him for being a member of the thesis committee.

I thank Wolfgang Kellerer for chairing the examination.

My co-authors contributed so much to this thesis. In particular, I thank Giuseppe Durisi,

Alexandre Graell i Amat, Thomas Jerkovits, Johan Östman, Michael Lentmaier, Joachim Neu, Marvin Xhemrishi and Peihong Yuan for the productive collaborations.

I thank the colleagues at DLR, among many other things, for the joyful "Research & Coffee" meetings, which broadened my perspective in many technical aspects. They made me conclude that the best way to learn is to spend time with people who know things. At TUM, I thank my fantastic office mates (Alessandro, Delcho, Lorenzo, Mahdi and Rawad) and each colleague for the fun work environment. In particular, I thank Peihong again for feeding my brain with enough glucose and many interesting discussions on polar codes.

During my visit at Duke, I enjoyed discussing on various technical topics ranging from topology to quantum error-correction with Elchanan Solomon and Narayanan Rengaswamy as well as nice trips and dinners with them. In addition, I did not only enjoy attending several classes of Robert Calderbank on algebra (which made Sundays more fun) but also discussing about the future of Man. Utd. with him :) Thank you all!

Munich has been so much fun thanks to good-hearted friends, who will be revealed in the following. First, I thank Hilal & Murat for being (literally) a home for me in the town, whose presence was keenly felt in the best and worst times. I thank Alper for the fun we had during his visits; Jessy for cheerful gatherings in BARiffi which lead to great friendships; Mehmet for being the companion during crazy and spontaneous adventures; Okan for being my peer all the way; and Umur for uncountably many occasions, e.g., grills, EURO2020, being the "engineer" whenever needed. I also thank Burak, Ceren, Çağatay, Ebru, Edo, Erdem, Eren, Fatih, Gökhan, Halil, Hilal A., Mahmut, Manuel, Marcel, Mustafa, Nesli, Özge, Pie, Serhan, Tuğçe and Vio, among many many others, especially, for keeping me around despite of my laziness in contributing to the events. Oh no! Almost forgetting... special thanks go to Gözde for the eternal memories.

It is probably a cliché to say "it would not have been possible without my family"; therefore, I paraphrase as follows. My mother has been so strong during any kind of difficulty even when we were away from home. For her, my dinner is literally much more important than her being hospitalized (c.f. proof by examples). **This thesis is dedicated to you!** Apologies to my brother Safa for letting him down at all his smart ideas as "I was more of a theoretician rather than an engineer" :) He is a true source of inspiration for what he has been achieving. My brother Veysi has been financing me in many fun-related occasions and more importantly taking most of the responsibility for the family matters. My younger brother Abdulsamet has always been the most selfless. He is usually like an older brother to me and has always been my closest friend. I am grateful to you all!

Munich, April 2022 M. C. C.

Contents

Zusammenfassung

Diese Dissertation fokussiert sich hauptsächlich auf die sukzessive Listen-Dekodierung (SCL Dekodierung) von gewissen Klassen von binären Blockcodes. Es werden verschiedene Codes für die verlässliche Kommunikation über gedächtnislose Kanäle mit binärer Eingabe (BMSCs) und über gedächtnislose zeitveränderliche Kanäle untersucht.

Zuerst wird die SCL Dekodierung über den binären Auslöschungskanal (BEC) betrachtet. Eine sukzessive Auslöschungs- und Inaktivierungs-Dekodierung (SCI Dekodierung) wird als effiziente Implementation von SCL Dekodierung vorgeschlagen. Falls es keine Komplexitätbeschränkung gibt, dann ist die SCI Dekodierung äquivalent zur Maximum-Likelihood (ML) Dekodierung. Für dieses Dekodierverfahren analysieren wir diverse Klassen von Codes in Bezug auf die durchschnittlich benötigte Anzahl der Inaktivierungen um die Leistungsfähigkeit de ML Dekodierung zu erreichen. Dynamische Reed–Muller Codes werden vorgeschlagen, welche beinahe so leistungsfähig sind wie erweiterte Bose–Chaudhuri–Hocquengham (eBCH) Codes sind und gleichzeitig unter der SCI Dekodierung eine kleinere Dekodierungskomplexität aufweisen.

In dem zweiten Teil dieser Dissertation betrachten wir die SCL Dekodierung für die Kommunikation über allgemeine BMSCs. Wir definieren die wichtigsten informationstechnischen Grössen, welche einen Proxy für das Design von SCL Dekodern bilden. Die vorgeschlagenen Codes übertreffen sowohl Polarcodes, welche als Standard für die 3GPP 5G Technologie gelten, als auch Codes nach aktuellem Forschungsstand, wie zum Beispiel polarisationsangepasste Faltungscodes (PAC) mit praktischen Listengrössen. Dies bildet einen wichtigen Schritt für das Konstruktive Design von modifizierten Polarcodes für SCL Dekodierung.

Als nächstese präsentieren wir Produktcodes aus erweiterten Hammingcodes und/oder einbit-Paritätsprüfcodes als (multi-Kernel) Polarcodes. Dies erlaubt es, die Werkzeuge von Polarcodes für die Analyse und das Dekodieren von solchen Produktcodes zu benützen. Zum Beispiel können die SCL und SCI Dekodierung erweitert werden und bieten damit eine nahezu optimale Dekodierung. Numerische Resultate werden durch eine Analyse der Weight-Enumerator ergänzt. Einige Produktcodes zeigen eine Leistung auf, die kompetitiv

zu der Leistung von 5G Codes ist.

Schliesslich entwickeln wir nicht-kohärente Dekodierungsstrategien basierend auf SCL Dekodierung für gedächtnislose zeitveränderliche Kanäle. Verschiedene Techniken zur Verbesserung der Effizienz der pilotgestützten sowie pilotfreien Übertragung werden entwickelt. Die vorgeschlagenen Techniken resultieren in einer erheblichen Verbesserung gegenüber standardmäßiger pilotgestützter Übertragung.

Abstract

This thesis focuses mainly on successive cancellation list (SCL) decoding of certain classes of binary block codes. Different codes are investigated for reliable communication over binary-input memoryless symmetric channels (BMSCs) as well as memoryless block-fading channels.

First, SCL decoding for communication over the binary erasure channel (BEC) is considered. Successive cancellation inactivation (SCI) decoding is proposed as an efficient implementation of SCL decoding. SCI decoding is equivalent to maximum likelihood (ML) decoding if there is no complexity constraint. Under SCI decoding, various code classes are analyzed in terms of the average number of inactivations required to implement ML decoding. Dynamic Reed–Muller codes are proposed, which perform close to extended Bose–Chaudhuri–Hocquengham (eBCH) codes with much less decoding complexity under SCI decoding.

Second, SCL decoding for communication over general BMSCs is studied. Information-theoretic quantities are defined that provide a proxy to design codes for SCL decoding. The proposed codes outperform the polar codes adopted by the 3GPP 5G cellular standard and state-of-the-art codes, e.g., polarization-adjusted convolutional (PAC) codes, with practical list sizes. This is an important step towards constructive designs of modified polar codes for SCL decoding.

Next, product codes with extended Hamming and/or single parity-check (SPC) component codes are represented as (multi-kernel) polar codes. This enables using the tools of polar codes to analyze and decode such product codes. For example, SCL and SCI decoding are extended and provide near-optimum decoding. Numerical results are complemented by weight enumerator analyses. Some product codes yield a performance that is competitive to that of 5G codes.

Finally, non-coherent decoding strategies are developed that use SCL decoding over memoryless block-fading channels. Several techniques are proposed to improve the efficiency of pilot-assisted and pilot-free transmission. The proposed techniques provide significant gains over classic pilot-assisted transmission methods.

1

Introduction

A point to point communication system has two parties: the *transmitter* and the *receiver*. The transmitter sends a *message* over a noisy medium called a *channel* that has an input and an output. The channel perturbs its input to produce its output, and the perturbation is usually modeled probabilistically. To achieve reliable communication, the message composed of K bits is mapped onto an N-dimensional *codeword*, where $N \geq K$. This mapping is referred to as *encoding* and the term *code* refers to the list of codewords. The receiver tries to guess the codeword upon observing the channel output, which is referred to as *decoding*. The *rate* of the code is measured in information bits per transmitted symbol, i.e., it is K/N bits per channel use for the described scheme.

Shannon established the largest rate at which reliable communication is possible, the *channel capacity* [1]. However, his work did not provide methods to design capacity-achieving channel codes with low encoding/decoding complexity. Elias suggested to use *product codes* [2] and he showed that one can achieve error probability close to zero[1] with a positive rate over the binary symmetric channel (BSC), a special type of *binary-input memoryless symmetric channel (BMSC)*. Although the rate of a product code is generally far below the channel capacity, the result was novel at the time. Roughly one year later, Elias extended the proof of [1] via randomly generated linear codes for transmission over the BSC and the *binary erasure channel (BEC)*, which is possibly the simplest noisy channel [3]. The proof was still non-constructive but it showed the existence of capacity-

[1] Shannon's proof showed that the block error probability can be made small while Elias' construction makes the bit error probability small, which is a weaker notion of reliability. These notions will be clarified in Section 2.5.2.

achieving codes with encoding complexity growing polynomially in N for transmission over BMSCs although decoding complexity was still exponential in N [4] [5, Theorem 6.2.1]. More than 50 years after the channel capacity was introduced, Arıkan [6] provided the first deterministic recipe to achieve the elusive goal of constructing codes, called *polar codes*, provably achieving capacity with simple encoding and decoding ($N \log N$ complexity) for BMSCs.

Arıkan proposed *successive cancellation (SC)* decoding. Unfortunately, at moderate codeword lengths, the performance of polar codes under SC decoding is not competitive with existing techniques such as low-density parity-check (LDPC) codes [7–9] and turbo codes [10]. To improve decoding, Tal and Vardy [11] proposed *successive cancellation list (SCL)* decoding, which achieves near-optimum decoding for a sufficiently large list size. They also observed that, even under SCL decoding, the performance of short and moderate-length polar codes is poor. The reason for this disappointing result can be found in the poor distance properties of the codes. To address this issue, in [11] the authors introduced a clever concatenation of an inner polar code with an outer error-detection code, namely a cyclic redundancy check (CRC) code. This method was the key enabler to improve the performance of polar codes. Thanks to this, polar codes were included in the fifth generation of wireless telecommunication standard (5G) as coding scheme for the control channel [12]. The beauty of the code structure combined with the practical relevance brought much more attention to polar codes, e.g., see [13–20]. Several polar code constructions were proposed, especially for small codeword lengths (up to $N = 128$). However, for intermediate or large codeword lengths the complexity of SCL decoding makes these methods non-competitive.

The adoption of polar codes in wireless communication systems calls for an analysis of the performance achievable by polar code constructions on channels affected by *fading*, i.e., the channels where the received signal energy fluctuates over time and across frequency bands. For example, wireless channels are often modelled by *memoryless block-fading channels* where the channel state is the same for a fixed number of channel uses (a *block*) and changes independently from block to block. These changes can be motivated by the mobility of the transmitter and receiver, as well as of objects in the environment. At the transmitter, knowledge of channel state information (CSI) allows the transmitter adapt its transmission strategy based on the channel state. At the receiver, CSI allows the receiver employ the correct decoding metric. Different assumptions on the availability of the CSI at the transmitter and/or at the receiver lead to different channel capacities (see [21, 22] and references therein for a thorough survey of the topic).

If the receiver cannot reliably estimate the current channel state, then communication is

said to be *non-coherent* [23, Section 10.7]. A common approach to address the lack of CSI is to embed in the transmitted *frame* training symbols (called *pilots*) that are known to the receiver. At the receiver, the CSI is then estimated based on the pilots, and it is used to decode the message. This approach is called *pilot-assisted transmission (PAT)* [24]. The performance of the decoder is highly impacted by the quality of the channel state estimation obtained through the pilots. In a noisy channel, a large number of pilots may be required to obtain an accurate estimation of the channel state [25]. Consider the case where the channel state stays constant for N channel uses (the length of a frame). Suppose that we allocate, e.g., $\sqrt{N}$ pilots to estimate the channel state accurately. The overhead introduced by the pilots is $\sqrt{N}/N = N^{-0.5}$ and it becomes negligible as $N \to \infty$. Hence, there is, asymptotically, no cost of estimating the channel state at the receiver, allowing to achieve the rates of the coherent setup (see, e.g., [21, Page 2632] for more details). On the contrary, when the frame length is small, the overhead required by a PAT scheme to estimate the channel state accurately may become large, calling for a modification of the plain PAT paradigm [22, 25–27]. Also, when channel changes rapidly, then periodic pilots required.

1.1. Contributions of the Thesis

This thesis investigates the potential of SCL decoders in the context of short-packet communications. First, we study SCL decoding of polar codes and their modifications and/or generalizations for transmission over BMSCs. We design codes for specified decoding complexities, that provide significant gains over the polar codes adopted by the 5G cellular standard, and over other the state-of-art short codes. Second, list decoders are used to address the problem of communicating over block-fading channels. We show that list decoders allow to dramatically reduce the channel estimation overhead, approaching the finite-length bounds on the block error probability (BLEP) for fading channels.

Most results in this thesis appeared in the following conference proceedings, journal publications, patents and patent applications:

- **M. C. Coşkun**, J. Neu and H. D. Pfister, "Successive cancellation inactivation decoding for modified Reed-Muller and eBCH Codes," *Proc. Int. Symp. Inf. Theory (ISIT)*, 2020, pp. 437-442. [28]

- **M. C. Coşkun** and H. D. Pfister, "Bounds on the list size of successive cancellation list decoding," *Proc. Int. Conf. Signal Process. and Commun. (SPCOM)*, 2020, pp. 1-5. [29]

- ▷ **M. C. Coşkun** and H. D. Pfister, "An information-theoretic perspective on successive cancellation list decoding and polar code design," *to appear IEEE Trans. Inf. Theory*, 2022. [30]
- ▷ **M. C. Coşkun**, T. Jerkovits and G. Liva, "Successive cancellation list decoding of product codes with Reed-Muller component codes," *IEEE Commun. Lett.*, vol. 23, no. 11, pp. 1972-1976, 2019. [31]
- ▷ **M. C. Coşkun**, G. Liva, A. Graell i Amat, M. Lentmaier and H. D. Pfister, "Successive cancellation decoding of single parity-check product codes: Analysis and Improved decoding," *IEEE Trans. Inf. Theory, minor revision*, 2020. [32]
- ▷ **M. C. Coşkun**, G. Liva, J. Östman, and G. Durisi, "Low-complexity joint channel estimation and list decoding of short codes" *Proc. ITG Int. Conf. Syst., Commun. and Coding*, 2019, pp. 269-273. [33]
- ▷ M. Xhemrishi, **M. C. Coşkun**, G. Liva, J. Östman and G. Durisi, "List decoding of short codes for communication over unknown fading channels," *Proc. Asilomar Conf. Signals, Systems, and Computers*, 2019, pp. 810-814. [34]
- ▷ P. Yuan, **M. C. Coşkun** and G. Kramer, "Polar-coded non-coherent communication," *IEEE Commun. Lett.*, vol. 25, no. 6, pp. 1786-1790, 2021. [35]
- ▷ **M. C. Coşkun** and T. Jerkovits, "Decoding method," *DE102019200941B4*, 2020.
- ▷ **M. C. Coşkun** and G. Liva, "Process for the transmission of data," *DE1020192004-83A1*, 2020.

1.2. Organization of the Thesis

Chapter 2 collects notation and basic definitions required for the remaining chapters.

Chapter 3 reviews binary linear error-correcting codes and their decoding. In particular, polar codes are reviewed in detail.

Chapter 4 deals with the BEC and block-wise maximum likelihood (ML) decoding for linear codes. A new algorithm called successive cancellation inactivation (SCI) decoding is introduced that is equivalent to ML decoding. An analysis of SCI decoding gives insight on the performance/complexity trade-off for various code classes, e.g., polar, Reed–Muller (RM), extended Bose–Chaudhuri–Hocquengham (eBCH) codes, as well as code ensembles

based on polar transforms with dynamic frozen bits. The average number of inactivations is characterized exactly via density evolution analysis. We also propose an accurate and easy-to-compute characterization of the decoding dynamics by deriving a tight approximation for the average number of unknown bits at each decoding stage. We introduce a new code ensemble, called the dynamic Reed–Muller (dRM) ensemble, and we show that codes from the dRM ensemble perform close to finite-length bounds even for short codeword lengths and with much smaller decoding complexity than, e.g., eBCH codes under SCI decoding. The results of this chapter are based on [28, 30].

Chapter 5 deals with general BMSCs and analyzes the average rank of the decoding path leading to the ML decision during SCL decoding with unbounded list size at each decoding stage. Key information-theoretic quantities are defined that characterize the rank by generalizing ideas from Chapter 4. Their calculation requires Monte-Carlo simulation, so we introduce easy-to-compute upper and lower bounds. The lower bound seems to be tight, especially for early decoding stages, and it can be used to guide the code design for SCL decoding with practical list sizes. The resulting codes outperform the polar codes adopted by the 3GPP 5G cellular standard and state-of-the art designs, e.g., polarization-adjusted convolutional (PAC) codes, for the considered list sizes over the binary-input additive white Gaussian noise channel (BAWGNC). The contributions of this chapter are based on [29, 30].

Chapter 6 focuses on a generalization of polar codes that bridges product codes with RM and/or single parity-check (SPC) component codes and (multi-kernel) polar codes. The code structure lets one use polar coding tools to describe, analyze and decode specific classes of product codes. In particular, SCL and SCI decoding naturally extend to the resulting product codes and their variants. This enables a low-complexity near-ML decoding of certain product codes that outperforms belief propagation (BP) decoding, especially when one concatenates the codes with high-rate outer codes. In addition, we propose a method to compute input-output weight enumerators (IOWEs) of short product codes with at least one SPC component code, which is particularly useful in choosing the outer code by means of a serially-concatenated code-ensemble analysis. Concatenated SPC product codes are competitive with polar codes, although they require larger list sizes, and they may outperform 5G LDPC codes for short codeword lengths. The contributions of this chapter are based on [31, 32]. The work presented in [32] extends my Master thesis that introduced SC and SCL decoding of SPC product codes, by introducing the connection to multi-kernel polar codes.

Chapter 7 deals with non-coherent block-fading channels. Joint channel estimation and decoding techniques are developed for polar-coded transmission schemes using SCL

decoding, as well as for general binary linear block codes with most reliable basis (MRB) decoding, in the case of PAT. Simulations show that the proposed algorithms perform close to the optimal decoder and to the best-known finite-length bounds. The results of this chapter are based on [33–35]. In [34], the ideas presented in [33] are applied to polar codes under SCL decoding to show gains over MRB decoding. For both [34] and [35], the first two authors contributed equally to the technical content.

1.3. Contributions Outside the Scope of the Thesis

The publications of the author during the thesis period which are not included in the main results of the thesis are as follows:

- J. Östman, G. Durisi, E. G. Ström, **M. C. Coşkun** and G. Liva, "Short packets over block-memoryless fading channels: Pilot-assisted or noncoherent transmission?," *IEEE Trans. Commun*, vol. 67, no. 2, pp. 1521-1536, 2020. [27]

- **M. C. Coşkun**, G. Durisi, T. Jerkovits, G. Liva, W. Ryan, B. Stein, and F. Steiner, "Efficient error-correcting codes in the short blocklength regime," *Elsevier Phys. Commun.*, vol. 34, pp. 66-79, 2019. [36]

- J. Neu, **M. C. Coşkun** and G. Liva, "Ternary quantized polar code decoders: Analysis and design," *Proc. Asilomar Conf. Signals, Systems, and Computers*, 2019, pp. 1724-1728. [37]

- P. Yuan and **M. C. Coşkun**, "Complexity-adaptive maximum-likelihood decoding," *Proc. Inf. Theory Workshop (ITW)*, 2021. [38]

- P. Yuan and **M. C. Coşkun**, "Successive cancellation ordered search decoding of modified $\boldsymbol{G}_n$-coset codes," *IEEE Trans. Commun., major revision*, 2021. [39]

- **M. C. Coşkun** and G. Liva, "Method for encoding and decoding packets in random access protocols," *DE102020101231A1*, 2021.

In the order of publication date, [27] developed the bounds for PAT schemes for block-fading channels and was the starting point for Chapter 7. The paper also implemented a PAT scheme employing a general binary linear block code using MRB decoding to demonstrate the accuracy of the bounds. In [36], we reviewed state-of-art channel codes and compared their performance to finite-length bounds and to traditional coding schemes. The work of [37] studied ternary-quantized SCL decoders for polar codes and proposed

low-complexity techniques for coarsely quantized decoders. Such decoders are particularly important for energy-efficient transceivers, e.g., for internet-of-things and wireless sensor networks. Finally, [38] introduced successive cancellation ordered search (SCOS) decoding, which is a complexity-adaptive tree search algorithm that implements an ML decoder. For short instances of dRM ensembles and polar codes, its average complexity is close to that of SC decoding for practical block error rates (BLERs). SCOS decoding is further studied in [39] by applying a post-decoding threshold test to reduce the number of undetected errors at the expense of coding gain as proposed in [40]. For more details, see [39].

2

Preliminaries

2.1. Sets, Vectors, and Matrices

The sets of *natural* and *real* numbers are denoted as $\mathbb{N}$ and $\mathbb{R}$, respectively. We write $\mathbb{F}_2$ for the *binary Galois field*. We use $[N]$, $N \in \mathbb{N}$, for the set of natural numbers up to N, i.e., $[N] \triangleq \{1, 2, \ldots, N\}$. We use calligraphic letters such as $\mathcal{X}$ for general sets. For any two sets $\mathcal{X}$ and $\mathcal{Y}$, their *Cartesian* product $\mathcal{X} \times \mathcal{Y}$ is defined as all tuples formed by the sets, i.e.,

$$\mathcal{X} \times \mathcal{Y} \triangleq \{(x, y) \,|\, x \in \mathcal{X}, y \in \mathcal{Y}\}. \tag{2.1}$$

The N-fold Cartesian product of $\mathcal{X}$ is denoted as $\mathcal{X}^N$. For example, the set $\mathbb{F}_2^N$ is the N-dimensional binary vector space.

Throughout the thesis, we use two different notations for row vectors. Mostly, x_i^j is preferred to denote the vector $(x_i, x_{i+1}, \ldots, x_j)$ where $j \geq i$. If $j < i$, then x_i^j is void. We prefer lower-case bold letters such as $\boldsymbol{x}$ if the length is clear from the context or when it is simply more convenient. Subvectors with indices in $\mathcal{X} \subseteq [N]$ are denoted by $x_{\mathcal{X}} = (x_{i_1}, \ldots, x_{i_{|\mathcal{X}|}})$ where $i_1 < \cdots < i_{|\mathcal{X}|}$ enumerates the elements in $\mathcal{X}$ with $|\mathcal{X}|$ being the cardinality of the set $\mathcal{X}$. The notation $x_{i,m}^j$ denotes the subvector with indices $\{k \in [j] : i = k \mod m\}$. For instance, $x_{1,2}^8 = (x_1, x_3, x_5, x_7)$. For the vector x_1^N, the notation $x_{\sim i}$ refers to the vector where the i-th entry is removed, i.e., $x_{\sim i} \triangleq x_{[N]\setminus\{i\}}$. For a given set $\mathcal{X}$, $\mathcal{X}^{(N)}$ represents an intersection set defined as

$$\mathcal{X}^{(N)} \triangleq \mathcal{X} \cap [N]. \tag{2.2}$$

The n-digit multibase representation of a decimal number a is denoted by $(a_1 a_2 \dots a_m)_{b_1^m}$, where the left-most digit is the most significant, and the conversion is done according to

$$a = \sum_{i=1}^{m} a_i \prod_{j=i+1}^{m} b_j \tag{2.3}$$

where b_j is the base of the j-th digit a_j with the left-most digit being the most significant one, and with $0 \leq a_j < b_j$ for all j. For example, the binary representation of a number is obtained by setting $b_j = 2$, $j = 1, \dots, m$. Finally, we write 0 and 1 for all-zero and all-one vectors, respectively, where the length is inferred from the context.

Let $\mathbb{1} : \{\text{false, true}\} \to \{0, 1\}$ be the indicator function with $\mathbb{1}(\text{true}) = 1$ and $\mathbb{1}(\text{false}) = 0$. The *Hamming weight* of x_1^N is defined as the number of its non-zero entries

$$w_{\mathrm{H}}\left(x_1^N\right) \triangleq \sum_{i=1}^{N} \mathbb{1}(x_i \neq 0). \tag{2.4}$$

The *Hamming distance* between two binary vectors $x_1^N, y_1^N \in \mathbb{F}_2^N$ is the number of entries in which they differ, i.e.,

$$d_{\mathrm{H}}\left(x_1^N, y_1^N\right) \triangleq \sum_{i=1}^{N} \mathbb{1}(x_i \neq y_i) \tag{2.5}$$

$$= w_{\mathrm{H}}\left(x_1^N \oplus y_1^N\right) \tag{2.6}$$

where $\oplus$ is the element-wise addition in $\mathbb{F}_2$.

Capital bold letters, e.g., $\boldsymbol{X} = [x_{i,j}]$, are used for matrices and $\boldsymbol{I}_N$ refers to an $N \times N$ identity matrix. For the transpose and inverse of $\boldsymbol{X}$, we use $\boldsymbol{X}^T$ and $\boldsymbol{X}^{-1}$, respectively. The *Kronecker* product of two matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$ is

$$\boldsymbol{X} \otimes \boldsymbol{Y} \triangleq \begin{bmatrix} x_{1,1}\boldsymbol{Y} & x_{1,2}\boldsymbol{Y} & \dots \\ x_{2,1}\boldsymbol{Y} & x_{2,2}\boldsymbol{Y} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}. \tag{2.7}$$

The Kronecker product is non-commutative, i.e., in general we have

$$\boldsymbol{X} \otimes \boldsymbol{Y} \neq \boldsymbol{Y} \otimes \boldsymbol{X}. \tag{2.8}$$

The N-fold Kronecker product of $\boldsymbol{X}$ is denoted as $\boldsymbol{X}^{\otimes N}$ with $\boldsymbol{X}^{\otimes 0} \triangleq 1$. The *mixed-product*

identity states

$$(\boldsymbol{X} \otimes \boldsymbol{Z})(\boldsymbol{Y} \otimes \boldsymbol{T}) = (\boldsymbol{XY}) \otimes (\boldsymbol{ZT}). \tag{2.9}$$

We define an $ab \times ab$ *perfect shuffle* matrix [41], denoted as $\boldsymbol{\Pi}_{a,b}$, by the following operation

$$(x_1, x_2, \ldots, x_{ab})\boldsymbol{\Pi}_{a,b} = (x_{1,b}^{ab}, x_{2,b}^{ab}, \ldots, x_{b-1,b}^{ab}, x_{b,b}^{ab}). \tag{2.10}$$

In particular, for the $2N \times 2N$ reverse shuffle permutation [6] matrix $\boldsymbol{\Pi}_{N,2}$, we have

$$(s_1, s_2, \ldots, s_{2N})\boldsymbol{\Pi}_{N,2} = (s_{1,2}^{2N}, s_{2,2}^{2N}) \tag{2.11}$$

$$= (s_1, s_3, \ldots, s_{2N-1}, s_2, s_4, \ldots, s_{2N}). \tag{2.12}$$

2.2. Probability

We denote *random variables (RVs)* with capital letters, e.g., X, and their *realizations* with the lower-case counterparts, e.g., x. For a discrete RV X with alphabet $\mathcal{X}$, the *probability mass function (PMF)* P_X assigns a probability that the variable takes on some value, i.e.,

$$P_X(x) \triangleq \Pr\{X = x\}, \quad x \in \mathcal{X}. \tag{2.13}$$

For a continuous and real-valued RV X, the cumulative distribution function (CDF) F_X assigns the probability that X takes on a value less than or equal to some specified value, i.e.,

$$F_X(x) \triangleq \Pr\{X \leq x\}, \quad x \in \mathbb{R}. \tag{2.14}$$

Then the *probability density function (PDF)* $p_X(x)$ is defined as

$$p_X(x) \triangleq \frac{\mathrm{d}F_X(x)}{\mathrm{d}x} \tag{2.15}$$

if the derivative exists. For instance, if X is a zero-mean *Gaussian* RV with variance σ^2, i.e., $X \sim \mathcal{N}(0, \sigma^2)$, then its PDF is given as

$$p_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2}{2\sigma^2}\right). \tag{2.16}$$

A PMF P_X fulfills

$$P_X(x) \geq 0, \quad \forall x \in \mathcal{X} \quad \text{and} \quad \sum_{x \in \mathcal{X}} P_X(x) = 1. \tag{2.17}$$

Similarly, a PDF p_X fulfills

$$p_X(x) \geq 0, \quad \forall x \in \mathcal{X} \quad \text{and} \quad \int_{x\in\mathcal{X}} p_X(x)\mathrm{d}x = 1. \tag{2.18}$$

The *support* of X is defined as

$$\mathrm{supp}(P_X) \triangleq \{x \in \mathcal{X} : P_X(x) > 0\} \quad \text{and} \quad \mathrm{supp}(p_X) \triangleq \{x \in \mathcal{X} : p_X(x) > 0\}. \tag{2.19}$$

The remaining notation will be given only for discrete RVs. Extensions to continuous RVs follow by replacing PMFs by PDFs and summation by integration as for (2.18).

The *joint* PMF P_{XY} for X and Y assigns a probability that the variables take on a value tuple, i.e.,

$$P_{XY}(x, y) \triangleq \Pr\{X = x, Y = y\} \tag{2.20}$$

and can be extended to an arbitrary number of RVs. The *law of total probability* is used to obtain the marginal distributions from the joint one as

$$P_X(x) = \sum_y P_{XY}(x, y). \tag{2.21}$$

The *conditional* PMF $P_{X|Y}$ assigns the probability of observing an event $x \in \mathcal{X}$ conditioned on another event $y \in \mathrm{supp}(P_Y)$, i.e., we have

$$P_{X|Y}(x|y) \triangleq \frac{P_{XY}(x, y)}{P_Y(y)} \quad \text{with} \quad P_Y(y) > 0. \tag{2.22}$$

Bayes' rule translates $P_{Y|X}$ into $P_{X|Y}$ as

$$P_{X|Y}(x|y) = \frac{P_{Y|X}(y|x)P_X(x)}{P_Y(y)} \quad \text{with} \quad P_Y(y) > 0. \tag{2.23}$$

Remark 2.1. The RVs X and Y are *stochastically independent* if and only if

$$P_{X,Y}(x, y) = P_X(x)P_Y(y) \quad \text{or} \quad P_{X|Y}(x|y) = P_X(x) \tag{2.24}$$

for all x and y for the first expression, and for all x and y with $P_Y(y) > 0$ for the second expression.

The *expected value* of a real-valued function $f : \mathcal{X} \to \mathbb{R}$ with respect to a discrete RV

X is its weighted average with the corresponding probabilities over $\mathrm{supp}(P_X)$, i.e.,

$$\mathbb{E}_X[f(X)] \triangleq \sum_{x \in \mathrm{supp}(P_X)} f(x) P_X(x). \tag{2.25}$$

Usually, the range of the summation will be discarded when it is clear from the context. The *mean* $\bar{X}$ of X is computed by setting $f(X) = X$ as

$$\bar{X} \triangleq \mathbb{E}_X[X]. \tag{2.26}$$

Similarly, the *variance* $\mathbb{V}[X]$ of a real-valued X is defined by setting $f(X) = (X - \mathbb{E}_X[X])^2$ as

$$\mathbb{V}[X] \triangleq \mathbb{E}_X[(X - \mathbb{E}_X[X])^2] = \mathbb{E}_X[X^2] - \mathbb{E}_X[X]^2. \tag{2.27}$$

In the case of complex-valued X, $\mathbb{V}[X]$ is obtained as

$$\mathbb{V}[X] = \mathbb{V}[\Re\{X\}] + \mathbb{V}[\Im\{X\}] \tag{2.28}$$

where $\Re\{X\}$ and $\Im\{X\}$ denote the real and imaginary parts of X.

Remark 2.2. From now on, the subscripts of PMFs, PDFs and expectations are omitted when they are clear from the context. For example, we write $P(x, y)$ instead of $P_{XY}(x, y)$.

Lemma 2.1 (Linearity of expectation)**.** Let X_1^N be a sequence of RVs with finite means. For any collection of functions $f_i : \mathcal{X} \to \mathbb{R}$, we have

$$\mathbb{E}\left[\sum_{i=1}^{N} f_i\left(X_i\right)\right] = \sum_{i=1}^{N} \mathbb{E}\left[f_i\left(X_i\right)\right]. \tag{2.29}$$

Corollary 2.2. For any constant c and any function $f : \mathcal{X} \to \mathbb{R}$, we have

$$\mathbb{E}[cf(X)] = c\mathbb{E}[f(X)]. \tag{2.30}$$

Next, the expectation of f conditioned on the event $y \in \mathrm{supp}(P(\cdot|y))$, i.e., *conditional* expectation, is defined as

$$\mathbb{E}[f(X)|Y = y] \triangleq \sum_{x} f(x) P(x|y). \tag{2.31}$$

Remark 2.3. From (2.31) the conditional expectation $\mathbb{E}[f(X)|Y = y]$ is a function of y. Hence, the expression $\mathbb{E}[f(X)|Y]$ is a function of the RV Y.

Combining Remark 2.3 with the definition (2.31), one obtains the following property.

Lemma 2.3 (Tower property)**.**

$$\mathbb{E}[\mathbb{E}[f(X)|Y]] = \mathbb{E}[f(X)]. \tag{2.32}$$

Proof.

$$\begin{aligned}
\mathbb{E}[\mathbb{E}[f(X)|Y]] &= \sum_y P(y)\mathbb{E}[f(X)|Y=y] && (2.33)\\
&= \sum_y P(y) \sum_x f(x)P(x|y) && (2.34)\\
&= \sum_x \sum_y f(x)P(x|y)P(y) && (2.35)\\
&= \mathbb{E}[f(X)] && (2.36)
\end{aligned}$$

which concludes the proof. ■

Definition 2.1 (Martingale)**.** A sequence of RVs $X_0, X_1, \ldots$ is called a *martingale* with respect to the sequence $Y_0, Y_1, \ldots$ if, $\forall N > 0$, we have

1. X_N is a function of Y_0^N
2. $\mathbb{E}[|X_N|] < \infty$
3. $\mathbb{E}[X_{N+1}|Y_0^N] = X_N$.

Example 2.1 (Doob's martingale)**.** A simple but important instance is formed by defining a sequence of RVs via conditional expectations as follows: let Z be a positive-valued RV with $\bar{Z} < \infty$ and let $Y_0, Y_1, \ldots$ be a sequence of RVs. Then the sequence of RVs $X_0, X_1, \ldots$ with $X_i \triangleq \mathbb{E}[Z|Y_0^i]$, $i = 0, 1, \ldots, N$, is a martingale. Condition 1 of a martingale follows from the definition of X_i and condition 2 from $\bar{Z} < \infty$. The last condition follows by writing

$$\begin{aligned}
\mathbb{E}[X_{N+1}|Y_0^N] &= \mathbb{E}[\mathbb{E}[Z|Y_0^{N+1}]|Y_0^N] && (2.37)\\
&= \mathbb{E}[Z|Y_0^N] && (2.38)\\
&= X_N && (2.39)
\end{aligned}$$

where (2.37) and (2.39) follow from the definition of X_i. The equality (2.38) uses the tower property (2.32).

Concentration Inequalities

We review basic techniques to bound the probability that a RV takes values far from its mean. The proofs can be found in standard textbooks [42].

Lemma 2.4 (Markov's inequality)**.** Let X be a nonnegative RV. Then we have

$$\Pr\{X \geq a\} \leq \frac{\bar{X}}{a}, \qquad \forall a > 0. \tag{2.40}$$

Markov's inequality is a good bound if, for a non-negative RV, only the mean is known. *Chebyshev's inequality* is a special case of Markov's inequality, and it is useful if both the mean and the variance of a RV are known.

Lemma 2.5 (Chebyshev's inequality)**.**

$$\Pr\{|X - \bar{X}| \geq a\} \leq \frac{\mathbb{V}[X]}{a^2}, \qquad \forall a > 0. \tag{2.41}$$

Observing the Chebyshev's inequality, the upper bound decays polynomial in parameter a. We are often interested in an exponential decay rather than a polynomial one. One family of bounds providing exponential decay is the one of Chernoff bounds [42, Chapter 4]. A related inequality is the *Azuma-Hoeffding inequality*, which applies to dependent RVs.

Lemma 2.6 (Azuma-Hoeffding inequality)**.** Let $X_0, X_1, \ldots, X_N$ be a martingale that satisfies

$$|X_i - X_{i-1}| \leq c_i \quad \text{for some} \quad c_i > 0. \tag{2.42}$$

We have

$$\Pr\{|X_n - X_0| \geq a\} \leq 2\exp\left(-\frac{a^2}{2\sum_{i=1}^{n} c_i^2}\right), \quad \forall n \in [N] \quad \text{and} \quad \forall a > 0. \tag{2.43}$$

The Azuma-Hoeffding inequality was used to analyze LDPC codes and decoding by forming a Doob's martingale [43–46]. For a list of applications, the reader is referred to [47]. Martingales provide useful convergence properties under quite general conditions, for example, without requiring (2.42), as stated next.

Lemma 2.7 (Martingale convergence theorem)**.** Let a sequence of RVs $X_0, X_1, \ldots$ form a martingale. Then, the sequence converges almost surely (a.s.) to a RV X_∞, i.e., we have

$$\Pr\left\{\lim_{t\to\infty} X_t - X_\infty = 0\right\} = 1. \tag{2.44}$$

In addition, the sequence converges in the mean, i.e., we also have

$$\lim_{t\to\infty} \mathbb{E}[|X_t - X_\infty|] = 0 \tag{2.45}$$

and

$$\mathbb{E}[X_\infty] = \mathbb{E}[X_i], \quad \forall i \in \{0, 1, \ldots, \}. \tag{2.46}$$

A fundamental concept called *channel polarization* [6] was proved using Lemma 2.7 after forming a suitable martingale. Later on, the proof was generalized to arbitrary kernels in [48] and to the mixture of kernels in [49] by following the same line of arguments, which proved a conjecture of Arıkan posed in [6, Section XI.B].

2.3. Big-$\mathcal{O}$ Notation and Complexity

The Big-$\mathcal{O}$ notation simplifies complexity analysis by focusing on the dominating behavior of a sequence. Let $T(N)$ denote the *running time* of a sequential algorithm where N is the size of the input. In particular, we are interested in upper-bounds on the complexity. To this end, given the function $f(\cdot)$ define the function set

$$\mathcal{O}(f(N)) \triangleq \{g(\cdot) : \exists\, c, N_0 > 0 \text{ such that } 0 \leq g(X) \leq cf(N), \forall N > N_0\}. \tag{2.47}$$

For a given $T(N)$, we write $T(N) \in \mathcal{O}(f(N))$ to indicate that $T(N)$ can be upper-bounded as in the definition of the set $\mathcal{O}(f(N))$. For an algorithm with a running time of $T(N) \in \mathcal{O}(f(N))$, we say it has a complexity of $\mathcal{O}(f(N))$. In addition, we use the same notation for the *space complexity* of an algorithm, which is defined as the memory required as a function of the input size.

2.4. Information Measures

The amount of information revealed by an event can be measured by taking the logarithm of its probability, i.e.,

$$i(x) = -\log_2 P(x) \tag{2.48}$$

where the logarithm base is chosen for convenience.

The *entropy* of a discrete RV X is the mean of $i(X)$, i.e., we define

$$H(X) \triangleq \mathbb{E}[i(X)] = \mathbb{E}[-\log_2 P(X)]. \tag{2.49}$$

For example, consider a binary RV X with distribution $P(0) = 1 - P(1) = p \in [0, 1/2]$. We define the *binary entropy function* $H_2 : [0, 1/2] \to [0, 1]$ as

$$H_2(p) \triangleq -p \log_2 p - (1-p) \log_2 (1-p). \tag{2.50}$$

We used the domain $[0, 1/2]$ so that the inverse $H_2^{-1} : [0, 1] \to [0, 1/2]$ is well-defined. Note that $H_2(p)$ is symmetric around $p^* = \frac{1}{2}$ when the definition (2.50) is extended to $p \in [0, 1]$.

The *conditional* entropy of a discrete RV X given the event y is defined as

$$H(X|Y = y) \triangleq \mathbb{E}[-\log_2 P(X|Y)|Y = y]. \tag{2.51}$$

Following Remark 2.3, it is clear that $\mathbb{E}[-\log_2 P(X|Y)|Y]$ is a RV. The conditional entropy of X given Y is defined as the expectation of $\mathbb{E}[-\log_2 P(X|Y)|Y]$, i.e.,

$$H(X|Y) \triangleq \mathbb{E}[\mathbb{E}[-\log_2 P(X|Y)|Y]] = \mathbb{E}[-\log_2 P(X|Y)]. \tag{2.52}$$

The *joint* entropy of two discrete RVs X and Y is defined as

$$H(XY) = \mathbb{E}[-\log_2 P(X, Y)]. \tag{2.53}$$

The following lemma stems from applying Bayes' rule and the entropy definitions.

Lemma 2.8 (Chain rule for entropy)**.**

$$H(X_1^N) = \sum_{i=1}^{N} H(X_i|X_1^{i-1}) \tag{2.54}$$

Another interesting quantity is *"how much information does X reveal about Y or vice versa"*. This is quantified by *mutual information* as

$$I(X;Y) \triangleq H(X) - H(X|Y) = H(Y) - H(Y|X). \tag{2.55}$$

The *conditional* mutual information between X and Y given Z is defined similar to (2.52)

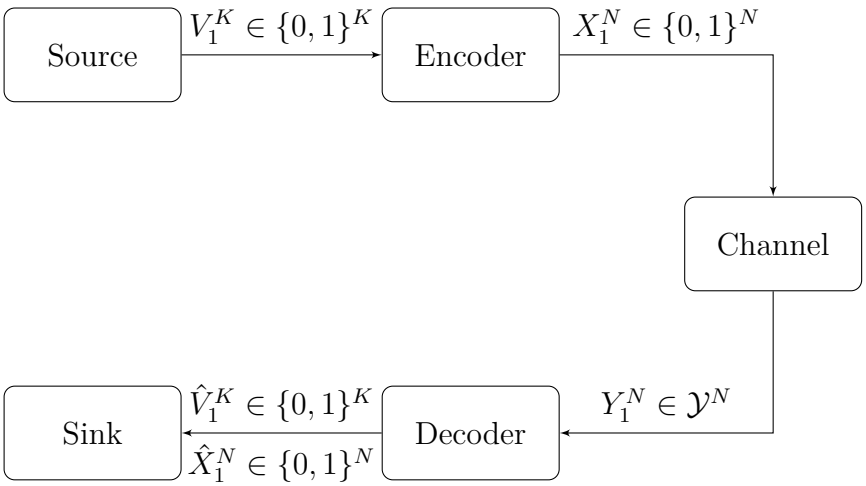

Figure 2.1.: Block diagram of a data transmission system.

as

$$I(X;Y|Z) \triangleq \sum_z P(z) I(X;Y|Z=z) \tag{2.56}$$

where

$$I(X;Y|Z=z) = H(X|Z=z) - H(X|Y,Z=z). \tag{2.57}$$

The chain rule for mutual information follows from (2.55) and (2.56).

Lemma 2.9 (Chain rule for mutual information)**.**

$$I(X_1^N;Y) = \sum_{i=1}^{N} I(X_i;Y|X_1^{i-1}). \tag{2.58}$$

2.5. Transmission System

Consider the *point-to-point* communication problem depicted in Figure 2.1. The source generates the K-bit source word V_1^K whose elements can be modelled as independent and identically distributed (i.i.d.) RVs with $P_{V_i}(0) = P_{V_i}(1) = 1/2$. The encoder is a bijective function $E : \mathbb{F}_2^K \to \mathbb{F}_2^N$ which maps V_1^K onto a length-N binary *codeword* X_1^N where $K \leq N$. The collection of all codewords associated to the 2^K different source blocks represents the binary *block code* $\mathcal{C}$. The channel W maps X_1^N to $Y_1^N \in \mathcal{Y}^N$ according to a conditional probability distribution $P(y_1^N|x_1^N)$. The decoder is the function $D : \mathcal{Y}^N \to \mathbb{F}_2^K$ which puts out an estimate $\hat{V}_1^K$, or alternatively $\hat{X}_1^N$ if the encoder E is bijective.

2.5.1. Error Probabilities and Rate

Several figures of merit are important for a transmission system. First of all, the *BLEP*

$$P_B \triangleq \Pr\left\{\hat{V}_1^K \neq V_1^K\right\} \tag{2.59}$$

should be as small as possible. Sometimes a receiver rejects the decision and outputs an *erasure*. Such events are called *detected* errors. For some applications, e.g., telecommand systems, *undetected* errors are especially harmful. A decoder is called *incomplete* if it has a mechanism to reject an estimate and is called *complete* otherwise. Although one can always convert a complete decoder into an incomplete one, e.g., by applying an outer error-detection mechanism or a threshold test [40], this comes at the cost of a higher overall BLEP. Although we will focus (almost) always on the overall BLEP, the definition of error probability naturally extends for the source and codeword bits as

$$P_{sb} \triangleq \frac{1}{K}\sum_{i=1}^{K}\Pr\left\{\hat{V}_i \neq V_i\right\} \tag{2.60}$$

$$P_{cb} \triangleq \frac{1}{N}\sum_{i=1}^{N}\Pr\left\{\hat{X}_i \neq X_i\right\}. \tag{2.61}$$

Since any bit error implies a block error, we have

$$P_{sb} \leq P_B \leq KP_{sb} \tag{2.62}$$

$$P_{cb} \leq P_B \leq NP_{cb}. \tag{2.63}$$

One way to assure a lower error probability is to map the source words onto longer codewords, i.e., by adding redundancy, if the amount of redundancy is already larger than a threshold dictated by the channel (this is related to the notion of the capacity, which will be clear in Section 2.5.3). The amount of redundancy introduced by the encoder is quantified by the *rate*

$$R \triangleq \frac{K}{N}. \tag{2.64}$$

Transmission is more *efficient* if it provides a lower error probability or a higher rate.

2.5.2. Decision Rules

The *bit-wise* maximum a-posteriori (MAP) decoding rule maximizes the *a-posteriori probability (APP)* $P(x_i|y_1^N)$ for each codeword bit x_i

$$\hat{x}_i = \underset{x_i \in \mathbb{F}_2}{\operatorname{argmax}} P(x_i|y_1^N). \tag{2.65}$$

Bit-wise MAP decoding minimizes P_{cb}. However, the decoder is not guaranteed to provide a sequence in $\mathcal{C}$, i.e., the output might not be a valid codeword (the decoder is incomplete). On the other hand, the bit-wise MAP decoding that maximizes the APP $P(v_i|y_1^N)$ for each source bit v_i

$$\hat{v}_i = \underset{v_i \in \mathbb{F}_2}{\operatorname{argmax}} P(v_i|y_1^N) \tag{2.66}$$

is complete. Similarly, the bit-wise ML decoder evaluates

$$\hat{v}_i = \underset{v_i \in \mathbb{F}_2}{\operatorname{argmax}} P(y_1^N|v_i). \tag{2.67}$$

If the source bits v_i are i.i.d. and uniform, i.e., $P(v_i) = 1/2$, $i \in [K]$, then (2.66) becomes

$$\hat{v}_i = \underset{v_i \in \mathbb{F}_2}{\operatorname{argmax}} P(v_i|y_1^N) \frac{P(y_1^N)}{P(v_i)} \tag{2.68}$$

$$= \underset{v_i \in \mathbb{F}_2}{\operatorname{argmax}} P(y_1^N|v_i) \tag{2.69}$$

which means that the bit-wise MAP rule coincides with the bit-wise ML rule.

The *block-wise* MAP decoding rule maximizes the APP $P(x_1^N|y_1^N)$ for each codeword $x_1^N \in \mathcal{C}$

$$\hat{x}_1^N = \underset{x_1^N \in \mathcal{C}}{\operatorname{argmax}} P(x_1^N|y_1^N) \tag{2.70}$$

which minimizes the BLEP. Similarly, the *block-wise* ML decoding rule maximizes the likelihood $P(y_1^N|x_1^N)$ for each codeword $x_1^N \in \mathcal{C}$

$$\hat{x}_1^N = \underset{x_1^N \in \mathcal{C}}{\operatorname{argmax}} P(y_1^N|x_1^N). \tag{2.71}$$

If the input to the channel is uniformly distributed among all codewords, i.e., $P(x_1^N) = 1/|\mathcal{C}|$,

then (2.70) becomes

$$\hat{x}_1^N = \underset{x_1^N \in \mathcal{C}}{\operatorname{argmax}} P(x_1^N|y_1^N)\frac{P(y_1^N)}{P(x_1^N)} \tag{2.72}$$

$$= \underset{x_1^N \in \mathcal{C}}{\operatorname{argmax}} P(y_1^N|x_1^N) \tag{2.73}$$

which means that MAP decoding is equivalent to ML decoding. Throughout the thesis, we consider a uniform distribution over the codewords so the MAP rule is the same as ML rule. Hence, we use MAP and ML interchangeably.

Remark 2.4. Optimum decoding via (2.70) requires computing conditional probabilities for all the codewords, and this requires in general exponential complexity in N.

2.5.3. Channel Coding Theorem

We wish to find encoder-decoder pairs that achieve the maximum rate R^* for a given BLEP P_B and blocklength N. Although characterizing R^* is difficult even for small values of N, Shannon realized that the problem simplifies significantly for vanishing BLEP ($P_B \to 0$) if the blocklength tends to infinity, i.e., $N \to \infty$ [1]. He reported the following lower bound on R^*.

Theorem 2.10 (Achievability for the channel coding theorem)**.** There exists a sequence of encoder-decoder pairs for which $P_B \to 0$ for $N \to \infty$ if $R < \max_{P_X} I(X;Y)$.

The proof relies on *random coding* arguments, i.e., each element of each codeword of a code is generated i.i.d. according to a distribution P_X. At the receiver, a suboptimal decoder called a *jointly typical decoder* is used. The BLEP is analyzed for an ensemble of codes consisting of all possible codes generated by the random coding experiment described above. Typicality arguments such as the *asymptotic equipartition property* show that the average BLEP decays to zero for large blocklengths and one concludes by observing that there exists at least one code performing at least as good as the average. A formal proof can be found, e.g., in [50, Chapter 7]. A simpler proof based on the law of large numbers and Markov's inequality can be found in [51]. We remark that Gallager used the notion of error exponents in [52] and showed that the decay in the BLEP can be made exponential as long as the rate is smaller than the mutual information [53]. The *converse* of Theorem 2.10 also holds, i.e., the BLEP is bounded away from zero as the blocklength increases if the rate is larger than the mutual information, where the proof follows by using Fano's inequality and can be found in [50, Chapter 7]. Hence, for a given channel W, the maximum rate at

which reliable communication is possible is called its *capacity*, denoted by $C(W)$, and it is given by

$$C(W) \triangleq \lim_{P_B \to 0} \lim_{N \to \infty} R^*(W) = \max_{P_X} I(X;Y). \tag{2.74}$$

2.6. Binary-Input Memoryless Symmetric Channels

We review the channel models considered in the rest of the thesis. For all channels, we assume a time discrete model where both the transmitter and receiver are synchronized. The channels have a binary input alphabet, except for the channel in Chapter 7 .

A channel (without feedback) is *memoryless* if

$$P(y_1^N | x_1^N) = \prod_{i=1}^{N} P(y_i | x_i). \tag{2.75}$$

Following the notation of [6], we take the convention of indicating the channel conditional probability distribution by the letter W. We write (2.75) as

$$W^N(y_1^N | x_1^N) = \prod_{i=1}^{N} W(y_i | x_i) \tag{2.76}$$

where $W(y|x) \triangleq P(y|x)$. The channel is (output) *symmetric* if there exists a permutation π of the output alphabet $\mathcal{Y}$ such that

$$\pi(\pi(y)) = y \quad \text{and} \quad W(\pi(y)|0) = W(y|1) \tag{2.77}$$

where the input alphabet is assumed to be $\mathbb{F}_2$.[1] A channel is called BMSC if it is both memoryless and symmetric. We define the *symmetric information rate (SIR)* of a binary-input channel W as the mutual information achieved by uniform inputs, i.e.,

$$I(W) \triangleq \sum_{y \in \mathcal{Y}} \sum_{x \in \mathcal{X}} \frac{1}{2} W(y|x) \log_2 \left(\frac{W(y|0)}{\frac{1}{2}W(y|0) + \frac{1}{2}W(y|1)} \right). \tag{2.78}$$

For BMSCs, the capacity is achieved by uniform inputs and $C(W) = I(W)$. In addition, we define the conditional entropy of uniform channel input X given the output Y as

$$H(W) \triangleq 1 - I(W). \tag{2.79}$$

[1] The choice of the input alphabet can be relaxed to any binary set. For example, we will make use of $\mathcal{X} = \{+1, -1\}$ for the BAWGNCs.

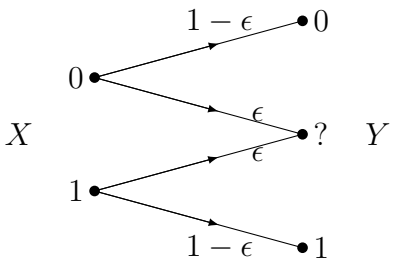

Figure 2.2.: The BEC with erasure probability ϵ, denoted as BEC(ϵ).

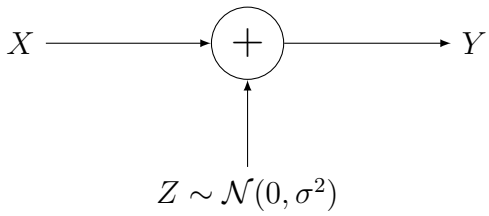

Figure 2.3.: The BAWGNC with noise variance σ^2, denoted as BAWGNC(σ).

2.6.1. Binary Erasure Channel

The BEC is illustrated in Figure 2.2. It is a BMSC with input alphabet $\mathcal{X} = \mathbb{F}_2$, discrete output alphabet $\mathcal{Y} = \{0, 1, ?\}$ and transition probabilities

$$W(0|0) = W(1|1) = 1 - \epsilon, \qquad W(?|0) = W(?|1) = \epsilon. \tag{2.80}$$

Here, ? denotes an *erasure*, which means that no information about the input is observed by the receiver. The quantity ϵ is called the channel erasure probability. We write BEC(ϵ) to denote the BEC with an erasure probability ϵ. As the channel is symmetric, the capacity is computed via (2.78) as

$$C(\mathrm{BEC}(\epsilon)) = 1 - \epsilon. \tag{2.81}$$

2.6.2. Binary-Input Additive White Gaussian Channel

Figure 2.3 illustrates a BAWGNC where $Z \sim \mathcal{N}(0, \sigma^2)$ is independent of the input RV X. The channel is denoted as BAWGNC(σ). The input alphabet is $\mathcal{X} = \{-1, +1\}$ while Y is real-valued. The input-output relation is $Y = X + Z$ with transition densities

$$W(y|x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-x)^2}{2\sigma^2}\right). \tag{2.82}$$

The channel can be characterized by a quantity called the *signal-to-noise ratio (SNR)* E_s/N_0 where E_s is the average energy per transmitted symbol and $N_0 = 2\sigma^2$ is single-sided noise power spectral density. For our alphabet, $E_s = \mathbb{E}[X^2] = 1$, hence, $E_s/N_0 = 1/2\sigma^2$. Alternatively, one can use the ratio E_b/N_0, where E_b is the average energy per transmitted information bit, i.e., $E_b = (N \cdot E_s)/K = E_s/R$. Therefore, the relation between SNRs is

$$\frac{E_b}{N_0} = \frac{E_s}{2R\sigma^2}. \tag{2.83}$$

An insightful discussion on when to prefer one characterization over the other can be found in [54, Section 4.1.11].

Since the BAWGNC is symmetric, uniform inputs give the capacity [55, Equation 2.2]

$$C(\mathrm{BAWGNC}(\sigma)) = 1 - \int_{-\infty}^{+\infty} W(y|+1) \log_2 \left(1 + \exp\left(-\frac{2}{\sigma^2} y\right)\right) \mathrm{d}y. \tag{2.84}$$

3

Product Codes and Polar Codes

In this chapter, we show how imposing a structure on a channel code can reduce the encoding and decoding complexity. We review a decoding algorithm called MRB decoding that mimics optimal decoding by exploiting only the linearity of a code. We briefly describe a class of linear codes, namely product codes, whose structure enables efficient decoding in an iterative manner. Finally, we review polar coding which provides asymptotically optimal performance with quasi-linear encoding and decoding complexity.

3.1. Binary Linear Codes

We here restrict attention to the binary field $\mathbb{F}_2$ and to binary codes.

Definition 3.1 (Linear codes)**.** An (N, K) binary linear block code is defined as a K-dimensional linear subspace of the N-dimensional space $\mathbb{F}_2^N$.

Let $\mathcal{C}$ be an (N, K) binary linear code with rate $R = K/N$. Since it is a linear subspace, it can be described by a *generator matrix* $\boldsymbol{G} \in \mathbb{F}_2^{K \times N}$. The rows of $\boldsymbol{G}$ span the K-dimensional subspace, i.e., the code space. Encoding can be performed as $c_1^N = v_1^K \boldsymbol{G}$. Thus, encoding requires approximately $N^2 R$ operations, and the code description can be conveniently summarized in $N^2 R$ bits by storing the generator matrix.

Equivalently, a binary linear block code can be defined using its $(N - K) \times N$ *parity check* matrix $\boldsymbol{H}$ as

$$\mathcal{C} \triangleq \{c_1^N \in \mathbb{F}_2 : c_1^N \boldsymbol{H}^T = 0\}. \tag{3.1}$$

This means that the rows of $\boldsymbol{H}$ span the subspace orthogonal to $\mathcal{C}$, which is called *dual code* of $\mathcal{C}$ and denoted by $\mathcal{C}^{\perp}$. Note that the intersection $\mathcal{C} \cap \mathcal{C}^{\perp}$ is not necessarily empty.

If the generator matrix is in the form where a subset of columns forms $\boldsymbol{I}_K$, then it is called a *systematic* generator matrix. Assume, without loss of generality, that the first K columns of $\boldsymbol{G}$ forms $\boldsymbol{I}_K$

$$\boldsymbol{G} = [\boldsymbol{I}_K, \boldsymbol{P}]. \tag{3.2}$$

Then a parity check matrix is obtained as $\boldsymbol{H} = [\boldsymbol{P}^T, \boldsymbol{I}_{N-K}]$.

Remark 3.1 (Affine codes)**.** An (N, K) binary affine block code $\mathcal{C}'$ is obtained by adding a constant N-dimensional vector $w_1^N \notin \mathcal{C}$ to all codewords of a linear block code $\mathcal{C}$, i.e., $\mathcal{C}' = \mathcal{C} + w_1^N$. Most of the properties of a linear block code can be extended with (almost) no effort to affine codes. However, for all the results in this thesis, linear codes suffice and we will (mostly) restrict attention to them.

The *minimum distance* of a code is defined as the minimum Hamming distance between two distinct codewords in the code, i.e.,

$$d_{\min}(\mathcal{C}) \triangleq \min\{d_{\mathrm{H}}\left(x_1^N, y_1^N\right) : x_1^N, y_1^N \in \mathcal{C}, x_1^N \neq y_1^N\}. \tag{3.3}$$

A binary linear code $\mathcal{C}$ is closed under element-wise addition in $\mathbb{F}_2$ and scalar multiplication, i.e., we have

$$\alpha x_1^N + \beta y_1^N \in \mathcal{C}, \quad \forall x_1^N, y_1^N \in \mathcal{C} \text{ and } \forall \alpha, \beta \in \mathbb{F}_2 \tag{3.4}$$

(3.3) can thus be re-written as

$$d_{\min}(\mathcal{C}) = \min\left\{w_{\mathrm{H}}\left(x_1^N\right) : x_1^N \in \mathcal{C}, x_1^N \neq 0\right\} \tag{3.5}$$

which means that, for a linear code, the problem of finding the minimum distance reduces to finding the codeword with minimum non-zero Hamming weight. There are many other useful properties of linear codes that can be found in standard books (see for instance [56, Chapter 1]).

Observe that a codeword c_1^N that is obtained by systematic encoding of v_1^K such that $w_{\mathrm{H}}\left(v_1^K\right) = 1$ satisfies $w_{\mathrm{H}}\left(c_1^N\right) \leq N - K + 1$. We formally state the implication of this observation through the following theorem (the statement can be generalized to non-linear and/or non-binary codes [57, Theorem 1]).

Theorem 3.1 (Singleton bound)**.** The minimum distance of an (N, K) binary code $\mathcal{C}$ satisfies

$$d_{\min}(\mathcal{C}) \leq N - K + 1. \tag{3.6}$$

Remark 3.2. A code whose minimum distance satisfies (3.6) with equality is called a *maximum distance separable (MDS)* code. Among all binary codes, there are only two classes of MDS codes: $(N, 1)$ repetition codes and $(N, N-1)$ SPC codes. The former repeats a single information bit N times and has a minimum distance of N, while the latter adds a single parity bit so that all codewords have an even Hamming weight and the minimum distance is 2.

More can be learned about a linear code if its *weight enumerating function (WEF)* is known. By introducing the dummy variable z, the WEF of $\mathcal{C}$ is

$$A_{\mathcal{C}}(z) \triangleq \sum_{i=0}^{N} \mathrm{A}_i z^i \tag{3.7}$$

where A_i is the number of codewords $x_1^N \in \mathcal{C}$ of weight $w_{\mathrm{H}}\left(x_1^N\right) = i$. The sequence $\mathrm{A}_0, \mathrm{A}_1, \ldots, \mathrm{A}_N$ is called the *weight enumerator (WE)* or *distance spectrum* of the code. WEs are particularly useful in deriving bounds on the performance of (binary) linear codes under ML decoding [58]. In the following, we use the terminology WE and distance spectrum interchangeably.

The *input-output weight enumerating function (IOWEF)* $A_{\mathcal{C}}^{\mathrm{IO}}(x, z)$ of an (N, K) code $\mathcal{C}$ is defined as

$$A_{\mathcal{C}}^{\mathrm{IO}}(x, z) \triangleq \sum_{i=0}^{k} \sum_{w=0}^{n} \mathrm{A}_{i,w}^{\mathrm{IO}} x^i z^w \tag{3.8}$$

where $\mathrm{A}_{i,w}^{\mathrm{IO}}$ is the number of codewords $x_1^N \in \mathcal{C}$ with $w_{\mathrm{H}}\left(u_1^K\right) = i$ and $w_{\mathrm{H}}\left(x_1^N\right) = w$. The IOWEF generalizes the WEF since we have $A_{\mathcal{C}}^{\mathrm{IO}}(z, z) = A_{\mathcal{C}}(z)$.

3.2. Most Reliable Basis Decoding

MRB decoding is a generic algorithm [59,60] that does not exploit any code structure except for linearity. The algorithm is tailored to channels that provide reliability information at their output. An example of such channels is BAWGNC, that we will use next as reference to describe MRB decoding.

Consider an (N, K) binary linear block code $\mathcal{C}$ with generator matrix $\boldsymbol{G}$. The algorithm works as follows. Suppose the codeword x_1^N is transmitted over the channel which provides y_1^N as the outputs. Note that if $|y_i| > |y_j|$, then we say that the observation y_i is more reliable than y_j [61, Section 10.2]. The outputs y_1^N are permuted via a permutation π_1, in an

increasing order of reliabilities. This yields a vector $\pi_1(y_1^N)$ whose first K components are the most reliable channel observations. The columns of $\boldsymbol{G}$ are permuted accordingly and the permuted generator matrix, denoted as $\pi_1(\boldsymbol{G})$, is put in systematic form (3.2). If the K left-most columns of $\pi_1(\boldsymbol{G})$ are not linearly independent, converting it to the systematic form may require additional column permutations, denoted by π_2. The permutation π_2 shall be applied to $\pi_1(y_1^N)$ also, which yields an overall permutation π defined via $\pi(y_1^N) = \pi_2(\pi_1(y_1^N))$. Next, hard-decisions are made for the first K bits as

$$\hat{v}_i = \begin{cases} 0 \text{ if } \pi(y_1^N)_i > 0 \\ 1 \text{ otherwise} \end{cases} \tag{3.9}$$

for $i = 1, \ldots, K$, where $\pi(y_1^N)_i$ is the i-th element of the permuted vector $\pi(y_1^N)$. The permuted decision for the codeword is obtained by first encoding $\hat{v}_1^K$ via $\pi(\boldsymbol{G})$ and then inverting the permutation, i.e., $\hat{x}_1^N = \pi^{-1}(\hat{v}_1^K \pi(\boldsymbol{G}))$. The complexity of this algorithm is roughly $\mathcal{O}(N^3)$, but it is highly sub-optimal even for codes as short as $N = 24$ bits [60, Figure 4].

The algorithm can be enhanced to provide near-optimum decoding at the expense of higher complexity. MRB decoding with reprocessing [60, Section IV.B] constructs a list after finding $\hat{v}_1^K$. To this end, all K-bit error patterns of Hamming weight up to t, where t is a predefined parameter, are added to $\hat{v}_1^K$, generating a set of vectors that are encoded via the systematic form of $\pi(\boldsymbol{G})$. yielding a list $\mathcal{L}$ of codewords with size

$$|\mathcal{L}| = \sum_{i=0}^{t} \binom{K}{i}. \tag{3.10}$$

Now the rule (2.72) is applied to the members of the list to obtain a decision

$$\hat{x}_1^N = \underset{x_1^N \in \mathcal{L}}{\operatorname{argmax}} P\left(y_1^N | x_1^N\right). \tag{3.11}$$

Note that (3.11) approaches to ML decoding for sufficiently large t and becomes equivalent if $t = K$. Observe that the list construction relies on the idea that, if one takes a hard decision on the most reliable channel observations, only few errors are typically observed. Recall Remark 2.4, which states that a naive implementation of optimal decoding has exponential complexity in N. MRB decoding with reprocessing reduces the complexity by limiting the search to the subset $\mathcal{L}$ of the codewords, controlled by the parameter t, by accepting sub-optimality.

MRB works remarkably well with short codes, enabling near-optimum decoding for even small t. However, as N grows, t must be increased to keep the decoder performance close to optimal. For instance, choosing $t = 1$ for the $(24, 12)$ Golay code suffices to approach the optimal performance [60, Figure 4] with $|\mathcal{L}| = 13$. On the other hand, t needs to be as large as 4 for close-to-optimum decoding when MRB decoding is applied to the $(128, 64)$ eBCH code [60, Figure 7] with $|\mathcal{L}| \approx 7 \cdot 10^5$.

Remark 3.3. A version of MRB decoding was first proposed by Dorsch for discrete output channels and applied to the quantized BAWGNC [59]. Approximately 20 years later, Fossorier and Lin rediscovered the method, called it *ordered statistics* decoding, analyzed it, and pointed out various directions to reduce complexity [60]. Since then MRB decoding has been studied mostly to reduce its complexity, e.g., see [62, 63]. In addition, MRB has been proposed as a part of hybrid schemes in the context of iterative decoding [64, 65]. The reader is referred to recent work [66] that reviews MRB decoding and suggests ways to further reduce the complexity.

3.3. Product Codes

Product codes are the first class of codes possessing a well-defined structure which were capable of achieving a vanishing bit error probability with a positive rate over the BSC [2]. The construction is based on smaller constituent codes, called *component* codes. The original decoder of [2] treats the product code as serial concatenation of several component codes. The component codesare decoded sequentially. Since the decoding task is divided into decoding smaller codes, its complexity is low compared to ML decoding applied to the whole product code. Much later, the suitability of product code constructions for iterative decoding algorithms [10] led to powerful classes of codes [67–72]. For an overview of product codes and their variants, we refer the reader to [73–75]. Product codes are usually constructed with high-rate algebraic component codes for which low-complexity soft-input soft-output (SISO) [68] or algebraic (e.g., bounded distance) [76–78] decoders are available. In the following, we review some important properties of product codes.

3.3.1. Parameters and Encoding

An m-dimensional (N, K) product code $\mathcal{C}$ is obtained by requiring that an m-dimensional array of bits satisfies a linear code constraint along each axis [2]. More precisely, denote by $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_m$ a set of linear block codes, where $\mathcal{C}_\ell$ has parameters (N_ℓ, K_ℓ). The infor-

mation bits are arranged in an m-dimensional hypercube, where the ℓ-th dimension is K_ℓ. Without loss of generality, encoding starts by encoding along the first dimension by replacing each K_1-bit vectors by its encoded version through the component code $\mathcal{C}_1$, yielding an m-dimensional hypercube of dimensions $N_1 \times K_2 \times \ldots \times K_m$. The K_2-bit vectors in second dimension is then encoded via the component code $\mathcal{C}_2$, resulting in the hypercube wit dimensions $N_1 \times N_2 \times K_3 \times \ldots \times K_m$. The encoding proceeds recursively till the m-th dimension, which outputs the hypercube with dimensions $N_1 \times N_2 \times \ldots \times N_m$. Hence, the parameters of the resulting product code satisfy

$$N = \prod_{\ell=1}^{m} N_\ell, \quad \text{and} \quad K = \prod_{\ell=1}^{m} K_\ell. \tag{3.12}$$

The rate of the product code is

$$R = \frac{K}{N} = \prod_{\ell=1}^{m} R_\ell \tag{3.13}$$

where R_ℓ is the rate of the ℓ-th component code. For example, a codeword of a 2-dimensional product code can be seen as a 2-dimensional array, where each row is a codeword of $\mathcal{C}_1$ and each column is a codeword of $\mathcal{C}_2$. Its structure is illustrated in Figure 3.1, where systematic encoding is adopted for the component codes and where the information bits v_1^K are placed at the top left corner. A generator matrix of the m-dimensional product code is

$$\boldsymbol{G} = \boldsymbol{G}_1 \otimes \boldsymbol{G}_2 \otimes \ldots \otimes \boldsymbol{G}_m \tag{3.14}$$

where $\boldsymbol{G}_\ell$ is the generator matrix of the ℓ-th component code.

3.3.2. Distance Spectrum

Characterizing the distance spectrum of an arbitrary product code is an open problem even if the distance spectra of its component codes are known [79–82]. Fortunately, the minimum distance and its multiplicity are known to be

$$d = \prod_{\ell=1}^{m} d_\ell, \quad \text{and} \quad A_d = \prod_{\ell=1}^{m} A_{d_\ell}^{(\ell)}. \tag{3.15}$$

Here d_ℓ and $A_{d_\ell}^{(\ell)}$ are the minimum distance of the ℓ-th component code and its multiplicity, respectively. Examples of minimum distances and multiplicities for some 2-dimensional product codes based on SPC and extended Hamming codes are provided in Table 3.1.

Although it may not be possible to obtain complete distance spectrum of an arbitrary

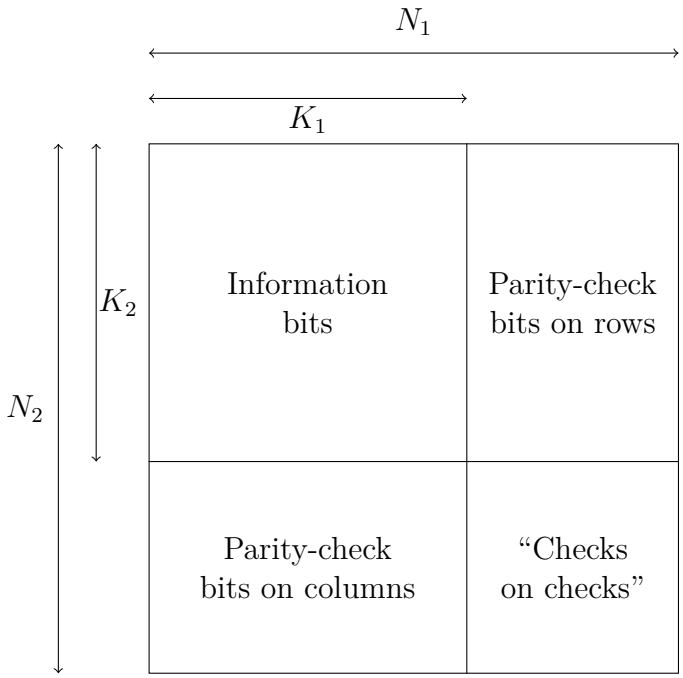

Figure 3.1.: Codeword structure of the product code $\mathcal{C}$ with component codes $\mathcal{C}_1$ and $\mathcal{C}_2$.

Table 3.1.: Minimum distances and multiplicities of some product codes (eH = extended Hamming code).

(N, K)	$\mathcal{C}_1$	$\mathcal{C}_2$	d	A_d
$(128, 77)$	eH $(16, 11)$	SPC $(8, 7)$	8	3920
$(128, 105)$	SPC $(16, 15)$	SPC $(8, 7)$	4	3360
$(256, 121)$	eH $(16, 11)$	eH $(16, 11)$	16	19600
$(256, 165)$	eH $(16, 11)$	SPC $(16, 15)$	8	16800
$(256, 225)$	SPC $(16, 15)$	SPC $(16, 15)$	4	14400
$(1024, 693)$	SPC $(64, 63)$	eH $(16, 11)$	8	282240
$(1024, 806)$	eH $(32, 26)$	SPC $(32, 31)$	8	615040
$(1024, 961)$	SPC $(32, 31)$	SPC $(32, 31)$	4	246016

product code, there are some exceptions. For instance, if both component codes are simplex codes (i.e., the dual of Hamming codes), then the distance spectrum of the product code is known [56, Page 571]. In addition, a closed-form expression for the WEF of 2-dimensional product codes with SPC component codes, namely SPC product codes, is given in [83, Appendix A]. Generalizing the result, the authors also propose a method to compute the WEF of 2-dimensional short product codes where at least one of the component codes is an SPC code. We will elaborate on the distance spectrum of SPC product codes in Chapter 6 and extend some of the previous results.

3.4. Polar Codes

Polar codes are the first class of provably capacity-achieving codes over any BMSC with low encoding and decoding complexity [6]. In addition to the theoretical interest, polar codes concatenated with an outer CRC code are attractive from a practical viewpoint [12, Chapter 5] due to their excellent performance under SCL decoding [11] in the short and moderate blocklength regime [36]. In the following, we review the concept of *channel polarization* briefly, which leads to the capacity-achieving scheme. Before concluding, generalizations of polar codes are reviewed, since the decoding algorithms under consideration naturally generalize to them. Proofs are reviewed in Appendix 3.5 which helps to understand not only polar codes but also the results presented in the thesis.

We motivate the underlying technique as follows. There are two types of channels over which the communication is *easy*: *useless* and *noiseless* channels. A channel is said to be *useless* if its output is independent from the input, i.e., if, for all $y \in \mathcal{Y}$, we have

$$W(y|0) = W(y|1). \tag{3.16}$$

The capacity of this channel is zero. A channel is said to be *noiseless* if its output determines the input, i.e., if, for all $y \in \mathcal{Y}$, we have

$$W(y|0)W(y|1) = 0. \tag{3.17}$$

The capacity of a noiseless channel is one, which means transmitting uncoded bits enables error-free communication. For both channels, there is no need to code. Channel polarization converts (in the limit at large blocklengths) any BMSC to a mixture of easy channels (plus vanishing-small fraction of mediocre channels). The main ingredients to achieve this result are a simple transform and a successive cancellation decoding algorithm.

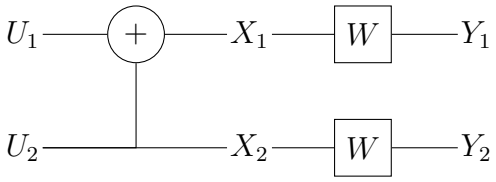

Figure 3.2.: The basic transform.

3.4.1. The Basic Transform

Let U_1^2 be uniform on $\mathbb{F}_2^2$ that is mapped onto X_1^2 as $X_1^2 = U_1^2 \boldsymbol{K}_2$ where

$$\boldsymbol{K}_2 \triangleq \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}. \tag{3.18}$$

The 2×2 matrix $\boldsymbol{K}_2$ is referred to as polarization *kernel*. Assume now that X_1 and X_2 are transmitted over two independent copies of a BMSC W resulting in the observations Y_1 and Y_2 (as in Figure 3.2). Since the channel is memoryless, we have $I(X_1; Y_1) = I(X_2; Y_2) = I(W)$. The following lemma holds (see Appendix 3.5.1 for the proof).

Lemma 3.2. [6, Proposition 4]

$$I(U_1; Y_1^2) + I(U_2; Y_1^2 U_1) = 2I(W) \tag{3.19}$$

$$I(U_2; Y_1^2 U_1) \geq I(W) \geq I(U_1; Y_1 Y_2) \tag{3.20}$$

with equality if and only if $I(W) \in \{0, 1\}$.

Inspecting the first inequality in (3.20) suggest that an optimum receiver (say bit-wise MAP) provides a more reliable estimate of U_2 by observing Y_1^2 with the "knowledge" of U_1 than that of X_2 by observing only Y_2. Similarly, observing Y_1 provides a more reliable estimate of X_1 than observing Y_1^2 for the estimate of U_1 according to the second inequality in (3.20). We then define two *synthesized* channels: the first has input U_1 and output Y_1^2, denoted as $W_{\boldsymbol{K}_2}^1 : \mathbb{F}_2 \to \mathcal{Y}^2$, and the second has input U_2 and output (Y_1^2, U_1), denoted as $W_{\boldsymbol{K}_2}^2 : \mathbb{F}_2 \to \mathcal{Y}^2 \times \mathbb{F}_2$. The former channel has a lower capacity than the original channel W, while the latter has a higher capacity. In the following, we will give the main results by generalizing this basic idea.

3.4.2. Recursive Application of the Basic Transform: Channel Polarization

The transform of size-2 defined by $\boldsymbol{K}_2$ provides all the necessary ingredients to describe a useful coding scheme and proving the main coding theorems.

A transform matrix of length-$N = 2^n$ is defined as $\boldsymbol{K}_2^{\otimes n}$, where $n \in \mathbb{N}$ and $\boldsymbol{K}_2^{\otimes n}$ is the m-fold Kronecker product with $\boldsymbol{K}_2^{\otimes 0} \triangleq 1$. This form of the transform matrix suffices to provide the desired results but it is tedious to describe. An alternative construction that enables successive decoding in a natural order, i.e., from U_1 to U_N, is available. To this end, the $N \times N$ transform matrix $\boldsymbol{G}^{[n]}$ is constructed recursively as

$$\boldsymbol{G}^{[n]} = (\boldsymbol{I}_{2^{n-1}} \otimes \boldsymbol{K}_2)\, \boldsymbol{\Pi}_{2^{n-1},2} \left(\boldsymbol{I}_2 \otimes \boldsymbol{G}^{[n-1]}\right) \tag{3.21}$$

where the recursion starts from $\boldsymbol{G}^{[0]} \triangleq 1$ and $\boldsymbol{\Pi}_{2^{n-1},2}$ is the reverse shuffle permutation (2.12). To see the relation between $\boldsymbol{K}_2^{\otimes n}$ and $\boldsymbol{G}^{[n]}$, we write

$$\begin{aligned}
\boldsymbol{G}^{[n]} &= \boldsymbol{\Pi}_{2^{n-1},2}\, (\boldsymbol{K}_2 \otimes \boldsymbol{I}_{2^{n-1}}) \left(\boldsymbol{I}_2 \otimes \boldsymbol{G}^{[n-1]}\right) && (3.22)\\
&= \boldsymbol{\Pi}_{2^{n-1},2} \left(\boldsymbol{K}_2 \otimes \boldsymbol{G}^{[n-1]}\right) && (3.23)\\
&= \left(\prod_{i=1}^{n} \boldsymbol{I}_{2^{i-1}} \otimes \boldsymbol{\Pi}_{2^{n-i},2}\right) (\boldsymbol{K}_2 \otimes \boldsymbol{K}_2 \otimes \ldots \otimes \boldsymbol{K}_2) && (3.24)\\
&= \boldsymbol{B}^{[n]} \boldsymbol{K}_2^{\otimes n} && (3.25)
\end{aligned}$$

where (3.22) follows by applying the identity

$$\boldsymbol{\Pi}_{2^{n-1},2} \left(\boldsymbol{I}_2 \otimes \boldsymbol{G}^{[n-1]}\right) = \left(\boldsymbol{G}^{[n-1]} \otimes \boldsymbol{I}_2\right) \boldsymbol{\Pi}_{2^{n-1},2} \tag{3.26}$$

and (3.23) from the mixed-product identity (2.9). Then (3.24) follows by re-writing $\boldsymbol{G}^{[n-1]}$ through (3.21) and by applying similar steps recursively. Finally, defining *bit-reversal permutation matrix* $\boldsymbol{B}^{[n]} \triangleq \prod_{i=1}^{n} \boldsymbol{I}_{2^{i-1}} \otimes \boldsymbol{\Pi}_{2^{n-i},2}$ [6, Section VII.B] leads to (3.25), which shows that $\boldsymbol{K}_2^{\otimes n}$ and $\boldsymbol{G}^{[n]}$ are equivalent up to a bit-reversed column permutation.

Let U_1^N be uniform on $\mathbb{F}_2^N$ and let $X_1^N = U_1^N \boldsymbol{G}^{[n]}$. The transition probabilities of the i-th synthesized channel, denoted by $W_{\boldsymbol{G}^{[n]}}^i : \mathbb{F}_2 \to \mathcal{Y}^N \times \mathbb{F}_2^{i-1}$, with the input U_i and the output (Y_1^N, U_1^{i-1}), are defined by

$$W_{\boldsymbol{G}^{[n]}}^i(y_1^N, u_1^{i-1}|u_i) \triangleq \sum_{u_{i+1}^N \in \mathbb{F}_2^{N-i}} \frac{1}{2^{N-1}} W_{\boldsymbol{G}^{[n]}}(y_1^N|u_1^N) \tag{3.27}$$

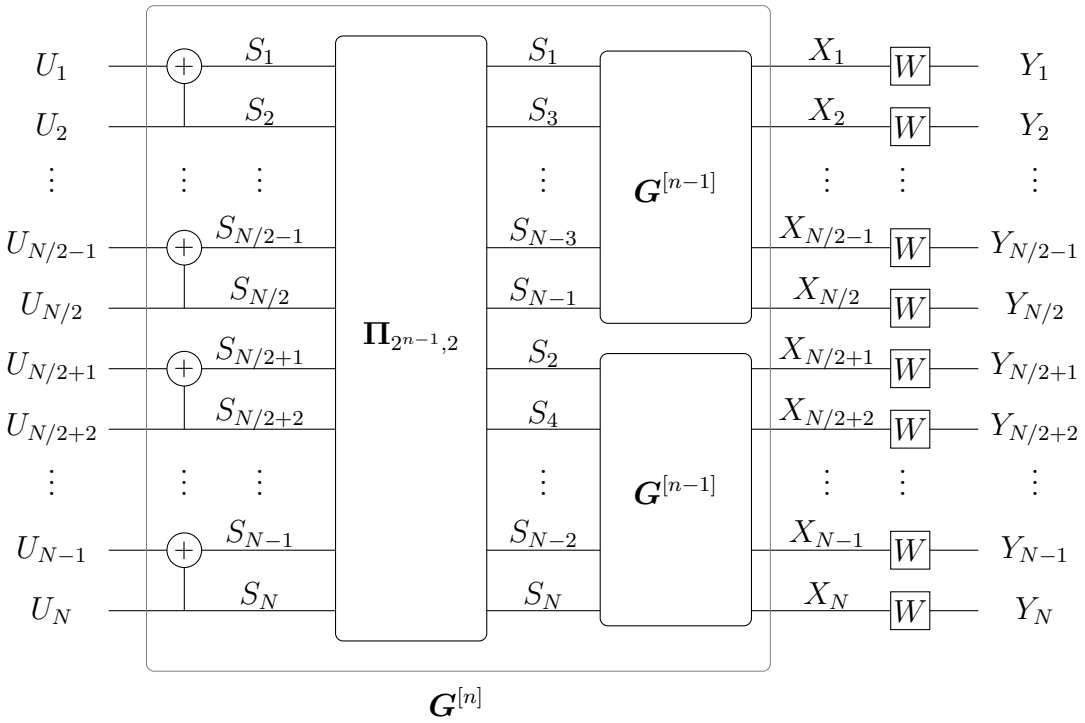

Figure 3.3.: Size-N polar transform $\boldsymbol{G}^{[n]}$, where $N = 2^n$.

where $W_{\boldsymbol{G}^{[n]}}(y_1^N|u_1^N) \triangleq W^N(y_1^N|u_1^N\boldsymbol{G}^{[n]})$. This mapping is illustrated in Figure 3.3. Next, we state the main theorem of channel polarization (the proof is given in Appendix 3.5.2).

Theorem 3.3. [6, Theorem 1] For any BMSC W, the synthesized channels $W^i_{\boldsymbol{G}^{[n]}}$, $i \in [N]$, polarize, i.e, the fraction of channels with $I\left(W^i_{\boldsymbol{G}^{[n]}}\right) > 1-\delta$ goes to $I(W)$ and the fraction of channels with $I\left(W^i_{\boldsymbol{G}^{[n]}}\right) < \delta$ to $1 - I(W)$ for any fixed $\delta \in (0,1)$ as $n \to \infty$.

The theorem states that, in the limit at large blocklengths, the transform $\boldsymbol{G}^{[n]}$ in conjunction with definition (3.27) creates a sequence of almost useless and almost noiseless channels $W^i_{\boldsymbol{G}^{[n]}}$ out of several copies of a BMSC W. In other words, the fraction of mediocre channels is vanishes with increasing N.

3.4.3. Polar Coding

We are now ready to define a first code construction that builds on Theorem 3.3. For any $\delta > 0$, define the two sets

$$\mathcal{A}_\delta \triangleq \left\{i \in [N] : \Pr\left\{U_i \neq \hat{U}_i | Y_1^N = y_1^N, U_1^{i-1} = u_1^{i-1}\right\} < \delta\right\} \tag{3.28}$$

$$\mathcal{F}_\delta \triangleq [N] \setminus \mathcal{A}_\delta \tag{3.29}$$

where the error events $\Pr\left\{U_i \neq \hat{U}_i | Y_1^N = y_1^N, U_1^{i-1} = u_1^{i-1}\right\}$ assume a MAP estimate $\hat{U}_i$ of U_i when observing (Y_1^N, U_1^{i-1}).

Code Construction: For a blocklength $N = 2^n$, $n \in \mathbb{N}$, and any BMSC, determine the set $\mathcal{F}_\delta$. Generate u_i, $i \in \mathcal{F}_\delta$, uniformly at random and reveal the vector $u_{\mathcal{F}_\delta}$ to the receiver. A $\boldsymbol{G}^{[n]}$-coset code is defined by an offset $u_{\mathcal{F}_\delta}\boldsymbol{G}^{[n]}_{\mathcal{F}_\delta}$ and a matrix $\boldsymbol{G}^{[n]}_{\mathcal{A}_\delta}$, where $\boldsymbol{G}^{[n]}_{\mathcal{S}}$ is the matrix formed by the rows of $\boldsymbol{G}^{[n]}$ indexed by the elements of set $\mathcal{S}$.

Encoding: Generate u_i, $i \in \mathcal{A}_\delta$, uniformly at random. Encode the message $u_{\mathcal{A}_\delta}$ using the (affine) code as

$$x_1^N = u_{\mathcal{F}_\delta}\boldsymbol{G}^{[n]}_{\mathcal{F}_\delta} \oplus u_{\mathcal{A}_\delta}\boldsymbol{G}^{[n]}_{\mathcal{A}_\delta} \tag{3.30}$$

and transmit x_1^N over the channel. Inspecting (3.21) and Figure 3.3 leads to a natural recursive implementation of encoding (3.30), which has a time complexity of $\mathcal{O}(N \log_2 N)$ and a space complexity of $\mathcal{O}(N)$ [6, Section VII.C].

Note that X_i, $i \in [N]$, are i.i.d. RVs. Hence, we conclude using Theorem 3.3 that the rate of the scheme satisfies

$$R = \frac{|\mathcal{A}_\delta|}{N} \to I(W) \tag{3.31}$$

since $I\left(W^i_{\boldsymbol{G}^{[n]}}\right) \to 1$ implies $\Pr\left\{U_i \neq \hat{U}_i | Y_1^N = y_1^N, U_1^{i-1} = u_1^{i-1}\right\} \to 0$.

Successive Cancellation Decoding: Upon observing the channel output y_1^N, SC decoding estimates the bit u_i starting from $i = 1$ to $i = N$ as

$$\hat{u}_i(u_1^N, y_1^N) = \begin{cases} u_i & \text{if } i \in \mathcal{F}_\delta \\ f_i(y_1^N, \hat{u}_1^{i-1}(u_1^N, y_1^N)) & \text{if } i \in \mathcal{A}_\delta \end{cases} \tag{3.32}$$

by using the previously estimated bits $\hat{u}_1^{i-1}$ in the function

$$f_i(y_1^N, \hat{u}_1^{i-1}(u_1^N, y_1^N)) \triangleq \begin{cases} 0 \text{ if } P_{U_i|Y_1^N,U_1^{i-1}}(0|y_1^N, \hat{u}_1^{i-1}) \geq P_{U_i|Y_1^N,U_1^{i-1}}(1|y_1^N, \hat{u}_1^{i-1}) \\ 1 \text{ otherwise.} \end{cases} \tag{3.33}$$

Note that the decision unit (3.32) seems to have u_1^N as an input, but this is not the case when (3.33) is inspected carefully. This notation is needed for the formalism in the error analysis. Also, the decision metric used in f_i is not even bit-wise MAP decoding (2.66) since the probabilities $P_{U_i|Y_1^N,U_1^{i-1}}(u_i|y_1^N, \hat{u}_1^{i-1})$ do not incorporate the knowledge of the frozen bits u_j, $j > i$ and $j \in \mathcal{F}_\delta$, which are known to the receiver. The computation of the

probabilities $P_{U_i|Y_1^N,U_1^{i-1}}(u_i|y_1^N,\hat{u}_1^{i-1})$ assumes that all the bits U_j, $j>i$, are uniform RVs, which enables an efficient computation. The notation will be refined when we explicitly provide equations to look SC decoding in detail.

Error Analysis: We are interested in the probability of block error events under SC decoding defined as

$$\mathcal{B} \triangleq \{(u_1^N, y_1^N) \in \mathbb{F}_2^N \times \mathcal{Y}^N : \hat{u}_{\mathcal{A}_\delta} \neq u_{\mathcal{A}_\delta}\} \tag{3.34}$$

since the decoder never makes an error in decoding $u_{\mathcal{F}_\delta}$. Let $P_B^{(\mathrm{SC})} \triangleq \Pr\{\mathcal{B}\}$ denote the BLEPs of SC decoding described above, which is what one can implement at the receiver. In addition, let $P_B^{(\mathrm{GA})}$ be the BLEPs of *genie-aided* SC decoding, which knows the bits u_1^{i-1} no matter what the decoding outputs for $\hat{u}_1^{i-1}$ are. The following lemma states an equivalence between the block error events of implementable SC decoding and genie-aided SC decoding, which will let us use the definition of set $\mathcal{A}_\delta$ to bound $P_B^{(\mathrm{SC})}$. The proof is given as Appendix 3.5.3, which is valid for any BMSC.

Lemma 3.4. [84, Lemma 1], [85, Proposition 2.1] For any fixed $\mathcal{A}_\delta$, $P_B^{(\mathrm{SC})} = P_B^{(\mathrm{GA})}$.

Using Lemma 3.4, we write

$$\begin{aligned} P_B^{(\mathrm{SC})} &= P_B^{(\mathrm{GA})} && (3.35)\\ &= \Pr\left\{\bigcup_{i=1}^{N} \mathcal{B}_i^{(\mathrm{GA})}\right\} && (3.36)\\ &\leq \sum_{i\in\mathcal{A}_\delta} \Pr\left\{\mathcal{B}_i^{(\mathrm{GA})}\right\} && (3.37)\\ &\leq |\mathcal{A}_\delta|\delta && (3.38)\\ &\leq N\delta && (3.39) \end{aligned}$$

where (3.36) follows by defining the bit-error event for u_i under a genie-aided SC decoder as

$$\mathcal{B}_i^{(\mathrm{GA})} \triangleq \{(u_1^N, y_1^N) \in \mathcal{T} : f_i(y_1^N, u_1^{i-1}) \neq u_i\} \tag{3.40}$$

and the sample space $\mathcal{T} \triangleq \{(u_1^N, y_1^N) \in \mathbb{F}_2^N \times \mathcal{Y}^N : U_{\mathcal{F}_\delta} = u_{\mathcal{F}_\delta}\}$. The bound (3.37) follows from the union bound and because SC decoding (3.32) does not make an error for bits $u_{\mathcal{F}_\delta}$, (3.38) from the definition of the set $\mathcal{A}_\delta$ as given in (3.28) and (3.39) from $\mathcal{A}_\delta \subseteq [N]$.

The upper bound (3.39) is not useful since δ is independent of N. Fortunately, [86,

Theorem 1] proves that the definition (3.28) can be modified, for any $0 < \beta < 1/2$, as

$$\mathcal{A}_{2^{-N^\beta}} \triangleq \left\{ i \in [N] : \Pr\left\{ U_i \neq \hat{U}_i | Y_1^N = y_1^N, U_1^{i-1} = u_1^{i-1} \right\} < 2^{-N^\beta} \right\} \tag{3.41}$$

with Theorem 3.3 still holding. We now rewrite (3.39) as

$$P_B^{(\mathrm{SC})} \leq N 2^{-N^\beta} \to 0 \quad \text{as} \quad N \to \infty. \tag{3.42}$$

Remark 3.4 (Rate of polarization)**.** In [6, Theorem 2], the definition (3.28) is modified by setting $\delta = \mathcal{O}(N^{-\frac{5}{4}})$, which still suffices to conclude that $P_B^{(\mathrm{SC})} \to 0$ although the convergence is slower than (3.42).

Note that the result (3.42) is derived for the $\boldsymbol{G}^{[n]}$-coset code ensemble where the bits $u_{\mathcal{F}_{2^{-N^\beta}}}$ are generated uniformly at random. It is easy to see that there is at least one choice of $u_{\mathcal{F}_{2^{-N^\beta}}}$ that leads to a capacity-achieving code.

Remark 3.5 (Polar code)**.** The method above is still not constructive, even if one finds the set $\mathcal{A}_{2^{-N^\beta}}$, since it is valid for a code ensemble. Fortunately, the selection of $u_{\mathcal{F}_{2^{-N^\beta}}}$ does not affect the error performance when transmission is over BMSCs [6, Corollary 1]. Hence, any selection leads to a capacity-achieving code. For any N, all codes defined by the set $\mathcal{A} \subset [N]$ minimizing the right-hand side (RHS) of (3.37) for a given channel W are called *polar codes*. The predetermined bits $u_{\mathcal{F}}$, $\mathcal{F} \triangleq [N] \setminus \mathcal{A}$, and free bits $u_{\mathcal{A}}$ are called *frozen* and *information bits*, respectively. Accordingly, the sets $\mathcal{A}$ and $\mathcal{F}$ are called *information* and *frozen sets*, respectively. Note also that setting $u_i = 0$, $i \in \mathcal{F}$, leads to a linear code. In this case, a generator matrix of the polar code is obtained by removing the rows of $\boldsymbol{G}^{[n]}$ with indices in $\mathcal{F}$.

3.4.4. A Detailed Look at Successive Cancellation Decoding

SC decoding has been described briefly with the decoding functions (3.32) and (3.33). We here rewrite these functions for an arbitrary information set $\mathcal{A} \subseteq [N]$ as

$$\hat{u}_i = \begin{cases} u_i & \text{if } i \in \mathcal{F} \\ f_i(y_1^N, \hat{u}_1^{i-1}) & \text{if } i \in \mathcal{A} \end{cases} \tag{3.43}$$

and

$$f_i(y_1^N, \hat{u}_1^{i-1}) \triangleq \begin{cases} 0 \text{ if } P^i_{\boldsymbol{G}^{[n]}}(0|y_1^N, \hat{u}_1^{i-1}) \geq P^i_{\boldsymbol{G}^{[n]}}(1|y_1^N, \hat{u}_1^{i-1}) \\ 1 \text{ otherwise} \end{cases} \tag{3.44}$$

where the quantities $P^{i}_{\boldsymbol{G}^{[n]}}(u_i|y_1^N, \hat{u}_1^{i-1})$, $u_i \in \mathbb{F}_2$, are computed recursively as

$$P^{2j-1}_{\boldsymbol{G}^{[n]}}(u_{2j-1}|y_1^N, u_1^{2j-2}) = \sum_{u_{2j}} P^{j}_{\boldsymbol{G}^{[n-1]}}(u_{2j-1} \oplus u_{2j}|y_1^{N/2}, u_{1,2}^{2j-2} \oplus u_{2,2}^{2j-2}) P^{j}_{\boldsymbol{G}^{[n-1]}}(u_{2j}|y_{N/2+1}^{N}, u_{2,2}^{2j-2}) \tag{3.45}$$

$$P^{2j}_{\boldsymbol{G}^{[n]}}(u_{2j}|y_1^N, u_1^{2j-1}) = \frac{P^{j}_{\boldsymbol{G}^{[n-1]}}(u_{2j-1} \oplus u_{2j}|y_1^{N/2}, u_{1,2}^{2j-2} \oplus u_{2,2}^{2j-2}) P^{j}_{\boldsymbol{G}^{[n-1]}}(u_{2j}|y_{N/2+1}^{N}, u_{2,2}^{2j-2})}{P^{2j-1}_{\boldsymbol{G}^{[n]}}(u_{2j-1}|y_1^N, u_1^{2j-2})} \tag{3.46}$$

where the notation $u_{1,2}^{2j}$ and $u_{2,2}^{2j}$ correspond to the subvectors of u_1^{2j} with odd and even indices, respectively, and the recursion continues down to length-1 computations as

$$P^{1}_{\boldsymbol{G}^{[0]}}(0|y_i) \triangleq \frac{W(y_i|0)}{\sum_x W(y_i|x)} \quad \text{and} \quad P^{1}_{\boldsymbol{G}^{[0]}}(1|y_i) \triangleq \frac{W(y_i|1)}{\sum_x W(y_i|x)}. \tag{3.47}$$

The recursive functions (3.45) and (3.46) can be efficiently implemented with time and space complexity $\mathcal{O}(N \log_2 N)$.

To understand where the efficiency arises, consider an alternative representation of the polar transform:

$$\boldsymbol{G}^{[n]} = \left(\boldsymbol{I}_2 \otimes \boldsymbol{G}^{[n-1]}\right) \boldsymbol{\Pi}_{2,2^{n-1}} \left(\boldsymbol{I}_{2^{n-1}} \otimes \boldsymbol{K}_2\right) \tag{3.48}$$

which can be verified via similar steps to conclude (3.25) starting from (3.21). This representation is illustrated in Figure 3.4. This decomposition is very similar to the efficient recursive encoding (recall Figure 3.3) and leads to a recursive call for two polar decoders of size-$N/2$ after performing a *soft inversion* of a single polar transform. Soft inversion corresponds to using equations (3.45) and (3.46) to obtain probabilistic information about the sequence $\tilde{S}_1^N$. Note that the second operation (3.46) requires a decision on the former bit. This implies that to decode the size-$N/2$ polar *constituent* code on the lower part of Figure 3.4, the decisions for the upper size-$N/2$ polar constituent code should be made first. Recursively, each polar decoder of size-$N/2$ calls two polar decoders of size-$N/4$. This recursion goes down to single size-1 decoders. Each decoder makes a decision according to the type of the bit, namely information or frozen bit, which either follows a simple threshold-based rule or the constraint imposed by the frozen bit. The reader is referred to [6, Section VIII.B] for more details on SC decoding and to [87] for efficient hardware implementations.

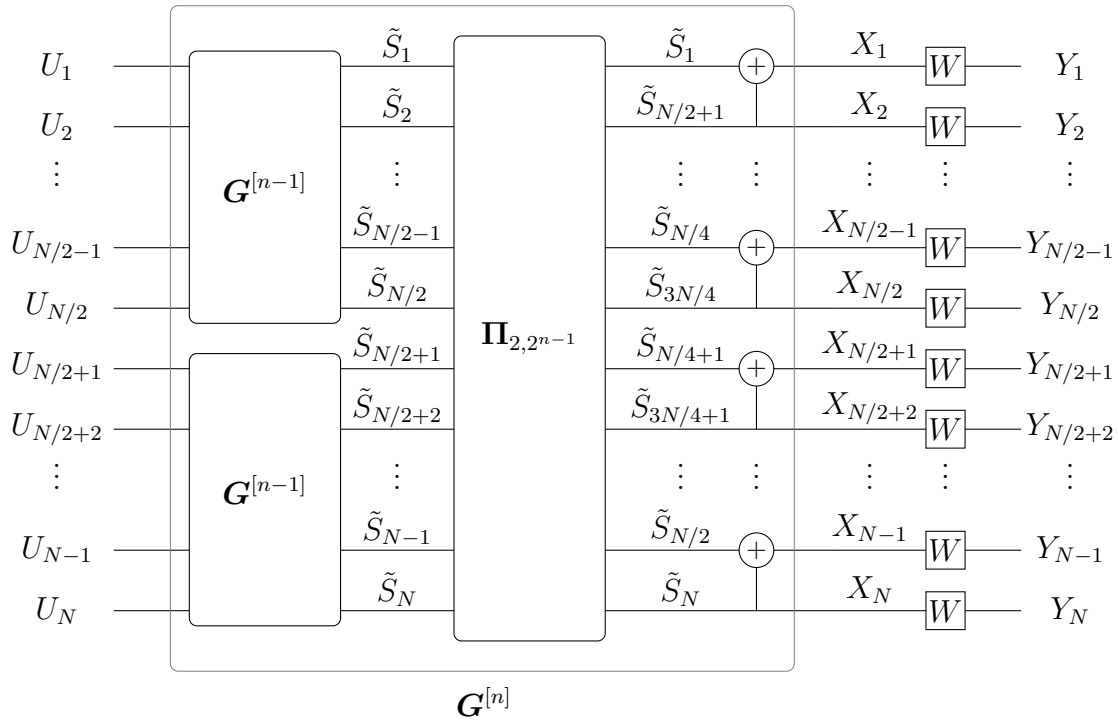

Figure 3.4.: Size-N polar transform $\boldsymbol{G}^{[n]}$, where $N = 2^n$, using (3.48).

3.4.5. Successive Cancellation List Decoding

The suboptimality of SC decoding in performance compared to ML decoding stems from the fact that any hard decision made for information bits leads to a block error and this cannot be corrected at a later stage. Fortunately, SCL decoding proposes a practical solution to this problem, closing gap to ML decoding with sufficiently large list sizes. In the following, we will briefly explain how the algorithm works.

For $i = 1, \ldots, N$, one can recursively compute

$$p^i_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^i) = p^{i-1}_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^{i-1}) P^i_{\boldsymbol{G}^{[n]}}(\tilde{u}_i | y_1^N, \tilde{u}_1^{i-1}) \tag{3.49}$$

where the right-most term can be computed efficiently via the SC decoding operations (3.45) and (3.46) for any partial input sequences $\tilde{u}_1^i \in \mathcal{U}_i \subseteq \{0,1\}^i$, with $\tilde{u}_{\mathcal{F}^{(i)}} = u_{\mathcal{F}^{(i)}}$ and $\mathcal{F}^{(i)} \triangleq \mathcal{F} \cap [i]$, by setting the initial value to

$$p^0_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^0) \triangleq 1. \tag{3.50}$$

Since they are computed by SC decoding operations, the quantities $p^i_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^i)$ facilitate the knowledge of $u_{\mathcal{F}^{(i)}}$, but they treat the future frozen bits $U_{\mathcal{F}\setminus\mathcal{F}^{(i)}}$ as uniform RVs.

Therefore, we refer to $p^i_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^i)$ as the *myopic probability* of the sequence $\tilde{u}_1^i$.[1]

The suboptimality of the SC decoding is overcome in practice by SCL decoding [11] which computes the quantities $p^i_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^i)$ for several partial input sequences, also called *decoding paths*, instead of making a hard decision directly after computing the quantities $P^i_{\boldsymbol{G}^{[n]}}(u_i|y_1^N, \hat{u}_1^{i-1})$, $u_i \in \mathbb{F}_2$. More formally, let $\mathcal{U}_{i-1} \subseteq \{0,1\}^{i-1}$ be a subset satisfying $|\mathcal{U}_{i-1}| = L$ and assume that $p^{i-1}_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^{i-1})$ is known for some $\tilde{u}_1^{i-1} \in \mathcal{U}_{i-1}$. For $\tilde{u}_i \in \mathbb{F}_2$, one can compute $p^i_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^i)$ using (3.49). This gives the values $p^i_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^i)$ for $2L$ partial sequences. One then prunes the list down to L sequences by keeping only the most likely paths according to (3.49) for an SCL decoder with list size L. Note that if u_i is frozen, then the decoder extends all paths with the correct frozen bit. After the N-th decoding stage, the estimate $\hat{u}_1^N$ is chosen as the candidate maximizing the function $p^N_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^N)$. The efficient data structures introduced in [11] let one implement an SCL decoder with list size L with time and space complexity of $\mathcal{O}(LN \log_2 N)$. Hence, keeping the parameter L small while having a good performance is one of the main targets for polar coding.

Lemma 3.5. For any $\tilde{u}_1^i \in \mathbb{F}_2^i$, SCL decoding implemented using the recursion (3.49) provides

$$p^i_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^i) = c_i P\left(\tilde{u}_{\mathcal{A}^{(i)}}|y_1^N, \tilde{u}_{\mathcal{F}^{(i)}}\right) \tag{3.51}$$

where $\mathcal{A}^{(i)} = \mathcal{A} \cap [i]$ as defined by (2.2), i.e., the myopic probabilities are equivalent to the APPs of the partial input sequence $\tilde{u}_1^i$ given the channel observation y_1^N and the previous frozen bits $\tilde{u}_{\mathcal{F}^{(i)}}$ up to a normalization factor $c_i > 0$.

Proof. See [11, Equations (9)-(12)] as stated in [11, Section IV.C] for the case $i = N$. ■

Corollary 3.6. Let γ denote the index of the last frozen bit, i.e.,

$$\gamma \triangleq \max \mathcal{F}. \tag{3.52}$$

Then for any $i > \gamma$ and any $\tilde{u}_1^i \in \mathbb{F}_2^i$, the myopic probabilities become equivalent to the APPs up to a normalization factor $c_i > 0$.

Proof. See Lemma 3.5 since there are no more frozen bits after γ. ■

The corollary implies that an SCL decoder with list size $L = 2^{NR}$ achieves a block-wise MAP decoding with an exponential complexity. However, a modification of the SCL decoding leads the same performance with a list size smaller than 2^{NR}. The modified SCL decoder works as follows: run an SCL decoder with unbounded list size until the

[1] We write the myopic probability of $\tilde{u}_1^i$ rather than $(y_1^N, \tilde{u}_1^i)$ since y_1^N is fixed once the channel is observed.

last frozen bit u_γ and then, for each decoding path $\tilde{u}_1^\gamma \in \mathbb{F}_2^\gamma$, find the MAP extension using *nearest coset coding* introduced in [88]. This leads to a list of candidates $\mathcal{L}$ such that $|\mathcal{L}| = 2^{NR-(N-\gamma)} = 2^{\gamma-N(1-R)}$. The final decision is the codeword in the final list with the largest probability. This decoder implements MAP decoding with a complexity of $\mathcal{O}(2^{\gamma-N(1-R)} N \log_2 N)$.

This result on the required list size to achieve block-wise optimum decoding was first given in [89, Theorem 1] for transmission over the BEC for standard SCL decoding and was extended to the case of arbitrary BMSCs in [88] by introducing the *hybrid* SCL decoding algorithm described above. This bound, however, has some disadvantages: first, it hinders the dependency of γ to the blocklength and appear to be dependent only on the position of last frozen bit.[2] Second and more importantly, it is far from being practically relevant. In particular, SCL decoding with list size $L = 32$ gets very close to the MAP performance of $(2048, 1024)$ polar codes [11, Figure 1], whereas $2^{\gamma-N(1-R)} = 2^{1795-1024} = 2^{771}$! In Chapter 5, we will relax our target by focusing on the average list size of SCL decoding to keep the transmitted codeword in the final list, which provides a result for the average case only; hence, SCL decoding whose list size is set to this average is not (even close to) a MAP decoder. This relaxation, however, will enable us not only to characterize the average list size at each decoding stage $i \in [N]$ for a given set $\mathcal{A}$ but also to provide heuristic guidelines on polar code design for list decoding with practical list sizes.

3.4.6. Reed-Muller Codes

RM codes are closely related to polar codes. Both RM and polar codes can be described using the Plotkin construction [90]. RM codes are named after Reed and Muller, where the former introduced them [91] and the latter proposed a related decoding algorithm [92]. An r-th order RM code of length $N = 2^n$ and dimension $K = 1 + \binom{n}{1} + \binom{n}{2} + \cdots + \binom{n}{r}$, denoted by $\mathrm{RM}(r, n)$ with $0 \leq r \leq n$, is defined using the matrix $\boldsymbol{G}^{[n]}$ as in the case of polar codes. A generator matrix for the RM code is obtained by removing the rows of $\boldsymbol{G}^{[n]}$ with weight lower than 2^{n-r}. This means they can be described by an information set $\mathcal{A}$ that consists of the indices $i \in [N]$ of rows having Hamming weight at least equal to $n - r$ for the binary expansion of $i - 1$. Encoding can be performed as for polar codes by setting $u_{\mathcal{F}} = 0$. For further similarities between polar and RM codes, see, e.g., [93].

Remark 3.6 (Minimum Distance)**.** An upper bound on the minimum distance of $\mathrm{RM}(r, n)$ is 2^{n-r}. Indeed, it can be shown by exploiting the Plotkin structure [90] that the minimum

[2] We improve this bound in Theorem 4.2 for transmission over the BEC in a straightforward manner by not restricting the analysis to the position of the last frozen bit.

distance is 2^{n-r} [56, Section 13.3, Theorem 3]. This means RM codes maximize the minimum distance among the codes obtained by removing rows from $\boldsymbol{G}^{[n]}$ for given parameters N and K.

Although asymptotically unreliable under SC decoding [6, Section X], RM codes achieve the capacity of the BEC under MAP decoding [94]. Motivated by this result with their better distance properties as compared to the polar codes, there has been a significant research effort recently to approach their MAP performance when transmission is over BMSCs [95–98]. Note also that several variants of SC-based decoders are have been proposed to decode RM codes in the past [99–102]. However, they are rather inefficient, especially if $N \geq 256$, because of the suboptimal selection of the frozen set $\mathcal{F}$ for SC decoding.

3.4.7. Methods to Choose the Frozen Set

For a given channel W, constructing an (N, K) polar code requires one to find the indices that minimize the upper bound (3.37) on the BLEP of SC decoding. The design is unfortunately not universal, i.e., the polar code design differs depending on the channel quality. There are several ways to obtain the indices. Monte Carlo-based designs were proposed in [6, 99], and a density evolution-based approach for the BEC channel was given in [6]. In the case of the BEC(ϵ), the ensity evolution[3] recursion is [6, Proposition 6]

$$\epsilon^{2i-1}_{\boldsymbol{G}^{[n]}} = 2\epsilon^{i}_{\boldsymbol{G}^{[n-1]}} - (\epsilon^{i}_{\boldsymbol{G}^{[n-1]}})^2 \tag{3.53}$$

$$\epsilon^{2i}_{\boldsymbol{G}^{[n]}} = (\epsilon^{i}_{\boldsymbol{G}^{[n-1]}})^2 \tag{3.54}$$

where the recursion starts at $\epsilon^{1}_{\boldsymbol{G}^{[0]}} \triangleq \epsilon$. It is not hard to derive these relations in 3.4.1: for U_1 to be erased, it suffices to observe one erasure in the channel output Y_1^2; hence, we have $\epsilon^{1}_{\boldsymbol{G}^{[1]}} = 2\epsilon - \epsilon^2$. On the other hand, U_2 is erased if and only if $Y_1^2 = (?, ?)$, which happens with probability ϵ^2; hence, we have $\epsilon^{2}_{\boldsymbol{G}^{[1]}} = \epsilon^2$. Recursive application of these equations as for (3.45) and (3.46) results in (3.53) and (3.54), respectively. The underlying assumption for these equations is the symmetry of the synthesized channels [6, Proposition 13]. Under this assumption, the error events $\mathcal{B}_i^{(\mathrm{GA})}$, $i \in [N]$, are independent from the transmitted codeword [6, Corollary 1] and we can assume transmission of the all-zero codeword. We may compute $\epsilon^{i}_{\boldsymbol{G}^{[n]}}$, $i \in [N]$, by writing the n-digit binary expansion b_1^n of $i-1$. Each 0

[3] In the case of the BEC, it suffices to track the evolution of a single parameter, namely the erasure probability, rather than densities. However, as we shall soon see, one must track the evolution of densities for general BMSCs.

and 1 in the expansion corresponds to applying (3.53) and (3.54), respectively, with an input erasure probability from the previous recursion. Note that the recursion starts and continues in the bit-reversed order, i.e., starting from b_n to b_1.

Example 3.1. Consider $N = 4$ and $\epsilon^2_{\boldsymbol{G}^{[2]}}$ when the channel erasure probability is $\epsilon = 0.5$. We have $b_1^2 = (0, 1)$; hence, we apply first (3.53) to obtain $\epsilon^1_{\boldsymbol{G}^{[1]}} = 0.75$. (3.54) is then applied with an input erasure probability of 0.75 from the previous recursion, resulting in $\epsilon^2_{\boldsymbol{G}^{[2]}} = 0.5625$.

For general BMSCs, a density evolution-based construction is introduced in [103]. Instead of tracking two densities, we define $L^i_{\boldsymbol{G}^{[n]}}(y_1^N)$ as the logarithm of their ratio, i.e.,

$$L^i_{\boldsymbol{G}^{[n]}}(y_1^N) \triangleq \log \frac{P^i_{\boldsymbol{G}^{[n]}}(U_i = 0 | y_1^N, U_1^{i-1} = 0)}{P^i_{\boldsymbol{G}^{[n]}}(U_i = 1 | y_1^N, U_1^{i-1} = 0)}. \tag{3.55}$$

Accordingly, we use $l^i_{\boldsymbol{G}^{[n]}}$ to denote the PDF of the RV $L^i_{\boldsymbol{G}^{[n]}}(Y_1^N)$. Extending the equations (3.53) and (3.54) to general BMSCs, the densities can be computed recursively as

$$l^{2i-1}_{\boldsymbol{G}^{[n]}} = l^i_{\boldsymbol{G}^{[n-1]}} \boxtimes l^i_{\boldsymbol{G}^{[n-1]}} \tag{3.56}$$

$$l^{2i}_{\boldsymbol{G}^{[n]}} = l^i_{\boldsymbol{G}^{[n-1]}} \circledast l^i_{\boldsymbol{G}^{[n-1]}} \tag{3.57}$$

where $\circledast$ denotes the variable node convolution and $\boxtimes$ the check node convolution as defined in [54, Chapter 4]. The probabilities $\Pr\{\mathcal{B}_i\}$ can be computed via $l^i_{\boldsymbol{G}^{[n]}}$ with $i \in [N]$ as

$$\Pr\{\mathcal{B}_i\} = \lim_{z \to 0} \left(\int_{-\infty}^{-z} l^{(i)}_{\boldsymbol{G}^{[n]}}(x) dx + \frac{1}{2} \int_{-z}^{+z} l^{(i)}_{\boldsymbol{G}^{[n]}}(x) dx \right). \tag{3.58}$$

The computation of (3.56), (3.57) and (3.58) can be carried out, for instance, via quantized density evolution [104]. A more accurate implementation for quantized density evolution is provided in [105] together with an analysis providing lower and upper bounds for the reliabilities of the synthesized channels. A Gaussian approximation of the density evolution for polar codes was proposed in [106], which approximates the densities $l^i_{\boldsymbol{G}^{[n]}}$ as Gaussian RVs enabling to track a single parameter, e.g., the mean $\mu^i_{\boldsymbol{G}^{[n]}}$, rather than densities [107] as for the BEC. In this case, the densities are assumed to be distributed as $L^i_{\boldsymbol{G}^{[n]}}(Y_1^N) \sim \mathcal{N}(\mu^i_{\boldsymbol{G}^{[n]}}, 2\mu^i_{\boldsymbol{G}^{[n]}})$, where the mean values are computed recursively as

$$\mu^{2i-1}_{\boldsymbol{G}^{[n]}} = \phi^{-1}\left(1 - \left[1 - \phi\left(\mu^i_{\boldsymbol{G}^{[n-1]}}\right)\right]^2\right) \tag{3.59}$$

$$\mu^{2i}_{\boldsymbol{G}^{[n]}} = 2\mu^i_{\boldsymbol{G}^{[n-1]}} \tag{3.60}$$

where

$$\phi(x) = \begin{cases} 1 - \frac{1}{\sqrt{4\pi x}} \int_{-\infty}^{+\infty} \tanh\left(\frac{u}{2}\right) \exp\left(-\frac{(u-x)^2}{4x}\right) \mathrm{d}u & \text{if } x > 0 \\ 1 & \text{if } x = 0. \end{cases} \tag{3.61}$$

The approximations for the function $\phi(x)$ can be used, e.g., see [107], or the equations can be rewritten using the J-function [108], which admits a tight and easy-to-compute approximation [109]. Note that a Gaussian approximation of density evolution can be implemented with $\mathcal{O}(N \log_2 N)$ time complexity.

Methods based on a *partial order* among the positions were first proposed in [110], and later used and generalized in several other works, e.g. [93,111–113], to reduce the complexity of polar code constructions. The basic idea is that ordering among some positions are universal irrespective of the underlying channel type or quality. Facilitating such relations enables sublinear complexity for finding the set $\mathcal{A}$ minimizing the RHS of (3.37) [112]. In addition to the low-complexity construction, these methods allow one to design frozen bit sequences that show a good behavior for a wide range of channel parameters and rates. This has been of particular importance during 5G standardization [114] with its strong emphasis on lowering the description complexity. A particularly efficient and compact description is the *polarization weight (PW)* [111] construction. For a given index $i \in [N]$, let b_1^n be the binary representation of $i-1$, where b_1 is the most-significant bit and b_n is the least significant one. The PW of the i-th synthesized channel is defined for $\beta > 1$ as

$$w_\beta(i) \triangleq \sum_{j=1}^{n} b_j \beta^j. \tag{3.62}$$

In this case, $\mathcal{A}$ consists of the K indices with the largest PWs for an (N, K) polar code constructed via PW. It has been observed in [111] that choosing $\beta = 2^{1/4}$ provides a robust construction performing well for a wide range of channel qualities and code rates.

So far, the target was to construct a polar code that performs well under SC decoding, and with low complexity. These techniques do not necessarily lead to codes performing well under decoding algorithms that mimic MAP decoding, e.g., SCL decoding. The reason is obvious: the MAP decoding performance of a code is highly related to the code distance spectrum and the procedure for selecting $\mathcal{F}$ does not take this aspect into account.

Remark 3.7 (Minimum Distance)**.** For a given set $\mathcal{A}$, the minimum distance of the resulting code is equal to the Hamming weight of the row i, $i \in \mathcal{A}$, of $\boldsymbol{G}^{[n]}$ with the lowest weight [115, Lemma 3]. The proof follows from observing that any polar code can be represented as a linear subspace[4] of the RM code having that row with the lowest weight

[4]A linear subspace of a code $\mathcal{C}$ is called a *subcode* of $\mathcal{C}$.

as a codeword.

It is not surprising from the minimum distance point of view that RM codes perform better than polar codes under MAP decoding [14] in general. The authors of [13] proposed a hybrid code construction "between" polar and RM codes, which are called RM-polar codes. The idea is as follows: to design an (N, K) RM-polar code, indices are chosen to minimize the RHS of (3.37) by using the standard techniques above, among the indices corresponding to that of $\mathrm{RM}(r, n)$ where r is the smallest integer such that

$$1 + \binom{n}{1} + \binom{n}{2} + \cdots + \binom{n}{r} \geq K. \tag{3.63}$$

In this way, the resulting code is guaranteed to have a minimum distance of 2^{n-r} as it is a subcode of $\mathrm{RM}(r, n)$. This method provides better performance under SCL decoding with practical list sizes, e.g., $L = 32$, for some code parameters. For instance, $(2048, 1024)$ RM-polar codes with a minimum distance of 32 outperform $(2048, 1024)$ polar codes with a minimum distance of 16 by 1.2 dB at a BLER of 10^{-4} [13]. Although choosing $\mathcal{F}$ is an interesting and (still) open problem, especially for SCL decoding, this limits the degrees of freedom for code design. Next, we review methods to improve the distance spectrum of polar codes, and more generally of $\boldsymbol{G}^{[n]}$-coset codes, via code concatenation.

3.4.8. Methods to Improve the Distance Spectrum

In addition to the efficient implementation of SCL decoding, one of the major contributions of [11] was to introduce a practical solution to improve the distance spectrum of polar codes via code concatenation. Remarkably, the resulting codes under SCL decoding with considerably small list sizes, e.g., $L = 32$, outperform state-of-art codes for short- to moderate-lengths [36]. The scheme works as follows: A systematic encoder[5] of an outer code, preferably a CRC code, is used to generate ℓ-bits of redundancy using the message v_1^K. This leads to a vector $w_1^{K+\ell}$ that has v_1^K as then first K elements. The vector $w_1^{K+\ell}$ is then encoded via an $(N, K + \ell)$ inner polar code, where $u_{\mathcal{A}} = w_1^{K+\ell}$ and $u_{\mathcal{F}} = 0$, followed by transmission. The received vector is decoded via an SCL decoding with list size L of the inner code, which ignores the constraints imposed by the outer code. Among the candidates fulfilling the outer code constraints, the one with the maximum probability is output as the estimate. Depending on the list size L and the chosen outer code, the

[5] Although the proposed scheme of [11] works also with non-systematic encoders, this will enable us to explain a generalization of the idea towards the end of this section.

decoder may not be able to output a valid codeword. In this sense, the initial scheme of [11] proposed an incomplete decoder. Later, researchers proposed methods to optimize the outer code [18] and the interleaver before the inner polar code [116, 117].

There is a simple modification to convert SCL decoding into a complete one as follows. Let $a_1^{K+\ell}$ be the vector, which contains the elements of $\mathcal{A}$ in an ascending order. Assuming the outer encoder to be in the systematic form, SCL decoding works as before until the K-th information bit u_{a_K} of the inner polar code. Then, all the decoding paths are extended using the outer code constraints since the upcoming information bits $u_{a_{K+1}}^{a_{K+\ell}}$ of the inner polar code are parity bits imposed by the outer code. This requires the computation of the probabilities for L paths according to (3.49) although the decisions are dictated by the outer code. In this way, all of the resulting paths are valid codewords and the one maximizing the probability (3.49) for $i = N$ is chosen as the estimate. The authors of [16] proposed to use ℓ parity-check bits not only at the end, but also scattered between the information bits of the inner polar code with the modified SCL decoding described above. This is a generalization of the modified polar codes of [11] with a complete decoder.

A careful reader will observe that the role of a parity bit imposed by the outer code in the modified SCL decoding is the same as that of a frozen bit. The only difference is that the value of a frozen bit is always set to a predetermined value, e.g., to 0, while that of a parity bit u_i varies depending on the preceding (information) bits $u_{\mathcal{A}^{(i-1)}}$, where $\mathcal{A}^{(i-1)} = \mathcal{A} \cap [i-1]$. This leads to the idea of *dynamic* frozen bits, which was proposed in [15, 118]. A frozen bit u_i is called dynamic if its value depends on a subset of (information) bits u_1^{i-1} preceding it. This concept unifies the concatenated polar code approach to improve the distance spectrum of polar codes and also enables one to represent any linear code as a polar code with dynamic frozen bits. Recently, there has been a significant research effort into optimizing code constructions with dynamic frozen bits for near-optimum decoding [18, 20, 119–122].

In particular, PAC codes received some attention due to their compact representation as *pre-transformed* polar codes [119]. For the description of an (N, K) pre-transformed polar code, one defines an upper-triangular $N \times N$ matrix $\boldsymbol{T}$ in addition to set $\mathcal{A} \subset [N]$. The matrix $\boldsymbol{T}$ imposes the dynamic frozen bit constraints through the encoding process: the vector u_1^N is described as before, where $u_{\mathcal{A}}$ carries the information and $u_{\mathcal{F}} = 0$. Encoding is performed as

$$c_1^N = u_1^N \boldsymbol{T} \boldsymbol{G}^{[n]}. \tag{3.64}$$

Encoding efficiency is achieved in [119] by choosing $\boldsymbol{T}$ with an upper-triangular Toeplitz

structure as

$$\boldsymbol{T} = \begin{bmatrix} t_0 & t_1 & \dots & t_m & 0 & \dots & \dots & \dots & 0 \\ 0 & t_0 & t_1 & \ddots & t_m & 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & t_0 & \dots & \ddots & t_m & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 & t_0 & t_1 \\ 0 & \dots & \dots & \dots & \dots & \dots & \dots & 0 & t_0 \end{bmatrix}. \tag{3.65}$$

Now the challenge is how to choose $\mathcal{A}$ and $\boldsymbol{T}$ such that the resulting code performs well under decoding algorithms that mimic optimum decoding with practical complexity, e.g., SCL decoding with small to medium list sizes. In the rest of thesis, we use the name concatenated or *modified* polar codes for the codes generated by an arbitrary selection of the set $\mathcal{A}$ and an arbitrary selection of dynamic frozen constraints (or outer code).

3.4.9. A Generalization: Multi-Kernel Polar Codes

Arıkan already mentioned in [6, Section XI] that generalizations of polar codes are possible by choosing different binary kernels than $\boldsymbol{K}_2$ and those kernels can even be mixed. Conditions for polarizing kernels were provided in [48] and corresponding error exponents were derived. Note that the mixed-kernel polar constructions were first proposed in [123] demonstrating gains over standard polar codes in error-correcting capability for finite blocklengths. The paper [49] extended the error exponent derivation to the construction of mixing kernels, as suggersted in [6], while [124] provided examples of constructions using this approach, namely multi-kernel polar codes.

The channel polarization of multi-kernel polar codes follows from the same ingredients as given in Lemma 3.2 for the standard polar codes, where the proof follows by forming a martingale similar to the one of Theorem 3.3. In Chapter 6, we will review the proof of polarization for a special type of kernels used in the description of SPC product codes using the tools of multi-kernel polar codes.

When it comes to SC/SCL decoding of the multi-kernel polar codes, the same principle works as for Arıkan's polar code; however, the complexity depends on the kernels chosen. Usually, simple operations as in (3.45) and (3.46) are hard to provide for a given kernel and a marginalization over all the upcoming bits within a local kernel is needed, see, e.g., [125, Equation 2]. Even if they are provided, the operations must change at each layer since the kernels are not necessarily the same. Therefore, we will not provide a generic decoding algorithm here, but describe it for the kernels chosen for the codes of interest in Chapter 6. Further results on the theory and applications of (modified) polar

codes based on larger (and possibly arbitrary) binary kernels is given in, e.g., [126–130].

Remark 3.8. Stolte proposed to construct $\mathcal{A}$ to minimize the BLEP under SC decoding in [99, Chapter 6] instead of choosing the set $\mathcal{A}$ of an RM code. This construction is equivalent to that of polar codes. The target was to design codes with finite-length performance superior to the one of RM codes under SC decoding. However, Stolte did not show the capacity-achieving properties of his construction [6]. In addition, SCL decoding was proposed in [102] and [99, Chapter 5]. However, SCL decoding became practically relevant only after Tal and Vardy introduced the CRC-concatenated polar codes with a more efficient implementation [11].

The SC decoding algorithm was provided in [6] by using log-likelihood ratios (LLRs), as these are numerically more stable for hardware than APPs. Instead, [11] used log-likelihoods (LLs) for SCL decoding since comparing different decoding paths in the pruning stage is readily available upon sorting with respect to their LLs. Later, [131] introduced an LLR-based SCL decoding of polar codes by introducing a quantity called path metric (PM) to compare paths based on LLRs. This turned out to be more hardware-friendly [132] when combined with the min-sum approximation [87].

3.5. Appendices

3.5.1. Proof of Lemma 3.2

The identity (3.19) follows via

$$\begin{aligned} 2I(W) &= I(X_1^2; Y_1^2) && (3.66)\\ &= I(U_1^2; Y_1^2) && (3.67)\\ &= I(U_1; Y_1^2) + I(U_2; Y_1^2|U_1) && (3.68)\\ &= I(U_1; Y_1^2) + I(U_2; Y_1^2 U_1). && (3.69) \end{aligned}$$

Since $I(X_2; Y_2) = I(U_2; Y_2) \leq I(U_2; Y_1^2 U_1)$, we have

$$I(U_2; Y_1^2 U_1) \geq I(W) \geq I(U_1; Y_1^2). \tag{3.70}$$

By (2.50), there exists a unique $p \in [0, 1/2]$ such that

$$p = H_2^{-1}\left(H(W)\right). \tag{3.71}$$

Define $p_{X|Y=y} \triangleq H_2^{-1}\left(H(X|Y=y)\right)$ which we consider to be a realization of a RV denoted as $p_{X|Y}$ (with abuse of notation), where the source of randomness is Y. We write

$$\begin{aligned}
I(U_1;Y_1^2) &= 1 - \mathbb{E}\left[H_2(p_{U_1|Y_1,Y_2})\right] && (3.72)\\
&= 1 - \mathbb{E}\left[H_2(p_{X_1|Y_1} * p_{X_2|Y_2})\right] && (3.73)\\
&= 1 - \mathbb{E}\left[\mathbb{E}\left[H_2(p_{X_1|Y_1} * p_{X_2|Y_2})\right]|Y_2\right] && (3.74)\\
&= 1 - \mathbb{E}\left[\mathbb{E}\left[H_2\left(H_2^{-1}\left(H_2(p_{X_1|Y_1})\right) * p_{X_2|Y_2}\right)\right]|Y_2\right] && (3.75)\\
&\leq 1 - \mathbb{E}\left[H_2\left(H_2^{-1}\left(\mathbb{E}\left[H_2(p_{X_1|Y_1})\right]\right) * p_{X_2|Y_2}\right)\right] && (3.76)\\
&= 1 - \mathbb{E}\left[H_2\left(p * p_{X_2|Y_2}\right)\right] && (3.77)\\
&= 1 - \mathbb{E}\left[H_2\left(p * H_2^{-1}\left(H_2(p_{X_2|Y_2})\right)\right)\right] && (3.78)\\
&\leq 1 - H_2\left(p * p\right) && (3.79)
\end{aligned}$$

with equality if and only if $p \in \{0, 1/2\}$, i.e., $H(W) \in \{0,1\}$. The first step (3.72) follows by (2.52) by defining $p_{U_1|Y_1^2=y_1^2} \triangleq H_2^{-1}\left(H(X|Y=y)\right)$, the step (3.73) because Y_1 and Y_2 are independent and by writing $a*b \triangleq ab+(1-a)(1-b)$, (3.74) by reordering the expectation, (3.74), (3.75), (3.77) and (3.78) from (3.71). Finally, (3.76) and (3.79) follow from Jensen's inequality using the convexity[6] of $H_2\left(H_2^{-1}(x) * p\right)$ in $x \in [0,1]$ for arbitrary $p \in [0, 1/2]$.

We combine (3.71) with (3.19), and use (3.79) to obtain (3.20) with equality iff $p \in \{0, 1/2\}$, i.e., $1 - I(W) = H(W) \in \{0,1\}$. ■

3.5.2. Proof of Theorem 3.3

We map the synthesized channel index $i \in [N]$ onto the n-bit binary representation b_1^n of $i-1$ and write, with slight abuse of notation, $I\left(W_{\boldsymbol{G}^{[n]}}^{(b_1^n)}\right) = I\left(W_{\boldsymbol{G}^{[n]}}^{i}\right)$. Define a random process

$$I_{\boldsymbol{G}^{[n]}} \triangleq I\left(W_{\boldsymbol{G}^{[n]}}^{(B_1^n)}\right) \tag{3.80}$$

where $I_{\boldsymbol{G}^{[0]}} \triangleq I\left(W\right)$ and the RVs B_i, $i = 1, 2, \ldots$, are i.i.d. with $P_{B_i}(0) = P_{B_i}(1) = 1/2$. Note that

$$\begin{aligned}
&I_{\boldsymbol{G}^{[n]}} \text{ is a function of } B_1^n && (3.81)\\
&\mathbb{E}[|I_{\boldsymbol{G}^{[n]}}|] < \infty && (3.82)\\
&\mathbb{E}[I_{\boldsymbol{G}^{[n+1]}}|B_1^n] = I_{\boldsymbol{G}^{[n]}} && (3.83)
\end{aligned}$$

[6] This result is known as Mrs. Gerber's Lemma [133].

where (3.81) follows from (3.80), (3.82) because $0 \leq I_{\boldsymbol{G}^{[n]}} \leq 1$ and, finally, (3.83) by writing

$$\mathbb{E}[I_{\boldsymbol{G}^{[n+1]}}|B_1^n] = \sum_{b_{n+1}\in\mathbb{F}_2} P(b_{n+1}) I\left(W_{\boldsymbol{G}^{[n+1]}}^{(B_1^n,b_{n+1})}\right) \tag{3.84}$$

$$= I_{\boldsymbol{G}^{[n]}} \tag{3.85}$$

where the last equality follows from (3.19) by replacing W by $W_{\boldsymbol{G}^{[n]}}^{(B_1^n)}$. From Definition 2.1, we conclude that the sequence $I_{\boldsymbol{G}^{[0]}}, I_{\boldsymbol{G}^{[1]}}, \ldots$ forms a martingale that converges a.s. to a RV $I_{\boldsymbol{G}^{[\infty]}}$. Hence, (2.45) implies

$$\mathbb{E}[|I_{\boldsymbol{G}^{[n+1]}} - I_{\boldsymbol{G}^{[n]}}|] = \frac{1}{2}\left(\mathbb{E}\left[I_{\boldsymbol{G}^{[n]}} - I\left(W_{\boldsymbol{G}^{[n+1]}}^{(B_1^n,0)}\right)\right] + \mathbb{E}\left[I\left(W_{\boldsymbol{G}^{[n+1]}}^{(B_1^n,1)}\right) - I_{\boldsymbol{G}^{[n]}}\right]\right) \to 0 \tag{3.86}$$

as $n \to \infty$. Now, replace W by $W_{\boldsymbol{G}^{[n]}}^{(B_1^n)}$ in (3.20) and observe (3.86). This quantity is non-negative and equal to 0 if and only if $I_{\boldsymbol{G}^{[\infty]}} \in \{0,1\}$. Using (2.46), we have $\mathbb{E}[I_{\boldsymbol{G}^{[\infty]}}] = I(W)$ which implies

$$\Pr\{I_{\boldsymbol{G}^{[\infty]}} = 1\} = 1 - \Pr\{I_{\boldsymbol{G}^{[\infty]}} = 0\} = I(W). \tag{3.87}$$

■

3.5.3. Proof of Lemma 3.4

Let $\mathcal{T} \triangleq \{(u_1^N, y_1^N) \in \mathbb{F}_2^N \times \mathcal{Y}^N : U_{\mathcal{F}_\delta} = u_{\mathcal{F}_\delta}\}$ be the sample space. Define the event of having the first bit-error at the i-th bit (information bit) under SC decoding as

$$\mathcal{B}_i^{(\mathrm{SC})} \triangleq \{(u_1^N, y_1^N) \in \mathcal{T} : \hat{u}_1^{i-1}(u_1^N, y_1^N) = u_1^{i-1}, f_i(y_1^N, \hat{u}_1^{i-1}(u_1^N, y_1^N)) \neq u_i\}. \tag{3.88}$$

The sets $\mathcal{B}_i^{(\mathrm{SC})}$ are disjoint for $i = 1, \ldots, N$ and we have $\mathcal{E} = \bigcup_{i=1}^{N} \mathcal{B}_i^{(\mathrm{SC})}$. On the other hand, the bit-error event for the same bit u_i under a genie-aided SC decoder is

$$\mathcal{B}_i^{(\mathrm{GA})} \triangleq \{(u_1^N, y_1^N) \in \mathcal{T} : f_i(y_1^N, u_1^{i-1}) \neq u_i\}. \tag{3.89}$$

Observe that $\mathcal{B}_i^{(\mathrm{SC})} \subseteq \mathcal{B}_i^{(\mathrm{GA})}$.[7]

[7] This is what is used in [6, Section V.B] to conclude $P_B^{(\mathrm{SC})} \leq P_B^{(\mathrm{GA})}$, which is enough to upper bound $P_B^{(\mathrm{SC})}$.

In the following, we show that

$$\bigcup_{i=1}^{N} \mathcal{B}_i^{(\mathrm{SC})} = \bigcup_{i=1}^{N} \mathcal{B}_i^{(\mathrm{GA})} \tag{3.90}$$

which concludes the proof. We use induction. Observe that $\mathcal{B}_1^{(\mathrm{SC})} = \mathcal{B}_1^{(\mathrm{GA})}$ and assume further that, for some $\ell \in [N]$, we have

$$\bigcup_{i=1}^{\ell} \mathcal{B}_i^{(\mathrm{SC})} = \bigcup_{i=1}^{\ell} \mathcal{B}_i^{(\mathrm{GA})}. \tag{3.91}$$

Since the sets $\mathcal{B}_i^{(\mathrm{SC})}$ are disjoint, it suffices to show $\mathcal{B}_{\ell+1}^{(\mathrm{SC})} = \mathcal{B}_{\ell+1}^{(\mathrm{GA})} \setminus \bigcup_{i=1}^{\ell} \mathcal{B}_i^{(\mathrm{GA})}$. To this end, we write

$$\begin{aligned}
\mathcal{B}_{\ell+1}^{(\mathrm{GA})} \setminus \bigcup_{i=1}^{\ell} \mathcal{B}_{\ell+1}^{(\mathrm{GA})} &= \mathcal{B}_{\ell+1}^{(\mathrm{GA})} \setminus \bigcup_{i=1}^{\ell} \mathcal{B}_i^{(\mathrm{SC})} && (3.92)\\
&= \mathcal{B}_{\ell+1}^{(\mathrm{GA})} \setminus \{(u_1^N, y_1^N) \in \mathcal{T} : (f_1(y_1^N, u_1^0), \ldots, f_\ell(y_1^N, u_1^{\ell-1})) \neq u_1^\ell\} && (3.93)\\
&= \mathcal{B}_{\ell+1}^{(\mathrm{SC})} && (3.94)
\end{aligned}$$

where (3.92) follows from the induction hypothesis (3.91), (3.93) from the unions starting from $i = 1$ to $i = \ell$, e.g., the first union is

$$\mathcal{B}_1^{(\mathrm{SC})} \cup \mathcal{B}_2^{(\mathrm{SC})} = \{(u_1^N, y_1^N) \in \mathcal{T} : (f_1(y_1^N, u_1^0), f_2(y_1^N, u_1^1)) \neq u_1^2\}. \tag{3.95}$$

Finally, we have (3.94) by combining the definitions of $\mathcal{B}_{\ell+1}^{(\mathrm{SC})}$ and $\mathcal{B}_{\ell+1}^{(\mathrm{GA})}$. ■

4

Successive Cancellation List Decoding over the BEC

The BEC is often used to model data networks where packets arrive either intact or are lost due to network congestion or detected errors. We first revisit block-wise MAP decoding of linear codes over the BEC and introduce code ensembles based on polar transform. Then, the chapter studies SCL decoding of an arbitrary binary linear block code when transmission is over the BEC. Thanks to the channel model, we then propose SCI decoding as an efficient alternative for SCL decoding, which is equivalent to MAP decoding when there is no complexity constraint. We rely on the average number of inactivations to analyze the complexity of various code classes. Numerical results show that dRM codes (one of the introduced ensembles) perform close to the Singleton bound for blocklengths for various blocklengths. Although the channel model is quite simple, the insights helps us to generalize some of the ideas to more general BMSCs in the next chapter.

4.1. Linear Codes under Block-wise MAP Decoding

Consider an (N, K) binary linear code $\mathcal{C}$ with generator matrix $\boldsymbol{G}$. Suppose the channel erases $\mathsf{e} \in [N]$ symbols of the transmitted vector x_1^N, yielding the output y_1^N. Let $\mathcal{E}$ and $\bar{\mathcal{E}}$ denote the sets of indices of the erased and non-erased entries, respectively, so that $|\mathcal{E}| = N - |\bar{\mathcal{E}}| = \mathsf{e}$. The vector $c_{\bar{\mathcal{E}}}$ collects the non-erased values of c_1^N while the matrix $\boldsymbol{G}_{:,\bar{\mathcal{E}}}$ collects the columns of $\boldsymbol{G}$ corresponding to $\bar{\mathcal{E}}$. The following equation thus has at

least one solution

$$v_1^K \boldsymbol{G}_{:,\bar{\mathcal{E}}} = c_{\bar{\mathcal{E}}}. \tag{4.1}$$

The solution(s) can be found with $\mathcal{O}(N^3)$ complexity using Gaussian elimination [134, Section 1.3]. Define $\rho \triangleq \operatorname{rank} \boldsymbol{G}_{:,\bar{\mathcal{E}}}$ and observe that the solutions of (4.1) form an affine subspace, denoted by $\mathcal{L}(y_1^N)$, where $|\mathcal{L}(y_1^N)| = 2^{K-\rho}$. If $\rho = K$ then (4.1) has a unique solution that is output as the MAP estimate. Otherwise, under the assumption that the a-priori distribution of the messages is uniform, each message in $\mathcal{L}(y_1^N)$ has the same likelihood. A MAP decoder would hence pick a message at random in $\mathcal{L}(y_1^N)$. Alternatively, when multiple solutions exist, we may declare a decoding failure. With slight abuse of notation, we will still refer to the latter approach as block-wise MAP decoding.

Remark 4.1. An alternative approach to MAP decoding, which makes use of the code parity-check matrix, works as follows [54, Section 3.2]. We group x_1^N into subvectors $x_{\mathcal{E}}$ and $x_{\bar{\mathcal{E}}}$. Similarly, the columns of parity-check matrix $\boldsymbol{H}$ of $\mathcal{C}$ are grouped to obtain $\boldsymbol{H}_{:,\mathcal{E}}$ and $\boldsymbol{H}_{:,\bar{\mathcal{E}}}$. Since $x_1^N \boldsymbol{H} = 0$ by definition, we write the linear system

$$x_{\mathcal{E}} \boldsymbol{H}_{:,\mathcal{E}}^T = x_{\bar{\mathcal{E}}} \boldsymbol{H}_{:,\bar{\mathcal{E}}}^T \tag{4.2}$$

where the RHS is known. Again, we declare a failure if the system does not provide a unique solution, i.e., if $\operatorname{rank} \boldsymbol{H}_{:,\mathcal{E}} < \mathsf{e}$. In terms of complexity, it may make sense to consider a generator matrix if $R < 1/2$ and a parity-check matrix otherwise.

4.1.1. The Singleton Bound

Consider transmission over a BEC(ϵ) with an (N, K) binary linear block code. A necessary condition for having a unique solution for (4.1) is to have a number of equations (i.e., a number of non-erased bits) at least as large as the number of unknowns K, that is, to have $N - \mathsf{e} \geq K$. By exploiting this observation, we can conclude that a lower bound on the BLEP achievable by any (N, K) binary linear block code over a BEC(ϵ) is

$$P_B \geq \sum_{i=N-K+1}^{N} \binom{N}{i} \epsilon^i (1-\epsilon)^{N-i} \tag{4.3}$$

where each summand corresponds the probability of having i erasures when x_1^N is transmitted. The bound (4.3) is commonly referred to as the Singleton bound, and it is achievable only by MDS codes. Recall that the only binary linear block codes that are MDS are $(N, 1)$ repetition codes and $(N, N-1)$ SPC codes. Besides these two binary linear code

classes, no other binary linear block code fulfills (4.3) with equality.

4.1.2. Berlekamp's Random Coding Bound

A tight upper bound on the average BLEP of a random binary linear code is known as *Berlekamp's random coding (BRC) bound* [135]. BRC bound can be used to prove the existence of binary linear block codes with a BLEP smaller than the average BLEP of random binary linear codes, as illustrated by the following theorem.

Theorem 4.1. For transmission over the BEC(ϵ), there exists an (N, K) binary linear code whose BLEP under MAP decoding satisfies

$$P_B < \sum_{i=N-K+1}^{N} \binom{N}{i} \epsilon^i (1-\epsilon)^{N-i} + \sum_{i=1}^{N-K} \binom{N}{i} \epsilon^i (1-\epsilon)^{N-i} 2^{-(N-K-i)}. \tag{4.4}$$

The RHS of (4.4) is composed of two terms, where the first term is the Singleton bound and the second term may be seen as the penalty paid by random linear codes with respect to MDS codes. A proof can be found in Appendix 4.5.1.

4.1.3. Linear Code Ensembles based on Polar Transforms

We introduce ensembles of modified polar codes with random choices of dynamic frozen bits after fixing the information set $\mathcal{A}$.

Definition 4.1 (Dynamic RM ensemble)**.** The *dRM* ensemble with parameters (r, m), denoted by $\mathcal{DRM}(r, m)$, is the set of all codes, specified by set $\mathcal{A}$ of the RM(r, m) code and choosing

$$u_i = \begin{cases} \sum_{j \in \mathcal{A}^{(i-1)}} v_{j,i} u_j & \text{if } \mathcal{A}^{(i-1)} \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

for $i \in \mathcal{F}$ with all possible $v_{j,i} \in \{0, 1\}$ and $\mathcal{A}^{(0)} \triangleq \emptyset$.

Recently, Arıkan introduced polarization-adjusted convolutional (PAC) codes [119] that can be represented as a polar code with dynamic frozen bits [136, 137]. PAC code rate-profiling is reflected in the frozen index set of its polar code representation [136]. Thus, if $\mathcal{A}$ of an RM(r, m) code is chosen via rate-profiling [119] then the corresponding PAC code becomes an instance from $\mathcal{DRM}(r, m)$. Another instance is the RM(r, m) code.

Definition 4.2 (Uniform generator ensemble based on polar transforms)**.** The uniform generator ensemble $\mathcal{G}(N, K)$ based on polar transform with parameters $N = 2^n$ and K is the set of polar codes specified by $\mathcal{A} = [K]$ and by choosing, for $i \in \mathcal{F}$,

$$u_i = \begin{cases} \sum_{j \in \mathcal{A}^{(i-1)}} v_{j,i} u_j & \text{if } \mathcal{A}^{(i-1)} \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

with all possible $v_{j,i} \in \{0, 1\}$ and $\mathcal{A}^{(0)} \triangleq \emptyset$.

4.2. SC and SCL Decoding

SC decoding works as in Section 3.4.4 with a slight modification in the decision function (3.44) to declare an error whenever an erasure is output for an information bit. We rewrite (3.44) for the BEC as

$$f_i(y_1^N, \hat{u}_1^{i-1}) \triangleq \begin{cases} 0 \text{ if } P^i_{\boldsymbol{G}^{[n]}}(0|y_1^N, \hat{u}_1^{i-1}) = 1 \\ ? \text{ if } P^i_{\boldsymbol{G}^{[n]}}(0|y_1^N, \hat{u}_1^{i-1}) = \frac{1}{2} \\ 1 \text{ otherwise} \end{cases} \tag{4.7}$$

by observing that the probabilities $P^i_{\boldsymbol{G}^{[n]}}(u_i|y_1^N, \hat{u}_1^{i-1})$, $u_i \in \mathbb{F}_2$, take values in a ternary alphabet $\{0, 1/2, 1\}$. The decision unit (3.43) remains the same, namely:

$$\hat{u}_i = \begin{cases} u_i & \text{if } i \in \mathcal{F} \\ f_i(y_1^N, \hat{u}_1^{i-1}) & \text{if } i \in \mathcal{A}. \end{cases} \tag{4.8}$$

The decoding process aborts with a frame error if $\hat{u}_i =?$ for any $i \in \mathcal{A}$.

SCL decoding uses the same principles as in Section 3.4.5, where the difference is that a BEC path is duplicated only if (4.7) provides an erasure for an information bit, i.e., for u_i with $i \in \mathcal{A}$ [138, Section V]. This is called a *branching event* [139, Appendix A] following the approach of [140]. Otherwise, the decoder follows the value $u_i \in \mathbb{F}_2$ for which $P^i_{\boldsymbol{G}^{[n]}}(u_i|y_1^N, \hat{u}_1^{i-1}) = 1$. When the algorithm encounters a frozen bit u_i, a path is pruned if it is not valid [138, Section V] and

$$P^i_{\boldsymbol{G}^{[n]}}(u_i|y_1^N, \hat{u}_1^{i-1}) = 0. \tag{4.9}$$

In this case, half of the existing paths are be pruned.[1] Suppose that the SCL decoding list size is L. We declare a frame error if, at any stage during the decoding process, the number of active paths exceeds L or if more than one path is active at the end of the process. SCL decoding with $L = 2^{NR}$ implements MAP decoding, i.e., it outputs all the solutions for (4.1) in the final list $\mathcal{L}$.

Remark 4.2 (MAP decoding for non-uniform inputs)**.** For a non-uniform distribution of the message bits $u_{\mathcal{A}}$, SCL decoding is modified as follows to implement MAP decoding. Output the final list as before using standard SCL decoding, i.e., list all possible solutions $\mathcal{L}$ of (4.1). Now choose the estimate that maximizes the a-priori probability in $\mathcal{L}$. Suppose without loss of generality that $P_{v_i}(0) = 1 - P_{v_i}(1) = p$ with $p > 1/2$. Then the MAP estimate is

$$\hat{v}_1^K = \underset{v_1^K \in \mathcal{L}}{\operatorname{argmin}} \, w_{\mathrm{H}}(v_1^K). \tag{4.10}$$

The question of "how large should $\mathcal{L}$ be for MAP decoding?" was addressed by [89, Theorem 1] for the case of the BEC, as reviewed in Section 3.4.5. We improve this bound with the following lemma that follows from a simple observation in Appendix 4.5.2.

Lemma 4.2 (Upper bound on the required list size for MAP decoding)**.** Let $L^*(\mathcal{C})$ be the smallest list size for an SCL decoding that implements a MAP decoder for an (N, K) binary linear code $\mathcal{C}$. Let ζ be the index of the last frozen bit before the first information bit, and let γ be the last (dynamic) frozen bit when $\mathcal{C}$ is represented as a polar code (with dynamic frozen bits), i.e., we have

$$\zeta \triangleq \min \mathcal{A} \quad \text{and} \quad \gamma \triangleq \max \mathcal{F}. \tag{4.11}$$

Then we have

$$L^*(\mathcal{C}) \leq \min \left\{ 2^{N(1-R)-(\zeta-1)}, 2^{\gamma - N(1-R)} \right\}. \tag{4.12}$$

Note that [89, Theorem 1] states that $L^*(\mathcal{C}) \leq 2^{\gamma - N(1-R)}$. Depending on the allocation of the frozen bit indices, (4.12) can improve the previous result significantly. Consider, for instance, the RM$(5, 7)$ code with parameters $N = 128$ and $K = 120$ where $\zeta = 4$ and $\gamma = 65$. The previous result states that $L^* \leq 2^{57}$ while (4.12) gives $L^* \leq 2^5$. The previous result is usually relevant for low rate codes, and simulations show its usefulness even for transmission over general BMSCs [89, Figure 15].[2] Lemma 4.2 tightens the bound for

[1] Although it might be difficult to see this here, it will be obvious after Proposition 4.3.

[2] The result does not hold for general BMSCs unless the SCL decoding is modified as suggested by [88], see Section 3.4.5.

high-rate codes. However, even this bound is far from being practical, especially for codes with rates $R \approx 0.5$. For instance, for the RM$(3,7)$ code the bound (4.12) is $L^* \leq 2^{49}$. In addition, the bounds are obviously independent of the channel quality since we are interested in exact MAP decoding. If ϵ is very low then one requires much shorter lists on average to decode successfully.

Recall that SCL decoding branches out the paths not for each information bit but whenever necessary. Therefore, we study the dynamics of SCL decoding with unbounded list size. This relaxation gives more understanding on the complexity vs. performance trade-offs on average. First, we provide a proposition that points towards a more efficient implementation of SCL decoding.

Proposition 4.3. Recall from (2.2) that $\mathcal{A}^{(i)} \triangleq \mathcal{A} \cap [i]$. On the BEC, the list of all valid partial input sequences u_1^m generated by SCL decoding with unbounded list size upon observing y_1^N form an affine subspace, denoted as $\mathcal{S}^{(i)}\left(y_1^N\right)$.

The proof is given as Appendix 4.5.3. Let $L_i\left(y_1^N\right)$ denote the list length after i-th decoding stage of SCL decoding with unbounded list size, i.e., $L_i\left(y_1^N\right) = \left|\mathcal{S}^{(i)}\left(y_1^N\right)\right|$. The proposition provides an immediate corollary, which bridges between the introduced quantity $d_i\left(y_1^N\right)$ and the list length $L_i\left(y_1^N\right)$ of SCL decoding with unbounded list size explicitly.

Corollary 4.4. At any decoding stage i, the list length $L_i\left(y_1^N\right)$, $y_1^N \in \{0,?,1\}^N$, of SCL decoding satisfies

$$\log_2 L_i\left(y_1^N\right) = d_i\left(y_1^N\right). \tag{4.13}$$

Hence, $L_i\left(y_1^N\right)$ is a non-negative integer power of 2.

Next, another corollary is provided, which follows from the fact that $L_i\left(y_1^N\right)$ is a non-negative integer power of 2 in combination with the error events of SCL decoding with a fixed list size L defined in Section 4.2, i.e., either $\left|\mathcal{S}^{(i)}\left(y_1^N\right)\right| > L$ for any $i \in [N]$ or $\left|\mathcal{S}^{(i)}\left(y_1^N\right)\right| \neq 1$.

Corollary 4.5. SCL decoding with list size L has the same performance as SCL decoding with $L' = 2^{\lfloor \log_2 L \rfloor}$.

A more interesting outcome of Proposition 4.3 is the new decoding algorithm, namely SCI decoding, which is introduced next. In essence, SCL decoding lists all possible sequences in an affine subspace of $\{0,1\}^{|\mathcal{A}^{(i)}|}$ after i decoding steps while SCI decoding stores a basis for this vector subspace.

4.3. Succesive Cancellation Inactivation Decoding

Suppose u_1^N is encoded and transmitted over the BEC(ϵ) with channel output y_1^N. Consider the case where (4.7) provides an erasure for an information bit, i.e., $\hat{u}_i = ?$, $i \in \mathcal{A}$. Instead of duplicating the path as in SCL decoding, the SCI decoder introduces a dummy variable $\tilde{u}_i$ and stores the decision as $\hat{u}_i = \tilde{u}_i$. This is called an *inactivation* event. It continues decoding with the next stages using the same schedule as for SC decoding. Now (4.7) is allowed to be a function of the previous inactivated variable. For example, if there is no other inactivation for the information bits in between, then it can output either an erasure or a linear combination of $\tilde{u}_0 \triangleq 1$ and the previous variable $\tilde{u}_i$, i.e., $a_0 \oplus a_i\tilde{u}_i$ with $a_{\{0,i\}} \in \{0,1\}^2$, for all bits u_j with $j > i$. We separate the cases where (i) u_j is an information bit and (ii) u_j is a frozen bit. In case (i), if the function (4.7) outputs an erasure, then the decoder inactivates another bit, namely $\hat{u}_j = \tilde{u}_j$. Otherwise, it continues by decoding the next bit knowing that $\hat{u}_j = a_0 \oplus a_i\tilde{u}_i$. In case (ii), if the decoder outputs an erasure or has a trivial combination, i.e., $a_{\{0,i\}} = 0$, then it sets $\hat{u}_j = 0$ and continues with the next bit. However, if it outputs a combination where $a_i = 1$, then it learns the value of the previously inactivated bit as $\tilde{u}_i = a_0$. The SCI decoder stores the equation separately and continues decoding with $\hat{u}_j = a_0 \oplus a_i\tilde{u}_i$. In the following, we initially do not impose constraints on the number of inactivations during the decoding process.

In general, the decoder can have g inactivations for the information bits $u_{\mathcal{I}\setminus\{0\}}$ with $\mathcal{G} \triangleq \{0, i_1, i_2, \ldots, i_g\}$, $0 < i_1 < i_2 < \cdots < i_g < i$ before decoding u_i, i.e., $\hat{u}_{\mathcal{G}\setminus\{0\}} = \tilde{u}_{\mathcal{G}\setminus\{0\}}$. For some binary vector $a_{\mathcal{G}}$, the function f_i is rewritten as

$$f_i(y_1^N, \hat{u}_1^{i-1}) \triangleq \begin{cases} a_{\mathcal{G}} \cdot \tilde{u}_{\mathcal{G}} \text{ if } P^i_{\boldsymbol{G}^{[n]}}(a_{\mathcal{G}} \cdot \tilde{u}_{\mathcal{G}} | y_1^N, \hat{u}_1^{i-1}) = 1 \\ ? \qquad \text{otherwise.} \end{cases} \tag{4.14}$$

Suppose that the decoder inactivates g bits in total during a decoding attempt. Then the final step of SCI decoding is to solve a system of linear equations in g unknowns. This will have a unique solution only if the equations obtained from the frozen bits have rank g. This algorithm is equivalent to an SCL decoder over the BEC with unbounded list size, and it thus implements MAP decoding. Next, we study the average behaviour of the SCI decoding with an unbounded number of inactivations.

4.3.1. Average Number of Inactivations for MAP Decoding

Let $\hat{u}_1^{i-1}$ denote the output of SC inactivation decoding with possible inactivations before estimating u_i.

Lemma 4.6. $f_i(y_1^N, u_1^{i-1}) = ?$ if and only if $f_i(y_1^N, \hat{u}_1^{i-1}) = ?$.

The proof is provided in Appendix 4.5.4. Recall now that density evolution computes the probabilities $\epsilon^i_{\boldsymbol{G}^{[n]}}$ exactly via (3.53) and (3.54), i.e., the erasure probabilities of the genie-aided SC decoder. The following lemma, whose proof is left for Appendix 4.5.5, characterizes the exact inactivation probabilities for u_i, $i \in [N]$.

Lemma 4.7. Let b_i be the probability of having an inactivation for u_i in the SC inactivation decoder. Then we have

$$b_i = \begin{cases} 0 & \text{if } i \in \mathcal{F} \\ \epsilon^i_{\boldsymbol{G}^{[n]}} & \text{otherwise.} \end{cases}$$

Corollary 4.8. Let $G(Y_1^N)$ be a RV equal to the total number of inactivations made by the decoder during a decoding attempt. Then we have

$$\mathbb{E}[G] = \sum_{i \in \mathcal{A}} \epsilon^i_{\boldsymbol{G}^{[n]}}. \tag{4.15}$$

Proof. The result follows from the expectation of inactivation indicator events and Lemma 4.7. ■

Corollary 4.8 describes the average number of inactivations required for MAP performance with a code defined by $\mathcal{A}$ when the transmission is over the BEC(ϵ), providing the expected number of unknowns for the resulting linear system.[3]

Remark 4.3. The performance improvement under MAP decoding when interpolating from polar to RM codes is driven by the weight spectrum improvement, e.g., the minimum distance increases [13, 14]. This comes at the cost of a higher MAP decoding complexity. The quantity $\mathbb{E}[G]$ is obtained from analyzing the inactivation decoder and this allows us to quantify this complexity increase. Another way to improve the distance spectrum is to embed dynamic frozen bits [15]. Although the number of inactivated bits remains unaffected by the use of dynamic frozen bits, they might add extra complexity to the decoder, especially for hardware implementations. Our analysis ignores this extra complexity, yet according to the simulations, it is still accurate since the number of inactivations is the dominating factor in the complexity.

[3] This relation was first observed in [139, Appendix A] for SCL decoding with unbounded list size, where an inactivation event is replaced by a branching event as an SCL decoder *branches* paths if it encounters an erasure.

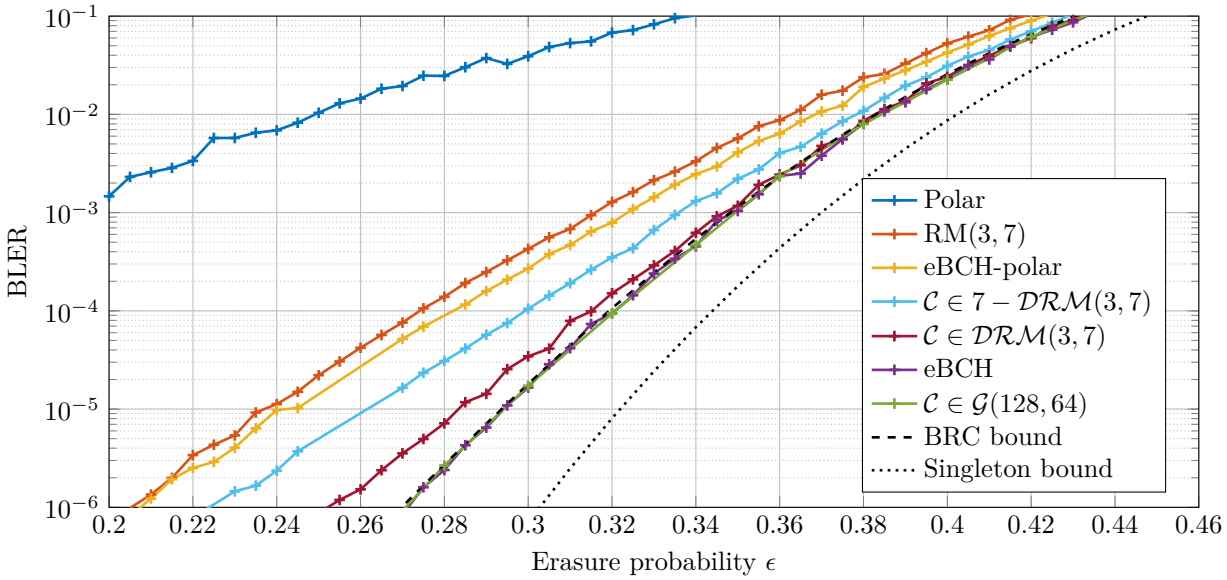

Figure 4.1.: BLER vs. ϵ for $(128, 64)$ codes.

4.3.2. Numerical Results

In this section, we consider (i) polar codes, (ii) RM codes, (iii) eBCH-polar subcodes (with 7 dynamic frozen bits) of [18] as well as (iv) codes from the introduced ensemble $\mathcal{DRM}(r, m)$. Numerical results are provided for rate $R = 1/2$ codes of length $N \in \{128, 512\}$ using a MAP decoding implemented via the SCI decoder. Note that the polar codes are designed for the erasure probability $\epsilon = 0.4$ via density evolution. The Singleton (4.3) and the BRC (4.4) bounds are provided as benchmarks.

In Figures 4.1 and 4.2, the BLERs are shown for length 128 and 512 codes, respectively. For any length, RM codes outperform polar codes. The $(128, 64)$ eBCH code performs very close to an instance from the $\mathcal{G}(128, 64)$ ensemble. The code is sampled from $\mathcal{G}(128, 64)$ with the idea explained in Definition 4.2. The BLER of the $(128, 64)$ eBCH-polar subcode of [18] is also provided as a reference and it performs slightly better than RM$(3, 7)$.

In the figures, the BLERs for two variants of RM codes with dynamic frozen bits are provided. Codes are sampled randomly from the ensemble $\mathcal{DRM}(r, n)$ as given in Definition 4.1. For $N = 128$, their performance is close to that of the eBCH code. The second variant, denoted as $7 - \mathcal{DRM}(3, 7)$, is constructed by declaring all but the last 7 frozen bits as static. The dynamic frozen bits are set to random linear combinations of the first 10 information bits since they are more likely to be erased. This code performs within an erasure probability gap of 0.04 from the eBCH code at a BLER $\approx 10^{-6}$, which outperforms the eBCH-polar subcode with 7 dynamic frozen bits also. For $N = 512$, the flattening in

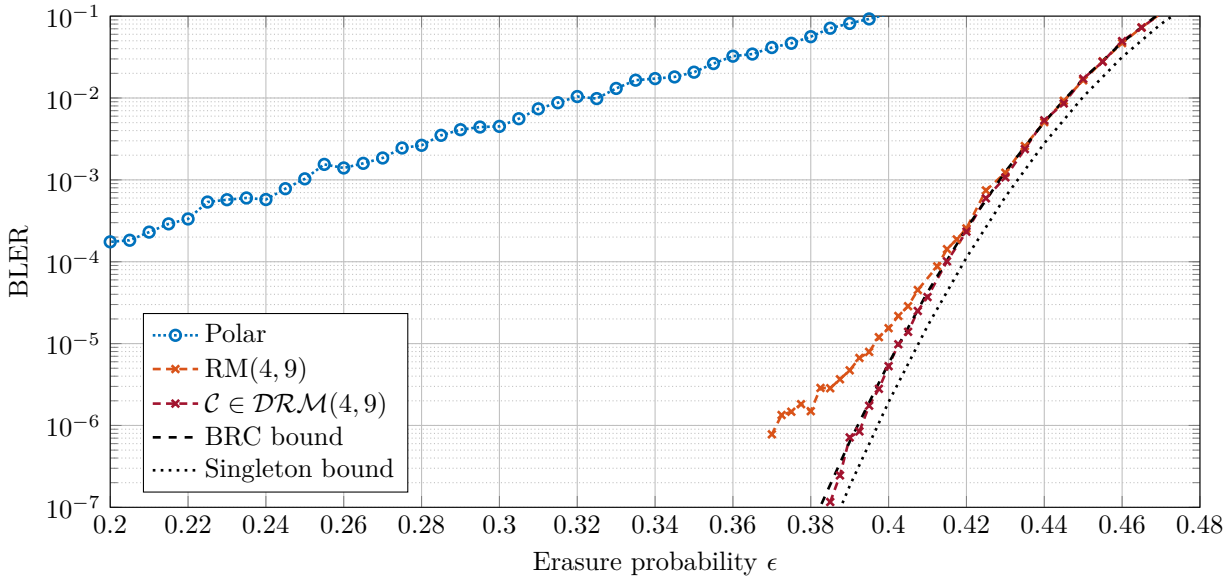

Figure 4.2.: BLER vs. ϵ for $(512, 256)$ codes.

the curve of the RM code at BLER $\approx 10^{-5}$ is avoided by the instances of $\mathcal{DRM}(4, 9)$, performing close to the Singleton bound down to a BLER $\approx 10^{-7}$.

In Figure 4.3, the expected numbers of inactivations $\mathbb{E}[G(Y_1^N)]$ from Corollary 4.8 are provided together with the results obtained from simulations for $N = 128$, demonstrating that the analysis is exact. MAP decoding of eBCH codes requires many more inactivations compared to the others, which results in a higher average decoding complexity. For this blocklength, surprisingly, $\mathbb{E}[G_{\text{RM}}]$ is close to $\mathbb{E}[G_{\text{eBCH}-\text{pol}}]$, where the eBCH-polar subcode has a (small) additional complexity due to dynamic frozen bits. Introducing the dynamic frozen bits does not affect the number of inactivations; hence, the complexity stays almost the same, but it improves the performance significantly.

SCI Decoding with Maximum Number of Inactivations

To limit the worst-case complexity, SCI decoding is extended by imposing a maximum number I of inactivations. An error is declared if the number of required inactivations exceeds I or there is no unique solution at the end. The following corollary is a counterpart of Theorem 4.2 for the SCI decoding with inactivation size I, where the proof follows by observing the equivalence between a branching event in SCL decoding and an inactivation event for SCI decoding.

Corollary 4.9 (Upper bound on the required number of inactivations for MAP decoding)**.** Let $I^*(\mathcal{C})$ be the smallest number of inactivations allowed for SCI decoding which imple-

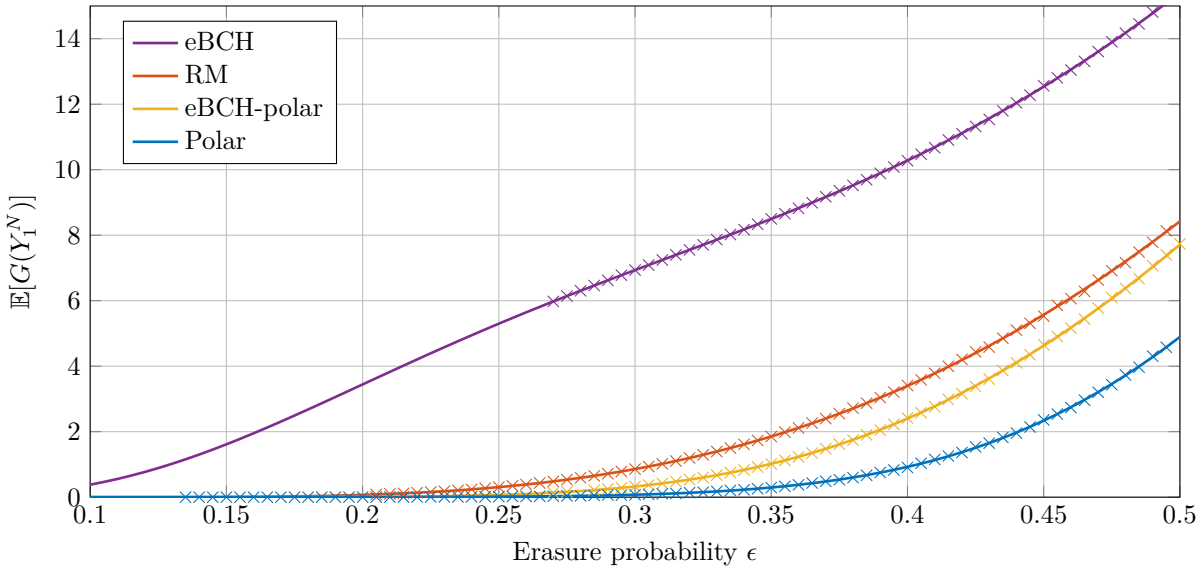

Figure 4.3.: $\mathbb{E}[G(Y_1^N)]$ vs. ϵ for the codes in Figure 4.1 with $N = 128$ (solid lines: from Corollary 4.8, markers: Monte-Carlo simulation averages).

ments a MAP decoder for an (N, K) binary linear code $\mathcal{C}$. Then $I^*(\mathcal{C})$ is upper bounded as

$$I^*(\mathcal{C}) \leq \min\left\{N(1-R) - (\zeta - 1), \gamma - N(1-R)\right\}. \tag{4.16}$$

Remark 4.4 (SCL vs. SCI decoding)**.** Let $P_B^{(\mathrm{SCL})}(L)$ and $P_B^{(\mathrm{SCI})}(I)$ denote the BLEPs for SCL decoding with list size $L = 2^I$ and SCI decoding with maximally I inactivations, respectively. Then we have

$$P_B^{(\mathrm{SCL})}(2^I) \leq P_B^{(\mathrm{SCI})}(I) \tag{4.17}$$

since SCL decoding can prune some invalid paths when it reaches a frozen bit. Note that both decoders would perform the same if one allows at most I branching events for the SCL decoding. Next, the SCI decoding is extended to solve for the inactivations along the way to behave similar to SCL decoding, which forms a basis for the analysis of SCL decoding.

4.4. SCI Decoding with Consolidations

We extend the inactivation decoder to include path pruning like in SCL decoding. The decoder's operation is unchanged whenever an information bit is encountered. For a frozen

bit $u_i = 0$, if (4.14) does not deliver an erasure, it provides the equation $a_{\mathcal{G}} \cdot \tilde{u}_{\mathcal{G}} = 0$.[4] If $a_{\mathcal{G}}$ has a non-zero term, the equation is solved for $\tilde{u}_{i_j}$ as $\tilde{u}_{i_j} = a_{\mathcal{G}\setminus\{i_j\}} \cdot \tilde{u}_{\mathcal{G}\setminus\{i_j\}}$, $i_j = \max\{i \in \mathcal{G} : i \neq 0\}$, and stored. This is called a *consolidation* event. The decoder continues with $\hat{u}_j = 0$. We declare an error if there remains any unresolved $\tilde{u}_i$ at the end. The pseudo codes for the proposed decoder is provided as Appendix 4.5.6. The following analysis of this decoder provides insights into the dynamics of the number of paths in SCL decoding for the BEC.

4.4.1. Dynamics of the Subspace Dimension

Recall Proposition 4.3 stating that the set of valid information sequences after m decoding stages is an affine subspace of $\{0,1\}^{|\mathcal{A}^{(m)}|}$. For any y_1^N, the subspace dimension, denoted as $d_m(y_1^N)$, is equal to a conditional entropy, i.e,

$$d_m(y_1^N) \triangleq H\left(U_{\mathcal{A}^{(m)}} | Y_1^N = y_1^N, U_{\mathcal{F}^{(m)}}\right) \tag{4.18}$$

and the corresponding RV is denoted as D_m that takes on the vaue $d_m(y_1^N)$ when $Y_1^N = y_1^N$.

Consider the decoding of information and frozen bits given the observed vector and preceding frozen bits. When an information bit u_m is decoded, one of following events occurs:

- ▷ The information bit is decoded as an erasure and the subspace dimension increases by one, i.e., $d_m(y_1^N) = d_{m-1}(y_1^N) + 1$. Averaged over all y_1^N, the probability of this event equals $\epsilon^m_{\boldsymbol{G}^{[n]}}$ (see Lemma 4.7).

- ▷ The information bit is decoded as an affine function of the previous information bits and the subspace dimension is unchanged, i.e., $d_m(y_1^N) = d_{m-1}(y_1^N)$. Averaged over all y_1^N, the probability of this event equals $1 - \epsilon^m_{\boldsymbol{G}^{[n]}}$.

If a frozen u_m is decoded, one of following events occurs:

- ▷ The decoder returns an erasure for the frozen bit. In this case, revealing the true value of the frozen bit allows decoding to continue, but no new information is provided about preceding information bits. We thus have $d_m(y_1^N) = d_{m-1}(y_1^N)$. Averaged over all y_1^N, the probability of this event equals $\epsilon^m_{\boldsymbol{G}^{[n]}}$.

[4] If it is a dynamic frozen bit, the RHS of the equation is the linear combination defining it (see Section 3.4.8). For simplicity, assume it is not.

▷ The frozen bit is decoded as an affine function of the previous information bits. Averaged over all y_1^N, the probability of this event equals $1 - \epsilon^m_{\boldsymbol{G}^{[n]}}$. In this case, revealing the true value of the frozen bit gives a linear equation for a subset of the preceding information bits. If the linear equation is informative, then the subspace dimension decreases by one via a consolidation event, i.e., we have $d_m(y_1^N) = d_{m-1}(y_1^N) - 1$. Otherwise, the dimension is unchanged, i.e., $d_m(y_1^N) = d_{m-1}(y_1^N)$.

At first glance, these rules might appear to tell the whole story. But the erasure rate $\epsilon_N^{(m)}$ is averaged over all y_1^N whereas predicting the value of D_m requires the conditional probability of erasure events given all past observations. More importantly, to understand consolidation events, one needs to compute the probability that the obtained equation is informative.

Since we do not have expressions for these quantities, we use two simplifying approximations. First, we approximate the probability of decoding an erasure for a frozen bit as independent of all past events, i.e., for any $d_1^{m-1} \triangleq (d_1\left(y_1^N\right), \ldots, d_{m-1}\left(y_1^N\right))$, we write

$$\Pr\left(P^m_{\boldsymbol{G}^{[n]}}(0|Y_1^N, U_1^{m-1}) = 1/2 \middle| D_1^{m-1} = d_1^{m-1}\right) \approx \epsilon_N^{(m)}. \tag{4.19}$$

Second, we approximate the probability that an informative equation obtained from consolidation by $1 - 2^{-D_{m-1}}$, independent of sequence $D_1, \ldots, D_{m-2}$. This means, for $m \in \mathcal{F}$, we write

$$\Pr\left(D_m = d_{m-1} \middle| D_1^{m-1} = d_1^{m-1}, P^m_{\boldsymbol{G}^{[n]}}(0|Y_1^N, U_1^{m-1}) \neq 1/2\right) \approx 2^{-d_{m-1}} \tag{4.20}$$

$$\Pr\left(D_m = d_{m-1} - 1 \middle| D_1^{m-1} = d_1^{m-1}, P^m_{\boldsymbol{G}^{[n]}}(0|Y_1^N, U_1^{m-1}) \neq 1/2\right) \approx 1 - 2^{-d_{m-1}} \tag{4.21}$$

which comes from modeling the obtained equation and the subset using a uniform random model. Under these assumptions, the random sequence $D_1, \ldots, D_N$ can be modelled by an inhomogeneous Markov chain with transition probabilities $P^{(m)}_{i,j} \triangleq \Pr\left(D_m = j \mid D_{m-1} = i\right)$ where

$$P^{(m)}_{i,j} \approx \begin{cases} \epsilon^m_{\boldsymbol{G}^{[n]}} & \text{if } m \in \mathcal{A},\ j = i+1 \\ 1 - \epsilon^m_{\boldsymbol{G}^{[n]}} & \text{if } m \in \mathcal{A},\ j = i \\ \epsilon^m_{\boldsymbol{G}^{[n]}} + \left(1 - \epsilon^m_{\boldsymbol{G}^{[n]}}\right) 2^{-D_{m-1}} & \text{if } m \in \mathcal{F},\ j = i \\ \left(1 - \epsilon^m_{\boldsymbol{G}^{[n]}}\right)\left(1 - 2^{-D_{m-1}}\right) & \text{if } m \in \mathcal{F},\ j = i-1. \end{cases} \tag{4.22}$$

Consider decoding of frozen bit u_m based on this Markov chain approximation. We write

$$\bar{D}_m = \mathbb{E}\left[D_{m-1} - \left(1-\epsilon_N^{(m)}\right)\left(1-2^{-D_{m-1}}\right)\right] \tag{4.23}$$

$$\approx \bar{D}_{m-1} - \left(1-\epsilon_N^{(m)}\right)\left(1-2^{-\bar{D}_{m-1}}\right) \tag{4.24}$$

where the last line follows from approximating $\mathbb{E}\left[2^{-D_m}\right]$ as $\approx 2^{-\bar{D}_m}$. In the case of information bit u_m, we have

$$\bar{D}_m = \mathbb{E}\left[\epsilon_N^{(m)}\left(D_{m-1}+1\right) + \left(1-\epsilon_N^{(m)}\right) D_{m-1}\right] \tag{4.25}$$

$$= \bar{D}_{m-1} + \epsilon_N^{(m)}. \tag{4.26}$$

By setting $\bar{D}_0 \triangleq 0$, (4.24) and (4.26) give the simple recursive approximation

$$\bar{D}_m \approx \begin{cases} \bar{D}_{m-1} + \epsilon_N^{(m)} & \text{if } m \in \mathcal{A} \\ \bar{D}_{m-1} - \left(1-\epsilon_N^{(m)}\right)\left(1-2^{-\bar{D}_{m-1}}\right) & \text{if } m \in \mathcal{F}. \end{cases} \tag{4.27}$$

Remark 4.5. Observe that $\bar{D}_m = H\left(U_{\mathcal{A}^{(m)}} | Y_1^N, U_{\mathcal{F}^{(m)}}\right)$. We use this entropy to extend some of the results presented in this chapter to more general BMSCs.

Before providing numerical results on the codes of, e.g., Figure 4.1, we study the stochastic convergence properties of the RV D_m, which motivates the mean analysis further.

4.4.2. Concentration of the Subspace Dimension

We form a Doob's Martingale by sequentially revealing information about the object of interest (e.g., see [43,46]), which is the conditional entropy in our case as given in Proposition 4.10. In N consecutive steps, we reveal the random channel realizations. Irrespective of the revealed realization, the change in the subspace dimension is bounded by some constant. This lets us use the Azuma-Hoeffding inequality, i.e., Lemma 2.6, since the channel under consideration is memoryless.

Remark 4.6. We assume, for the upcoming analysis, that the values of frozen bits are also uniformly distributed and SCI decoding learns them causally. In the case of dynamic frozen bits, uniform random constants are added, which are revealed causally as in the case of other frozen bits.

Proposition 4.10. The sequence of RVs $H_0^{(m)}, H_1^{(m)}, \ldots, H_N^{(m)}$ where $H_i^{(m)} \triangleq \mathbb{E}\left[D_m | Y_1^i\right]$ is a Doob's Martingale, i.e., we have

$$H_i^{(m)} \text{is a function of } Y_1^i \tag{4.28}$$

$$\mathbb{E}\left[|D_m|\right] < \infty \tag{4.29}$$

$$H_{i-1}^{(m)} = \mathbb{E}\left[H_i^{(m)} | Y_1^{i-1}\right]. \tag{4.30}$$

Proof. The statement (4.28) follows from the construction of the RVs $H_i^{(m)}$ and the definition of conditional expectation (recall Remark 2.3). The inequality (4.29) follows from the non-negativity of D_m and $\mathbb{E}\left[D_m\right] = H\left(U_{\mathcal{A}^{(m)}} | Y_1^N, U_{\mathcal{F}^{(m)}}\right)$. Finally, (4.30) follows by

$$\mathbb{E}\left[H_i^{(m)} | Y_1^{i-1}\right] = \mathbb{E}\left[\mathbb{E}\left[D_m | Y_1^i\right] | Y_1^{i-1}\right] \tag{4.31}$$

$$= \mathbb{E}\left[D_m | Y_1^{i-1}\right] \tag{4.32}$$

$$= H_{i-1}^{(m)} \tag{4.33}$$

where (4.31) and (4.33) follow from the definition of $H_i^{(m)}$, and (4.32) from the tower property, i.e., Lemma 2.3. ■

Proposition 4.11. For transmission over the BEC, the subspace dimension satisfies the Lipschitz-1 condition: for all $i \in [N]$ and all values y_1^N and $\tilde{y}_1^N$ such that $y_{\sim i} = \tilde{y}_{\sim i}$ and $y_i \neq \tilde{y}_i$, the subspace dimension satisfies

$$|d_m(y_1^N) - d_m(\tilde{y}_1^N)| \leq 1. \tag{4.34}$$

Proposition 4.11, whose proof is given as Appendix 4.5.7, will enable us to write the following.

Corollary 4.12. The subspace dimension D_m concentrates around its mean d_m for sufficiently large block lengths, i.e., for any $\beta > 0$, we have

$$\Pr\left\{\frac{1}{N}|D_m - \bar{D}_m| > \beta\right\} \leq 2\exp\left(-\frac{\beta^2}{2}N\right). \tag{4.35}$$

Proof. Since the channel under consideration is memoryless, Y_i, $i \in [N]$, are independent due to the uniform U_1^N implied by Remark 4.6. Hence, Proposition 4.11 implies

$$\left|H_i^{(m)} - H_{i-1}^{(m)}\right| \leq 1, \ i \in [N]. \tag{4.36}$$

Then, we apply the Azuma-Hoeffding inequality, i.e., Lemma 2.6, by observing that the first element in the martingale is the expectation of D_m and the last one is the RV itself, i.e., $H_0^{(m)} = \bar{D}_m$ and $H_N^{(m)} = D_m$. ■

Remark 4.7. Let $\rho = \lceil \log_2 m \rceil$ and $N_0 = 2^\rho$. Due to the recursive structure of the SCL decoder, the statistics of D_m are the same for all $N \geq N_0$ if the first N_0 frozen bits are the same. Thus, Corollary 4.12 remains valid if we replace (4.35) by

$$\Pr\left\{\frac{1}{N_0}|D_m - d_m| > \beta\right\} \leq 2\exp\left(-\frac{\beta^2}{2}N_0\right). \tag{4.37}$$

This provides a significant improvement when $N_0 \ll N$.

Remark 4.8. Note that the bounds of the form (4.35) are typically loose (see [46, Section IV] for a discussion on tightness of the concentration results for the performance of a randomly chosen LDPC code around the ensemble average). Nevertheless, such analysis shows that the mean $\bar{D}_m$ under consideration is meaningful.

4.4.3. Numerical Results

Consider a *dRM ensemble sequence*, which is introduced as the sequence of rate-1/2 dRM ensembles (see Definition 4.1). The ℓ-th ensemble in the sequence is $\mathcal{DRM}(\ell, 2\ell + 1)$. To understand the accuracy of the mean analysis and approximations, we simulated SCI decoding with consolidations for instances of $\mathcal{DRM}(\ell, 2\ell+1)$, $\ell \in \{4, 5, 6\}$. The results of these simulations are realizations of the random process $D_1, \ldots, D_N$ and we compare their mean to the theoretical predictions (4.22) and (4.27) in Figures 4.4-4.6. Note that as the blocklength changes, a random instance from the corresponding ensemble is picked for the simulations. The numerical results are quite similar for different instances chosen randomly. These results show that for a random code in $\mathcal{DRM}(\ell, 2\ell+1)$ the simulation mean is close to the analysis, where Markov approximation (4.22) matches the mean slightly better for large values of m. With an increasing blocklength, (4.22) and (4.27) match the simulations better for large values of m.

One weakness of these bounds is that the channel variation (e.g., in the number of erasures) significantly increases the variation in D_1^N, especially for small to medium blocklengths, e.g., $N = 512$. Consider a fixed-weight BEC that chooses a random pattern with exactly $\texttt{round}(N\epsilon)$ erasures. To motivate this, note that density evolution naturally captures the typical behavior of the analyzed system [54]. Figure 4.7 shows simulation results for realizations of D_1^N and compares these with their average and the theoretical prediction

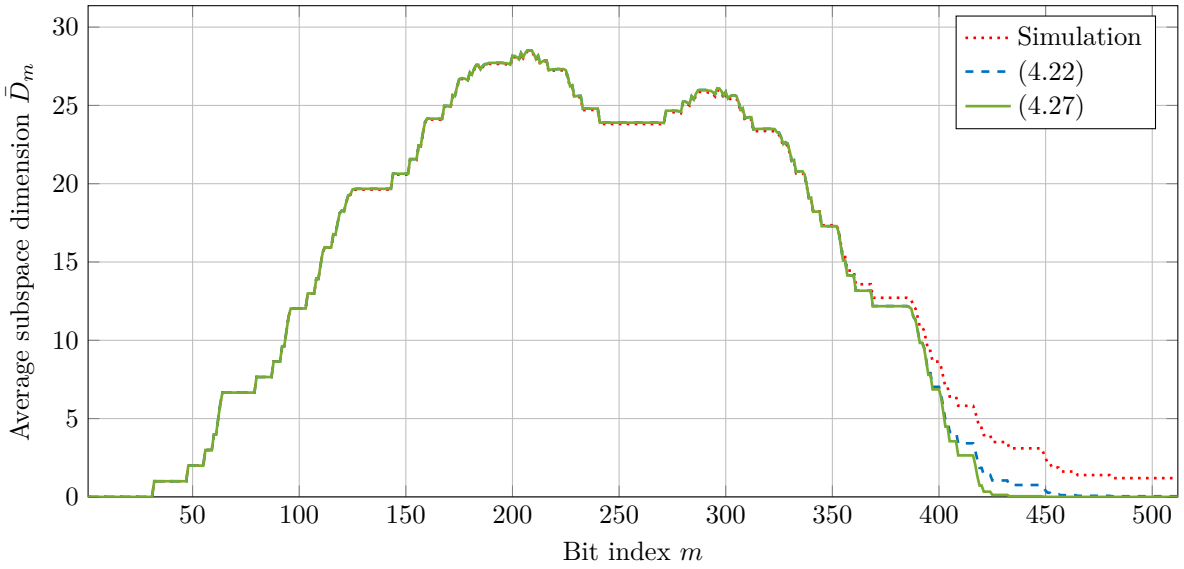

Figure 4.4.: $\bar{D}_m$ vs. m at $\epsilon = 0.48$ for an instance from $\mathcal{DRM}(4, 9)$.

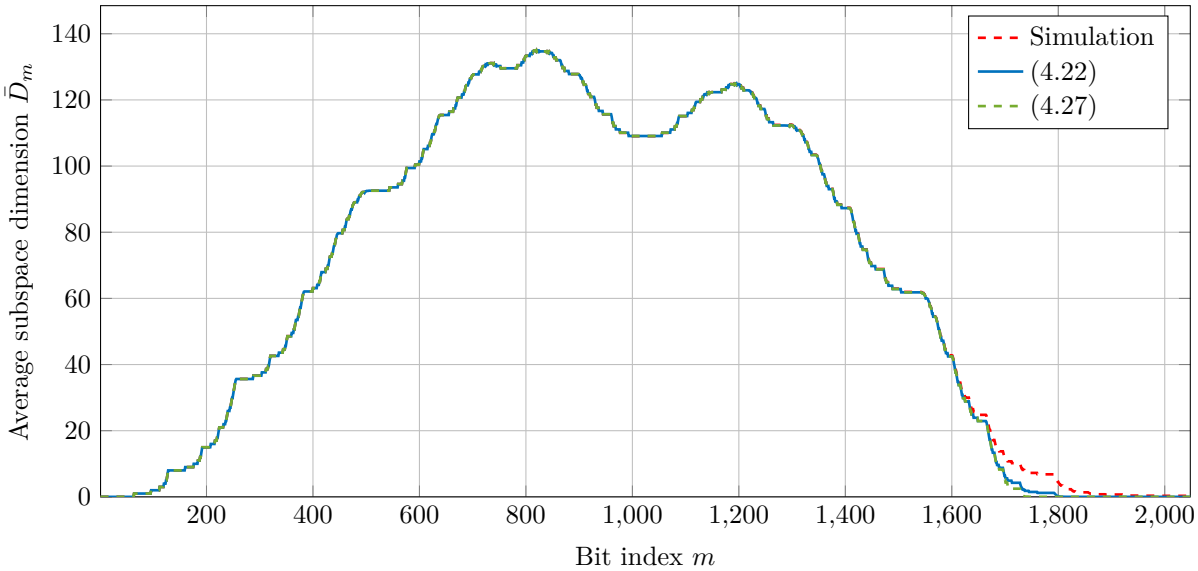

Figure 4.5.: $\bar{D}_m$ vs. m at $\epsilon = 0.48$ for an instance from $\mathcal{DRM}(5, 11)$.

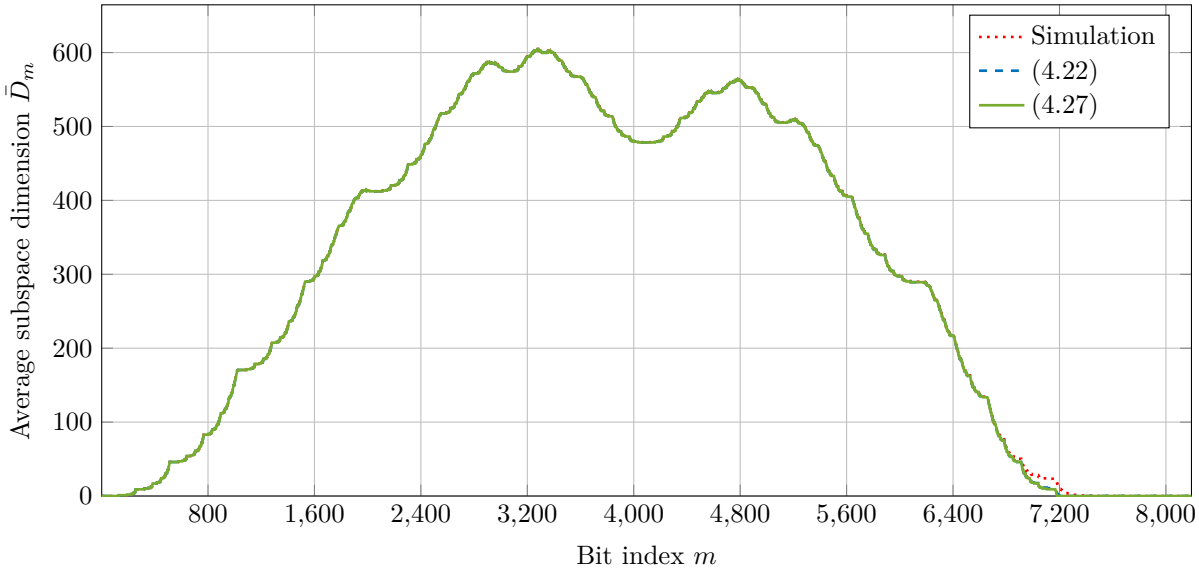

Figure 4.6.: $\bar{D}_m$ vs. m at $\epsilon = 0.48$ for an instance from $\mathcal{DRM}(6, 13)$.

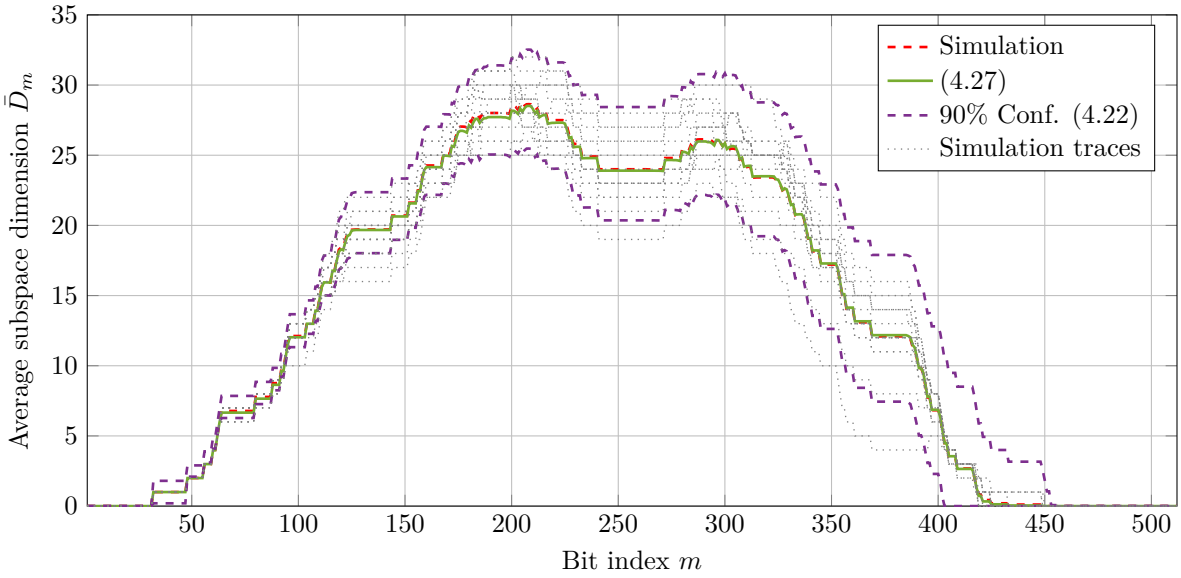

Figure 4.7.: $\bar{D}_m$ vs. m for randomly permuted 246 erasures for an instance from $\mathcal{DRM}(4, 9)$.

(4.27) for a random code in $\mathcal{DRM}(4,9)$ as for Figure 4.4. The results show that even for a random code in $\mathcal{DRM}(4,9)$ the simulation mean is close to the analysis for the entire range of m. The 15 random simulation traces lie largely within the 90% confidence range of the Markov chain analysis.

Next, we study how the subspace dimension behaves as the blocklength increases for the dRM codes. Let $w \triangleq \frac{m}{N}$, $m \in [N]$, be the normalized decoding stage. Figure 4.8 provides the normalized dimension $\frac{1}{N}\bar{D}_{wN}$ as a function of w for the samples of dRM ensemble sequence with different blocklengths, from $N = 2^9$ up to $N = 2^{31}$. The match between the approximation (4.27) and the simulation up to $N = 2^{13}$ shows the accuracy of the analysis and we believe that the results for larger blocklengths are also accurate. The asymptotic behavior of $\frac{1}{N}\bar{D}_{wN}$ gives the asymptotic decoding complexity of an ML decoder implemented via an SCI decoder. This provides insight into the asymptotic decoding complexity of RM codes to achieve the capacity over the BEC [94] for two reasons: first, $\mathrm{RM}(\ell, 2\ell+1)$ is a member of the ensemble, and second, the simulation results look very similar for RM codes up to $N = 2^{13}$. Other decoding algorithms might improve the complexity, but we are not aware of a lower-complexity ML decoder than SCI decoding for RM codes. Figure 4.8 shows that the convergence is rather slow for the defined sequence. Interesting directions include understanding what happens to $\frac{1}{N}\bar{D}_{wN}$ as $N \to \infty$ analytically and trying to find code sequences where $\max_w \frac{1}{N}\bar{D}_{wN}$ is significantly better than that of dRM codes, but that still perform competitively.

Figure 4.9 provides the PMF for $\frac{1}{N}D_{[0.4N]}$, where $[wN]$ is the nearest integer to wN, $w \in (0,1]$. The parameter w is set to 0.4 since the mean analysis given in Figure 4.8 shows that, for the considered codes, the mean reaches to its maximum around $w = 0.4$. Interestingly, the PMFs concentrates around the mean $\frac{1}{N}\bar{D}_{[0.4N]}$ with increasing blocklength as suggested by Corollary 4.12.

To further highlight the performance vs. complexity trade-off, Figure 4.10 shows the average subspace dimension $\bar{D}_m$ as the SCI decoder with consolidations proceeds from $m = 1$ to $m = N$ with $N = 128$ at $\epsilon = 0.4$ for the polar and RM code of Figure 4.1. The code with a better performance, i.e., RM code (see Figure 4.1), has more inactivations at early stages due to more information bits at unreliable positions; hence, a larger decoding complexity. The frozen bits placed at fairly reliable positions help resolve the inactivations, yielding a better performance. The decoder is not able to resolve the inactivated bits for the polar code because of the lack of frozen bits at reliable positions appearing after inactivations. Hence, $\bar{D}_m$ provides a measure to quantify performance vs. complexity trade-off. On the one hand, many unresolved inactivations increase complexity. On the other hand, too few inactivations to begin with do not make best use of the information

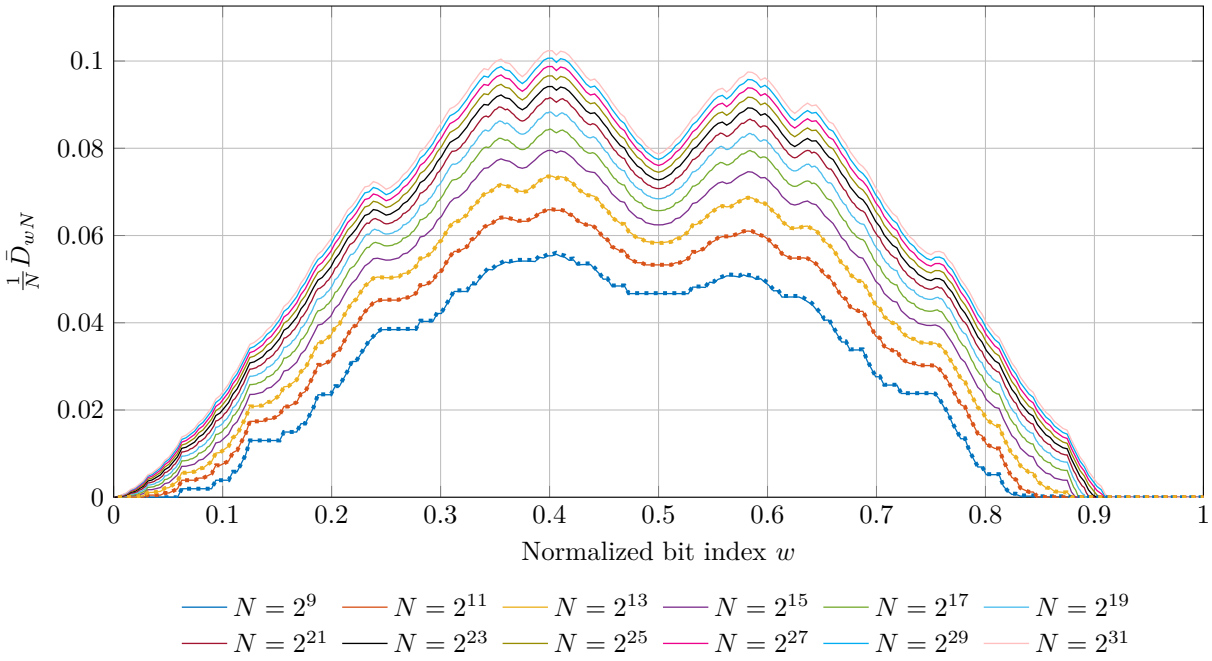

Figure 4.8.: $\frac{1}{N}\bar{D}_{wN}$ vs. w (solid: using (4.27) with an erasure probability $\epsilon = 0.48$, dashed: simulations for the fixed-weight BEC with $\texttt{round}(N\epsilon)$ erasures) for instances from dRM code ensembles $\mathcal{DRM}(\ell, 2\ell+1)$, $\ell \in \{4, 5, \ldots, 15\}$.

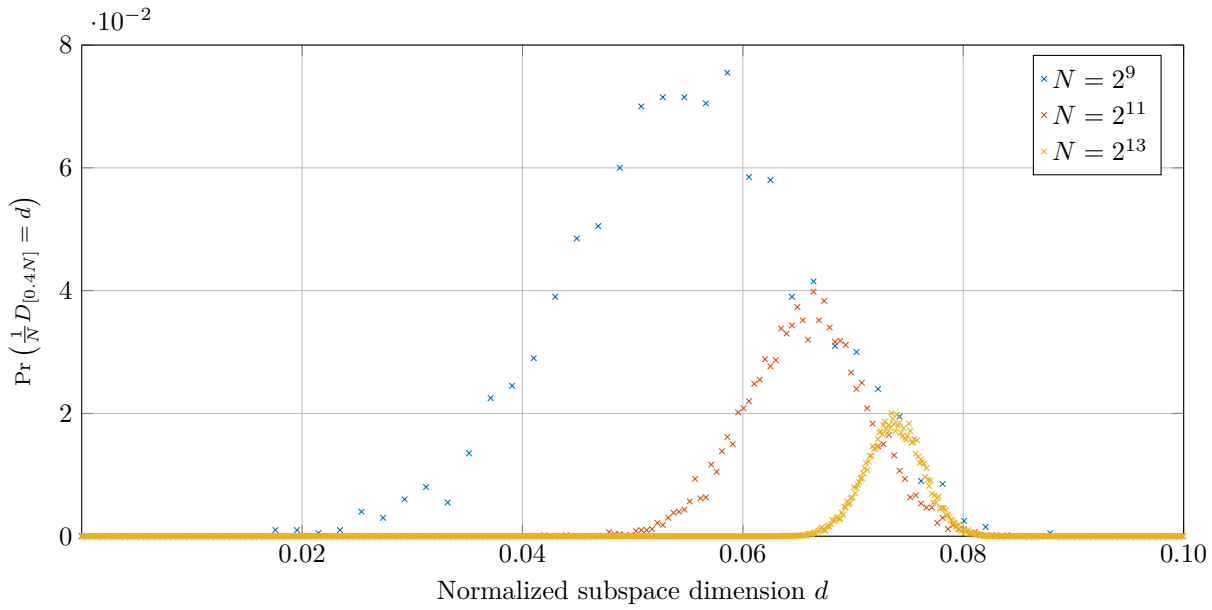

Figure 4.9.: $\Pr\left(\frac{1}{N}D_{[0.4N]} = d\right)$ vs. d for an erasure probability $\epsilon = 0.48$ for instances from dRM code ensembles $\mathcal{DRM}(\ell, 2\ell+1)$, $\ell \in \{4, 5, 6\}$.

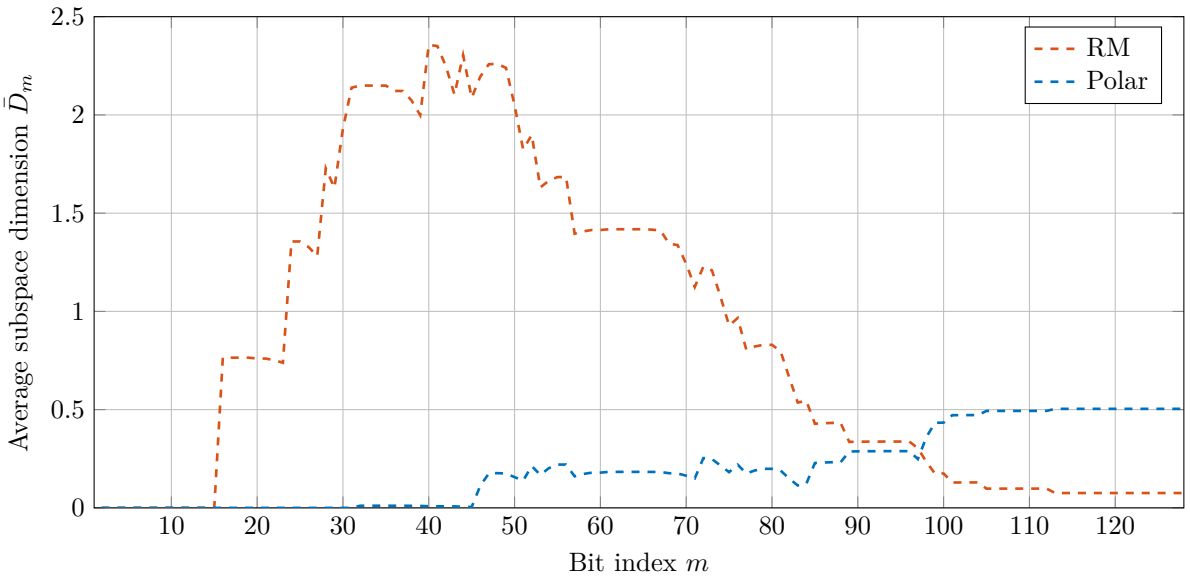

Figure 4.10.: $\bar{D}_m$ vs. m at $\epsilon = 0.4$ for $(128, 64)$ polar code and RM$(3, 7)$.

provided by frozen bits when it comes resolving inactivations.

In Figure 4.11, $\bar{D}_m$ is provided for codes with dynamic frozen bits. Observe the large number of inactivations for the random code from $\mathcal{G}(128, 64)$ that are mostly resolved at the end. The eBCH code and the instance from $\mathcal{DRM}(3, 7)$ provide a similar performance (see Figure 4.1) with a lower complexity compared to the random code. In addition, the eBCH-polar code has the lowest complexity but with a degraded performance (see Figure 4.1). The instance from $7 - \mathcal{DRM}(3, 7)$ is an exemplary construction for a code performing halfway between the eBCH-polar and dRM codes (see Figure 4.1). An analysis of the additional complexity due to dynamic frozen bits is left to future work.

4.5. Appendices

4.5.1. Proof of Theorem 4.1

Assume that a $K \times N$ generator matrix $\boldsymbol{G}$ is constructed by generating all the entries $\{g_{i,j}\}$ uniformly at random over $\mathbb{F}_2$. Let $P_B(\boldsymbol{G})$ denote the block error probability of the random code defined by $\boldsymbol{G}$ under MAP decoding, over the BEC(ϵ). We are interested in upper-bounding the average BLEP, i.e., in upper-bounding $\mathbb{E}[P_B(\boldsymbol{G})]$.

We start by computing the probability of having a rank P smaller than K when generating $K \times (N - \mathsf{e})$ random matrices with $N - \mathsf{e} \geq K$, where all the entries are generated

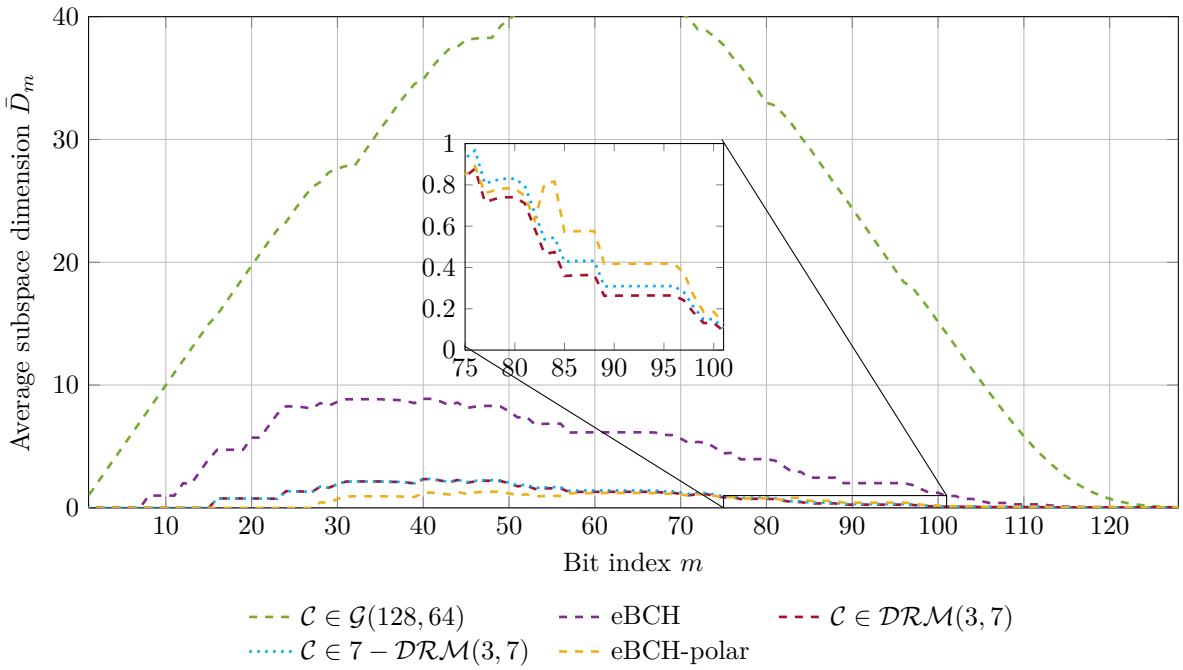

Figure 4.11.: $\bar{D}_m$ vs. m at $\epsilon = 0.4$ for $(128, 64)$ codes with dynamic frozen bits.

uniformly at random. We have [141]

$$\Pr\{P < K\} = 1 - \prod_{i=0}^{K-1} (1 - 2^{-(N-i-\mathsf{e})}). \tag{4.38}$$

The result can be obtained recursively. We generate the $K \times (N - \mathsf{e})$ binary submatrix of $\boldsymbol{G}$ induced by e erasures in row-by-row fashion and we compute the probability that a newly generated row falls in the linear subspace spanned by the preceding rows. Denote by $\boldsymbol{G}_{:,\bar{\mathcal{E}}}^{(i)}$ the matrix obtained after generating the first i rows, with $\boldsymbol{G}_{:,\bar{\mathcal{E}}}^{(K)} = \boldsymbol{G}_{:,\bar{\mathcal{E}}}$. We have that

$$\Pr\{\operatorname{rank} \boldsymbol{G}_{:,\bar{\mathcal{E}}}^{(i)} = i | \boldsymbol{G}_{:,\bar{\mathcal{E}}}^{(i-1)} = i - 1\} = 1 - \frac{2^{i-1}}{2^{N-\mathsf{e}}} \tag{4.39}$$

with

$$\Pr\{\operatorname{rank} \boldsymbol{G}_{:,\bar{\mathcal{E}}}^{(1)} = 1\} = 1 - \frac{1}{2^{N-\mathsf{e}}}. \tag{4.40}$$

by applying (4.39) recursively, we obtain

$$\Pr\{\operatorname{rank} \boldsymbol{G}_{:,\bar{\mathcal{E}}} = K\} = \prod_{i=0}^{K-1} \left(1 - \frac{1}{2^{N-i-\mathsf{e}}}\right). \tag{4.41}$$

from which (4.38) follows. One can use induction to obtain a tight bound [142, Proposition

1] as

$$\Pr\{P < K\} = 2^{-(N-K-\mathtt{e})} \tag{4.42}$$

which is easier to compute. We now write

$$\mathbb{E}\left[P_B\left(\boldsymbol{G}\right)\right] \leq \sum_{i=N-K+1}^{N} \binom{N}{i} \epsilon^i (1-\epsilon)^{N-i} + \sum_{i=1}^{N-K} \binom{N}{i} \epsilon^i (1-\epsilon)^{N-i} 2^{-(N-K-i)} \tag{4.43}$$

where the first term on the RHS follows by observing that if there are more than $N-K$ erasures, then a unique solution is not possible and noting that the probability of observing i erasures is

$$\Pr\{\mathtt{E} = i\} = \binom{N}{i} \epsilon^i (1-\epsilon)^{N-i} \tag{4.44}$$

The second term follows from (4.38) and (4.44), where $\mathtt{e}$ is replaced by i in the former. The result now follows by observing that there exists at least one code performing better than the average. ■

4.5.2. Proof of Lemma 4.2

The SCL decoder can branch up to maximally $2^{K-(N-\gamma)} = 2^{\gamma - N(1-R)}$ paths until the last frozen bit u_γ. Over the BEC, each path is equally likely and the only way to differentiate between the paths is to cancel some of them whenever we obtain zero probability for the paths. Now suppose that (a) the SCL decoder delivers a unique solution at the end and suppose further that (b) there was a branching event after u_γ. Since there is no frozen bit left after u_γ, (b) would mean that there are at least two solutions, hence contradicting (a). Therefore, for an SCL decoder providing a unique solution, it is not possible to have a list length larger than $2^{\gamma - N(1-R)}$ at any decoding stage.

Observe also that bits $u_1^{\zeta-1}$ are frozen and they appear before the first information bit. This means they are of no use for cancelling any path. This would mean there are at most $N-K-(\zeta-1) = N(1-R)-(\zeta-1)$ frozen bits that have potential to prune half of the existing paths. If there are more than $2^{N(1-R)-(\zeta-1)}$ paths at any decoding stage during SCL decoding, then there will be at least 2 solutions at the end, concluding the proof. ■

4.5.3. Proof of Proposition 4.3

Let $\mathcal{E}$ denote the set of erased positions in the realization y_1^N. We can write

$$(u_{\mathcal{A}^{(i)}}, u_{i+1}^N)\boldsymbol{G}_{[N]\setminus\mathcal{F}^{(i)},\bar{\mathcal{E}}}^{[n]} = y_{[N]\setminus\mathcal{E}} \oplus u_{\mathcal{F}^{(i)}}\boldsymbol{G}_{\mathcal{F}^{(i)},\bar{\mathcal{E}}}^{[n]} \tag{4.45}$$

where $\boldsymbol{G}^{[n]}_{\mathcal{S},\bar{\mathcal{E}}}$ is the matrix formed by the rows of $\boldsymbol{G}^{[n]}$ indexed in $\mathcal{S}$ and then removing its columns indexed in $\mathcal{E}$. This equation enables to use the frozen bits $u_{\mathcal{F}^{(i)}}$ as side information. Let $\mathcal{C}$ denote the set of all possible solutions for $(u_{\mathcal{A}^{(i)}}, u_{i+1}^N)$, which is an affine subspace. We are interested in all compatible partial information sequences $u_{\mathcal{A}^{(i)}}$ with (4.45) (hence, u_1^i as $u_{\mathcal{F}^{(i)}}$ is a linear transform of $u_{\mathcal{A}^{(i)}}$). To this end, we define the mapping $\Pi_{\mathcal{A}^{(i)}} : \mathbb{F}_2^{N-|\mathcal{F}^{(i)}|} \to \mathbb{F}_2^{|\mathcal{A}^{(i)}|}$ as

$$\Pi_{\mathcal{A}^{(i)}}(\mathcal{C}) \triangleq \left\{ v_1^{|\mathcal{A}^{(i)}|} : v_1^{N-|\mathcal{F}^{(i)}|} \in \mathcal{C} \right\} \tag{4.46}$$

which is a linear mapping since it can be represented as a multiplication of the input by a matrix formed by stacking an $|\mathcal{A}^{(i)}| \times |\mathcal{A}^{(i)}|$ identity matrix and an $(N-i) \times |\mathcal{A}^{(i)}|$ all-zero matrix. The result now follows by noting that a linear transform of an affine subspace is affine. ■

4.5.4. Proof of Lemma 4.6

The case where $\hat{u}_1^{i-1}$ does not contain any inactivation, i.e., $\hat{u}_1^{i-1} = u_1^{i-1}$, is trivial. Thus, we assume that the decoder inactivated some information bits, i.e., $\hat{u}_j = \tilde{u}_j$ for some j, $1 \leq j < i$.

Now suppose that

$$f_i(y_1^N, u_1^{i-1}) = ? \longrightarrow P^i_{\boldsymbol{G}^{[n]}}(u_i | y_1^N, u_1^{i-1}) = 1/2 \quad \forall u_i \in \mathbb{F}_2. \tag{4.47}$$

Suppose also that we have a vector $a_{\mathcal{G}}$ such that

$$P^i_{\boldsymbol{G}^{[n]}}(a_{\mathcal{G}} \cdot \tilde{u}_{\mathcal{G}} | y_1^N, \hat{u}_1^{i-1}) = 1 \longrightarrow f_i(y_1^N, \hat{u}_1^{i-1}) \neq ? \tag{4.48}$$

But (4.48) implies

$$P^i_{\boldsymbol{G}^{[n]}}(a_{\mathcal{G}} \cdot u_{\mathcal{G}} | y_1^N, u_1^{i-1}) = 1 \tag{4.49}$$

by replacing inactivated bits with their values and having $u_0 = \tilde{u}_0$ (equivalently, $u_0 = 1$). This contradicts (4.47).

Now, consider the other direction, i.e., suppose that for some $u_i \in \mathbb{F}_2$ we have

$$f_i(y_1^N, u_1^{i-1}) = u_i \longrightarrow P^i_{\boldsymbol{G}^{[n]}}(u_i | y_1^N, u_1^{i-1}) = 1 \tag{4.50}$$

and that

$$f_i(y_1^N, \hat{u}_1^{i-1}) = ?. \tag{4.51}$$

Then, (4.50) implies that there exists a vector $a_{\mathcal{G}}$ for which we have

$$P^{i}_{\boldsymbol{G}^{[n]}}(a_{\mathcal{G}} \cdot \tilde{u}_{\mathcal{G}} | y_1^N, \hat{u}_1^{i-1})\Big|_{a_{\mathcal{G}} \cdot \tilde{u}_{\mathcal{G}} = u_i, \hat{u}_1^{i-1} = u_1^{i-1}} = 1 \tag{4.52}$$

and this contradicts (4.51). ■

4.5.5. Proof of Lemma 4.7

Let $b'_i = \Pr\left\{f_i(y_1^N, \hat{u}_1^{i-1}) = ?\right\}$. We have

$$b'_i \overset{\text{(a)}}{=} \mathbb{E}[\mathbb{1}\{f_i(y_1^N, \hat{u}_1^{i-1}) = ?\}] \overset{\text{(b)}}{=} \mathbb{E}[\mathbb{1}\{f_i(y_1^N, u_1^{i-1}) = ?\}] \overset{\text{(c)}}{=} \epsilon^{i}_{\boldsymbol{G}^{[n]}} \tag{4.53}$$

where (a) and (c) follow from the definition of expectation and (b) from Lemma 4.6. The result follows from (4.8). ■

4.5.6. Algorithms for SCI Decoding with Consolidations

We first provide the data structures required, where the space-efficient SC decoding of [11] is followed closely. We first have vectors $P[\lambda]$, for each layer $\lambda \in [n+1]$, of length $2^{n-\lambda+1}$, which consists of elements from the ternary alphabet $\{0, ?, 1\}$. Secondly, we have 2-dimensional bit arrays $C[\lambda]$ of size $2^{n-\lambda+1} \times 2$, whose elements are from $\{0, 1\}$. For every pair of $\lambda \in [n+1]$ and $\beta \in [2^{n-\lambda+1}]$, we have binary sparse vectors $L[\lambda][\beta]$ and $R[\lambda][\beta, \phi]$, $\phi \in [2]$, of size N. Finally, we have bit vectors $\hat{u}$ and $F[i]$, $i \in [N]$, of size N. They can be initialized as in Algorithm 1 with the caveat that the initialization of the sparse vectors $F[i]$ should take into account the dynamic frozen bit constraints. In the following we provide an example.

Example 4.1. Assume that we have an $(N = 8, K = 4)$ polar code, where $\mathcal{A} = \{4, 6, 7, 8\}$ and $u_5 = u_4$ is a dynamic frozen bit. Then, the sparse vectors $F[i]$, $i \in [8] \setminus \{5\}$, are initialized as the all-zero vectors of length 8, but $F[5]$ has non-zero entry as its 4-th element, i.e., $F[5][4] = 1$. This represents the dynamic frozen via a tuple $(\hat{u}[5], F[5])$. In other words, we have

$$u_5 = \hat{u}[5] \oplus u_1^N F[5]^{\mathrm{T}} \tag{4.54}$$

where $F[5]^{\mathrm{T}}$ is the transpose of $F[5]$.

Algorithm 2 is the main loop of SCI decoding with consolidations. In addition to the representation of the data structures as sparse vectors, the main difference of the algorithm

Algorithm 1: initializeDataStructures()

```
1  û = zeros(N)
2  for i = 1, ..., N do
3      F[i] = sparse(N) // initiated according to dynamic frozen bit constraints
4  for λ = 1, ..., n + 1 do
5      P[λ] = array(2^{n−λ+1})
6      C[λ] = array(2^{n−λ+1}, 2)
7      for β = 1, ..., 2^{n−λ+1} do
8          L[λ][β] = sparse(N)
9          for φ = 1, 2 do
10             R[λ][β, φ] = sparse(N)
```

from SC decoding is due to the inactivation event as given in line 16 and the subroutine consolidate(i) given as Algorithm 3. For any $i \in \mathcal{A}$, if $P[n+1][1]$ is computed as an erasure, then the decoder inactivates its value as u_i by setting $F[i][i] = 1$. As in Example 4.1, u_i is represented by a tuple $(\hat{u}_i, F[i])$, which simply reads as

$$u_i = \hat{u}[i] \oplus u_1^N F[i]^T. \tag{4.55}$$

For any $i \in \mathcal{F}$, if the combination provided by the data structures $P[n+1]$ and $L[n+1]$ do not match that of $\hat{u}[i]$ and $F[i]$ in the case that the decoder does not provide an erasure, this means that the frozen bit provides information about the former inactivations since there cannot be any error over the BEC. Then, the subroutine consolidate(i) solves the equation for the unresolved variable with the smallest index[5] and updates the data structures affected by previous inactivations. The variable d initiated as the all-zero vector in line 5 of Algorithm 2 and $d[m]$ is equivalent to the subspace dimension $d_m\left(y_1^N\right)$. This means there is a unique solution if $d[N] = 0$, i.e., if $d_N\left(y_1^N\right) = 0$, but the decoder outputs an error in case there is more solutions.[6]

Algorithms 4-5 operates recursively as usual for SC decoding [11], where the former is responsible of computing $P[\lambda]$ and $L[\lambda]$ and the latter updates the decisions $C[\lambda]$ and $R[\lambda]$ including the inactivations. An important aspect to note is that if $P[\lambda][\beta]$ is computed as erasure one does not need to update $L[\lambda][\beta]$ (see lines 10 and 23 of Algorithm 4). The reason is that the tuple $(P[\lambda][\beta], L[\lambda][\beta])$ represents an erasure in this case no matter what

[5]The equation can be solved for any unresolved variable at this point. We choose to solve it for the smallest by assuming that it has higher probability to be involved in many data structures.

[6]The algorithm can be modified to output the tuple $(\hat{u}, F)$, representing the solution set compactly.

Algorithm 2: SCI Decoding with Consolidations

Input: y_1^N
Output: $\hat{u}_1^N$
for $i = 1, \ldots, N$ **do**
    $P[1][i] = y_i$
$d = \text{zeros}(N)$
for $i = 1, \ldots, N$ **do**
    recursivelyCalcPL$(n+1, i-1)$
    if $i \notin \mathcal{A}$ **then**
        if $P[n+1][1] \neq ?$ **then**
            if $P[n+1][1] \neq \hat{u}[i]$ || $L[n+1][1] \neq F[i]$ **then**
                consolidate(i)
                $d[i] = d[i-1] - 1$
    else
        if $P[n+1][1] \neq ?$ **then**
            $\hat{u}[i] = P[n+1][1]$
            $F[i] = L[n+1][1]$
        else
            $F[i][i] = 1$ `// inactivate` u_i
            $d[i] = d[i] + 1$
    $C[n+1][1, (i-1) \mod 2 + 1] = \hat{u}[i]$
    $R[n+1][1, (i-1) \mod 2 + 1] = F[i]$
    if $i \mod 2 = 0$ **then**
        recursivelyCalcCR$(n+1, i-1)$
if $d[N] = 0$ **then**
    return $\hat{u}$
else
    return *erasures*(N)

Algorithm 3: consolidate(i)

```
1  û[i] = û[i] ⊕ P[n+1][1]
2  F[i] = F[i] ⊕ L[n+1][1]
3  t = firstNonZeroElement(F[i])
4  for j = 1, ..., N do
5      if F[j][t] = 1 then
6          û[j] = û[j] ⊕ û[i]
7          F[j] = F[j] ⊕ F[i]
8  a = i
9  while a ≠ 0 do
10     λ = n + 1 − ⌊log2 a⌋
11     for β = 1, ..., 2^{n−λ+1} do
12         if L[λ][β][t] = 1 then
13             P[λ][β] = P[λ][β] ⊕ û[i]
14             L[λ][β] = L[λ][β] ⊕ F[i]
15         for φ = 1, 2 do
16             if R[λ][β, φ][t] = 1 then
17                 C[λ][β, φ] = C[λ][β, φ] ⊕ û[i]
18                 R[λ][β, φ] = R[λ][β, φ] ⊕ F[i]
19     a = a − 2^{⌊log2 a⌋}
20 F[i][t] = 0
```

$L[\lambda][\beta]$ is.

Remark 4.9. We consider upper-bounding the number of XORs required by the algorithm for a given channel output y_1^N. This means, for example, we are interested in the number of XORs due to lines 12, 13, 20 and 21 of Algorithm 4, but the assignments like in lines 10, 16 and 17 are omitted. The number of XORs required $C_N^{\mathrm{SCI}}\left(y_1^N\right)$ by the proposed decoder is upper bounded as

$$C_N^{\mathrm{SCI}}\left(y_1^N\right) \leq \frac{3}{2}\left(1 + d_{\max}\left(y_1^N\right)\right) N \log_2 N + \sum_{\substack{m \in \mathcal{F}, \\ \text{cons.} \\ \text{happens}}} (3m + N + 1)\left(1 + d_{m-1}\left(y_1^N\right)\right) \quad (4.56)$$

where $d_{\max}\left(y_1^N\right) \triangleq \max_m d_m\left(y_1^N\right)$. The first term follows from observing that SC decoding operations are introduced for the sparse vectors which can have at most $d_{\max}\left(y_1^N\right)$ non-zero entries at any stage of decoding and observing that standard SC decoding requires at most $\frac{3}{2} N \log_2 N$ XORs over the BEC. The second term stems from subroutine consolidation(m),

Algorithm 4: recursivelyCalcPL(λ, i)

Input: λ and i
if $\lambda = 1$ **then**
  return
$\psi = \lfloor i/2 \rfloor$
if $i \mod 2 = 0$ **then**
  recursivelyCalcPL($\lambda - 1, \psi$)
for $\beta = 1, \ldots, 2^{n-\lambda+1}$ **do**
  if $i \mod 2 = 0$ **then**
    if $P[\lambda-1][2\beta-1] =? \;||\; P[\lambda-1][2\beta] =?$ **then**
      $P[\lambda][\beta] =?$
    else
      $P[\lambda][\beta] = P[\lambda-1][2\beta-1] \oplus P[\lambda-1][2\beta]$
      $L[\lambda][\beta] = L[\lambda-1][2\beta-1] \oplus L[\lambda-1][2\beta]$
  else
    if $P[\lambda-1][2\beta] \neq?$ **then**
      $P[\lambda][\beta] = P[\lambda-1][2\beta]$
      $L[\lambda][\beta] = L[\lambda-1][2\beta]$
    else
      if $P[\lambda-1][2\beta-1] \neq?$ **then**
        $P[\lambda][\beta] = P[\lambda-1][2\beta-1] \oplus C[\lambda][\beta, 1]$
        $L[\lambda][\beta] = L[\lambda-1][2\beta-1] \oplus R[\lambda][\beta, 1]$
      else
        $P[\lambda][\beta] =?$

Algorithm 5: recursivelyCalcCR(λ, i)

Input: λ and i
$\psi = \lfloor i/2 \rfloor$
for $\beta = 1, \ldots, 2^{n-\lambda+1}$ **do**
  $C[\lambda-1][2\beta-1, \psi \mod 2+1] = C[\lambda][\beta, 1] \oplus C[\lambda][\beta, 2]$
  $R[\lambda-1][2\beta-1, \psi \mod 2+1] = R[\lambda][\beta, 1] \oplus R[\lambda][\beta, 2]$
  $C[\lambda-1][2\beta, \psi \mod 2+1] = C[\lambda][\beta, 2]$
  $R[\lambda-1][2\beta, \psi \mod 2+1] = R[\lambda][\beta, 2]$
if $\psi \mod 2 = 1$ **then**
  recursivelyCalcCR($\lambda - 1, \psi$)

which can be seen as a form of backward substitution. If called, it requires to update $3m$ tuples between lines 9 and 19, where each tuple consists of, e.g., $P[\lambda][\beta]$ or $C[\lambda][\beta,\phi]$, and a binary vector, e.g., $L[\lambda][\beta]$ or $R[\lambda][\beta,\phi]$, of at most $d_{\max}\left(y_1^N\right)$ non-zero entries. Hence, this update requires at most $3m\left(1+d_{m-1}\left(y_1^N\right)\right)$ XORs. In addition, lines 1 and 2 require at most $\left(1+d_{m-1}\left(y_1^N\right)\right)$ XORs and the number of XORs between lines 4 and 7 might update N tuples, which require at most $N\left(1+d_{m-1}\left(y_1^N\right)\right)$ XORs. Hence, we have an upper bound (4.56). Then, we write

$$C_N^{\mathrm{SCI}}\left(y_1^N\right) \leq \frac{3}{2}\left(1+d_{\max}\left(y_1^N\right)\right) N\log_2 N + g\left(y_1^N\right)(4N+1)\left(1+d_{\max}\left(y_1^N\right)\right) \tag{4.57}$$

where $g\left(y_1^N\right)$ denotes the total number of inactivations during decoding. Then, the RHS follows by noting that there can be at most as many consolidations as inactivations. By observing that $d_{\max}\left(y_1^N\right) \leq g\left(y_1^N\right) \leq NR$, the bound (4.57) scales at most cubic in N independent of the code, which gets typically loose for wide range of codes, e.g., polar codes.

When no constraint on the list size is imposed, the worst-case complexity $C_N^{\mathrm{SCL}}\left(y_1^N\right)$ of SCL decoding is upper bounded as

$$C_N^{\mathrm{SCL}}\left(y_1^N\right) \leq \frac{3}{2} 2^{d_{\max}\left(y_1^N\right)} N\log_2 N \tag{4.58}$$

where the RHS scales exponentially in $d_{\max}\left(y_1^N\right)$. For $d_{\max}\left(y_1^N\right)=0$, bounds (4.57) and (4.58) both become $\frac{3}{2}N\log_2 N$ as expected. For large values of $d_{\max}\left(y_1^N\right)$, two bounds behave differently in favour of the former.

Figure 4.12 provides the probabilities $\Pr\left(\frac{1}{N}d_{\max}\left(Y_1^N\right)\leq d\right)$, $d\in[0,NR]$, for instances from dRM code ensembles with different block lengths over the BEC with erasure probability $\epsilon=0.48$. For $N=2^{13}$, we have $\Pr\left(\frac{1}{N}d_{\max}\left(Y_1^N\right)\leq 0.065\right)\approx 0$. This means, for an SCL decoder with list size $L<2^{532}$, the list length would exceed the list size with very high probability, which yields block error rates close to 1. For the same case, we have $\Pr\left(\frac{1}{N}d_{\max}\left(Y_1^N\right)\leq 0.074\right)\approx 0.5$ and $\Pr\left(\frac{1}{N}d_{\max}\left(Y_1^N\right)\leq 0.083\right)\approx 1$. This means that the list length of SCL decoding with unbounded list size would reach 2^{606} roughly half of the time and to 2^{680} with a non-zero probability, which may not be feasible to implement due to its complexity.

For completeness, Figure 4.13 provides the ML performance for instances of dRM codes of Figure 4.12, obtained via SCI decoding. For $N\in\{2^{11},2^{13}\}$, they perform remarkably close to the Singleton bound.

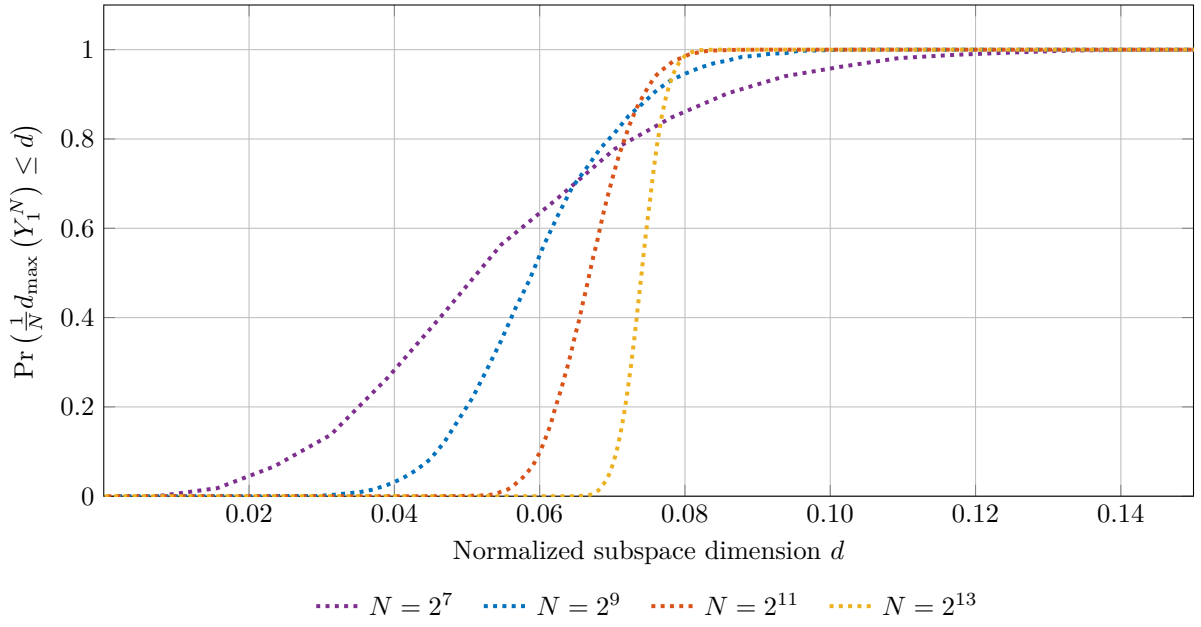

Figure 4.12.: $\Pr\left(\frac{1}{N}d_{\max}\left(Y_1^N\right) \leq d\right)$ vs. d for an erasure probability $\epsilon = 0.48$ for instances from dRM code ensembles $\mathcal{DRM}(\ell, 2\ell+1)$, $\ell \in \{3,4,5,6\}$.

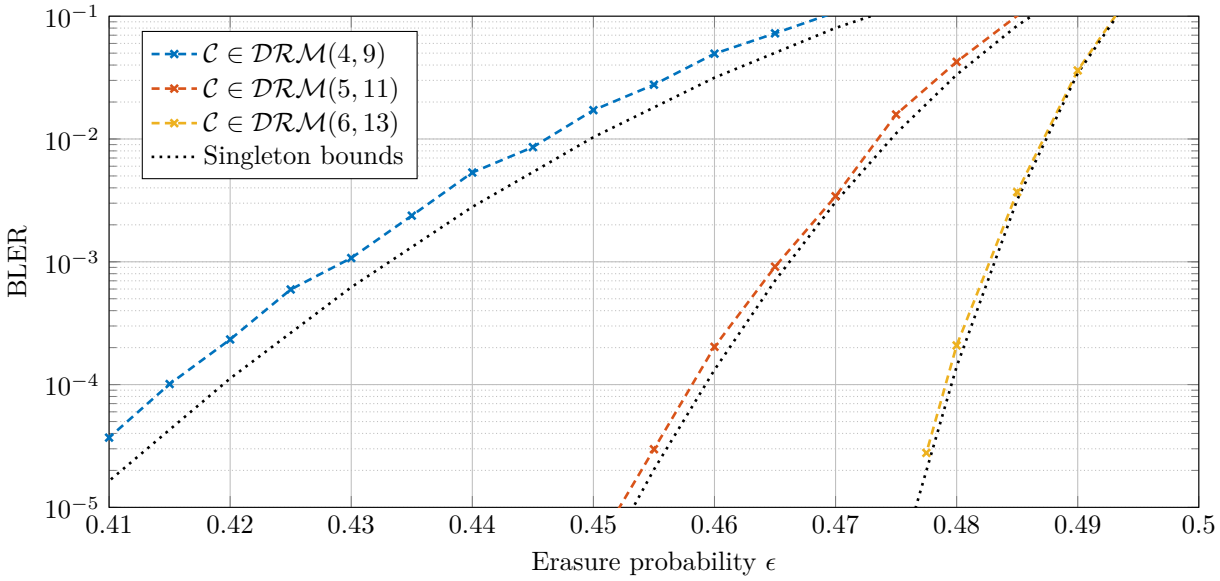

Figure 4.13.: BLER vs. ϵ for instances from dRM code ensembles $\mathcal{DRM}(\ell, 2\ell+1)$, $\ell \in \{4,5,6\}$.

4.5.7. Proof of Proposition 4.11

It suffices to consider the case where y_i is not erased but $\tilde{y}_i$ is an erasure (i.e., $y_i = x_i$ and $\tilde{y}_i = ?$). Observe that SCI decoding with consolidations over the BEC is equivalent to solving a system of linear equations with side information depending on the decoding stage. In other words, the decoder knows the frozen bits $u_{\mathcal{F}^{(m)}}$ after decoding stage m as side information. Now recall the linear system given as (4.45). All compatible vectors $(u_{\mathcal{A}^{(m)}}, u_{m+1}^N)$, $m \in [N]$, with (4.45) form an affine subspace. The dimension of this subspace is

$$d'_N(y_1^N) = N - |\mathcal{F}^{(m)}| - \operatorname{rank}(\boldsymbol{G}^{[n]}_{[N],\bar{\mathcal{E}}}). \tag{4.59}$$

Since removing one more column of $\boldsymbol{G}^{[n]}_{[N],\bar{\mathcal{E}}}$ (and also of $\boldsymbol{G}^{[n]}_{\mathcal{F}^{(m)},\bar{\mathcal{E}}}$) cannot decrease the rank by more than one, we have

$$d'_N(y_1^N) \leq d'_N(\tilde{y}_1^N) \leq d'_N(y_1^N) + 1. \tag{4.60}$$

Hence, the number of compatible vectors $(u_{\mathcal{A}^{(m)}}, u_{m+1}^N)$ with (4.45) is (at most) doubled or unchanged.

We are interested in the subspace dimension $d_m(y_1^N)$. This is equal to the number of different subvectors $u_{\mathcal{A}^{(m)}}$ of all compatible $(u_{\mathcal{A}^{(m)}}, u_{m+1}^N)$ with (4.45). Using (4.60), one concludes that the number of different vectors $u_{\mathcal{A}^{(m)}}$ either increases by a factor of 2 or does not change, resulting in (4.34). ■

5

Successive Cancellation List Decoding over General BMSCs

This chapter develops an information-theoretic perspective for analyzing the SCL decoding of polar codes when transmission is over BMSCs by generalizing the ideas presented in Chapter 4 for the BEC. An important property of SCL decoding is that, if the correct codeword is on the list at the end of decoding, then the BLEP is upper bounded by that of the MAP decoder. We study how large the list should be on average at each decoding stage on average so that the correct codeword is likely to be on the list. Intuitively, such an average list size would be small for codes which are more suited for SC decoding, e.g., polar codes, while large for others, e.g., RM codes. A closed form approximation for this average, that is easy to compute, provides a first-order proxy for good code designs for SCL decoding over general BMSCs.

The performance of random codes from the dRM ensemble $\mathcal{RM}(3,7)$ is close to the random coding union (RCU) bound [143, Theorem 16] down to the BLER 10^{-5} under SCL decoding with list size $L = 128$. The analysis shows how to modify the design to improve the performance when a more practical list size (e.g., $L = 32$) is adopted, while keeping the performance with $L = 128$ unchanged. For a blocklength of $N = 512$, a design performing within 0.4 dB from the RCU bound down to the BLER 10^{-6} under an SCL decoder with list size $L = 1024$ is provided. The design is modified using the new guidelines derived in this chapter, so that the performance improves with practical list sizes (e.g., $L \in \{8, 32, 128\}$), outperforming the polar codes included in the 3GPP 5G cellular standard.

5.1. Information-Theoretic Analysis of SCL Decoding

Consider a length-N polar code with SCL decoding after the m-th decoding stage. Since SCL decoding does not use future frozen bits, we focus on the subset of length-m input patterns that have significant conditional entropy given the channel observation. An important insight is that, after observing Y_1^N, the uncertainty in U_1^m is quantified by the entropy

$$H\left(U_1^m|Y_1^N\right) = \sum_{i=1}^{m} H\left(U_i|U_1^{i-1}, Y_1^N\right) \tag{5.1}$$

where U_1^N is assumed to be uniform over $\{0,1\}^N$. This is exactly true if the first m bits are all information bits, i.e., if $[m] \subseteq \mathcal{A}$. If $[m]$ contains also frozen indices, however, then the situation is more complicated.

Recall that $\mathcal{A}^{(m)} = \mathcal{A} \cap [m]$ and $\mathcal{F}^{(m)} = \mathcal{F} \cap [m]$, i.e., these are the sets containing information and frozen indices within the first m input bits, respectively. Now consider an experiment where the frozen bits U_i with $i \in \mathcal{F}^{(m)}$ are uniform and independent of U_1^{i-1}. Using (5.1) naively with the assumption that $U_{\mathcal{F}^{(m)}}$ is not known to the receiver would cause an overestimate of $H\left(U_1^m|Y_1^N\right)$ by an amount of at least $\sum_{i\in\mathcal{F}^{(m)}} H\left(U_i|U_1^{i-1}, Y_1^N\right)$. In addition, the frozen bits $U_{\mathcal{F}^{(m)}}$ may reveal additional information about the previous information bits.

To better understand the uncertainty of the first m input bits during SCL decoding, recall the quantity

$$\bar{D}_m = H\left(U_{\mathcal{A}^{(m)}} \middle| Y_1^N, U_{\mathcal{F}^{(m)}}\right) \tag{5.2}$$

and define a difference sequence

$$\Delta_m \triangleq \bar{D}_m - \bar{D}_{m-1}. \tag{5.3}$$

Observe that, if U_m is an information bit, then we have

$$\begin{aligned}
\Delta_m &= H\left(U_{\mathcal{A}^{(m)}}|Y_1^N, U_{\mathcal{F}^{(m)}}\right) - H\left(U_{\mathcal{A}^{(m-1)}}|Y_1^N, U_{\mathcal{F}^{(m-1)}}\right) \\
&= H\left(U_{\mathcal{A}^{(m)}}|Y_1^N, U_{\mathcal{F}^{(m-1)}}\right) - H\left(U_{\mathcal{A}^{(m-1)}}|Y_1^N, U_{\mathcal{F}^{(m-1)}}\right) \\
&= H\left(U_{\mathcal{A}^{(m)}}, U_{\mathcal{F}^{(m-1)}}|Y_1^N\right) - H\left(U_{\mathcal{A}^{(m-1)}}, U_{\mathcal{F}^{(m-1)}}|Y_1^N\right) \\
&= H\left(U_1^{m-1}|Y_1^N\right) + H\left(U_m|Y_1^N, U_1^{m-1}\right) - H\left(U_1^{m-1}|Y_1^N\right) \\
&= H(U_m|Y_1^N, U_1^{m-1})
\end{aligned} \tag{5.4}$$

which is exactly what one would expect from the naive analysis given by (5.1).

If U_m is a frozen bit, then consider a model where U_m is not known to the receiver at the time of transmission.[1] The act of revealing U_m to the receiver changes the conditional uncertainty about $U_{\mathcal{A}^{(m-1)}}$ by

$$\begin{aligned}\Delta_m &= H\left(U_{\mathcal{A}^{(m)}}|Y_1^N, U_{\mathcal{F}^{(m)}}\right) - H\left(U_{\mathcal{A}^{(m-1)}}|Y_1^N, U_{\mathcal{F}^{(m-1)}}\right)\\ &= H\left(U_{\mathcal{A}^{(m-1)}}|Y_1^N, U_{\mathcal{F}^{(m-1)}}, U_m\right) - H\left(U_{\mathcal{A}^{(m-1)}}|Y_1^N, U_{\mathcal{F}^{(m-1)}}\right)\\ &= -I\left(U_m; U_{\mathcal{A}^{(m-1)}}|Y_1^N, U_{\mathcal{F}^{(m-1)}}\right)\\ &= H\left(U_m|Y_1^N, U_1^{m-1}\right) - H\left(U_m|Y_1^N, U_{\mathcal{F}^{(m-1)}}\right)\\ &\geq H\left(U_m|Y_1^N, U_1^{m-1}\right) - 1. \end{aligned} \tag{5.5}$$

This expression quantifies the effect of revealing the new frozen bit as a reduction in the conditional entropy of the information bits preceding it. A large reduction may occur when the channel $W^m_{\boldsymbol{G}^{[n]}}$ has low entropy (i.e., a low-entropy effective channel is essentially frozen) and the reduction will be small if the channel entropy is high (i.e., the input is unpredictable from Y_1^N and U_1^{m-1}).

For BMS channels, we can combine (5.4) and (5.5) to understand the dynamics of $\bar{D}_m$. This gives a proxy for the uncertainty in the SCL decoding after m steps. We have

$$\sum_{i\in\mathcal{A}^{(m)}} H\left(W^i_{\boldsymbol{G}^{[n]}}\right) - \sum_{i\in\mathcal{F}^{(m)}} \left(1 - H\left(W^i_{\boldsymbol{G}^{[n]}}\right)\right) \leq \bar{D}_m \tag{5.6}$$

$$\leq \sum_{i\in\mathcal{A}^{(m)}} H\left(W^i_{\boldsymbol{G}^{[n]}}\right). \tag{5.7}$$

The lower bound assumes that frozen bits (when perfectly observed) always reduce the entropy. The following theorem, whose proof is left for Appendix 5.2.1, provides a tighter upper bound than (5.7).

Theorem 5.1. Upon observing y_1^N when u_1^N is transmitted, the set of partial sequences $\tilde{u}_1^m$ with a larger likelihood than some fraction, determined by a positive number $\alpha \leq 1$, of that for true sequence u_1^m after m stages of SCL decoding is given by

$$\mathcal{S}^{(m)}_\alpha\left(u_1^m, y_1^N\right) \triangleq \{\tilde{u}_1^m : p^i_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^m) \geq \alpha p^i_{\boldsymbol{G}^{[n]}}(y_1^N, u_1^m)\}. \tag{5.8}$$

[1] This reflects how the SCL decoder operates, i.e., it does not use the knowledge of any frozen bit U_m until reaching the end of its decoding stage m. The soft estimate $p^m_{\boldsymbol{G}^{[n]}}(y_1^N, u_1^{m-1})$ provides an additional information to separate the hypotheses (i.e, paths) although the hard estimate is chosen as $\hat{u}_m = u_m$ independent of $p^m_{\boldsymbol{G}^{[n]}}(y_1^N, u_1^{m-1})$.

On average, the logarithm of its cardinality is upper bounded by

$$\mathbb{E}\left[\log_2 |\mathcal{S}_\alpha^{(m)}|\right] \leq \bar{D}_m + \log_2 \alpha^{-1} \tag{5.9}$$

$$= H\left(U_{\mathcal{A}^{(m)}} | Y_1^N, U_{\mathcal{F}^{(m)}}\right) + \log_2 \alpha^{-1}. \tag{5.10}$$

Note that α is used as a tuning parameter to catch near misses. Making it too small will keep many partial sequences with low probabilities in the set while choosing it as $\alpha = 1$ will exclude those with probabilities close to (but slightly smaller than) that of the correct path. For Monte-Carlo simulations validating (5.10), we choose it as close as possible to 1, but still keeping $\mathbb{E}\left[\log_2 |\mathcal{S}_\alpha^{(m)}|\right]$ close to $\bar{D}_m$ especially for small values of m. Observe that making α too small will keep many partial sequences with low probabilities in $\mathcal{S}_\alpha^{(m)}$ and choosing it as $\alpha = 1$ misses.

Now consider an SCL decoder whose list size is L_m during the m-th decoding step. Then the decoder should satisfy $L_m \geq |\mathcal{S}_1^{(m)}|$ for the true u_1^m to be in the set $\mathcal{S}_1^{(m)}$. Using (5.10) and (5.7) yields the simple upper bound

$$\mathbb{E}\left[\log_2 |\mathcal{S}_\alpha^{(m)}|\right] \leq \sum_{i \in \mathcal{A}^{(m)}} H\left(W_N^{(i)}\right) + \log_2 \alpha^{-1}. \tag{5.11}$$

Remark 5.1. The analysis in terms of $\log_2 L_m$ has two weaknesses. First, the entropy $\bar{D}_m$ characterizes only typical events, e.g., ensuring that the correct codeword stays on the list at least half of the time, whereas coding typically focuses on rarer events, e.g., BLERs less than 10^{-2}. Second, the sequence $\bar{D}_m$ is averaged over Y_1^N but the actual decoder sees a random realization $D_m(y_1^N) = H\left(U_{\mathcal{A}^{(m)}} | Y_1^N = y_1^N, U_{\mathcal{F}^{(m)}}\right)$. Nevertheless, we believe the results provide a useful step towards a theoretical analysis of SCL decoding. In the following, we provide numerical results that illustrate the accuracy of the analysis. After the numerical results, we study the convergence properties of the RV D_m as for the BEC to motivate further analysis.

Remark 5.2. The results have significance for code design. To achieve good performance with under SCL decoding whose list size is L_m during the m-th decoding step, a reasonable first-order design criterion is that $\log_2 L_m \geq \bar{D}_m$. This observation implies, in principle, that frozen bits should be allocated to prevent $\bar{D}_m$ from exceeding $\log_2 L_m$. Since computing $\bar{D}_m$ requires simulations with huge list sizes (if not unbounded) and the upper bound (5.7) ignores the affect of frozen bits, we use the lower bound (5.6) as the proxy for designs. Exemplary designs are provided in Section 5.1.2.

Figure 5.1 shows simulation results for a random instance from $\mathcal{RM}(3, 7)$ under SCL

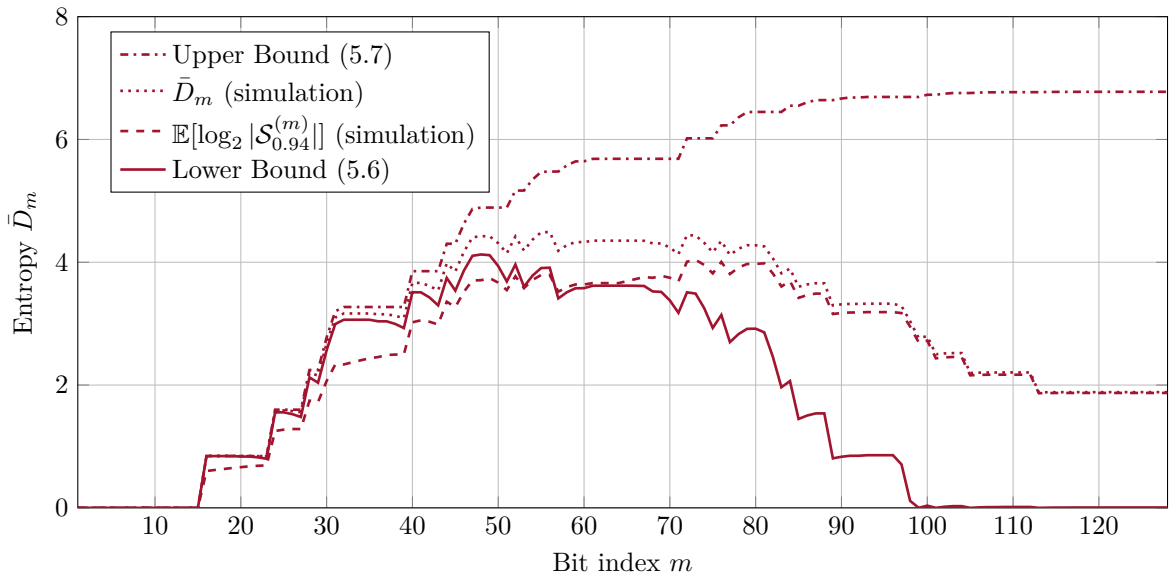

Figure 5.1.: $\bar{D}_m$ vs. m at $E_b/N_0 = 0.5$ dB for an instance from $\mathcal{DRM}(3, 7)$.

decoding with $L = 2^{14}$ at an SNR of $E_b/N_0 = 0.5$ together with the lower bound (5.6) and the upper bound (5.7) on $\bar{D}_m$. The figure validates the bounds (5.6), (5.7) and (5.10). Note that we set the parameter $\alpha = 0.94$ in (5.10) to provide a robust estimate by capturing the near misses, which happen if there are decoding paths with probabilities slightly smaller than that of the correct path. To understand this better, consider the proposed design where u_1 is an information bit. If one sets $\alpha = 1$, we get

$$\mathbb{E}\left[\log_2 |\mathcal{S}_1^{(1)}|\right] \approx 0.5 \log_2 1 + 0.5 \log_2 2 = 0.5 \tag{5.12}$$

which follows from $\bar{D}_m \approx 1$. Observing Figure 5.1, we obtain $\mathbb{E}\left[\log_2 |\mathcal{S}_{0.94}^{(1)}|\right] \approx 1$. Therefore, tightness of (5.10), especially at early decoding stages, is impacted by the choice of α.[2] Our numerical results show that the curve for $\mathbb{E}\left[\log_2 |\mathcal{S}_{\alpha}^{(m)}|\right]$ is more robust to changes in α at late decoding stages, i.e., for larger values of m. This means that the near misses happen at early decoding stages more often. In addition, observe that (5.6) closely tracks the simulation for $m \leq 50$ and it is easy to compute via standard methods, e.g., we used Gaussian approximation of density evolution (see Section 3.4.7), which further motivate using it for code design.

[2]One may further reduce the threshold α for inclusion to find a better match of $\mathbb{E}\left[\log_2 |\mathcal{S}_{\alpha}^{(m)}|\right]$ to $\bar{D}_m$ for the entire range.

5.1.1. Concentration of the Required Entropy in the List

We follow the steps of Section 4.4.2 to show that the required uncertainty accumulated by an SCL decoder to keep the correct path on the list concentrates around its mean $\bar{D}_m$ for sufficiently large block lengths. Observe that the sequence $H_0^{(m)}, H_1^{(m)}, \ldots, H_N^{(m)}$ defined by Proposition 4.10 forms a Doob's Martingale independent of the channel under consideration.

Proposition 5.2. Consider transmission over a BMSC satisfying $W(y|x) \geq \delta > 0, \forall y \in \mathcal{Y}$, $\forall x \in \{0,1\}$. Then for all $i \in [N]$ and all values y_1^N and $\tilde{y}_1^N$ such that $y_{\sim i} = \tilde{y}_{\sim i}$ and $y_i \neq \tilde{y}_i$, the conditional entropy satisfies

$$\left| d_m(y_1^N) - d_m(\tilde{y}_1^N) \right| \leq 4 \left| \log_2 \delta \right| . \tag{5.13}$$

As a result of Proposition 5.2, the following corollary provides a concentration for the logarithm of the list size required to achieve the performance of a code under MAP decoding when the transmission is over discrete output BMSCs. More precisely, we consider the normalized (with respect to the block length) deviation of the logarithm of the random list size, required to keep the correct codeword in the list.

Corollary 5.3. For transmission over a BMS channel satisfying $W(y|x) \geq \delta > 0, \forall y \in \mathcal{Y}$, $\forall x \in \{0,1\}$, the RV D_m, $m \in [N]$, for a particular random realization Y_1^N concentrates around its mean $\bar{D}_m$ for sufficiently large block lengths, i.e., for any $\beta > 0$, we have

$$\Pr\left\{ \frac{1}{N} |D_m - \bar{D}_m| > \beta \right\} \leq 2 \exp\left(- \frac{\beta^2}{32 \left| \log_2 \delta \right|^2} N \right) . \tag{5.14}$$

Proof. As for Corollary 4.12, apply the Azuma-Hoeffding inequality [42, Theorem 12.6] via Proposition 5.2 since the channel is memoryless. ■

For the case where $W(y|0)$ is a continuous probability density function on a compact set $\mathcal{Y} \subset \mathbb{R}$, the same proof applies with $\delta = \min_{y \in \mathcal{Y}} W(y|0)$. The proof does not extend to unbounded output alphabets.

Note also that the same idea in Remark 4.7 can also be applied to Corollary 5.3, but the value of δ must be modified as well. Still, the bounds of the type (5.14) are expected to be loose as mentioned in Remark 4.8.

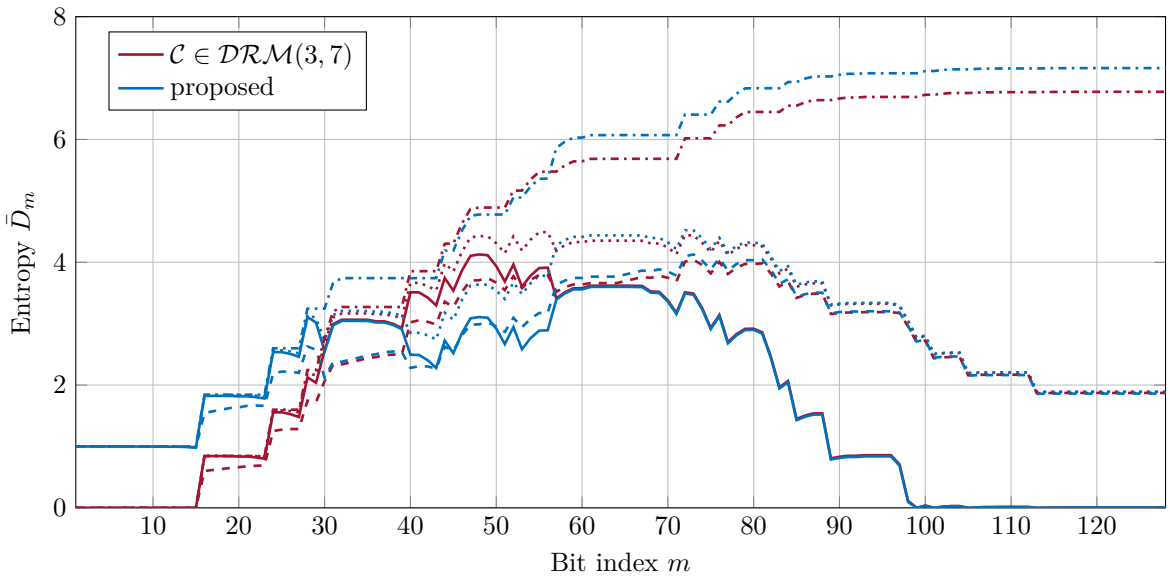

Figure 5.2.: $\bar{D}_m$ vs. m at $E_b/N_0 = 0.5$ dB for $(128, 64)$ codes (dash-dotted: upper bound (5.7), solid: lower bound (5.6), dotted: $\bar{D}_m$ via simulation, dashed: $\mathbb{E}[\log_2 |\mathcal{S}_{0.94}^{(m)}|]$ via simulation).

5.1.2. Numerical Results

This section provides simulation results for some constructions with dynamic frozen bits. Figure 5.2 shows simulation results for a random instance from $\mathcal{RM}(3, 7)$ (as in Figure 5.1) and a novel design (based on suggestions in Remark 5.2) under SCL decoding with $L = 2^{14}$ and $E_b/N_0 = 0.5$ together with the upper bound (5.7) and the lower bound (5.6) on $\bar{D}_m$. The proposed code takes the set $\mathcal{A}_{\text{RM}}$ of the $(128, 64)$ RM code and obtains a new set as $\mathcal{A} = (\mathcal{A}_{\text{RM}} \setminus \{30, 40\}) \cup \{1, 57\}$, i.e., $u_{\{30,40\}}$ are frozen and $u_{\{1,57\}}$ are unfrozen, where each frozen bit is still set to a random linear combination of preceding information bit(s). This helps especially for the considered list size $L = 32$ by inspecting Figure 5.3. The reason is illustrated by the lower bounds on $\bar{D}_m$ in Figure 5.2. In addition to having a smaller peak value, this peak occurs for the proposed design later than for the instance from $\mathcal{RM}(3, 7)$. This helps the proposed code to not loose the correct path at early decoding stages, and hence, to keep the correct path in the list towards the end for small list sizes, e.g., $L = 32$. We choose a relatively small E_b/N_0 for the analysis, e.g., close to the Shannon limit (~ 0.189 dB) for rate-$1/2$ codes, since we are after the events where the channel realization is not good enough to keep the correct path in the list with small list lengths, which happen rarely. If the list size is further decreased, then having u_1 as information bit can cause a

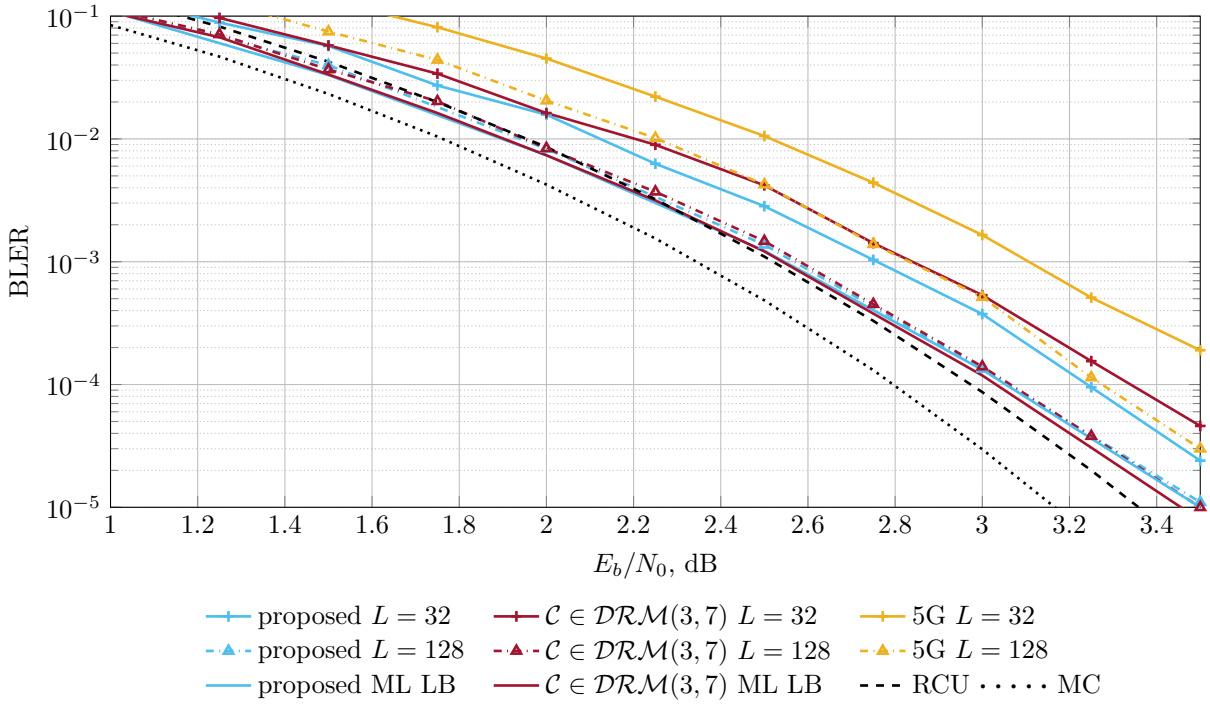

Figure 5.3.: BLER vs. SNR for $(128, 64)$ codes.

degradation.[3]

Figure 5.3 compares the performance of an instance from $\mathcal{RM}(3,7)$ and the proposed code. When an SCL decoder with $L = 128$ is considered, both codes perform within 0.15 dB of the RCU bound [143, Theorem 16] at a BLER of 10^{-5} and they almost match the simulation-based ML lower bounds [11], denoted as ML LB in the figure. When a smaller list size, e.g., $L = 32$, is adopted, the proposed code outperforms the instance from $\mathcal{RM}(3,7)$ especially at higher SNR values. This validates the analysis illustrated in Figure 5.2. The performance for the 5G design employing the CRC-11 defined by the generator polynomial $g(x) = x^{11} + x^{10} + x^9 + x^5 + 1$ [12, Section 5.1], [114] under SCL decoding with $L = 32$ is 0.4 dB worse than the proposed design at a BLER around 10^{-4}. When SCL decoding with $L = 128$ is considered, both codes outperform the 5G design by no less than 0.25 dB at all BLERs considered. Note that the PAC code perform very close to the dRM code under SCL decoding with $L \in \{32, 128\}$ [122, Figure 1]. The metaconverse (MC) bound [143, Theorem 28] is also provided.

Next, consider moderate-length codes, e.g., $(512, 256)$ codes, which are more challenging to design if the decoders are restricted to be of low- to moderate-complexity, i.e., $L \leq 1024$

[3] In particular, if $L = 1$, (5.12) suggests that the correct path would be lost roughly half of the time already after first decoding stage.

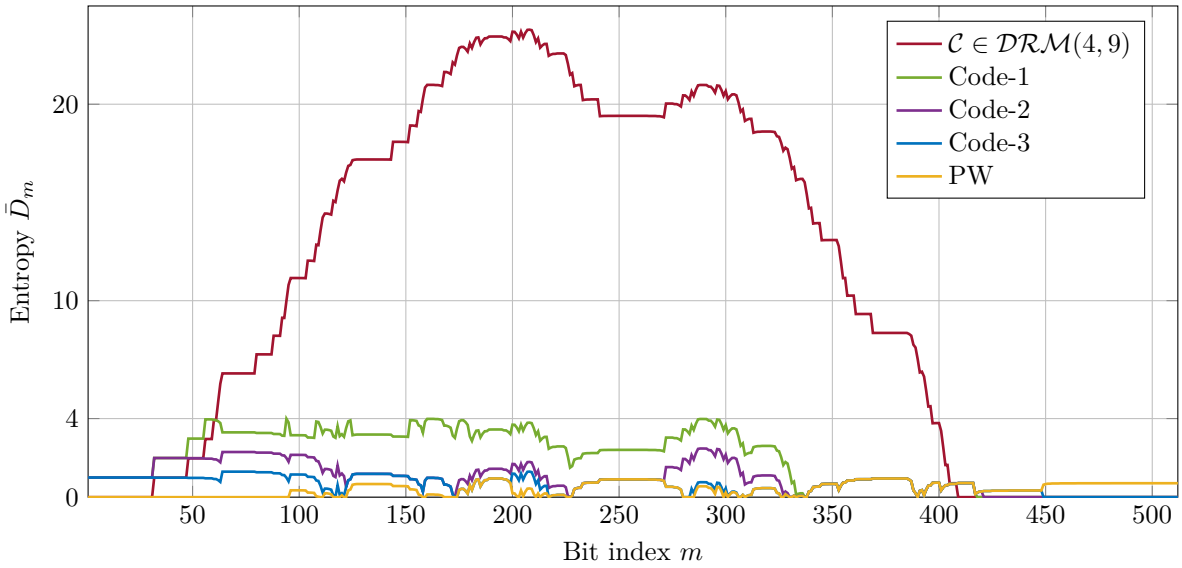

Figure 5.4.: Lower bound (5.6) on $\bar{D}_m$ at $E_b/N_0 = 0.5$ dB for $(512, 256)$ codes.

[36, Section 5.2] [144]. Figure 5.4 provides the bounds (5.6) for instances from $\mathcal{RM}(4, 9)$ and three novel designs. The peak of the lower bound corresponding to the $\mathcal{RM}(4, 9)$ gets close to 25 and recall that this quantity is related to the logarithm of the required list size. This explains why SCL decoding needs very large list sizes for a good performance when used for the RM$(4, 9)$ code (or any other instance from $\mathcal{RM}(4, 9)$) [14]. At the other extreme, the lower bound is provided for the construction based on the PW method with $\beta = 2^{1/4}$ [111], which is more suitable for SCL decoding with small list sizes. The idea behind the designs is similar to the length-128 case: we start from the information positions of an RM code, modify the positions to lower the peak value and keep the curve flat so that there is enough entropy kept on the list to make use of reliable frozen positions for a good performance. To this end, we also introduced u_1 as information bit in all three designs. This would harm the performance if the list size is very small, e.g., $L \leq 4$. In modifying the designs, we used the information positions from the PW construction. The information positions for the designs are provided in Appendix 5.2.

Figure 5.5 compares the performance of three designs under different list sizes. Code-1 requires the largest list size to get closer to its ML performance. When a large list size is adopted, e.g., $L \in \{512, 1024\}$, it performs within 0.4 dB of the RCU bound at BLERs around 10^{-6}, outperforming the non-binary LDPC code defined over $\mathbb{F}_{256}$ which has a higher decoding complexity [144]. Nevertheless, even with $L = 1024$, there is a non-negligible gap to the ML lower bound at BLERs above 10^{-6}. Code-2 is competitive for a wide range of list sizes, i.e., $L \in \{8, 32, 128, 1024\}$. In particular, it performs within 0.75

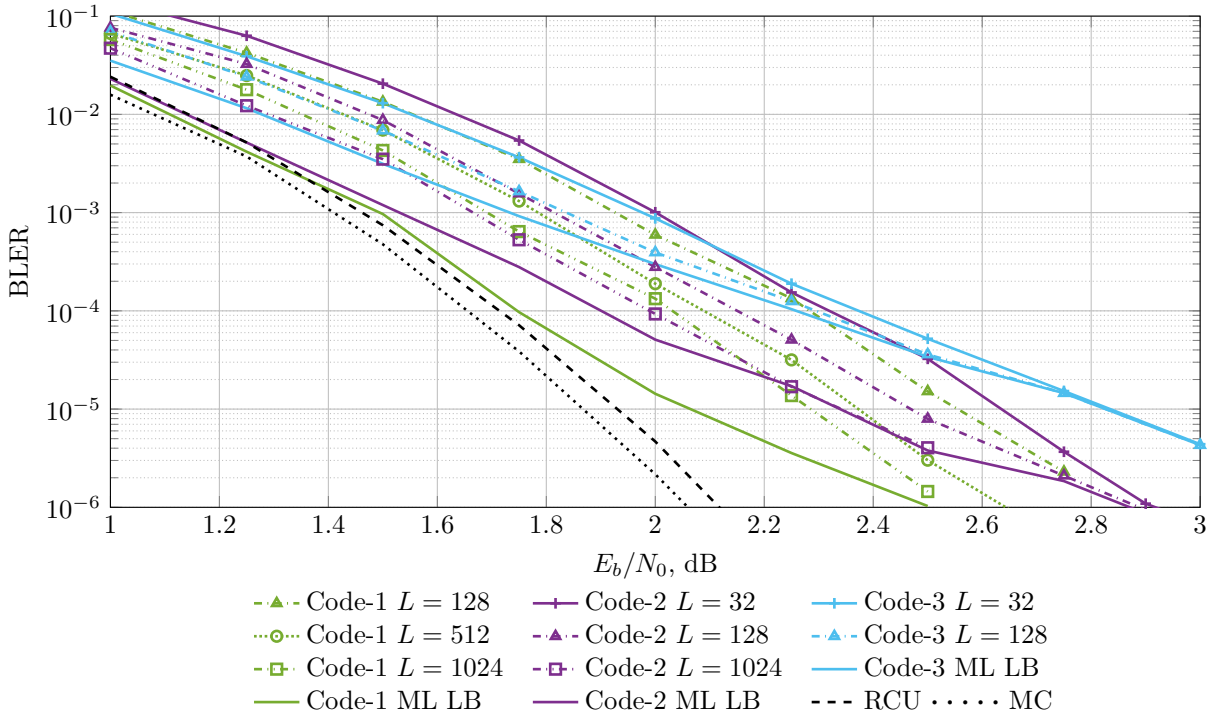

Figure 5.5.: BLER vs. SNR for $(512, 256)$ codes.

dB from the RCU bound down to the BLER of 10^{-6} under SCL decoding with $L = 32$. When an even smaller list size considered, e.g., $L = 8$, Code-3 performs the best, which is expected to perform well under small list sizes (see Figure 5.4), at all SNR values shown. With a relatively small list size, e.g., $L = 32$, Code-3 reaches to its ML performance at a BLER of 10^{-5} or less.

Figure 5.6 compares Code-1 to a polar code concatenated with the CRC-16 specified in [12, Section 5.1] with the generating polynomial $g_{16}(x) = x^{16} + x^{12} + x^5 + 1$. For the considered list sizes, Code-1 outperforms the modified polar code. In addition, Code-1 performs very similar to the 5G design when $L = 128$, and it provides sizeable gains, e.g., 0.35 dB, at BLERs close to 10^{-6} when $L = 1024$.

Figure 5.7 compares Code-2 to the 5G design based on PW with CRC-11, defined by polynomial $g(x) = x^{11} + x^{10} + x^9 + x^5 + 1$ [145, 146], where the former outperforms the latter under SCL decoding with the same list size $L \in [8, 1024]$ at all BLERs considered (down to 10^{-6}). Remarkably, Code-2 provides small improvement, (around 0.1 dB) even if the considered list size is $L = 32$ compared to the 5G code under SCL decoding with $L = 128$ at a BLER of 10^{-6}.

In Figure 5.8, Code-3 is also compared to a polar code concatenated with the CRC-7 with the generator polynomial $g_7(x) = x^7 + x^6 + x^5 + x^2 + 1$. The polynomial is taken

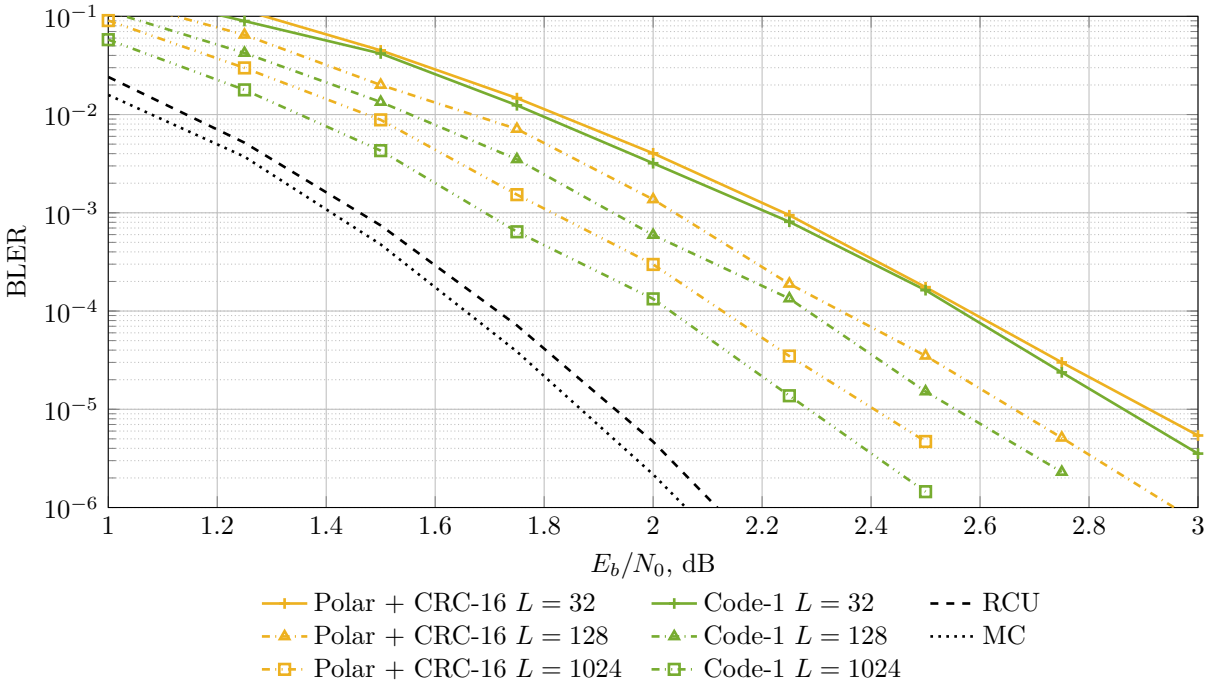

Figure 5.6.: BLER vs. SNR for $(512, 256)$ concatenated polar codes with CRC-16 compared to Code-1.

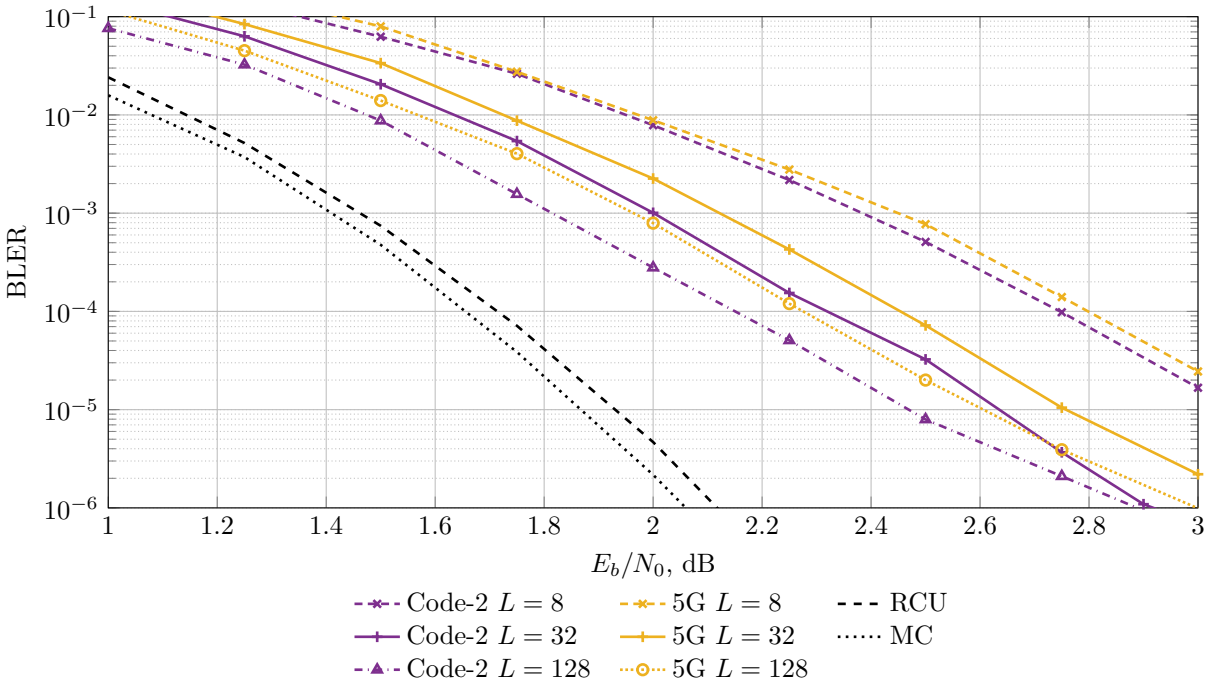

Figure 5.7.: BLER vs. SNR for $(512, 256)$ 5G codes compared to new designs.

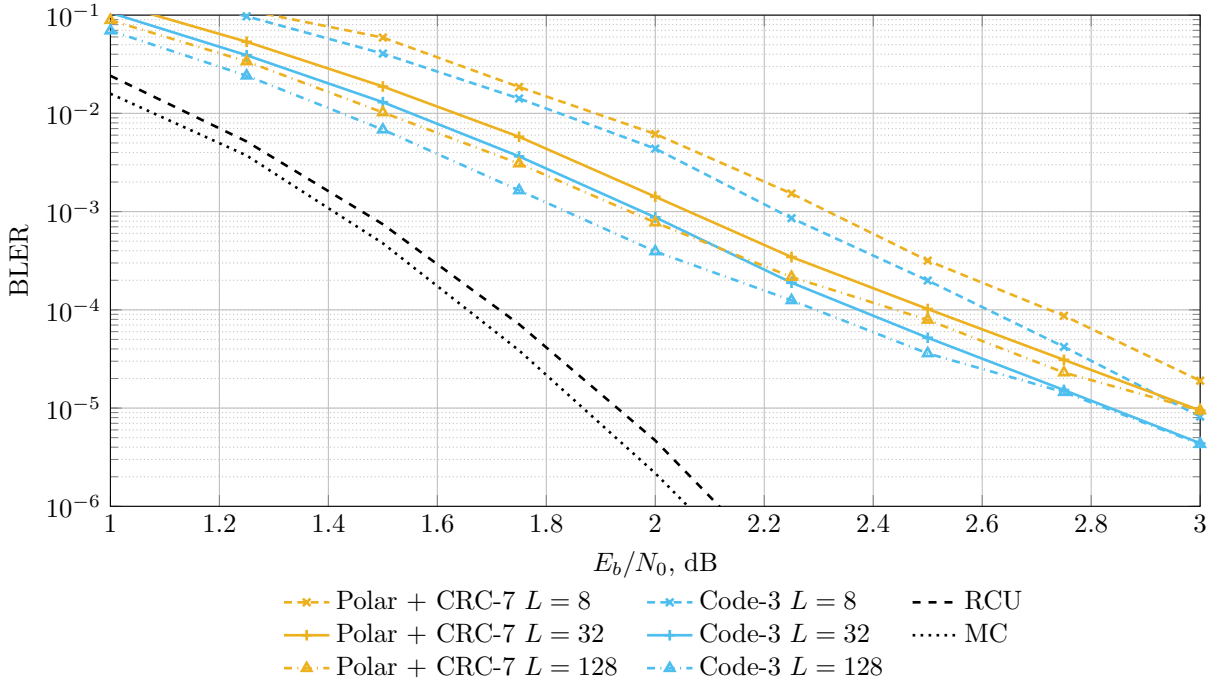

Figure 5.8.: BLER vs. SNR for $(512, 256)$ concatenated polar codes with CRC-7 compared to Code-3.

from [18] as it provides the best performance for the $(128, 64)$ case although it may not be optimal for the $(512, 256)$ code. Note finally that for all polar codes (irrespective of the chosen CRC length) provided as reference in this work the indices of frozen bits are selected according to the 5G standard [12, Section 5.1], [114].

5.2. Appendices

5.2.1. Proof of Theorem 5.1

Assume, w.l.o.g., that u_1^N and y_1^N are transmitted and observed, respectively. Then we have

$$\log_2 |\mathcal{S}_\alpha^{(m)}| = \log_2 \sum_{\tilde{u}_1^m} \mathbb{1}_{\left(P(\tilde{u}_{\mathcal{A}^{(m)}}|y_1^N, u_{\mathcal{F}^{(m)}}) \geq \alpha \cdot P(u_{\mathcal{A}^{(m)}}|y_1^N, u_{\mathcal{F}^{(m)}})\right)} \tag{5.15}$$

$$\leq -\log_2 \alpha \cdot P\left(u_{\mathcal{A}^{(m)}}|y_1^N, u_{\mathcal{F}^{(m)}}\right) \tag{5.16}$$

where (5.15) follows from Lemma 3.5, i.e., $p^m_{\boldsymbol{G}^{[n]}}(y_1^N, \tilde{u}_1^m) \propto P\left(\tilde{u}_{\mathcal{A}^{(m)}}|y_1^N, \tilde{u}_{\mathcal{F}^{(m)}}\right)$, and Bayes' rule, and (5.16) because if there are more than $\left(\alpha \cdot P\left(u_{\mathcal{A}^{(m)}}|y_1^N, u_{\mathcal{F}^{(m)}}\right)\right)^{-1}$ sequences $\tilde{u}_{\mathcal{A}}$

with probability $\alpha \cdot P\left(u_{\mathcal{A}^{(m)}}|y_1^N, u_{\mathcal{F}^{(m)}}\right)$ then the total probability exceeds 1. As the inequality is valid for any pair u_1^N and y_1^N, taking the expectation over all u_1^m and y_1^N yields the stated result. ■

5.2.2. Proof of Proposition 5.2

In the following, the probabilities are denoted as

$$p\left(u_1^m, x_1^N, y_1^N\right) = \Pr\left\{U_1^m = u_1^m, X_1^N = x_1^N, Y_1^N = y_1^N\right\}.$$

The proof starts by writing

$$\frac{P\left(u_{\mathcal{A}^{(m)}}|y_1^N, u_{\mathcal{F}^{(m)}}\right)}{P\left(u_{\mathcal{A}^{(m)}}|\tilde{y}_1^N, u_{\mathcal{F}^{(m)}}\right)} = \frac{p\left(u_1^m, y_1^N\right)}{p\left(y_1^N, u_{\mathcal{F}^{(m)}}\right)} \cdot \frac{p\left(\tilde{y}_1^N, u_{\mathcal{F}^{(m)}}\right)}{p\left(u_1^m, \tilde{y}_1^N\right)} \tag{5.17}$$

$$= \frac{\sum_{x_1^N} p\left(u_1^m, x_1^N, y_1^N\right)}{\sum_{x_1^N} p\left(y_1^N, x_1^N, u_{\mathcal{F}^{(m)}}\right)} \cdot \frac{\sum_{x_1^N} p\left(\tilde{y}_1^N, x_1^N, u_{\mathcal{F}^{(m)}}\right)}{\sum_{x_1^N} p\left(u_1^m, x_1^N, \tilde{y}_1^N\right)} \tag{5.18}$$

$$= \frac{\sum_{x_i} W(y_i|x_i) \sum_{x_{\sim i}} p\left(u_1^m, x_1^N, y_{\sim i}\right)}{\sum_{x_i} W(\tilde{y}_i|x_i) \sum_{x_{\sim i}} p\left(u_1^m, x_1^N, y_{\sim i}\right)} \cdot \frac{\sum_{x_i} W(\tilde{y}_i|x_i) \sum_{x_{\sim i}} p(y_{\sim i}, x_1^N, u_{\mathcal{F}^{(m)}})}{\sum_{x_i} W(y_i|x_i) \sum_{x_{\sim i}} p(y_{\sim i}, x_1^N, u_{\mathcal{F}^{(m)}})} \tag{5.19}$$

$$= \frac{\sum_{x_i} W(y_i|x_i) p(u_1^m, x_i, y_{\sim i})}{\sum_{x_i} W(\tilde{y}_i|x_i) p(u_1^m, x_i, y_{\sim i})} \cdot \frac{\sum_{x_i} W(\tilde{y}_i|x_i) p\left(y_{\sim i}, x_i, u_{\mathcal{F}^{(m)}}\right)}{\sum_{x_i} W(y_i|x_i) p\left(y_{\sim i}, x_i, u_{\mathcal{F}^{(m)}}\right)} \tag{5.20}$$

where (5.17) follows from Bayes' rule, (5.18) and (5.20) from the law of total probability, and (5.19) from rearranging the sum, Bayes' rule and noting that $Y_i - X_i - (U_1^m, X_{\sim i}, Y_{\sim i})$ forms a Markov chain. We now take the logarithm and absolute value of both sides in (5.20). Applying the triangle inequality $|a+b| \leq |a| + |b|$ and upper bounding

$$\left|\log_2 \sum_{x_i} W(y_i|x_i) p(u_1^m, x_i, y_{\sim i})\right| \leq \max_y \left|\log_2 W(y|0)\right| \tag{5.21}$$

we conclude that

$$\left|\log_2 \frac{P\left(u_{\mathcal{A}^{(m)}}|y_1^N, u_{\mathcal{F}^{(m)}}\right)}{P\left(u_{\mathcal{A}^{(m)}}|\tilde{y}_1^N, u_{\mathcal{F}^{(m)}}\right)}\right| \leq 4 \max_y \left|\log_2 W(y|0)\right|. \tag{5.22}$$

(5.22) is valid for any u_1^m, and averaging over all u_1^m, combined with the Jensen's inequality, leads to (5.13) since $W(y|0) \geq \delta, \forall y \in \mathcal{Y}$. ■

5.2.3. Code Designs

Let $\mathcal{A}_{\mathrm{PW}}$ denote the set of information positions of the $(512, 256)$ polar code designed according to PW with $\beta = 2^{1/4}$ [111]. The sets $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ corresponding to Code-1, Code-2 and Code-3, respectively, are given as follows:

$$\mathcal{A}_3 = (\mathcal{A}_{\mathrm{PW}} \setminus \{449, 450, 451, 453, 457, 465, 481\}) \tag{5.23}$$

$$\cup \{1, 64, 118, 122, 159, 200, 284\}, \tag{5.24}$$

$$\mathcal{A}_2 = (\mathcal{A}_3 \setminus \{122, 421, 425, 433\}) \cup \{32, 174, 272, 280\}, \tag{5.25}$$

$$\mathcal{A}_1 = (\mathcal{A}_2 \setminus \{64, 96, 125, 180, 418, 419\}) \tag{5.26}$$

$$\cup \{48, 56, 94, 108, 122, 152\}. \tag{5.27}$$

For all codes, each frozen bit is set to a random linear combination of the preceding information bit(s). The performance curves will also be available on the website [145].

6

Successive Cancellation List Decoding of Product Codes

In this chapter, we focus on product codes based on RM and/or SPC component codes by establishing connections to (multi-kernel) polar codes. Product codes based on RM component codes, namely RM product codes, have been considered in [68, 147] where the component codes are extended Hamming and SPC codes whose length is a power of 2. This choice of component codes has been considered in wireless communication systems (see, e.g., [73, 148, 149]) thanks to the availability of low-complexity SISO decoders for SPC and extended Hamming codes.[1] Product codes with SPC component codes, namely SPC product codes, are considered in [83, 150, 151], where the interest was mainly their performance and WEs.

6.1. A First Connection: Recursive Encoding

Recall the generator matrix of an m-dimensional product code given by (3.14). Alternatively, we can define a generator matrix similar to (3.48) recursively as follows. Let the binary vectors v_1^K and x_1^N be the K-bit message to be encoded and the corresponding N-bit codeword, respectively, where the relation between them is $x_1^N = v_1^K \mathbf{G}^{[n]}$, where $\mathbf{G}^{[n]}$ is the generator matrix of the product code with dimension n. We obtain $\mathbf{G}^{[n]}$ recursively

[1] Extended Hamming codes can be efficiently decoded by exploiting their trellis representation or by employing the sub-optimum Chase-Pyndiah decoder proposed in [68].

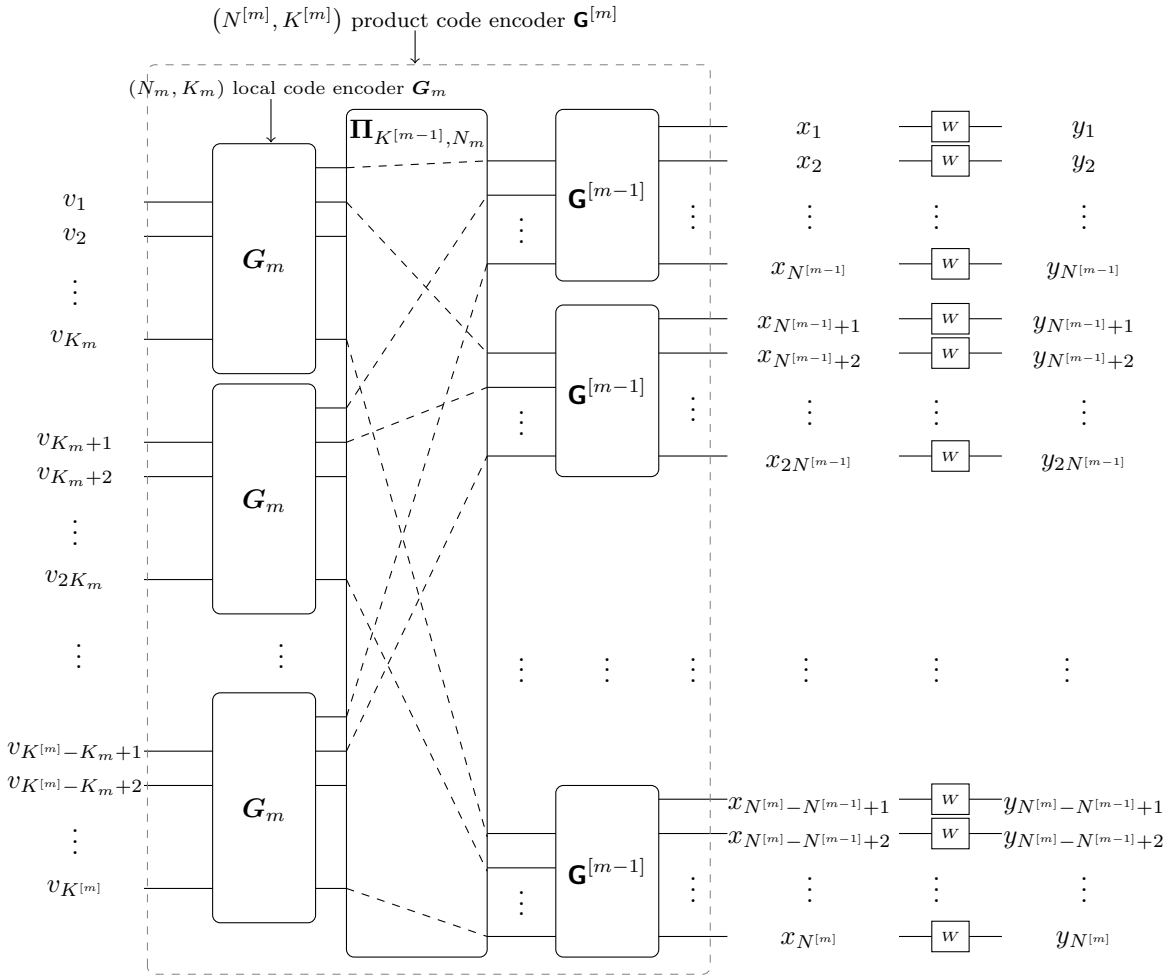

Figure 6.1.: Encoding using an (N, K) product code with m dimensions.

as

$$\mathbf{G}^{[n]} = \left(\boldsymbol{I}_{K^{[n-1]}} \otimes \boldsymbol{G}_n\right) \boldsymbol{\Pi}_{K^{[n-1]},N_n} \left(\boldsymbol{I}_{N_n} \otimes \mathbf{G}^{[n-1]}\right) \tag{6.1}$$

where $\mathbf{G}^{[0]} \triangleq 1$, $K^{[m-1]} \triangleq \prod_{i=1}^{m-1} K_i$ with $K^{[0]} = 1$ (observe that $K^{[m]} = K$) and $\boldsymbol{\Pi}_{a,b}$ is the $ab \times ab$ perfect shuffle matrix as defined by (2.10). Note also that $N = N^{[m]} \triangleq \prod_{i=1}^{m} N_i$ with $N^{[0]} = 1$. Figure 6.1 depicts the encoding with product codes where the encoding recursion is based on (6.1).

To see the relation between $\boldsymbol{G}$ and $\boldsymbol{G}^{[m]}$, write

$$\begin{aligned} \mathbf{G}^{[m]} &= \left(\boldsymbol{I}_{K^{[m-1]}} \otimes \boldsymbol{G}_m\right) \left(\mathbf{G}^{[m-1]} \otimes \boldsymbol{I}_{N_m}\right) \boldsymbol{\Pi}_{N^{[m-1]},N_m} && (6.2) \\ &= \left(\mathbf{G}^{[m-1]} \otimes \boldsymbol{G}_m\right) \boldsymbol{\Pi}_{N^{[m-1]},N_m} && (6.3) \end{aligned}$$

$$= \left((\boldsymbol{I}_{K^{[m-2]}} \otimes \boldsymbol{G}_{m-1}) \left(\mathbf{G}^{[m-2]} \otimes \boldsymbol{I}_{N_{m-1}} \right) \boldsymbol{\Pi}_{N^{[m-2]},N_{m-1}} \otimes \boldsymbol{G}_m \right) \boldsymbol{\Pi}_{N^{[m-1]},N_m} \tag{6.4}$$

$$= \left(\mathbf{G}^{[m-2]} \otimes \boldsymbol{G}_{m-1} \otimes \boldsymbol{G}_m \right) \left(\boldsymbol{\Pi}_{N^{[m-2]},N_{m-1}} \otimes \boldsymbol{I}_{N_m} \right) \boldsymbol{\Pi}_{N^{[m-1]},N_m} \tag{6.5}$$

$$= (\boldsymbol{G}_1 \otimes \boldsymbol{G}_2 \otimes \ldots \otimes \boldsymbol{G}_m) \prod_{i=1}^{m} \left(\boldsymbol{\Pi}_{N^{[i-1]},N_i} \otimes \boldsymbol{I}_{N^{[m]}/N^{[i]}} \right) \tag{6.6}$$

where (6.2) follows from applying the identity

$$\boldsymbol{\Pi}_{K^{[m-1]},N_m} \left(\boldsymbol{I}_{N_m} \otimes \mathbf{G}^{[m-1]} \right) = \left(\mathbf{G}^{[m-1]} \otimes \boldsymbol{I}_{N_m} \right) \boldsymbol{\Pi}_{N^{[m-1]},N_m}. \tag{6.7}$$

Equation (6.3) follows from the mixed-product identity and (6.4) from writing $\mathbf{G}^{[m-1]}$ as (6.1). Equations (6.5) and (6.6) follow by applying similar steps recursively. Since the product of an arbitrary number of permutation matrices yields another permutation matrix, we conclude that $\boldsymbol{G}$ and $\mathbf{G}^{[m]}$ are equivalent up to a column permutation for all $m \geq 1$.

Remark 6.1. An inspection of Figure 6.1 suggests an SC-based decoding of product codes by using their recursive structure combined with local decoding operations imposed by the component codes.[2] The motivation is because the SCL decoding with large list size (or SCI decoding for the BEC) enables near-optimum decoding of polar codes [11]. This was demonstrated by computing a numerical lower bound on the ML decoding BLEP via Monte Carlo simulation, where the correct codeword is introduced artificially in the final list prior to the final selection. If the simulated BLEP is close to the numerical ML decoding lower bound, then increasing the list size L would not yield much improvement. We shall see in Sections 6.2 and 6.3 that the same principle applies to SCL decoding of product codes. In particular, we start with RM product codes, which can be represented as subcodes of larger RM codes, and then we study SPC product codes and their relations to multi-kernel polar codes.

The relation between $\boldsymbol{G}$ as described in (3.14) and $\mathbf{G}^{[m]}$ is similar to the relation between the polar transform matrices $\boldsymbol{K}_2^{\otimes n}$ and $\boldsymbol{G}^{[n]}$, which are equivalent up to the bit-reversal permutation of the columns. This means that they can be both used to describe polar codes as discussed in Section 3.4.2. In the description of RM product codes, we use $\boldsymbol{K}_2^{\otimes n}$ for notational convenience.

[2]The efficiency of such SC decoding depends on the component code structure. In general, it is not efficient to decode a linear block code using SC decoding [152] or even a polar code based on an arbitrary kernel, e.g., see [125] and references therein.

6.2. Reed-Muller Product Codes

In this section, our main interest is RM product codes. This class of codes includes product codes with extended Hamming component codes, which are used for high-rate applications requiring very low bit or block error rates [147]. Remarkably, a 2-dimensional product code with a rate ≈ 0.981 with simple Hamming component codes was already suggested in [67] and operates 0.27 dB away from the Shannon limit, and this code can be seen as a punctured RM product code.

We mention that RM product codes were studied by [153] explicitly. The authors showed that the resulting product code is a subcode of a longer RM code, which enabled methods to modify the product code to obtain higher rate codes with the same minimum distance. In the following, we revisit this representation, that we call a polar code representation, to introduce SCL decoding for RM product codes, which approaches the ML performance of the codes with small to moderate list sizes up to blocklength in the order of 1024 bits. The simulation results are given for plain RM product codes and also for concatenations via high-rate outer codes. The simulations are accompanied by a distance spectrum analysis restricted to the minimum distance terms.

The results in this section can be extended to product codes with polar component codes, which are studied, e.g., in [154, 155]. In [154], the authors proposed a construction that allows interpreting a polar code as a 2-dimensional product code. The construction allows using SC decoders row- and column-wise to reduce the complexity/latency with respect to the case where SC decoding is performed over the larger polar code. Similarly, [155] proposed product code designs where the component codes are polar codes. The focus was on reducing the latency by proposing a two-stage decoding where the SC decoder of the large polar code is used only if the iterative product code decoder does not converge to a valid codeword.

6.2.1. Polar Code Representation

Consider an RM(r, n) code and let w_1^K contain the indices of its frozen set $\mathcal{F}$ in an ascending order, i.e., $w_1 < w_2 < \ldots < w_K$. We define the entry at the ith row and jth column of the $K \times N$ matrix $\boldsymbol{F}$ as

$$F_{i,j} = \begin{cases} 1 \text{ if } j = w_i \\ 0 \text{ otherwise.} \end{cases} \tag{6.8}$$

A generator matrix $\boldsymbol{G}$ of RM(r, n) is

$$\boldsymbol{G} = \boldsymbol{F}\boldsymbol{K}_2^{\otimes n}. \tag{6.9}$$

The 2^n-dimensional frozen bit vector $\boldsymbol{f}$, where $f_i = 0$ for all $i \in \mathcal{F}$ and $f_i = 1$ otherwise, is also obtained by summing all the rows of $\boldsymbol{F}$, i.e.,

$$\boldsymbol{f} = \boldsymbol{e}\boldsymbol{F} \tag{6.10}$$

where $\boldsymbol{e}$ is the length-K all-ones vector.

Consider an m-dimensional product code where the ℓ-th component code $\mathcal{C}_\ell$ is an RM(r_ℓ, n_ℓ) code with the corresponding row-selecting matrix $\boldsymbol{F}_\ell$ (frozen bit vector $\boldsymbol{f}_\ell$). This is a subcode of the RM$(r_1 + r_2 + \ldots + r_m, n_1 + n_2 + \ldots + n_m)$ code [153, Corollary 2]. For the ℓ-th component code, the generator matrix is obtained via (6.9) as

$$\boldsymbol{G}_\ell = \boldsymbol{F}_\ell \boldsymbol{K}_2^{\otimes n_\ell}. \tag{6.11}$$

The frozen bit vector for the resulting product code is determined by those of the component codes, as stated next.

Proposition 6.1. The generator matrix of the product code obtained by iterating m RM codes RM(r_1, n_1), …, RM(r_m, n_m) is given by

$$\boldsymbol{G} = \boldsymbol{F}\boldsymbol{K}_2^{\otimes(n_1+\ldots+n_m)} \tag{6.12}$$

where $\boldsymbol{F} = \boldsymbol{F}_1 \otimes \boldsymbol{F}_2 \otimes \ldots \otimes \boldsymbol{F}_m$, resulting in

$$\boldsymbol{f} = \boldsymbol{f}_1 \otimes \ldots \otimes \boldsymbol{f}_m. \tag{6.13}$$

Proof. Apply the mixed-product property (2.9) to the combination of (3.14), (6.10) and (6.11):

$$(\boldsymbol{F}_1\boldsymbol{K}_2^{\otimes n_1}) \otimes \cdots \otimes (\boldsymbol{F}_m\boldsymbol{K}_2^{\otimes n_m}) = (\boldsymbol{F}_1 \otimes \cdots \otimes \boldsymbol{F}_m)(\boldsymbol{K}_2^{\otimes n_1} \otimes \cdots \otimes \boldsymbol{K}_2^{\otimes n_m}). \tag{6.14}$$

■

Example 6.1. Consider a two-dimensional product code with a $(2, 1)$ repetition code and

a $(4, 3)$ SPC code as component codes with $\boldsymbol{F}_1 = \begin{bmatrix} 0 & 1 \end{bmatrix}$ yielding $\boldsymbol{f}_1 = (0, 1)$ and

$$\boldsymbol{F}_2 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.15}$$

yielding $\boldsymbol{f}_2 = (0, 1, 1, 1)$. The product code generator matrix is obtained via (6.12) where

$$\boldsymbol{F} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.16}$$

with $\boldsymbol{f} = (0, 0, 0, 0, 0, 1, 1, 1)$ via (6.13).

Proposition 1 provides an interpretation of RM product codes as RM subcodes, where the frozen bit positions are given as (6.13). This enables using the SC-based decoding algorithms for RM and polar codes, e.g., SCL, SCI or SCOS decoding [6, 11, 28, 38, 99], to decode this class of product codes.

Remark 6.2. Equivalent codes, defined by different frozen bit vectors, can be obtained via the Kronecker product which is not commutative. The definition of the different frozen bits vectors is related to the order with which the component codes' generator matrices are iterated in (3.14). We follow Remark 5.2 and choose a representation that exhibits good performance under SCL decoding with small list sizes to decode the resulting code.

Numerical Results

We provide simulation results for two RM product codes over the BAWGNC. As a reference, the performance under BP is provided with a maximum number of iterations set to 100. The component codes are decoded by MAP SISO decoding over the component code trellis. The truncated union bound (TUB) in the tighter form of [147, Equation 3] is also provided. Note that when using an RM product code to transmit over a BMSC, the BLEP under ML decoding can be well approximated by the TUB already at moderate BLERs [147].

The first product code is the $(128, 77)$ code from Table 3.1, whose performance is depicted in Figure 6.2. In particular, the component codes $\mathcal{C}_1$ and $\mathcal{C}_2$ are $(16, 11)$ extended Hamming and $(8, 7)$ SPC codes, respectively. List decoding with $L = 4$ suffices to approach the performance of BP decoding. With $L = 8$, the SCL decoder tightly matches the ML lower

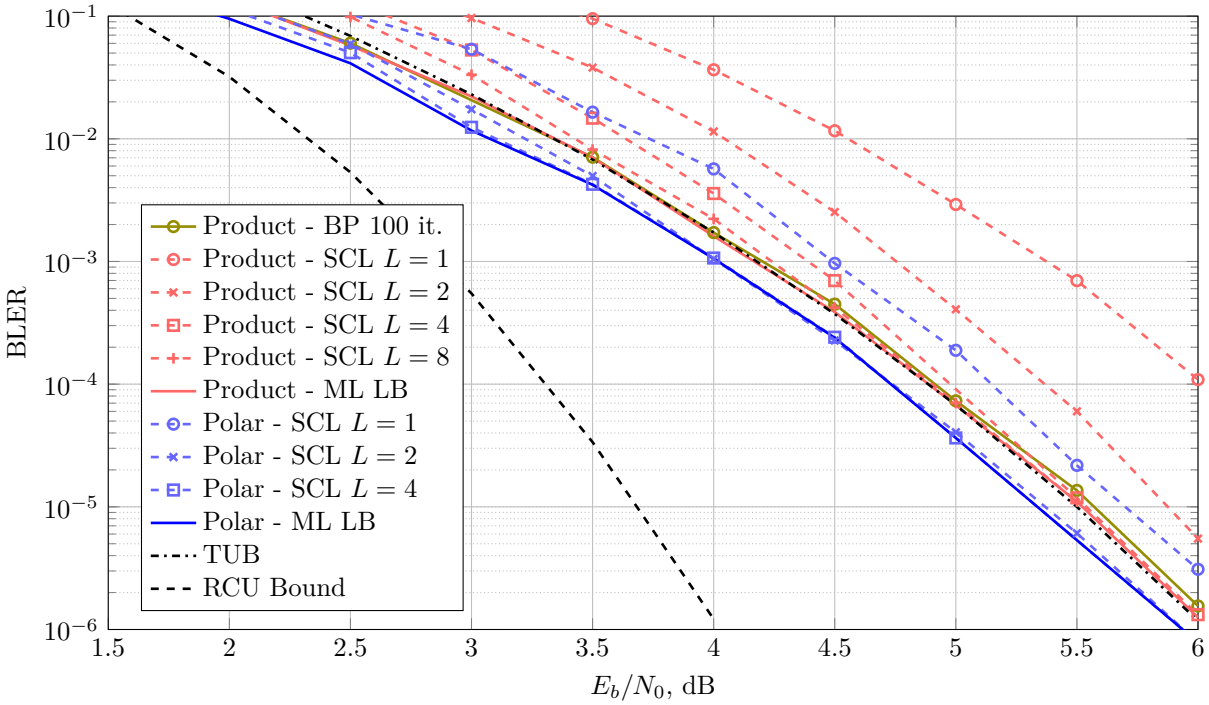

Figure 6.2.: BLER vs. SNR under SCL decoding with various list sizes for the $(128, 77)$ product code (with the $(16, 11)$ extended Hamming and the $(8, 7)$ SPC component codes) compared with BP decoding. As reference, the performance of a $(128, 77)$ polar code is also provided under SCL decoding with various list sizes.

bound below BLER 10^{-2}. The gap to the RCU bound is limited to 2 dB at BLER 10^{-6}. In the same figure, the performance of a $(128, 77)$ polar code under SCL decoding is provided. While the ML decoding performance of the polar code and of the product code are very close, the polar code requires a smaller list size to saturate.

A longer $(1024, 693)$ product code has been constructed by choosing $\mathcal{C}_1$ and $\mathcal{C}_2$ to be the $(16, 11)$ extended Hamming and the $(64, 63)$ SPC codes, respectively. The results of the product codes are shown in Figure 6.3. In particular, the required list size increases due to the sub-optimal choice of a large number of non-frozen bits enforced by the specific product code construction for long blocklengths.

6.2.2. Concatenation with a High-Rate Outer Code

Following [11], we analyze the performance under SCL decoding of RM product codes concatenated with a high-rate outer code. One reason to analyze such concatenation is

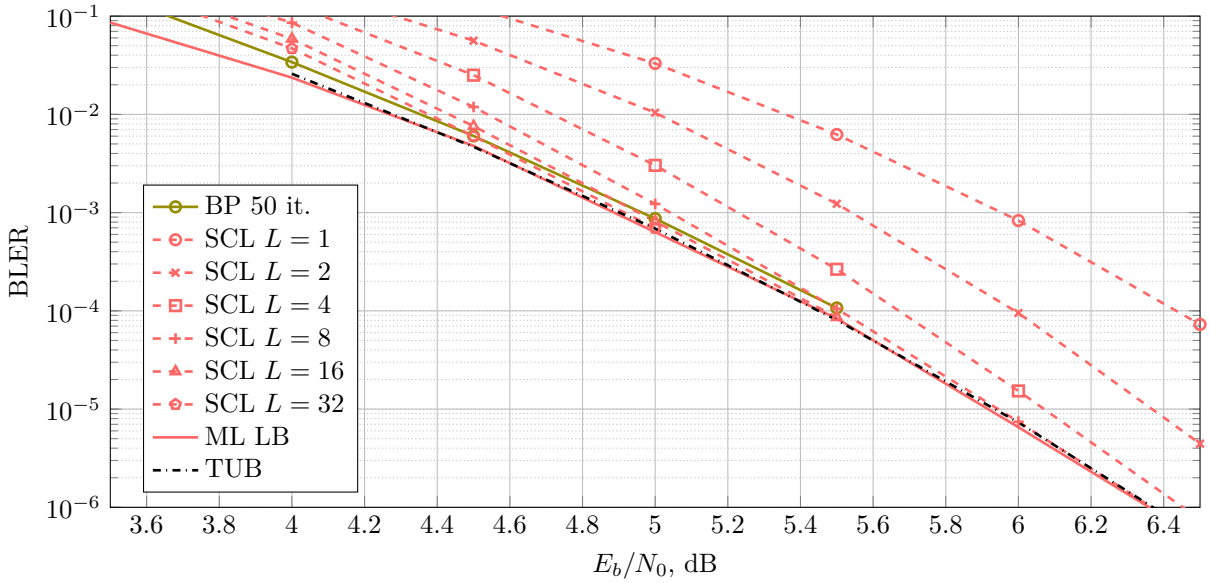

Figure 6.3.: BLER vs. SNR under SCL decoding with various list sizes for $(1024, 693)$ product code (with the $(16, 11)$ extended Hamming and the $(64, 63)$ SPC component codes), compared with the BP decoding.

(besides in the obviously expected performance improvement) because practical schemes employing product codes make use of an error detection code to protect the product code information message. We may hence sacrifice (part of) the error detection capability for a coding gain. Following the construction adopted in the IEEE 802.16 standard [149], we consider product codes with systematic encoding.

For the product codes we consider, large gains are expected by adding a high-rate outer code, especially at moderate to low BLERs. This follows because RM product codes have a fairly large multiplicity of minimum weight codewords, as already observed in Table 3.1. By a suitable choice of the outer code, the multiplicity of minimum weight codewords is lowered considerably. This can yield significant gains with sub-optimum BP/SCL decoding. We analyze the impact of the outer code for a concatenated ensemble by studying the weight distribution and focusing on the minimum weight terms only.

Average Weight Distribution of Concatenated Ensembles

Consider the concatenation of an $(N_\mathsf{i}, K_\mathsf{i})$ inner product code $\mathcal{C}_\mathsf{i}$ with an $(N_\mathsf{o}, K_\mathsf{o})$ high-rate outer code $\mathcal{C}_\mathsf{o}$. Note that $K_\mathsf{i} = N_\mathsf{o}$. Let d be the minimum distance of the inner product code. We further define the generator matrices of $\mathcal{C}_\mathsf{i}$ and $\mathcal{C}_\mathsf{o}$ as $\boldsymbol{G}_\mathsf{i}$ and $\boldsymbol{G}_\mathsf{o}$, respectively.

Definition 6.1 (Concatenated Ensemble)**.** The (serially) concatenated ensemble $\mathscr{C}(\mathcal{C}_\mathsf{o}, \mathcal{C}_\mathsf{i})$

is the set of all codes with a generator matrix of the form $\boldsymbol{G} = \boldsymbol{G}_{\mathsf{o}}\boldsymbol{\Pi}\boldsymbol{G}_{\mathsf{i}}$, where $\boldsymbol{\Pi}$ is an $N_{\mathsf{o}} \times N_{\mathsf{o}}$ permutation matrix.

Suppose the outer code WE A^{o}_j and the IOWE of the inner product code $A^{\mathrm{IO,i}}_{j,w}$ are known. We are interested in the WE of the concatenated code for a given permutation matrix $\boldsymbol{\Pi}$ (e.g., $\boldsymbol{\Pi} = \boldsymbol{I}_{N_{\mathsf{o}}}$) that interleaves the output of the outer encoder. This requires enumerating all possible input vectors of length K_{o}, which is not possible in practice. For this reason, the outer code is typically analyzed for a concatenated ensemble by assuming that the interleaver is distributed uniformly over all possible $\binom{N_{\mathsf{o}}}{i}$ permutations [156]. The average WE of the ensemble is then

$$\bar{A}_w = \sum_{j=0}^{N_{\mathsf{o}}} \frac{A^{\mathsf{o}}_j \cdot A^{\mathrm{IO,i}}_{j,w}}{\binom{N_{\mathsf{o}}}{j}} \tag{6.17}$$

where $\bar{A}_w$ is the average multiplicity of codewords $\boldsymbol{x}$ with $w(\boldsymbol{x}) = w$.

For the considered product codes, the IOWEs $A^{\mathrm{IO,i}}_{j,w}$ are not known in general.[3] Fortunately, the minimum-weight IOWE $A^{\mathrm{IO,i}}_{j,d}$ is easily computed from the IOWEs of the component codes if $\boldsymbol{G}_{\mathsf{i}}$ is in systematic form [82, Theorem 1]. Then, the expected multiplicity of weight-d codewords $\bar{A}_d$ can be used in TUB to obtain an estimate of the ensemble average BLEP in the low BLEP regime.

Example 6.2. Consider the $(128, 77)$ systematic product code with $(16, 11)$ extended Hamming and $(8, 7)$ SPC component codes, which has minimum distance 8 with a multiplicity of 3920. The code is concatenated with an outer CRC-7 code with generator polynomial $g(x) = x^7 + x^3 + 1$. The resulting code is a member of a concatenated ensemble with an expected number of weight-8 codewords given by $\bar{A}_8 \simeq 26.4$, i.e., the multiplicity of weight-8 codewords is reduced, on average, by two orders of magnitude. The contribution of these codewords to the ensemble average BLEP is reduced significantly. Hence, the TUB shall be approached only at low BLERs.

The generator matrix of the product code constructed according to Proposition 6.1 is not in systematic form. Suppose the inner code generator matrix $\boldsymbol{G}_{\mathsf{i}}$ is systematic. The overall code generator matrix can be written as

$$\boldsymbol{G} = \boldsymbol{G}_{\mathsf{o}}\boldsymbol{\Pi}\boldsymbol{G}_{\mathsf{i}} = \boldsymbol{G}_{\mathsf{o}}\boldsymbol{\Pi}\boldsymbol{S}\boldsymbol{G}_{\mathsf{i,nsys}} = \boldsymbol{G}_{\mathsf{mo}}\boldsymbol{G}_{\mathsf{i,nsys}} \tag{6.18}$$

[3] For short product codes where at least one of the component codes is an SPC code, we will develop an efficient method to compute the IOWEs in Section 6.3.5.

where $\boldsymbol{\Pi}$ is the interleaver matrix, $\boldsymbol{S}$ is a $K_{\mathsf{i}} \times K_{\mathsf{i}}$ non-singular matrix and $\boldsymbol{G}_{\mathsf{i,nsys}} = \boldsymbol{S}^{-1}\boldsymbol{G}_{\mathsf{i}}$ is the non-systematic generator matrix according to Proposition 6.1. Furthermore, $\boldsymbol{G}_{\mathsf{mo}}$ is defined to be the product $\boldsymbol{G}_{\mathsf{o}}\boldsymbol{\Pi}\boldsymbol{S}$. Thus, the SCL decoding can be used for the inner product code, where the modified outer code with generator matrix $\boldsymbol{G}_{\mathsf{mo}}$ is used to test the codewords of the final list prior to a decision.

Numerical Results

Figure 6.4 shows the performance by concatenating the $(128, 77)$ product code of Figure 6.2 with an outer CRC code with generator polynomial $g(x) = x^7+x^3+1$, leading to a $(128, 70)$ code. The performance of the concatenated scheme is provided for two interleavers between the inner and outer code. The label "no interleaver" denotes the trivial interleaver, i.e., $\boldsymbol{\Pi}$ is chosen to be the $K_{\mathsf{i}} \times K_{\mathsf{i}}$ identity matrix, while in the second case a random interleaver is used. The concatenation with the trivial interleaver performs remarkably well under SCL decoding. At a BLER of 10^{-6}, SCL decoding of the concatenated code achieves gains up to 1.4 dB over the original product code. The gains attained by SCL decoding over BP decoding[4] range from 1 dB at a BLER of 10^{-2} to 1.4 dB at a BLER $\approx 10^{-5}$. The gap to the RCU bound is 0.5 dB at a BLER $\approx 10^{-7}$. In this case, omitting an interleaving stage yields a code performing better than the ensemble average. For the sake of completeness, we also provide the performance of a concatenation employing a randomly generated interleaver. The performance tightly approaches, in this case, the expected ensemble performance approximated by the TUB.

Figure 6.5 shows the performance by concatenating the $(1024, 693)$ product code of Figure 6.3 with an outer CRC code with generator polynomial $g(x) = x^{10} + x^9 + x^5 + x^4 + x + 1$, leading to a $(1024, 683)$ code.[5] The concatenated schemes needs a larger list than the one required by the product code alone to approach the ML lower bound, especially at high BLERs.

[4] BP decoding of the concatenated scheme operates on the Tanner graph modified by adding a check node representing the outer code constraints (as for the component codes, the outer code is decoded within the node by a MAP SISO decoder).

[5] The performance under BP is not provided. The reason is that the addition of the outer code check node in the product code Tanner graph resulted in large performance degradation due to the emergence of a number of small trapping sets for the BP decoder.

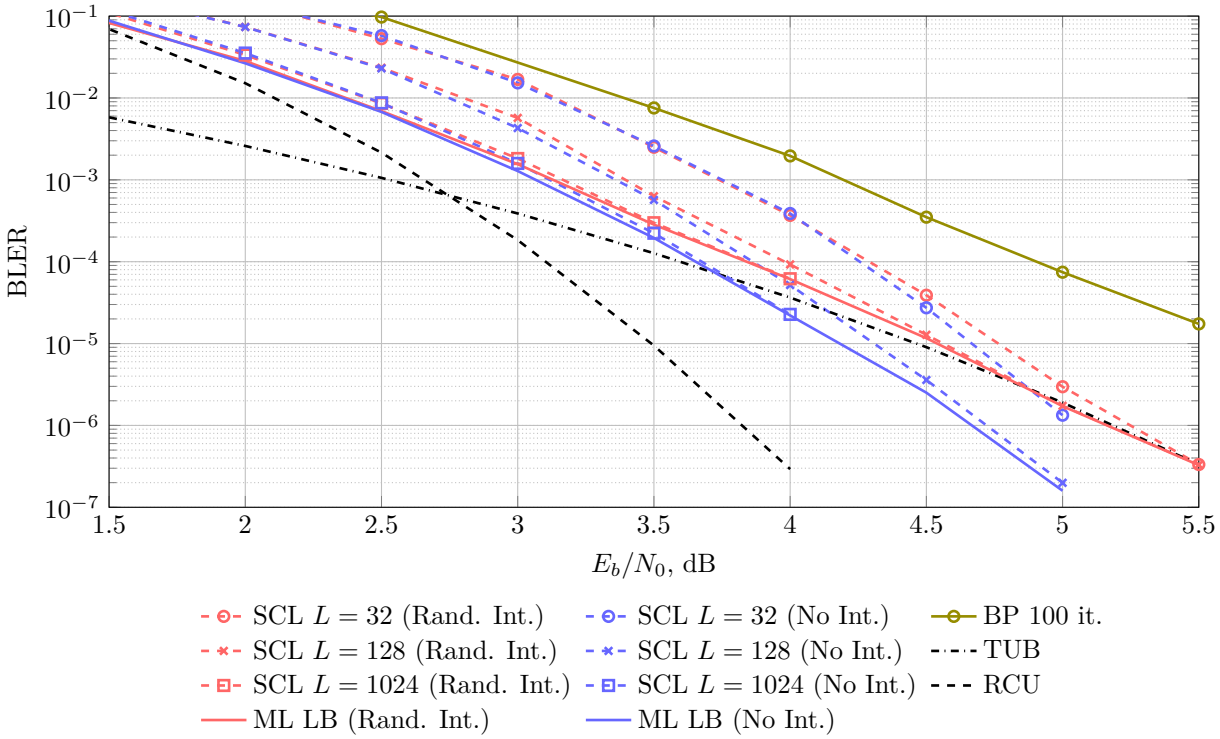

Figure 6.4.: BLER vs. SNR under SCL decoding with various list sizes for the $(128, 70)$ concatenated product codes compared with BP decoding.

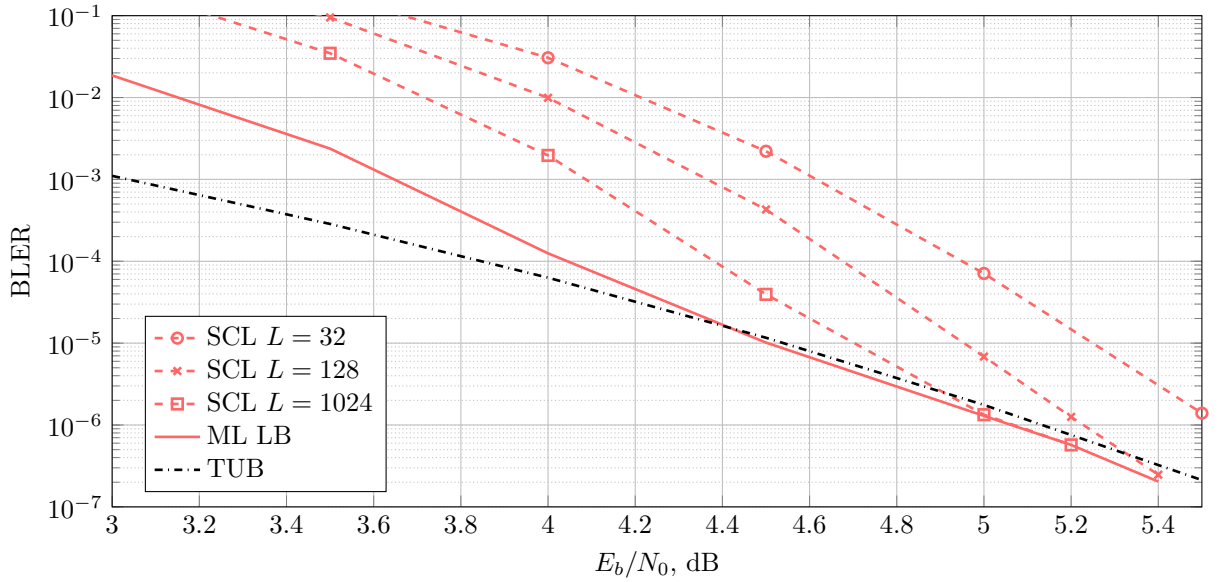

Figure 6.5.: BLER vs. SNR under SCL decoding with various list sizes for the $(1024, 683)$ concatenated product codes.

6.3. SPC Product Codes

The rate of product codes tends to be low if there are many component codes, see (3.13). SPC product codes provide the maximum possible overall rate. Moreover, SPC product codes perform close to the channel cutoff rate, as shown by Battail [157]. He argued that a better criterion for a code to perform well is having a distance spectrum close to that of random coding rather than a good minimum distance [158]. Later, it was shown that the distance spectrum of SPC product codes approaches that of a random code if each component code length tends to infinity [83, 159]. Since then, SPC product codes and their variants have been studied for BECs [160], BAWGNCs [150, 151, 161–163], and Rayleigh fading [161]. The main interest was on large blocklengths (with a few exceptions, e.g., [163] suggests MRB decoding of short SPC product codes). Recently, an instance of SPC product codes was shown to attain channel capacity with a vanishing bit error probability under MAP decoding relying on the code structure rather than the distance spectrum [164] although the query for capacity-achieving SPC product codes with vanishing BLEP remains open.

In the following, we bridge SPC product codes and multi-kernel polar codes so that the advanced tools of the latter can be used to obtain good performance using the former when short blocklengths considered. We start with a multi-kernel polar code representation of the SPC product codes similar to Section 6.2.1 for the RM product codes. Then the SC decoding equations are formalized for the multi-kernel polar representation of the code. SPC product codes are analyzed under SC decoding for transmission over the BEC and general BMSCs for asymptotic as well as finite-length cases. SCL decoding [165] is revisited by using the new equations and SCI decoding is introduced. The numerical results are provided together with a distance spectrum analysis.

In [83], a closed form expression is provided to compute the WE of 2-dimensional SPC product codes, relying on the MacWilliams identity for joint weight enumerators (JWEs) [56, Page 147] [166]. In [167], a closed form solution is extended to compute the IOWE of 2-dimensional SPC product codes by converting the dual code into a systematic form. This method does not seem applicable for higher-dimensional constructions as it is not trivial how to get to a systematic form of the dual code in such cases. We will present an alternative approach to that of [83] to compute the WE that avoids using JWEs. This approach is then extended to accommodate the IOWE of 2-dimensional product codes where one component code is an SPC code. The method is used to compute the IOWE of the exemplary short 3-dimensional SPC product code that can be seen as a 2-dimensional product code where one component code is an SPC code. By combining this result with

the uniform interleaver approach, the average IOWE of the concatenated code ensemble is computed, which is then used to compute tight bounds on the BLEPs [168, 169], e.g., via the Poltyrev tangential-sphere bound (TSB).

6.3.1. Multi-Kernel Polar Code Representation

Consider $(N_\ell \times N_\ell)$ kernels $\boldsymbol{K}_{N_\ell}$, $N_\ell \geq 2$, $\ell \in [m]$, of the form

$$\boldsymbol{K}_{N_\ell} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 1 & & & \\ \vdots & & \boldsymbol{I}_{N_\ell} & \\ 1 & & & \end{bmatrix} \tag{6.19}$$

with $K_\ell = N_\ell - 1$. Similar to (3.21), an $N^{[m]} \times N^{[m]}$ transform matrix $\boldsymbol{G}^{[m]}$ is obtained recursively as

$$\boldsymbol{G}^{[m]} = \left(\boldsymbol{I}_{N^{[m-1]}} \otimes \boldsymbol{K}_{N_m}\right) \boldsymbol{\Pi}_{N^{[m-1]},N_m} \left(\boldsymbol{I}_{N_m} \otimes \boldsymbol{G}^{[m-1]}\right) \tag{6.20}$$

where $\boldsymbol{G}^{[0]} \triangleq 1$. Recall the synthesized channels $W^i_{\boldsymbol{G}^{[m]}}$ defined in (3.27). The proof of the following lemma is given as Appendix 6.4.1.

Lemma 6.2. The multi-kernel construction (6.20) with a sequence of kernels of the form (6.19) polarizes. More formally, the fraction of channels with $I\left(W^i_{\boldsymbol{G}^{[m]}}\right) > 1 - \delta$ goes to $I(W)$ and the fraction of channels with $I\left(W^i_{\boldsymbol{G}^{[m]}}\right) < \delta$ to $1 - I(W)$ for any $\delta \in (0, 1)$ as $m \to \infty$ for any selection of $\{N_1, N_2, \dots\}$ where each N_i is finite, i.e., $N_i < \infty$.

Note that Lemma 6.2 shows that polarization occurs, but it is not sufficient to claim a capacity-achieving multi-kernel polar code. For a capacity-achieving construction, one may show that the rate of convergence is positive when the indices for frozen bits are chosen according to the rule imposed by [6, section I.C.4], i.e., showing that δ can be chosen as $\delta = 2^{-N^\beta}$ with $\beta > 0$. The exponent β can be computed via [49, Theorem 2] after fixing the relative frequencies of kernels in the construction. In the following, however, we provide a selection procedure for the frozen bit indices yielding an SPC product code, which does not take into account the quality of the synthesized channels. This hinders the possibility to achieve capacity for the SPC product codes under SC decoding.

Recall the multibase representation (2.3) of a decimal number i, denoted by

$$(i_1 i_2 \dots i_m)_{N_1 N_2 \dots N_m}. \tag{6.21}$$

The generator matrix $\mathbf{G}^{[m]}$ is obtained by removing the rows of $\boldsymbol{G}^{[m]}$ with indices in set $\mathcal{F}_{\mathrm{PC}} \subset [N]$ of frozen bits, which is chosen as

$$\mathcal{F}_{\mathrm{PC}} = [N] \setminus \{i+1 \in [N] : i_j \neq 0, \quad \forall j = 1, 2, \ldots, m\}\,. \tag{6.22}$$

Encoding can be done either by using (6.1) as $x_1^N = v_1^K \mathbf{G}^{[m]}$, or by using (6.20) as $x_1^N = u_1^N \boldsymbol{G}^{[m]}$ with $u_i = 0$ for all $i \in \mathcal{F}_{\mathrm{PC}}$ and the remaining positions are allocated for the information bits as for polar codes. In other words, (6.20) generalizes (3.21) to generate the mother code for multi-kernel polar codes generated by m kernels $\boldsymbol{K}_{N_\ell}$ in dimensions $\ell \in [m]$. Observe that (6.20) recovers (3.21) by setting $N_\ell = 2$ for all $\ell \in [m]$.

Example 6.3. Consider the (3×3) kernels

$$\boldsymbol{K}_{N_1} = \boldsymbol{K}_{N_2} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}. \tag{6.23}$$

We construct $\boldsymbol{G}^{[2]}$ by using (6.20):

$$\boldsymbol{G}^{[2]} = \begin{bmatrix} 1&0&0&0&0&0&0&0&0 \\ 1&0&0&1&0&0&0&0&0 \\ 1&0&0&0&0&0&1&0&0 \\ 1&1&0&0&0&0&0&0&0 \\ 1&1&0&1&1&0&0&0&0 \\ 1&1&0&0&0&0&1&1&0 \\ 1&0&1&0&0&0&0&0&0 \\ 1&0&1&1&0&1&0&0&0 \\ 1&0&1&0&0&0&1&0&1 \end{bmatrix}. \tag{6.24}$$

The generator matrix is

$$\mathbf{G}^{[2]} = \begin{bmatrix} 1&1&0&1&1&0&0&0&0 \\ 1&1&0&0&0&0&1&1&0 \\ 1&0&1&1&0&1&0&0&0 \\ 1&0&1&0&0&0&1&0&1 \end{bmatrix} \tag{6.25}$$

by removing the rows with indices given by (6.22), i.e., $\mathcal{A}_{\mathrm{PC}} = \{1, 2, 3, 4, 7\}$, as depicted in

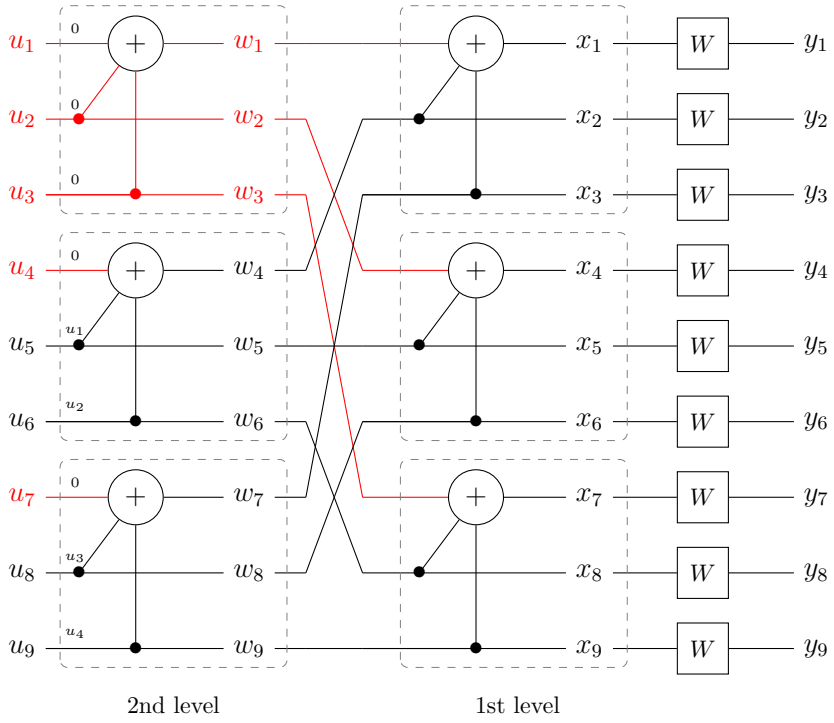

Figure 6.6.: How to choose the frozen bits to obtain a SPC product code, where red edges show those carrying frozen bit values and red variables are set to 0.

Figure 6.6. Equivalently, $\mathbf{G}^{[2]}$ can be formed by using (6.1) after removing the first rows of the kernels to get the generator matrices $\boldsymbol{G}_1$ and $\boldsymbol{G}_2$ defining SPC component codes, i.e.,

$$\boldsymbol{G}_1 = \boldsymbol{G}_2 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}. \tag{6.26}$$

6.3.2. SC Decoding

Consider transmission over a BMSC W using an m-dimensional (N, K) systematic SPC product code $\mathcal{C}$. Suppose we interpret the SPC product code by the multi-kernel polar code perspective discussed in Section 6.3 and depicted in Figure 6.6. SC decoding then follows the schedule of Section 3.4.4 for polar codes. Explicitly, decision $\hat{u}_1^N$ is made successively as

$$\hat{u}_i = \begin{cases} u_i & \text{if } \ i \in \mathcal{F}_{\mathrm{PC}} \\ f_i(y_1^N, \hat{u}_1^{i-1}) & \text{if } \ \text{otherwise} \end{cases} \tag{6.27}$$

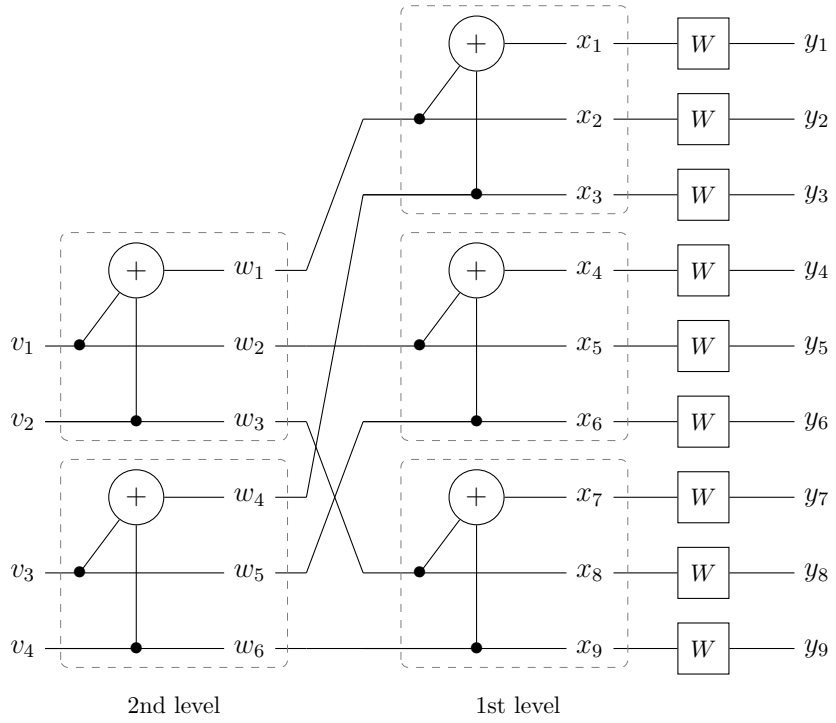

Figure 6.7.: The frozen bits and the corresponding edges are removed, providing a graphical representation of a 2-dimensional SPC product code as in Figure 6.1.

and

$$f_i(y_1^N, \hat{u}_1^{i-1}) \triangleq \begin{cases} 0 \text{ if } P^i_{\boldsymbol{G}^{[m]}}(0|y_1^N, \hat{u}_1^{i-1}) \geq P^i_{\boldsymbol{G}^{[m]}}(1|y_1^N, \hat{u}_1^{i-1}) \\ 1 \text{ otherwise.} \end{cases} \tag{6.28}$$

where the quantities $P^i_{\boldsymbol{G}^{[m]}}(u_i|y_1^N, \hat{u}_1^{i-1})$, $u_i \in \mathbb{F}_2$, $1 \leq i \leq N$ are computed as follows. For any $m \geq 0$, $N = N^{[m]}$, $1 \leq j \leq N^{[m-1]}$, $1 \leq k \leq N_m$, $t \triangleq N_m(j-1)$, the terms

$$P^{t+k}_{\boldsymbol{G}^{[m]}}\left(u_{t+k}|y_1^{N^{[m]}}, u_1^{t+k-1}\right) \tag{6.29}$$

are computed recursively as

$$\frac{\sum_{u_{t+k+1}^{t+N_m}} P^j_{\boldsymbol{G}^{[m-1]}}\left(\bigoplus_{\ell=t+1}^{t+N_m} u_\ell | y_1^{N^{[m-1]}}, \bigoplus_{\ell=1}^{N_m} u_{\ell,N_m}^t\right) \prod_{\ell=2}^{N_m} P^j_{\boldsymbol{G}^{[m-1]}}\left(u_{t+\ell}|y_{(\ell-1)N^{[m-1]}+1}^{\ell N^{[m-1]}}, u_{\ell,N_m}^t\right)}{\prod_{\ell=1}^{k-1} P^{t+\ell}_{\boldsymbol{G}^{[m]}}\left(u_{t+\ell}|y_{(\ell-1)N^{[m-1]}+1}^{\ell N^{[m-1]}}, u_1^{t+\ell-1}\right)} \tag{6.30}$$

where the denominator is set to 1 if $k = 1$. The notation u_{ℓ,N_m}^t denotes the subvector of u_1^t with indices in set $\{k \in [t] : \ell = k \mod N_m\}$ and the recursion (6.30) continues down

to length-1 computations as

$$P^1_{\boldsymbol{G}^{[0]}}(0|y_i) \triangleq \frac{W(y_i|0)}{\sum_x W(y_i|x)} \quad \text{and} \quad P^1_{\boldsymbol{G}^{[0]}}(1|y_i) \triangleq \frac{W(y_i|1)}{\sum_x W(y_i|x)}. \tag{6.31}$$

A block error event occurs if $\hat{u}_1^N \neq u_1^N$.

To gain insight on (6.30), consider the simple case of a length-5 kernel $\boldsymbol{K}_5$ with a single dimension, i.e., $m = 1$ and $\boldsymbol{G}^{[m]} = \boldsymbol{K}_5$. Suppose we are interested in $P^3_{\boldsymbol{G}^{[1]}}(u_3|y_1^5, u_1^2)$, for every $u_3 \in \{0, 1\}$, by assuming that the previous bits are given as $u_1^2 = (1, 0)$. Using (6.30), the computation is performed as

$$P^3_{\boldsymbol{G}^{[1]}}\left(u_3|y_1^5, u_1^2\right) = \frac{\sum_{u_4^5} P^1_{\boldsymbol{G}^{[0]}}\left(\bigoplus_{\ell=1}^5 u_\ell|y_1\right) \prod_{\ell=2}^5 P^1_{\boldsymbol{G}^{[0]}}(u_\ell|y_\ell)}{\prod_{\ell=1}^2 P^\ell_{\boldsymbol{G}^{[1]}}\left(u_\ell|y_1^5, u_1^{\ell-1}\right)} \tag{6.32}$$

$$= \frac{\sum_{u_4^5} P^1_{\boldsymbol{G}^{[0]}}\left(1 \oplus \bigoplus_{\ell=3}^5 u_\ell|y_1\right) P^1_{\boldsymbol{G}^{[0]}}(0|y_2) \prod_{\ell=3}^5 P^1_{\boldsymbol{G}^{[0]}}(u_\ell|y_\ell)}{P^1_{\boldsymbol{G}^{[1]}}(1|y_1^5)\, P^2_{\boldsymbol{G}^{[1]}}(0|y_1^5, 1)} \tag{6.33}$$

where (6.33) follows by inserting the values of bits u_1^2.

The block error event of the SC decoding is equal to that of the genie-aided SC decoding for the SPC product codes as stated in Lemma 3.4. A proof follows from Appendix 3.5.3 by suitably modifying the definition of the error events. The BLEP under SC decoding is thus bounded as

$$P_B^{(\mathrm{SC})} \leq \sum_{i \in \mathcal{A}_{\mathrm{PC}}} \Pr\left\{\mathcal{B}_i^{(\mathrm{GA})}\right\} \tag{6.34}$$

where the upper bound follows by applying the union bound as for (3.37).

6.3.3. Binary Erasure Channel

Over the BEC, ties are not broken towards any decision by revising (6.27) as (4.7). This means that we declare an error whenever an erasure is output for an information bit. In the following, we analyze the SC decoder of SPC product codes over the BEC. To this end, we derive general density evolution equations for the multi-kernel constructions (6.20) with kernels of the form (6.19) and analyze the performance when the frozen set is restricted to the choice $\mathcal{F}_{\mathrm{PC}}$. We do so to gain a deeper understanding on the behavior of the SC decoder when applied to the code construction under investigation.

We start by analyzing the behavior of the erasure probabilities of the synthesized channels when a kernel $\boldsymbol{K}_N$ is used for the BEC(ϵ). We denote by $\epsilon^i_{\boldsymbol{K}_N}$ the erasure probability for the i-th bit after SC decoding conditioned on the knowledge of the $i-1$ preceding bits,

$i = 1, \ldots, N$. We distinguish two cases. The first bit, i.e., u_1, is not erased only if there is no erasure at the output vector, i.e., we have

$$\epsilon^1_{\boldsymbol{K}_N} = 1 - (1-\epsilon)^N. \tag{6.35}$$

When the knowledge of the $i-1$ preceding bits is available, then decoding the i-th bit is successful either when $y_i \neq ?$ or there is no erasure in the subvector (y_1, y_{i+1}^N). Hence, the relationship between the input-output erasure probabilities, for $i \in \{2, \ldots, N\}$, is

$$\epsilon^i_{\boldsymbol{K}_N} = \epsilon \left(1 - (1-\epsilon)^{N-i+1}\right). \tag{6.36}$$

Based on the relation given in (6.36), we can derive the erasure probability $\epsilon^i_{\boldsymbol{G}^{[m]}}$ associated with bit u_i of the multi-kernel construction (6.20) under the genie-aided SC decoding by iterating (6.36). More precisely, for any $m \geq 0$, $N = N^{[m]}$, $1 \leq j \leq N^{[m-1]}$, $1 \leq k \leq N_m$, $t \triangleq N_m(j-1)$, we have the recursion in m as

$$\epsilon^{t+k}_{\boldsymbol{G}^{[m]}} = \begin{cases} \left(1 - \left(1 - \epsilon^{(j)}_{\boldsymbol{G}^{[m-1]}}\right)^{N_m}\right) & \text{if } k = 1 \\ \epsilon^{(j)}_{\boldsymbol{G}^{[m-1]}} \left(1 - \left(1 - \epsilon^{(j)}_{\boldsymbol{G}^{[m-1]}}\right)^{N_m - k + 1}\right) & \text{otherwise.} \end{cases} \tag{6.37}$$

Observe that the recursion can be implemented similar to the density evolution equations of polar codes with (3.53) and (3.54) as illustrated via Example 3.1. The only difference is that we use the m-digit multi-base representation (2.3) of $i-1$, say i_1^m, to compute $\epsilon^i_{\boldsymbol{G}^{[m]}}$, $i \in [N]$. Then we apply the operation (6.37) according to the values of i_ℓ in the digit reversed order, i.e., starting from i_m to i_1, where $k = i_\ell + 1$.

Example 6.4. Consider the size-9 transform in Figure 6.6. Suppose we want to compute $\epsilon^5_{\boldsymbol{G}^{[2]}}$ where the channel erasure probability is $\epsilon = 0.5$. Then $i_1^2 = (1,1)$ and we apply (6.37) with input erasure probability 0.5 by setting $k = 2$ two times recursively, which yields $\epsilon^5_{\boldsymbol{G}^{[2]}} \approx 0.228$.

Remark 6.3 (Ordering of kernels)**.** For a given sequence of kernels, the order in which they are used in the transform (6.20) affects the reliabilities of the synthetic channels [170, Section II.B] as implied by (6.37). Numerical evaluations suggest to use these in descending order according to the rate of polarization for the kernels, i.e., the kernel with the largest rate of polarization is used in the first level (see Figure 6.6). The rate of polarization for a given kernel is easily computed via [48, Theorem 11]. For kernels of the form (6.19),

the larger the size of the kernel, the lower the rate of polarization. Therefore, the largest kernel is used in the first level and the smallest is used in the last level (see Figure 6.6). The rate of polarization in an asymptotic setting for the resulting multi-kernel polar code is indifferent to this ordering.

Remark 6.4 (Ordering of component codes)**.** Given the blocklength and rate of an SPC product code, there is a unique sequence of component codes satisfying the parameters (3.12). For this sequence, when the blocklengths of component codes are different, an important question is what decoding order should be adopted. The natural approach is to start the decoding from the lowest rate SPC component code, i.e., to treat this code as the component code in the first level as in Figure 6.7, because a code with a lower rate has a higher error-correction capability. This ordering has been verified via numerical computation for an exemplary construction provided in Example 6.5, where a larger threshold is obtained if the decoding is performed in the reverse order of the component code rates. This observation is in line with Remark 6.3.

As stated, we are interested in the behaviour of SPC product codes under SC decoding. In particular, we proceed by bounding the performance of an (N, K) SPC product code $\mathcal{C}$ via (6.34). Hence, we restrict attention to computing the erasure probabilities corresponding to the set $\mathcal{A}_{\mathrm{PC}}$. This means that the density evolution equation corresponding to the case where $k = 1$ in (6.37) is never activated for the computation (see (6.22)). Since the RHS of (6.37) is monotonically increasing in the input erasure probability $\epsilon^{j}_{\boldsymbol{G}^{[m-1]}}$ and monotonically decreasing in $k \in \{2, \ldots, N_m\}$, the largest information bit erasure probability is equal to that of the first decoded information bit, i.e., we have

$$\epsilon_{\max} \triangleq \max_{i \in \mathcal{A}_{\mathrm{PC}}} \epsilon^{i}_{\boldsymbol{G}^{[m]}} = \epsilon^{\alpha}_{\boldsymbol{G}^{[m]}} \tag{6.38}$$

where $\alpha \triangleq \min \mathcal{A}_{\mathrm{PC}}$. By rewriting (6.34) in terms of $\epsilon^{i}_{\boldsymbol{G}^{[m]}}$, we obtain

$$P_B^{(\mathrm{SC})} \leq \sum_{i \in \mathcal{A}_{\mathrm{PC}}} \epsilon^{i}_{\boldsymbol{G}^{[m]}}. \tag{6.39}$$

A loose upper bound follows by tracking only the largest erasure probability for $i = \alpha$:

$$P_B^{(\mathrm{SC})} \leq K \epsilon_{\max}. \tag{6.40}$$

Figure 6.8 illustrates the simulation results for the 3-dimensional $(125, 64)$ SPC product code, obtained by iterating $(5, 4)$ SPC codes, over the BEC. The results are provided in

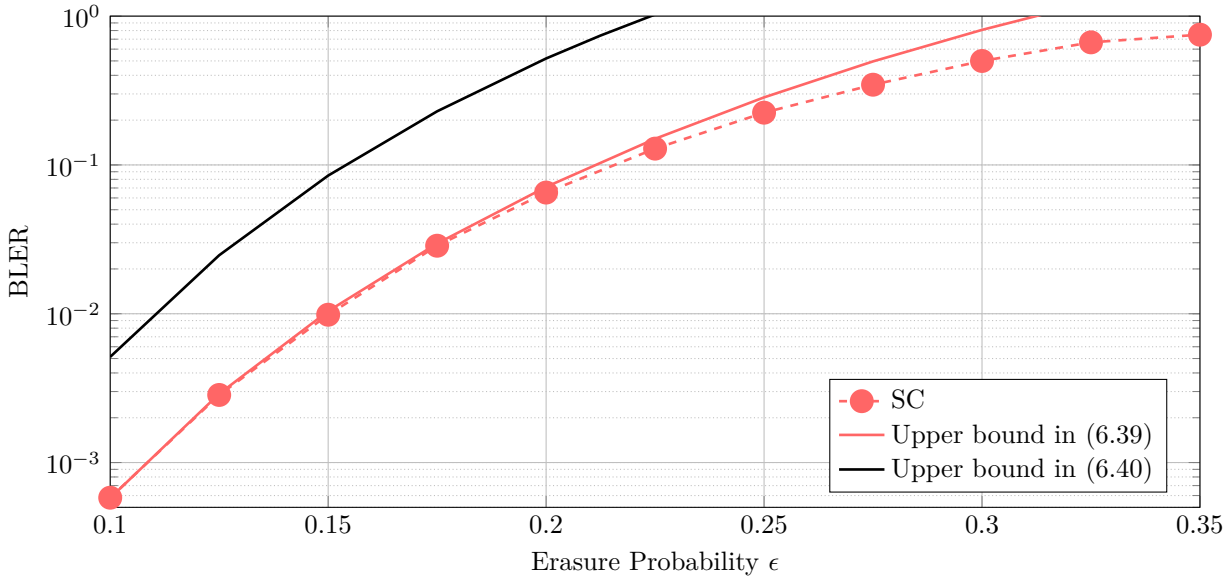

Figure 6.8.: BLER vs. ϵ over the BEC for the $(125, 64)$ product code under SC decoding.

terms of BLER vs. channel erasure probability ϵ. The upper bounds on SC decoding, computed via the RHS of (6.39) and (6.40), are also provided. The former turns out to be tight while the latter is loose as expected.

Asymptotic Performance Analysis

We now consider the asymptotic performance of SPC product codes. We analyze the BLEP of a *product code sequence* defined by an ordered sequence of component code sets

$$\mathscr{C}^{[m]} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_m\} \tag{6.41}$$

where we constrain $\left|\mathscr{C}^{[m]}\right| = m$, i.e., the number of component codes for the m-th product code in the sequence is set to m, and the component code rates satisfy $R_i \leq R_j$ for $i < j$. Let $\mathcal{C}^{[m]}$ be the m-th product code in the sequence corresponding to the set $\mathscr{C}^{[m]}$. We study the behavior of the BLEP as the dimension m tends to infinity when the SC decoding starts from $\mathcal{C}_1$ up to $\mathcal{C}_m$. We remark that, as m changes, the component codes used to construct the product code are allowed to change, i.e., the sequence of product codes is defined by the set of component codes employed for each value of m. Observe that the rate of $\mathcal{C}^{[m]}$ may vanish as m grows large if the choice of the component codes forming the sets $\mathscr{C}^{[m]}$ is not performed carefully.

We proceed by analyzing the limiting behavior in terms of *block erasure thresholds* for

different product code sequences with positive rates under SC decoding. Recall that we consider m-dimensional systematic SPC product code constructions where the frozen set $\mathcal{F}_{\text{PC}}$ is given by (6.22) after specifying the kernels $\boldsymbol{K}_{N_1}, \ldots, \boldsymbol{K}_{N_m}$ of the form (6.19).

Definition 6.2. The SC decoding block erasure threshold of an SPC product code sequence defined by the component code sets $\mathscr{C}^{[1]}, \ldots, \mathscr{C}^{[m]}$ is the largest channel erasure probability $\epsilon^\star$ for which the BLEP $P_B^{(\text{SC})}\left(\mathcal{C}^{[m]}\right)$ converges to 0 asymptotically in m if the limit exists, i.e.,

$$\epsilon^\star = \sup_{\epsilon \in [0,1)} \{\epsilon : \lim_{m \to \infty} P_B^{(\text{SC})}\left(\mathcal{C}^{[m]}\right) = 0\}. \tag{6.42}$$

As it is not possible to evaluate $P_B^{(\text{SC})}\left(\mathcal{C}^{[m]}\right)$ exactly, we rely on the upper bound (6.40) to obtain a lower bound $\epsilon_{\text{LB}}^\star$ on the block erasure threshold in the form

$$\epsilon_{\text{LB}}^\star = \sup_{\epsilon \in [0,1)} \{\epsilon : \lim_{m \to \infty} K\left(\mathcal{C}^{[m]}\right) \epsilon_{\max}\left(\mathcal{C}^{[m]}\right) = 0\} \tag{6.43}$$

where $K\left(\mathcal{C}^{[m]}\right)$ and $\epsilon_{\max}\left(\mathcal{C}^{[m]}\right)$ are the dimension of code $\mathcal{C}^{[m]}$ and the maximum information bit erasure probability after SC decoding, respectively, which depend on the sequence of product codes. We next provide two examples of product code sequences whose rates converge to a positive value. The first sequence exhibits a positive block erasure threshold (lower bound), which is however arguably far from the Shannon limit. We then analyze a product code sequence that achieves the BEC capacity under bit-wise MAP decoding [164, 171].

Example 6.5 (Euler's infinite-product representation of the sine function as an SPC product code)**.** Consider an SPC product code sequence with an $(a^2\ell^2, a^2\ell^2 - 1)$ SPC component code at the ℓ-th dimension, yielding $R_\ell = (1 - (a\ell)^{-2})$, with $\ell = 1, \ldots, m$. The asymptotic rate is computed via Euler's infinite-product representation of the sine function:

$$\sin\left(\frac{\pi}{a}\right) = \frac{\pi}{a} \prod_{\ell=1}^{\infty} \left(1 - \frac{1}{a^2\ell^2}\right) \tag{6.44}$$

yielding an asymptotic rate $R = \frac{a}{\pi} \sin\left(\frac{\pi}{a}\right)$. Different product code sequences can be obtained for various choices of the parameter a.

The lower bounds on the block erasure thresholds are provided in Table 6.1 for several values of a. The second column in Table 6.1 provides the asymptotic rate of the SPC product code sequence defined by the parameter a (whose squared value is reported in the first column). The third column reports the lower bound on the block erasure threshold. The fourth column gives the Shannon limit for the given asymptotic rate, while the last

Table 6.1.: Lower bounds on the block erasure thresholds for some SPC product code sequences based on Euler's infinite-product representation of the sine function

a^2	R	$\epsilon^\star_{\mathrm{LB}}$	Limit, $\epsilon = 1 - R$	$\epsilon^\star_{\mathrm{LB}}/\epsilon$
2	0.3582	0.3308	0.6418	0.5154
4	0.6366	0.1440	0.3634	0.3963
8	0.8067	0.0681	0.1933	0.3523
16	0.9003	0.0332	0.0997	0.3331
32	0.9494	0.0164	0.0506	0.3241
64	0.9745	0.0081	0.0255	0.3176

column shows the fraction of the Shannon limit achieved by each construction. The thresholds achieved by the different product code sequences lie relatively far from the Shannon limit. In relative terms, the lowest-rate construction (obtained for $a^2 = 2$) achieves the largest fraction (above 1/2) of the limit, while the efficiency of the sequences decreases as the rate grows.

Example 6.6 (Product of $(m, m-1)$ SPC product codes in m dimensions)**.** Consider the product code obtained by iterating $(m, m-1)$ SPC codes in m dimensions, i.e., the resulting code is an $(m^m, (m-1)^m, 2^m)$ code. The rate of the m-th product code in the sequence is

$$R\left(\mathcal{C}^{[m]}\right) = \left(1 - \frac{1}{m}\right)^m \tag{6.45}$$

which converges to e^{-1} for $m \to \infty$. As presented in [164], this product code sequence is capacity-achieving over the BEC under bit-wise MAP decoding. This observation follows from results derived in [171]. Unfortunately, the block-wise erasure threshold under SC decoding turns out to be zero. This negative result is provided by the following theorem, whose proof is given in Appendix 6.4.2.

Theorem 6.3. Under SC decoding, the block erasure threshold of the product code sequence defined by the component code sets $\mathscr{C}^{[m]} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_m\}$, where $\mathcal{C}_i$, $i = 1, \ldots, m$, are $(m, m-1)$ SPC codes, is zero.

6.3.4. Binary Memoryless Symmetric Channels

In the following, the BLEP of SPC product codes is upper bounded using density evolution when the transmission takes place over a BMSC. Due to the channel symmetry and the linearity of the codes, we assume that the all-zeroes codeword is transmitted. Recall the

definition (3.55) of the LLR $L^i_{\boldsymbol{G}^{[m]}}(y_1^N)$ for u_i where all the previous bit-values are provided as zeros to the decoder. Similarly, recall that $l^i_{\boldsymbol{G}^{[m]}}$ is the PDF of the RV $L^i_{\boldsymbol{G}^{[m]}}(Y_1^N)$. Extending the equations (6.36) and (6.37) to general BMSCs, the densities, for any $m \geq 0$, $N = N^{[m]}$, $1 \leq j \leq N^{[m-1]}$, $1 \leq k \leq N_m$, $t \triangleq N_m(j-1)$, can be computed recursively as

$$l^{t+k}_{\boldsymbol{G}^{[m]}} = \begin{cases} (l^{(j)}_{\boldsymbol{G}^{[m-1]}})^{\circledast N_m} & \text{if } k = 1 \\ l^{(j)}_{\boldsymbol{G}^{[m-1]}} \circledast \left(l^{(j)}_{\boldsymbol{G}^{[m-1]}}\right)^{\circledast N_m - k + 1} & \text{otherwise} \end{cases} \tag{6.46}$$

The RHS of (6.34) can be computed as

$$\sum_{i \in \mathcal{A}_{\mathrm{PC}}} \lim_{z \to 0} \left(\int_{-\infty}^{-z} l^i_{\boldsymbol{G}^{[m]}}(x)dx + \frac{1}{2} \int_{-z}^{+z} l^{(i)}_{\boldsymbol{G}^{[m]}}(x)dx \right). \tag{6.47}$$

The computation of (6.46) and (6.47) can be carried out, for instance, via quantized density evolution [104] as discussed in Section 3.4.7, yielding an accurate estimate of the RHS of (6.34).

In Figure 6.9, we provide simulation results for the $(125, 64)$ SPC product code over the BAWGNC. The SC decoding performance is compared to the performance under BP decoding. BP decoding with a maximum number of iterations set to 100 outperforms the SC decoding significantly, which motivates us to introduce SCL decoding in the next section. The upper bound on the SC decoding, computed via (6.34) and (6.47), is also provided and appears to be tight.

6.3.5. SCI and SCL Decoding

While the asymptotic analysis provided in Section 6.3.3 provides insights on the SPC product codes with many component codes, we are ultimately interested in the performance of product codes in the practical setting where the number of component codes is small and the blocklength is moderate (or small). Like polar codes, SC decoding of SPC product codes performs poorly in this regime, e.g., see Figure 6.9. Hence, following [11], we investigate the BLEP of SPC product codes under SCL decoding.

The SCL decoding of SPC product code works the same as for polar codes described in Section 3.4.5 with the frozen set $\mathcal{F}$, where the myopic probabilities $p^i_{\boldsymbol{G}^{[m]}}(y_1^N, \tilde{u}_1^i)$ are computed using (3.49) with recursive functions (6.30) instead of (3.45) and (3.46) in the case of polar codes. Similarly, SCI decoding of SPC product codes can be described as in Section 4.3. In addition, the analysis of Chapter 4 for the SCI decoding of polar codes naturally extends to the SCI decoding of SPC product codes.

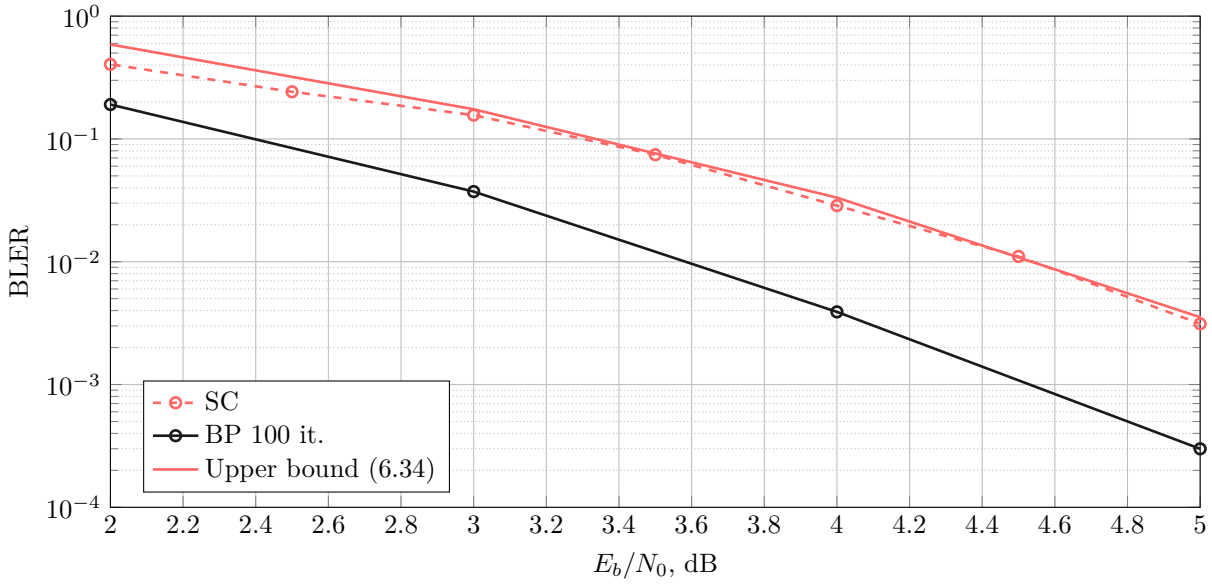

Figure 6.9.: BLER vs. SNR over the BAWGNC for a $(125, 64)$ product code under SC decoding, compared to a BP decoding with 100 iterations.

Following Remark 6.1, we first study the performance of SPC product codes under ML decoding by developing a weight enumerator analysis. Similarly, inspired by the concatenated RM product code construction of Section 6.2.2, we also study the performance of a concatenation of a high-rate outer code with an inner SPC product code under ML decoding. For both cases, the analysis is complemented by the SCL decoding simulations. We will see that the ML decoding performance is attainable for some short product codes by SCL decoding with small list sizes, e.g., $L \leq 8$, while larger list sizes, e.g., $128 \leq L \leq 1024$, are required when SPC product codes are concatenated with an outer code.

Finite-length Performance Analysis via Weight Distribution

Computing the weight enumerator of SPC product codes for small constructions is feasible using the method presented in [83, Appendix A]. We first provide an alternative derivation to the WEF of a 2-dimensional product code $\mathcal{C}$ with systematic arbitrary binary linear component code $\mathcal{C}_1$ and systematic $(\nu, \nu - 1)$ SPC code $\mathcal{S}_\nu$ as the second component code. The proof does not require an explicit use of MacWilliams identities, and is presented in Appendix 6.4.3.

Theorem 6.4. Let $\mathcal{C}_1$ and $\mathcal{S}_\nu$ be an arbitrary (N_1, K_1) systematic code with a generator matrix $\boldsymbol{G}$ and a length-ν systematic SPC code, respectively. Then the WEF of the product

code $\mathcal{C}$ with component codes $\mathcal{C}_1$ and $\mathcal{S}_\nu$ is

$$A_{\mathcal{C}}(z) = 2^{-K_1} \sum_{\boldsymbol{v} \in \{0,1\}^{K_1}} \left(\sum_{\boldsymbol{u} \in \{0,1\}^{K_1}} (-1)^{\boldsymbol{u} \cdot \boldsymbol{v}^{\mathrm{T}}} z^{w_{\mathrm{H}}(\boldsymbol{u}\boldsymbol{G})} \right)^{\nu}. \tag{6.48}$$

Thanks to Theorem 6.4, one can compute the WEF of short and moderate-length SPC product codes, by simply choosing one component code to be a SPC product code $\mathcal{C}_1$ and the other one to be a SPC code $\mathcal{S}_\nu$. Given the weight enumerator of a product code, upper bounds on the ML decoding error probability can be obtained. For example, a tight bound on the BLEP over the BAWGNC is provided by Poltyrev's TSB [169]. Another example of a tight bound on the BLEP of a code based on its weight enumerator is Di's union bound over the BEC [168, Lemma B.2]. We refer the interested reader to [58] for an extensive survey on performance bounds under ML decoding.

We next provide simulation results for the 3-dimensional $(125, 64)$ SPC product code under ML decoding implemented over the BEC via SCI decoding with $I = \infty$. The results are shown in Figure 6.10. As a reference, simulation results for a $(125, 64)$ punctured polar code [172] are also provided, where the selection of frozen bits for the polar code design follows the guidelines of the 5G standard (without outer CRC code) [114]. The polar code slightly outperforms the considered SPC product code. Note that Di's union bound tightly approaches the performance of the SPC product code.

The same $(125, 64)$ SPC product code is simulated over the BAWGNC under SCL decoding with various list sizes. The results are given in Figure 6.11. The TSB is computed using the weight enumerator analysis. Remarkably, SCL decoding with $L = 4$ suffices to operate close to the TSB and to outperform BP decoding. With $L = 8$, the SCL decoder approaches the ML lower bound, which is not the case for BP decoding. The RCU and the MC bounds are here plotted as references. The gap to the RCU bound reaches to 1.7 dB at BLER 10^{-3}.

The performance of SPC product codes is compared to that of the $(125, 64)$ polar code in Figure 6.12. For $L = 4$, the performance of the polar code matches its ML lower bound and outperforms the SPC product code by approximately 0.3 dB at BLER 10^{-3}. The gap between their ML performance is about 0.25 dB. Note that the polar code requires a smaller list size to approach its ML performance.

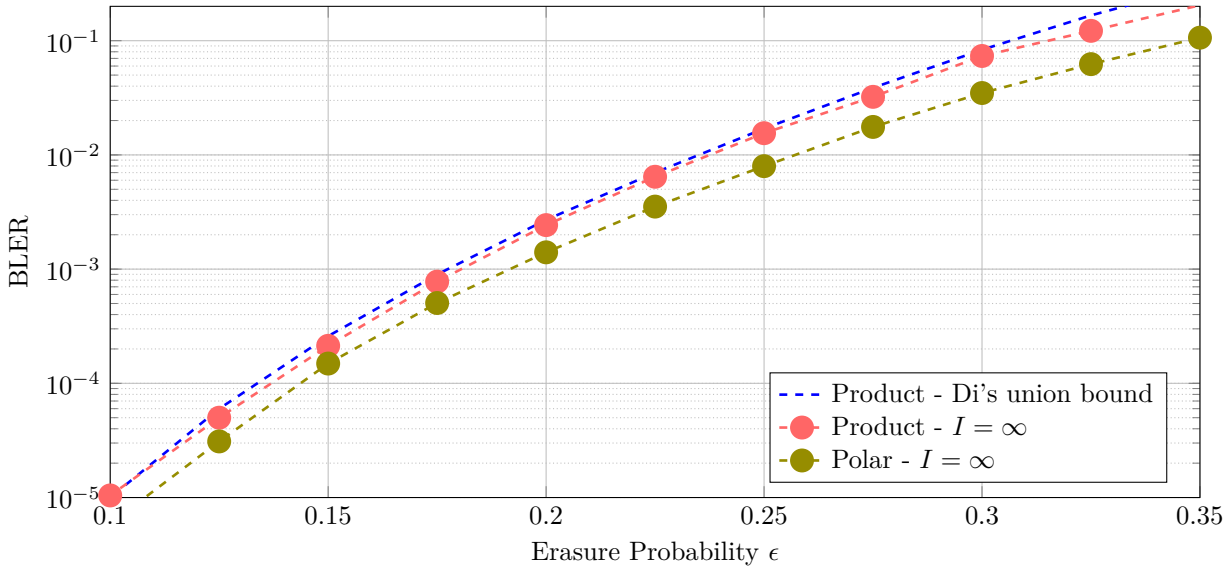

Figure 6.10.: BLER vs. ϵ over the BEC for the $(125, 64)$ product code under ML decoding, implemented via SCI decoding with $I = \infty$, compared to a $(125, 64)$ punctured polar code.

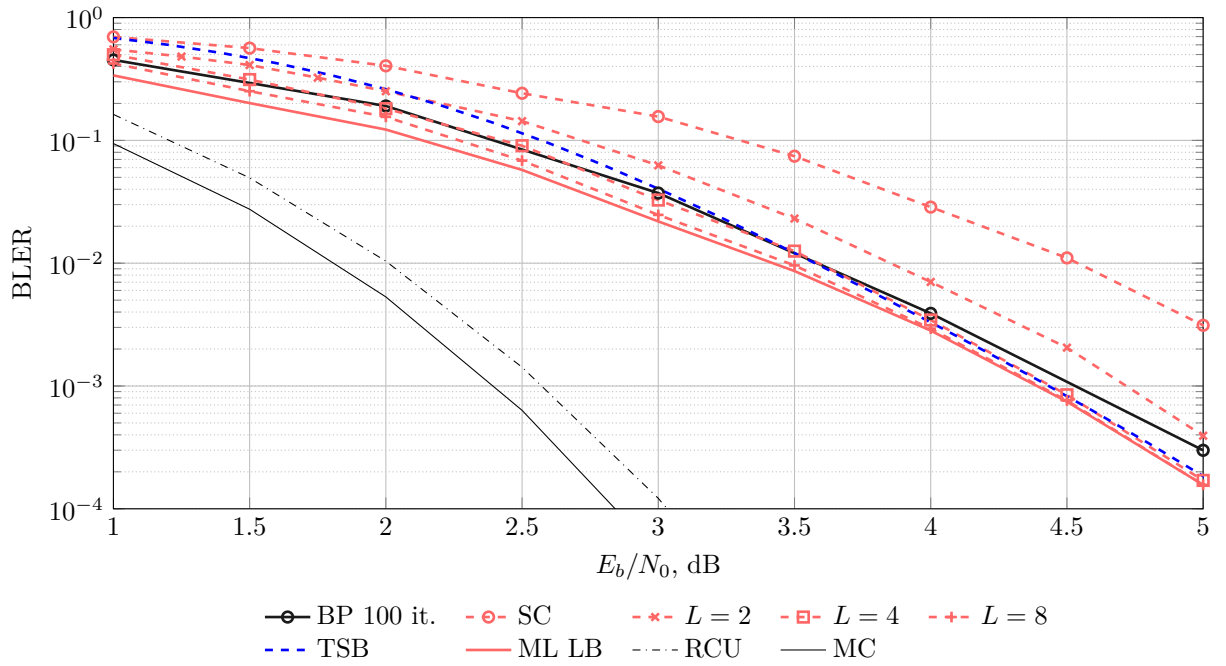

Figure 6.11.: BLER vs. SNR under SCL decoding for the $(125, 64)$ product code with various list sizes, compared to a BP decoding with 100 iterations.

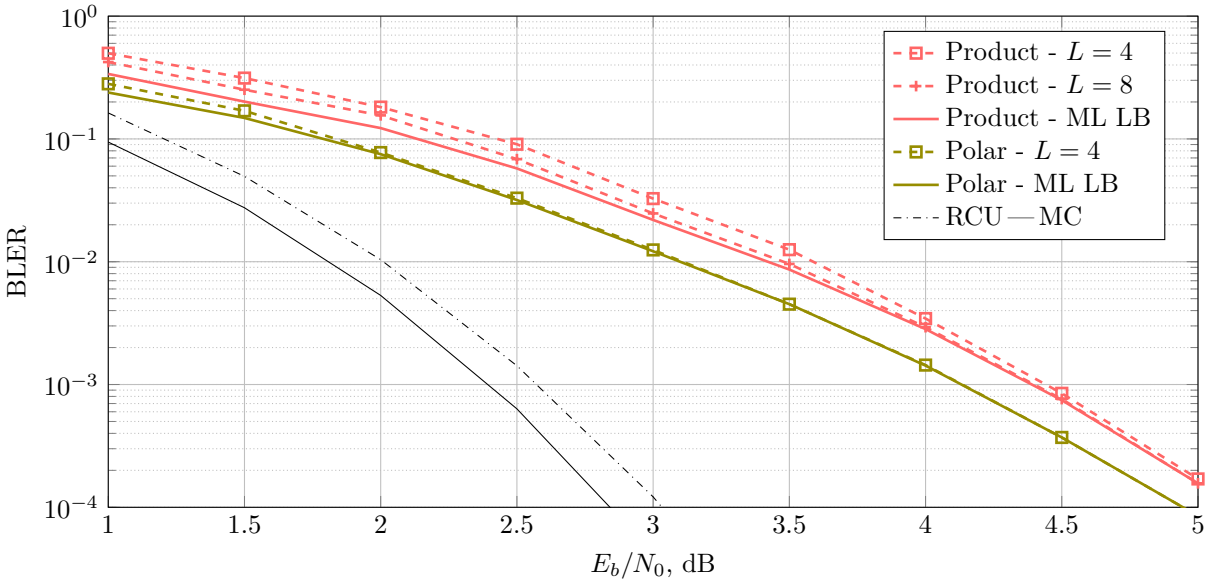

Figure 6.12.: BLER vs. SNR over the BAWGNC for the $(125, 64)$ product code under SCL decoding with various list sizes, compared to a $(125, 64)$ punctured polar code.

Finite-length Performance Analysis via Average Weight Distribution of Concatenated Ensembles

Motivated by the results of Section 6.2.2, we now concatenate SPC product codes with a high-rate outer code to improve the distance profile. To analyze the ML decoding performance of such a concatenation, we first derive the weight enumerator of product codes concatenated with an outer code. The weight enumerators are then used to derive the TSB which helps to choose an outer code based on an ensemble analysis.

Computing (6.17) requires knowing the input-output weight enumerator of the inner SPC product code. Therefore, we extend the result of Theorem 6.4 to derive the IOWEF in Appendix 6.4.4.

Theorem 6.5. Let $\mathcal{C}_1$ and $\mathcal{S}_\nu$ be a (N_1, K_1) systematic code with generator matrix $\boldsymbol{G}$ and a systematic SPC code, respectively. Then the IOWEF of the product code $\mathcal{C}$ with component codes $\mathcal{C}_1$ and $\mathcal{S}_\nu$ is

$$A_{\mathcal{C}}^{\mathrm{IO}}(x,z) = 2^{-K_1} \sum_{\boldsymbol{v}\in\{0,1\}^{k_1}} \left(\sum_{\boldsymbol{u}\in\{0,1\}^{k_1}} (-1)^{\boldsymbol{u}\cdot\boldsymbol{v}^{\mathrm{T}}} x^{w_{\mathrm{H}}(\boldsymbol{u})} z^{w_{\mathrm{H}}(\boldsymbol{u}\boldsymbol{G})} \right)^{\nu-1} \left(\sum_{\boldsymbol{u}\in\{0,1\}^{k_1}} (-1)^{\boldsymbol{u}\cdot\boldsymbol{v}^{\mathrm{T}}} z^{w_{\mathrm{H}}(\boldsymbol{u}\boldsymbol{G})} \right). \tag{6.49}$$

Similar to the WEF, one can compute the IOWEF of short and moderate-length SPC

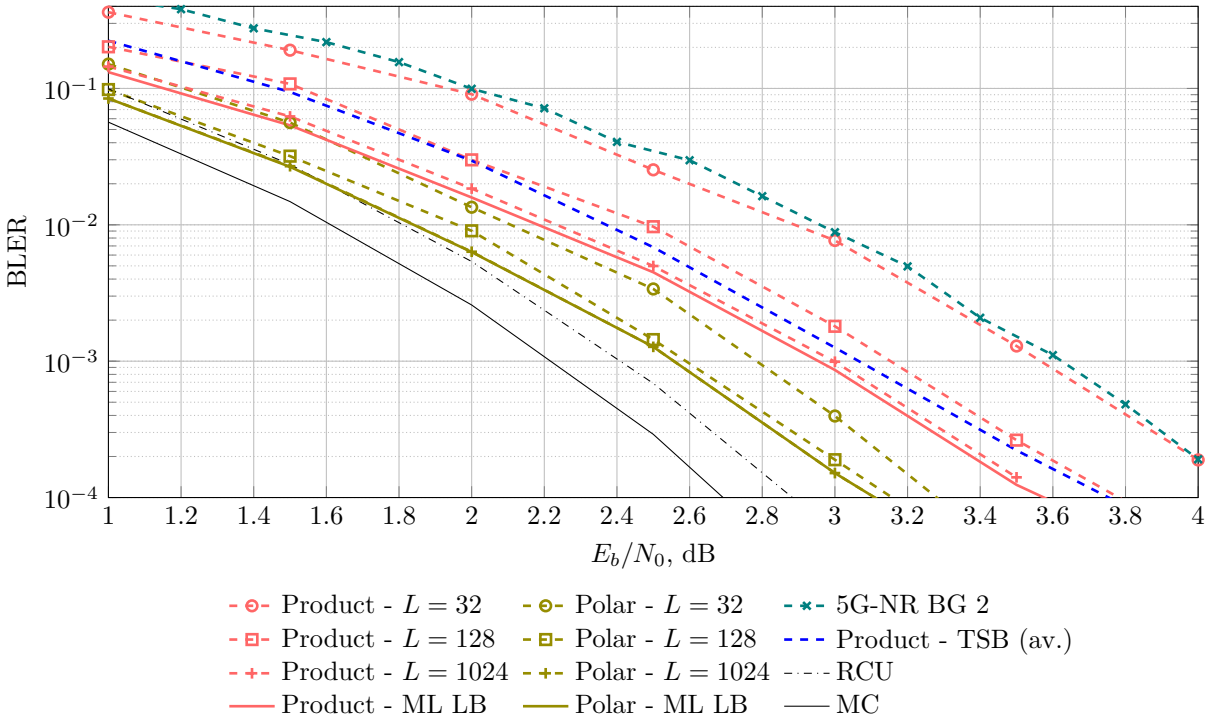

Figure 6.13.: BLER vs. SNR over the BAWGNC for a $(125, 64)$ product code concatenated with a $(64, 56)$ CRC code under SCL decoding with various list sizes. The performance is compared to the one of a $(125, 56)$ CRC-concatenated polar code with various list sizes, where the generator polynomial of the outer code is optimized for SCL decoding [18], and the one of the $(128, 64)$ 5G-NR LDPC code (base graph 2, see [36]) under BP decoding where the maximum number of iterations is set to 100.

product codes, by choosing one component code to be a SPC product code and the other one to be a SPC code. Given the average weight enumerator of a concatenated ensemble, upper bounds on the ML decoding error probability can be obtained as in Section 6.3.5.

Figure 6.13 shows the performance of concatenating the $(125, 64)$ product code with a 8-bit outer CRC code with generator polynomial $g(x) = x^8 + x^6 + x^5 + x^4 + x^2 + x + 1$, where the interleaver between the codes is the trivial one defined by an identity matrix. This concatenation leads to a $(125, 56)$ code. Since the code distance properties are improved (observed directly via the TSB on the average performance of the code ensemble), the performance improvement under ML decoding is expected to be significant. The CRC polynomial is selected to provide the best TSB, obtained using the average weight enumerator of a concatenated ensemble with a uniform interleaver. The gain achieved by SCL decoding is remarkably large, operating below the TSB. At a BLER of 10^{-2}, SCL decod-

ing of the concatenated code achieves gains up to 1.25 dB over the original product code, reaching up to 1.5 dB at a BLER $\approx 10^{-4}$. The gap to the RCU bound is approximately 0.7 dB at a BLER of 10^{-4} and less for higher BLERs, providing a competitive performance for similar parameters [36]. For example, the performance of a $(128, 64)$ 5G-NR LDPC code (base graph 2, see [36]) is reported. The code is decoded with BP by setting the maximum number of iterations to 100. The concatenation of the outer CRC code with the inner SPC product code yields a remarkable gain of ≈ 0.6 dB with respect to the 5G-NR LDPC code (it shall be noted, however, that the concatenated SPC code possesses a slightly lower code rate). It is not always possible to attain a performance close to the ML performance of concatenated codes using BP decoding (see, e.g., Section 6.2.2 or [173]). In that sense, SCL decoding provides a low-complexity solution to approach the ML performance of the concatenated SPC product code scheme. As another reference, a (125, 56) CRC-concatenated polar code is constructed by using the $(125, 64)$ polar code with an 8-bit outer CRC code optimized using the guidelines of [18], which takes into account the exact code concatenation under SCL decoding rather than an ensemble performance. The generator polynomial of the CRC code is $g(x) = x^8 + x^7 + x^6 + x^5 + 1$. The gap between the ML performance of two codes is less than 0.5 dB in the considered regime. A careful optimization of the interleaver of the CRC-concatenated SPC product code might provide further gains as for polar codes [116], but this is not in the scope of this thesis.

6.4. Appendices

We first provide a proposition, which will be used to prove Lemma 6.2.

Proposition 1. Let $H_2 : [0, 1/2] \to [0, 1]$ be the binary entropy function and $a * b \triangleq a(1-b) + (1-a)b$. Then, for all $a, b \in [0, 1/2]$, it satisfies

$$(1 - H_2(a))\, H_2(b) \geq H_2(a * b) - H_2(a) \geq (1 - H_2(a))\, H_2^2(b) \tag{6.50}$$

with equality at the left-hand side (LHS) of (6.50) if and only if $a \in \{0, 1/2\}$ or $b \in \{0, 1/2\}$ and with equality at the RHS if and only if $a = 1/2$ or $b \in \{0, 1/2\}$.

Proof. We define a function $f_b : [0, 1] \to [0, 1]$ parameterized by $b \in [0, 1/2]$ as

$$f_b(x) = H_2\left(H_2^{-1}(x) * b\right) \tag{6.51}$$

which is convex in x for all b due to Mrs. Gerber's Lemma [133]. Then, for a Bernoulli

RV X satisfying $a = H_2^{-1}(P(X=1))$, we write

$$H_2(a*b) = f_b(\mathbb{E}[X]) \tag{6.52}$$
$$\leq \mathbb{E}[f_b(X)] \tag{6.53}$$
$$= (1 - H_2(a))H_2(b) + H_2(a) \tag{6.54}$$

where (6.52) follows from (6.51) and $\mathbb{E}[X] = H_2(a)$ and (6.53) from the Jensen's inequality with equality if and only if X is deterministic, i.e., $a \in \{0, 1/2\}$, or f_b is linear, i.e., $b \in \{0, 1/2\}$. This concludes the proof for the LHS of (6.50).

To find a lower bound, we observe that the Taylor expansion around $a = 1/2$ gives

$$H_2(a) = \sum_{i=1}^{\infty} c_i \left[1 - (1-2a)^{2i}\right] \tag{6.55}$$

where $c_i = (2i(2i-1)\ln 2)^{-1}$. Since $1 - 2(a*b) = (1-2a)(1-2b)$, (6.55) yields

$$H_2(a*b) = \sum_{i=1}^{\infty} c_i \left[1 - (1-2a)^{2i}(1-2b)^{2i}\right]. \tag{6.56}$$

Then, we write

$$H_2(a*b) - H_2(a) = \sum_{i=1}^{\infty} c_i (1-2a)^{2i} \left[1 - (1-2b)^{2i}\right] \tag{6.57}$$
$$\geq \left[1 - (1-2b)^2\right] \sum_{i=1}^{\infty} c_i (1-2a)^{2i} \tag{6.58}$$
$$= 4b(1-b)(1 - H_2(a)) \tag{6.59}$$
$$\geq (1 - H_2(a))\, H_2^2(b) \tag{6.60}$$

where (6.58) follows from $(1-2b)^2 \geq (1-2b)^{2i}$ for $i > 1$ with equality if and only if $b \in \{0, 1/2\}$ and (6.59) from combining (6.55) to the convergent series $\sum_{i=1}^{\infty} c_i = 1$. Finally, (6.60) is due to the inequality $H_2(b) \leq 2\sqrt{b(1-b)}$ with equality if and only if $b \in \{0, 1/2\}$ [6, Prop. 11]. Both (6.58) and (6.60) become equality if and only if $a = 1/2$ or $b \in \{0, 1/2\}$, which concludes the proof. ■

Note finally that the proposition still holds for all $a, b \in [0,1]$ when the preimage of the binary entropy function is extended to $[0,1]$ since $H_2(a) = H_2(1-a)$ and $a * b = 1 - ((1-a) * b)$. Now, we are ready to prove Lemma 6.2.

6.4.1. Proof of Lemma 6.2

Let $U_1^{N_\ell}$ be a-priori uniform on $\mathcal{X}^{N_\ell}$ and let $X_1^{N_\ell}$ as $X_1^{N_\ell} = U_1^{N_\ell} \boldsymbol{K}_{N_\ell}$. Due to [49, Theorem 1], it is sufficient to show that there exists $\alpha, \beta > 0$ for all $i \in [N_\ell]$ such that

$$\left| I\left(W^i_{\boldsymbol{K}_{N_\ell}}\right) - I\left(W\right)\right| \geq I\left(W\right)^\alpha \left(1 - I\left(W\right)\right)^\beta \tag{6.61}$$

which can be translated to

$$\left| H\left(U_i \middle| Y_1^{N_\ell}, U_1^{i-1}\right) - H(W)\right| \geq \left(1 - H(W)\right)^\alpha H^\beta(W). \tag{6.62}$$

Note that $H\left(U_i \middle| Y_1^{N_\ell}, U_1^{i-1}\right)$ is decreasing in i for $i \geq 2$ due to the kernel structure. This means we have, for $i \in \{2, \ldots, N_\ell\}$,

$$H\left(U_i \middle| Y_1^{N_\ell}, U_1^{i-1}\right) \leq H\left(U_2 \middle| Y_1^{N_\ell}, U_1\right). \tag{6.63}$$

We no focus on the RHS of (6.63). We have $U_1 = X_1 \oplus \ldots \oplus X_{N_\ell}$ and $U_2 = X_2$ and setting $S = U_1 \oplus X_2 = X_1 \oplus X_3 \ldots \oplus X_{N_\ell}$, we write the RHS of (6.63) as

$$\begin{aligned} H\left(X_2 \middle| Y_1^{N_\ell}, S \oplus X_2\right) &= H(X_2|Y_1^{N_\ell}) + H(S \oplus X_2|Y_1^{N_\ell}, X_2) - H(S \oplus X_2|Y_1^{N_\ell}) && (6.64)\\ &= H(X_2|Y_2) + H(S|Y_{\sim 2}) - H(S \oplus X_2|Y_1^{N_\ell}) && (6.65)\end{aligned}$$

where (6.64) follows from the chain rule of entropy and (6.65) because $X_2 - Y_2 - Y_{\sim 2}$ and $T - Y_{\sim 2} - Y_2$ form Markov chains with $Y_{\sim i}$ being the random vector where the i-th element is removed. A lower bound on the last term in the RHS of (6.65) is

$$\begin{aligned} H(S \oplus X_2|Y_1^{N_\ell}) &\geq H_2\left(H_2^{-1}\left(H(S|Y_{\sim 2})\right) * H_2^{-1}\left(H(X_2|Y_2)\right)\right) && (6.66)\\ &\geq H(S|Y_{\sim 2}) + [1 - H(S|Y_{\sim 2})]H^2\left(X_2|Y_2\right) && (6.67)\end{aligned}$$

where (6.66) is due to the Jensen's inequaltiy applied twice via Mrs. Gerber's Lemma [133] by noting that $(S, Y_{\sim 2})$ and (X_2, Y_2) are mutually independent, and (6.67) follows from the RHS of (6.50). Then, we subtract $H(W)$ from the both sides of (6.65) to obtain

$$\begin{aligned} H\left(X_2 \middle| Y_1^{N_\ell}, S \oplus X_2\right) - H(W) &\leq H(S|Y_{\sim 2}) - H(S \oplus X_2|Y_1^{N_\ell}) && (6.68)\\ &\leq -[1 - H(S|Y_{\sim 2})]H^2(W) && (6.69)\\ &= -[1 - H(X_1 \oplus X_3 \oplus \ldots \oplus X_{N_\ell}|Y_{\sim 2})]H^2(W) && (6.70)\end{aligned}$$

where (6.69) follows from (6.67), and (6.70) by recalling $S = X_1 \oplus X_3 \oplus \ldots \oplus X_{N_\ell}$. For an upper-bound on (6.70), we write

$$H(X_1 \oplus X_3 \oplus \ldots \oplus X_{N_\ell}|Y_{\sim 2}) = \sum_{y_{\sim 2}} p(y_{\sim 2}) H(X_1 \oplus X_3 \oplus \ldots \oplus X_{N_\ell}|Y_{\sim 2} = y_{\sim 2}) \tag{6.71}$$

$$= \sum_{y_{\sim 2}} p(y_1) p(y_3^{N_\ell}) H_2\left(p_{X_1 \oplus X_3 \oplus \ldots \oplus X_{N_\ell}|Y_{\sim 2} = y_{\sim 2}}\right) \tag{6.72}$$

$$= \sum_{y_{\sim 2}} p(y_1) p(y_3^{N_\ell}) H_2\left(p_{X_1|Y_1=y_1} * p_{X_3 \oplus \ldots \oplus X_{N_\ell}|Y_3^{N_\ell} = y_3^{N_\ell}}\right) \tag{6.73}$$

$$\leq H(X_1|Y_1) + [1 - H(X_1|Y_1)] H(X_3 \oplus \ldots \oplus X_{N_\ell}|Y_3^{N_\ell}) \tag{6.74}$$

$$= H(W) + [1 - H(W)] H(X_3 \oplus \ldots \oplus X_{N_\ell}|Y_3^{N_\ell}) \tag{6.75}$$

where (6.72) and (6.73) follows by defining quantities $p_{X|Y=y} \triangleq H_2^{-1}(H(X|Y=y))$ and the independence of $\{Y_i\}$, $i \in [N_\ell]$, and (6.74) from the LHS of (6.50). Since $X_2 = U_2$ and $U_1 = S \oplus X_2$, combining (6.75) and (6.70) gives

$$H\left(U_2 \middle| Y_1^{N_\ell}, U_1\right) - H(W) \leq -\left[1 - \left(H(W) + [1 - H(W)] H(X_3 \oplus \ldots \oplus X_{N_\ell}|Y_3^{N_\ell})\right)\right] H^2(W) \tag{6.76}$$

$$= -\left(1 - H(X_3 \oplus \ldots \oplus X_{N_\ell}|Y_3^{N_\ell})\right) [1 - H(W)] H^2(W) \tag{6.77}$$

where (6.77) follows from algebraic manipulation. Now recall (6.63). By recursively applying the steps to reach (6.77), one obtains

$$H\left(U_i \middle| Y_1^{N_\ell}, U_1^{i-1}\right) - H(W) \leq -[1 - H(W)]^{N_\ell - 1} H^2(W) \tag{6.78}$$

for $i = 2, \ldots, N_\ell$.

Using the chain rule for conditional entropy, we have

$$\sum_{i=1}^{N_\ell} \left[H(U_i|Y_1^{N_\ell}, U_1^{i-1}) - H(W)\right] = H(U_1^{N_\ell}|Y_1^{N_\ell}) - N_\ell H(W) = 0. \tag{6.79}$$

Combining (6.78) and (6.79) gives

$$H\left(U_1 \middle| Y_1^{N_\ell}\right) - H(W) \geq (N_\ell - 1)[1 - H(W)]^{N_\ell - 1} H^2(W) \tag{6.80}$$

$$\geq [1 - H(W)]^{N_\ell - 1} H^2(W) \tag{6.81}$$

where (6.81) follows because $N_\ell \geq 2$. We obtain (6.62) by setting $\alpha = N_\ell - 1$ and $\beta = 2$, which concludes the proof. ■

6.4.2. Proof of Theorem 6.3

The proof uses the following idea: For a product code with $(m, m-1)$ SPC component codes in m dimensions over the BEC, the largest information bit erasure probability $\epsilon_{\max}$ under SC decoding is the channel erasure probability $\epsilon \in (0,1)$, i.e., $\epsilon_{\max} = \epsilon$ as $m \to \infty$, which is proved below. Since $\epsilon_{\max}$ is a lower bound on the BLEP under SC decoding (see (6.39)), we have $\epsilon^\star = 0$.

We rewrite the recursion (6.37), for any $m \geq 0$, $N = N^{[m]}$, $1 \leq j \leq N^{[m-1]}$, $1 \leq k \leq N_m$, $t \triangleq N_m(j-1)$, as a mutual information:

$$I^{t+k}_{\boldsymbol{G}^{[m]}} = \begin{cases} 1 - \left(1 - \left(I^j_{\boldsymbol{G}^{[m-1]}}\right)^{N_m}\right) & \text{if } k = 1 \\ 1 - \left(1 - I^j_{\boldsymbol{G}^{[m-1]}}\right)\left(1 - \left(I^j_{\boldsymbol{G}^{[m-1]}}\right)^{N_m - k + 1}\right) & \text{otherwise.} \end{cases} \tag{6.82}$$

Note that $I^i_{\boldsymbol{G}^{[m]}} = 1 - \epsilon^i_{\boldsymbol{G}^{[m]}}$ where $I^i_{\boldsymbol{G}^{[m]}}$ is the mutual information of the BEC with an erasure probability of $\epsilon^i_{\boldsymbol{G}^{[m]}}$ with uniform inputs. We are interested in

$$I_{\min} \triangleq 1 - \epsilon_{\max} = 1 - \epsilon_\alpha = I_\alpha \tag{6.83}$$

for BEC(ϵ), with $\epsilon \in [0,1)$, which can be calculated recursively via (6.82). To simplify the recursion, observe that the multi-base binary representation α_1^m of $\alpha - 1$ is the length-m all-ones vector (recall (6.22) and (6.38)). We now rewrite the first (and single) recursion, using the RHS of (6.82) with $N_m = m$ as

$$f(I) \triangleq 1 - (1 - I)\left(1 - I^{m-1}\right) \tag{6.84}$$

$$= I + I^{m-1} - I^m \tag{6.85}$$

by setting $I = 1 - \epsilon$. This recursion is illustrated in the "EXIT chart" in Figure 6.14 for $N_\ell = m$, $\ell \in [m]$, $m = 10$, and $\epsilon = 0.37$. For the considered construction, the top curve in the figure shifts down for a larger m, resulting in a narrower tunnel between the two curves, although the number of recursions m increases. One must have $I_{\min} \to 1$, i.e., one reaches the $(1,1)$ point with m recursions in the figure for $P_B^{(\mathrm{SC})} \to 0$. In the following, we provide an answer for the question on the dominating effect (narrower tunnel or more recursions) with increasing m.

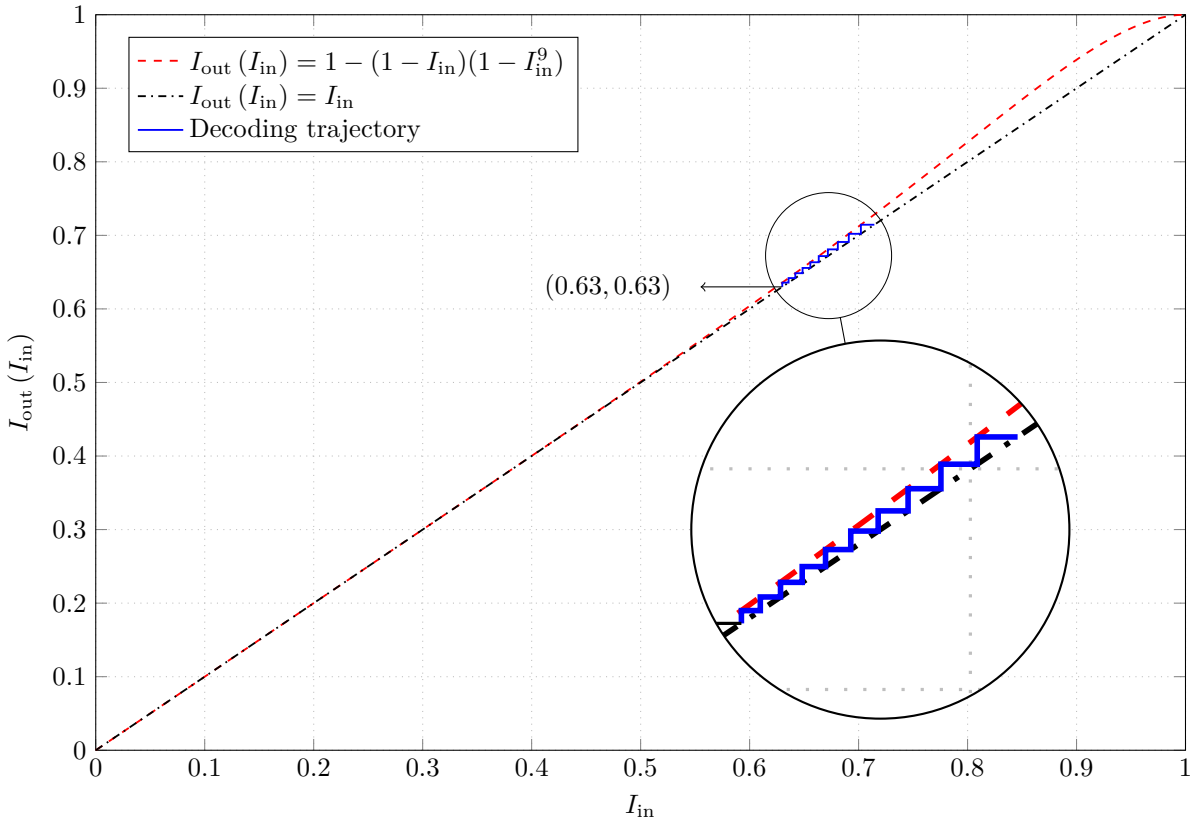

Figure 6.14.: Decoding trajectory for the bit u_α of the 10-dimensional SPC product code, where the component codes are $(10, 9)$ SPC codes, over the BEC(0.37).

Using (6.85) and $\alpha_1^m = 1$, we know that the mutual information after m iterations is $I_\alpha = f^{\circ m}(I)$ where $f^{\circ m}(I) = f\left(f^{\circ m-1}(I)\right)$ denotes the m-th iteration of function f with $f^{\circ 1}(I) \triangleq f(I)$. We are interested in the smallest channel mutual information for which the block error rate converges to zero asymptotically in m, i.e.,

$$I^\star = \inf_{I\in(0,1]}\{I : \lim_{m\to\infty} P_B^{(\mathrm{SC})} \to 0\}. \tag{6.86}$$

Consider an arbitrary $\delta > 0$. For any positive $\gamma < 1$, there is a sufficiently large m such that $m\gamma^{m-2} \leq \delta$. Then for any non-negative $I \leq \gamma$, we write

$$\begin{aligned} f(I) &= I + I^{m-1} - I^m && (6.87)\\ &\leq I\left(1 + I^{m-2}\right) && (6.88)\\ &\leq I\left(1 + \frac{\delta}{m}\right) && (6.89) \end{aligned}$$

where (6.88) follows because $I \geq 0$ and (6.89) because $I \leq \gamma$. Combined with $m\gamma^{m-2} \leq \delta$, this leads to $I^{m-2} \leq \frac{\delta}{m}$. For any initial $I_{\mathrm{ch}} \leq \mathsf{e}^{-\delta}\gamma$ and any $m' \leq m$, we have

$$f^{\circ m'}(I) \leq \mathsf{e}^{-\delta}\gamma\left(1 + \frac{\delta}{m}\right)^{m'} \leq \gamma \tag{6.90}$$

which ensures that the condition $I \leq \gamma$ for (6.89) is not violated with m' iterations. Therefore, for any positive $\gamma < 1$, any $\delta > 0$ and any $I_{\mathrm{ch}} \leq \mathsf{e}^{-\delta}\gamma$, we have

$$\lim_{m\to\infty} f^{\circ m}(I) \leq I\mathsf{e}^{\delta}. \tag{6.91}$$

The result follows by choosing δ small and γ close to 1. ■

6.4.3. Proof of Theorem 6.4

Let $f(\boldsymbol{Z}) = \prod_{i=1}^{\nu} f_i(z_{i,1}, \ldots, z_{i,N})$ be a multinomial in the variables $\boldsymbol{Z} = \{z_{i,j}\}$, $1 \leq i \leq \nu$, $1 \leq j \leq N$, where each factor f_i is a multinomial only in the variables $z_{i,1}, \ldots, z_{i,N}$. Assume that each variable $z_{i,j}$ appears with exponent either 0 or 1 in the multinomial $f_i(z_{i,1}, \ldots, z_{i,N})$, and, hence, in $f(\boldsymbol{Z})$. Suppose that we wish to remove from $f(\boldsymbol{Z})$ all the terms in the form

$$\prod_{i\in\mathcal{T}} z_{i,j} \tag{6.92}$$

where $|\mathcal{T}|$ is odd. The remaining terms can be obtained by computing

$$\frac{1}{2}\sum_{y\in\{+1,-1\}}\prod_{i=1}^{\nu} f_i(z_{i,1},\ldots,z_{i,j-1},yz_{i,j},z_{i,j+1},\ldots,z_{i,N}). \tag{6.93}$$

Similarly, to remove all the terms in the form

$$\prod_{i\in\mathcal{T}_j} z_{i,j}, \quad j\in[N] \tag{6.94}$$

where $|\mathcal{T}_j|$ is odd, it suffices to evaluate

$$2^{-N}\sum_{\boldsymbol{y}\in\{+1,-1\}^N}\prod_{i=1}^{\nu} f_i(y_1z_{i,1},\ldots,y_Nz_{i,N}). \tag{6.95}$$

For a given (N,K) binary linear block code $\mathcal{C}$, its complete weight enumerating function (CWEF) is

$$A_{\mathcal{C}}(\boldsymbol{z}) \triangleq \sum_{\boldsymbol{x}\in\mathcal{C}} \boldsymbol{z}^{\boldsymbol{x}} \tag{6.96}$$

where $\boldsymbol{z}^{\boldsymbol{x}} \triangleq \prod_{i=1}^{n} z_i^{x_i}$. Consider first a product code composed of $\nu\times N_1$ arrays whose rows and columns are codewords of $\mathcal{C}_1$ and a trivial rate-1 code $\mathcal{I}$ with a generator matrix $\boldsymbol{I}_\nu$, respectively. Generalizing the definition (6.96), the CWEF $A_{\mathcal{C}_1\otimes\mathcal{I}}(\boldsymbol{Z})$ of the product code uses the dummy variables $\boldsymbol{Z}=\{z_{i,j}\}$, $1\le i\le\nu$, $1\le j\le N_1$, to track bits by their (i,j) coordinate in the codeword. This is obtained simply by multiplying the CWEFs given as (6.96) for the codes corresponding to each row i, i.e., we have

$$A_{\mathcal{C}_1\otimes\mathcal{I}}(\boldsymbol{Z}) = \prod_{i=1}^{\nu} A_{\mathcal{C}_1}(z_{i,1},\ldots,z_{i,N_1}). \tag{6.97}$$

Recall now that the codewords of $\mathcal{C}$ are $\nu\times N_1$ arrays whose rows and columns are codewords of $\mathcal{C}_1$ and $\mathcal{S}_\nu$, respectively. Then the CWEF $A_{\mathcal{C}}(\boldsymbol{Z})$ of the product code $\mathcal{C}$ is derived from $A_{\mathcal{C}_1\otimes\mathcal{I}}(\boldsymbol{Z})$ by requiring that each column word has even weight, i.e., we have

$$A_{\mathcal{C}}(\boldsymbol{Z}) = 2^{-N_1}\sum_{\boldsymbol{y}\in\{+1,-1\}^{N_1}}\prod_{i=1}^{\nu} A_{\mathcal{C}_1}(y_1z_{i,1},\ldots,y_{N_1}z_{i,N_1}). \tag{6.98}$$

The WEF of the product code $\mathcal{C}$ is obtained by setting $z_{i,j}=z$, yielding

$$A_{\mathcal{C}}(z) = 2^{-N_1}\sum_{\boldsymbol{y}\in\{1,-1\}^{N_1}} \left(A_{\mathcal{C}_1}(y_1z,\ldots,y_{N_1}z)\right)^{\nu} \tag{6.99}$$

$$= 2^{-N_1} \sum_{\boldsymbol{y}\in\{1,-1\}^{N_1}} \left(\sum_{\boldsymbol{x}\in\mathcal{C}_1} z^{w_{\mathrm{H}}(\boldsymbol{x})} \prod_{i'=1}^{N_1} y_{i'}^{x_{i'}} \right)^{\nu} \tag{6.100}$$

$$= 2^{-N_1} \sum_{\boldsymbol{y}\in\{0,1\}^{N_1}} \left(\sum_{\boldsymbol{x}\in\mathcal{C}_1} z^{w_{\mathrm{H}}(\boldsymbol{x})} (-1)^{\boldsymbol{x}\cdot\boldsymbol{y}^{\mathrm{T}}} \right)^{\nu} \tag{6.101}$$

$$= 2^{-N_1} \sum_{\boldsymbol{y}\in\{0,1\}^{N_1}} \left(\sum_{\boldsymbol{u}\in\{0,1\}^{K_1}} z^{w_{\mathrm{H}}(\boldsymbol{uG})} (-1)^{\boldsymbol{uG}\boldsymbol{y}^{\mathrm{T}}} \right)^{\nu} \tag{6.102}$$

$$= 2^{-N_1} \sum_{\boldsymbol{y}\in\{0,1\}^{N_1}} \left(\sum_{\boldsymbol{u}\in\{0,1\}^{K_1}} z^{w_{\mathrm{H}}(\boldsymbol{uG})} (-1)^{\boldsymbol{uG}\tilde{\boldsymbol{H}}\boldsymbol{y}^{\mathrm{T}}} \right)^{\nu} \tag{6.103}$$

$$= 2^{-N_1} \sum_{y_1^{N_1-K_1}} \sum_{\boldsymbol{v}\in\{0,1\}^{K_1}} \left(\sum_{\boldsymbol{u}\in\{0,1\}^{K_1}} z^{w_{\mathrm{H}}(\boldsymbol{uG})} (-1)^{\boldsymbol{uG}\bar{\boldsymbol{H}}^{\mathrm{T}}\boldsymbol{v}^{\mathrm{T}}} \right)^{\nu} \tag{6.104}$$

with $\tilde{\boldsymbol{H}} \triangleq \left[\boldsymbol{H}^{\mathrm{T}}, \bar{\boldsymbol{H}}^{\mathrm{T}}\right]$ where $\boldsymbol{H}$ and $\bar{\boldsymbol{H}}$ are the parity-check matrix of $\mathcal{C}_1$ and a complementary matrix such that $\tilde{\boldsymbol{H}}$ is non-singular, respectively, and $\boldsymbol{v} \triangleq y_{N_1-K_1+1}^{N_1}$. Equation (6.100) follows from (6.96) by noting that $(y_1 z, \ldots, y_{N_1} z)^{\boldsymbol{x}} = z^{w_{\mathrm{H}}(\boldsymbol{x})} \boldsymbol{y}^{\boldsymbol{x}}$, (6.101) from re-defining the dummy vector $\boldsymbol{y}$, (6.102) from $\boldsymbol{x} = \boldsymbol{uG}$ and performing the sum over $\boldsymbol{u} \in \{0,1\}^{K_1}$ instead of $\boldsymbol{x} \in \mathcal{C}_1$, (6.103) from the nonsingularity of $\tilde{\boldsymbol{H}}$ and the sum being over all possible $\boldsymbol{y} \in \{0,1\}^{N_1}$, and (6.104) from dividing the outer sum into two parts, namely over $y_1^{N_1-K_1} \in \{0,1\}^{N_1-K_1}$ and $\boldsymbol{v} \in \{0,1\}^{K_1}$, and because the product $\boldsymbol{GH}^{\mathrm{T}}$ results in the all-zeroes matrix. Finally, (6.48) follows because the outer sum can be removed by multiplying the remaining term by $2^{N_1-K_1}$ and because the product $\boldsymbol{G}\bar{\boldsymbol{H}}^{\mathrm{T}}$ is non-singular.[6] ■

6.4.4. Proof of Theorem 6.5

Recall (6.98) and assume, without loss of generality, that the component code generator matrices are of the form $[\boldsymbol{I}_{k_i}|\boldsymbol{P}_i]$. We set $z_{i,j} = xz$ for $1 \le i \le K_1$ and $1 \le j \le K_2 = \nu - 1$, and $z_{i,j} = z$ otherwise, to obtain

$$A_{\mathcal{C}}(x,z) = 2^{-N_1} \sum_{\boldsymbol{y}\in\{1,-1\}^{N_1}} \left(\sum_{\boldsymbol{u}\in\{0,1\}^{K_1}} x^{w_{\mathrm{H}}(\boldsymbol{u})} z^{w_{\mathrm{H}}(\boldsymbol{uG})} \boldsymbol{y}^{\boldsymbol{uG}} \right)^{\nu-1} \left(\sum_{\boldsymbol{u}\in\{0,1\}^{K_1}} z^{w_{\mathrm{H}}(\boldsymbol{uG})} \boldsymbol{y}^{\boldsymbol{uG}} \right) \tag{6.105}$$

which follows by applying similar steps as in the proof of Theorem 6.4. ■

[6] The rows of $\bar{\boldsymbol{H}}^{\mathrm{T}}$ are linearly independent of the rows of $\boldsymbol{H}$ by definition.

7

List Decoding over Block-Fading Channels

This chapter deals with non-coherent communications over fading channels, where the CSI is known neither at the transmitter nor at the receiver. The lack of CSI is usually addressed via PAT so that the receiver can operate as if communication was coherent by replacing the CSI with a noisy estimate. The quality of the channel state estimation impacts the performance significantly. For a reasonably good estimate, one may require embedding large number of pilots. When the system requires using short block, i.e., fading is fast, embedding large number of pilots costs large overhead, which causes a reduction in the resources allocated for error correction. Hence, classic PAT methods perform poorly when short blocks are used [22,25,26]. The rates achievable without CSI were investigated in [174–176]. Bounds on the BLEPs are provided in [27,177,178] not only for non-coherent transmission but also for various PAT strategies.

This chapter extends the work of [176] by introducing a PAT scheme that, thanks to the use of list decoders, drastically reduces the pilots overhead. More specifically, the pilot symbols are used to obtain a (potentially rough) channel estimate which is then employed by a list decoder to explore the neighborhood (in the codebook) of the channel observation, i.e., to construct a list of candidate codewords that achieve a large likelihood given the available channel estimate. The final decision is performed by selecting the codeword in the list according to a suitably defined non-coherent decoding metric. The pilot symbols thus enable the construction of a good list – a task that is less challenging than deriving directly a decision on the transmitted codeword. This principle can be applied to list

decoders in general and to various slow fading channels. The effectiveness of the method is illustrated via polar codes under SCL decoding and for general short binary linear block codes under MRB decoding, providing large gains compared to traditional PAT schemes over single-input single-output Rayleigh block-fading channels.

The second part of the chapter proposes a pilot-free two-stage polar-coded transmission (PCT) scheme to jointly estimate the CSI and data with an adjustable complexity that can be made comparable to PAT, especially when the number of diversity branches is small. In the first stage, SCL decoding and the polar code constraints are used to estimate the CSI. In the second stage, mismatched SCL decoding proceeds with this estimate. Remarkably, PCT performs within a few tenths of a dB form the performance of a coherent receiver with perfect CSI.

Although the ideas extend to higher-order modulation as well as to transmission over various fading channels, e.g., Rician block-fading channel, our focus will be on the quadrature phase-shift keying (QPSK) and Rayleigh fading channels.

7.1. Notation and System Model

This section requires extra notation: the natural logarithm is denoted as $\log(\cdot)$. We use $\mathbb{C}$ for the field of complex numbers. For a given $x \in \mathbb{C}$, $\Re\{x\}$ and $|x|$ denote the real part and the magnitude of x, respectively. Let $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}^N$. We use $\|\boldsymbol{x}\|$ for the l^2-norm of $\boldsymbol{x}$, $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ for the inner product. We write $\mathcal{CN}(\mu, \sigma^2)$ to denote a complex Gaussian distribution with mean μ and variance σ^2. A bold letter with underline (e.g., $\underline{\boldsymbol{X}}$) is used for random vectors.

We consider a single-input single-output block-fading channel, where the fading coefficient H is constant for N_c channel uses and changes independently across B coherence blocks that are called diversity branches. The frame size is thus $N = BN_c$ symbols. Such a setup is relevant, for instance, for orthogonal frequency-division mutiplexing (OFDM) systems. The channel output of the i-th coherence block is

$$\boldsymbol{y}_i = h_i \boldsymbol{x}_i + \boldsymbol{z}_i, \quad i = 1, \ldots, B \tag{7.1}$$

where $\boldsymbol{x}_i \in \mathcal{X}^{N_c}$ and $\boldsymbol{y}_i \in \mathbb{C}^{N_c}$ are the transmitted and received vectors, $h_i \in \mathbb{C}$ is a realization of H_i and $\boldsymbol{z}_i$ is an additive white Gaussian noise (AWGN) term whose entries are i.i.d. as $\mathcal{CN}(0, \sigma^2)$. We assume that the noise variance σ^2 is known to the receiver; this may be justified by the slow time scale of receiver device variations as compared to fading due to mobility. The mutually independent RVs H_i and $\boldsymbol{N}_i$ are assumed to be independent over i. A vector without subscripts denotes a concatenation of vectors or

scalars, e.g., $\boldsymbol{y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_B)$, $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_B)$ and $\boldsymbol{h} = (h_1, \ldots, h_B)$.

Consider QPSK with Gray labeling where the energy per symbol is normalized to 1. This means that the input alphabet is

$$\mathcal{X} = \{\pm\Delta \pm j\Delta\}, \quad \Delta = \frac{1}{\sqrt{2}}, \tag{7.2}$$

and we map the binary vector (a codeword) c_1^{2m} to $x_1^m \in \mathcal{X}^m$ via $\chi : \{0,1\}^{2m} \mapsto \mathcal{X}^m$ as

$$\chi\left(c_1^{2m}\right) = (\chi_g(c_1, c_2), \chi_g(c_3, c_4), \ldots, \chi_g(c_{2m-1}, c_{2m})) \tag{7.3}$$

where

$$\chi_g\left(c_1, c_2\right) = (-1)^{c_1}\Delta + j(-1)^{c_2}\Delta. \tag{7.4}$$

The mapping (7.3) is *symmetric*, i.e., if $\chi\left(c_1^{2m}\right) = \boldsymbol{x}$, then $\chi\left(c_1^{2m} \oplus 1\right) = -\boldsymbol{x}$.

7.2. Decoding Rules over Fading Channels

We revisit various block-wise decoding rules for different assumptions on the availability of the CSI and on the frame structure. This will help us understand the relevant finite-length performance bounds used as benchmarks, as well as possible ways to improve the performance.

7.2.1. Decoding with Perfect CSI

If the channel coefficients are known to the receiver, the (coherent) ML decoding rule is

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmax}}\, p_{\underline{\boldsymbol{Y}}|\underline{\boldsymbol{X}},\underline{\boldsymbol{H}}}(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{h}) \tag{7.5}$$

$$= \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmax}} \prod_{i=1}^{B} p_{\underline{\boldsymbol{Y}}_i|\underline{\boldsymbol{X}}_i,H_i}(\boldsymbol{y}_i|\boldsymbol{x}_i, h_i) \tag{7.6}$$

$$= \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmin}} \sum_{i=1}^{B} ||\boldsymbol{y}_i - h_i\boldsymbol{x}_i||^2 \tag{7.7}$$

$$= \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmax}} \sum_{i=1}^{B} 2\Re\{\langle \boldsymbol{y}_i, h_i\boldsymbol{x}_i\rangle\} - |h_i|^2 \|\boldsymbol{x}_i\|^2 \tag{7.8}$$

where $\mathcal{C}$ is the set of transmitted signal vectors induced by the chosen channel code and modulation. When $\|\boldsymbol{x}_i\|$ is constant across codewords and blocks, we have

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmax}} \sum_{i=1}^{B} \Re\{\langle \boldsymbol{y}_i, h_i \boldsymbol{x}_i \rangle\} \tag{7.9}$$

which is the case, for instance, if the modulation is QPSK.

7.2.2. Decoding without CSI

Assume next that the decoder does not have access to the channel coefficients. We refer to this setting as non-coherent.

Blind Approach

if no pilots are embedded in the transmitted sequence, we rely on a *blind* approach. If the receiver does not possess information on the distribution of the channel coefficients then the problem can be tackled by designing a generalized likelihood-ratio test (GLRT) as in [179] yielding

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmax}} \max_{\boldsymbol{h}} p_{\underline{\boldsymbol{Y}}|\underline{\boldsymbol{X}},\underline{\boldsymbol{H}}}(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{h}) \tag{7.10}$$

$$= \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmin}} \sum_{i=1}^{B} \min_{h_i} ||\boldsymbol{y}_i - h_i \boldsymbol{x}_i||^2 \tag{7.11}$$

$$= \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmax}} \sum_{i=1}^{B} \frac{|\langle \boldsymbol{y}_i, \boldsymbol{x}_i \rangle|^2}{\|\boldsymbol{x}_i\|^2}. \tag{7.12}$$

where (7.12) follows by observing that the ML channel estimate (7.23) given $\boldsymbol{y}_i$ and $\boldsymbol{x}_i$ minimizes the objective function at RHS of (7.11). For QPSK, (7.12) reduces to

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmax}} \sum_{i=1}^{B} |\langle \boldsymbol{y}_i, \boldsymbol{x}_i \rangle|^2. \tag{7.13}$$

If the receiver knows the channel coefficients distribution, then the non-coherent ML estimate is

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmax}} \prod_{i=1}^{B} \mathbb{E}_{H_i}[p_{\underline{\boldsymbol{Y}}_i|\underline{\boldsymbol{X}}_i,H_i}(\boldsymbol{y}_i|\boldsymbol{x}_i, H_i)] \tag{7.14}$$

$$= \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmax}} \sum_{i=1}^{B} p_{\underline{\boldsymbol{Y}}_i|\underline{\boldsymbol{X}}_i}(\boldsymbol{y}_i|\boldsymbol{x}_i). \tag{7.15}$$

Example 7.1 (Non-coherent decoding metrics for Rician and Rayleigh block-fading channels)**.** Suppose the channel coefficients are distributed as $H_i \sim \mathcal{CN}(\mu_{\mathrm{H}}, \sigma_{\mathrm{H}}^2)$, $i \in [B]$. Observe that $p_{\underline{\boldsymbol{Y}}_i|\underline{\boldsymbol{X}}_i}(\boldsymbol{y}_i|\boldsymbol{x}_i)$ is the PDF of a conditionally Gaussian distribution with mean $\mu_{\boldsymbol{y}_i|\boldsymbol{x}_i} \triangleq \mathbb{E}\left[\boldsymbol{y}_i|\boldsymbol{x}_i\right]$ and covariance matrix $\boldsymbol{C}_{\boldsymbol{y}_i|\boldsymbol{x}_i} \triangleq \mathbb{E}\left[\boldsymbol{y}_i^{\mathrm{H}}\boldsymbol{y}_i|\boldsymbol{x}_i\right]$; hence, we rewrite (7.15) as

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmax}} \prod_{i=1}^{B} \frac{1}{\pi^{N_c}\left|\boldsymbol{C}_{\boldsymbol{y}_i|\boldsymbol{x}_i}\right|} \exp\left(-\left(\boldsymbol{y}_i - \mu_{\boldsymbol{y}_i|\boldsymbol{x}_i}\boldsymbol{x}_i\right)\boldsymbol{C}_{\boldsymbol{y}_i|\boldsymbol{x}_i}^{-1}\left(\boldsymbol{y}_i - \mu_{\boldsymbol{y}_i|\boldsymbol{x}_i}\boldsymbol{x}_i\right)^{\mathrm{H}}\right) \tag{7.16}$$

$$= \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmax}} \sum_{i=1}^{B} \frac{\sigma_{\mathrm{H}}^2\left|\langle\boldsymbol{y}_i, \boldsymbol{x}_i\rangle\right|^2 + 2\sigma^2\Re\left\{\langle\boldsymbol{y}_i, \mu_{\mathrm{H}}\boldsymbol{x}_i\rangle\right\} + \sigma^2\left|\mu_{\mathrm{H}}\right|^2\|\boldsymbol{x}_i\|^2}{\sigma^2\left(\sigma^2 + \sigma_{\mathrm{H}}^2\|\boldsymbol{x}_i\|^2\right)} - \log\left(1 + \frac{\sigma_{\mathrm{H}}^2}{\sigma^2}\|\boldsymbol{x}_i\|^2\right) \tag{7.17}$$

where (7.17) follows from taking the logarithm of the objective, inserting $\mu_{\boldsymbol{y}_i|\boldsymbol{x}_i} = \mu_{\mathrm{H}}\boldsymbol{x}_i$, $\boldsymbol{C}_{\boldsymbol{y}_i|\boldsymbol{x}_i} = \sigma_{\mathrm{H}}^2\boldsymbol{x}_i^{\mathrm{H}}\boldsymbol{x}_i + \sigma^2\boldsymbol{I}_{N_c}$ in (7.16) and applying algebraic manipulations. Note that $\left|\boldsymbol{C}_{\boldsymbol{y}_i|\boldsymbol{x}_i}\right|$ can be found using matrix determinant lemma [180, Lemma 1.1] as

$$\left|\boldsymbol{C}_{\boldsymbol{y}_i|\boldsymbol{x}_i}\right| = \sigma^{2N_c}\left(1 + \|\boldsymbol{x}_i\|^2\frac{\sigma_{\mathrm{H}}^2}{\sigma^2}\right) \tag{7.18}$$

and $\boldsymbol{C}_{\boldsymbol{y}_i|\boldsymbol{x}_i}^{-1}$ via the formula [181, Equation 2] as

$$\boldsymbol{C}_{\boldsymbol{y}_i|\boldsymbol{x}_i}^{-1} = \frac{1}{\sigma^2}\left(\boldsymbol{I}_{N_c} - \frac{\sigma_{\mathrm{H}}^2}{\sigma^2 + \sigma_{\mathrm{H}}^2\|\boldsymbol{x}_i\|^2}\boldsymbol{x}_i^{\mathrm{H}}\boldsymbol{x}_i\right). \tag{7.19}$$

The model (7.1) yields a Rician block-fading channel when $\mu_{\mathrm{H}} = \sqrt{\kappa/1+\kappa}$ and $\sigma_{\mathrm{H}}^2 = (1+\kappa)^{-1}$, $\kappa \geq 0$, and Rayleigh block-fading if $\kappa = 0$. With increasing κ, the channel behaves like a standard additive white Gaussian noise channel (AWGNC), and exactly so for $\kappa = \infty$. In the following, we mainly investigate Rayleigh fading. For $\kappa = \infty$, (7.17) recovers (7.8). For $\kappa = 0$, (7.17) yields

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmax}} \sum_{i=1}^{B} \frac{\left|\langle\boldsymbol{y}_i, \boldsymbol{x}_i\rangle\right|^2}{\sigma^2\left(\sigma^2 + \|\boldsymbol{x}_i\|^2\right)} - \log\left(1 + \frac{1}{\sigma^2}\|\boldsymbol{x}_i\|^2\right). \tag{7.20}$$

For QPSK, we recover (7.13).

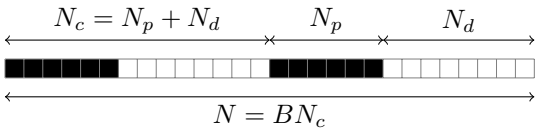

Figure 7.1.: A PAT frame structure with $B = 2$ coherence blocks. Dark and white boxes represent pilot and coded symbols, respectively.

Pilot-Assisted Channel Estimation

The idealized setting described in Section 7.2.1 is often approximated by including pilot symbols in the transmitted sequence which are used to estimate the channel coefficients. Consider, without loss of generality, PAT as shown in Figure 7.1 where the first N_p symbols in each coherence block are pilot symbols $\boldsymbol{x}_i^{\mathsf{p}}$ and the remaining $N_d = N_c - N_p$ symbols $\boldsymbol{x}_i^{\mathsf{d}}$ are coded. Upon observing $\boldsymbol{y}$, an ML estimate of the CSI is

$$\hat{h}_i = \operatorname*{argmax}_{h\in\mathbb{C}} p_{\underline{\boldsymbol{Y}}_i^{\mathsf{p}}|\underline{\boldsymbol{X}}_i^{\mathsf{p}},H_i}(\boldsymbol{y}_i^{\mathsf{p}}|\boldsymbol{x}_i^{\mathsf{p}}, h) \tag{7.21}$$

$$= \operatorname*{argmin}_{h\in\mathbb{C}} \|\boldsymbol{y}_i^{\mathsf{p}} - h_i\boldsymbol{x}_i^{\mathsf{p}}\|^2 \tag{7.22}$$

$$= \frac{\langle \boldsymbol{y}_i^{\mathsf{p}}, \boldsymbol{x}_i^{\mathsf{p}}\rangle}{\|\boldsymbol{x}_i^{\mathsf{p}}\|^2}. \tag{7.23}$$

where (7.23) follows from taking the derivative of the objective in (7.22) with respect to h and setting it to 0. This estimate $\hat{\boldsymbol{h}}$ is treated as ideal by the decoder, yielding the mismatched decoding rule

$$\hat{\boldsymbol{x}}^{\mathsf{d}} = \operatorname*{argmax}_{\boldsymbol{x}^{\mathsf{d}}\in\mathcal{C}} p_{\underline{\boldsymbol{Y}}^{\mathsf{d}}|\underline{\boldsymbol{X}}^{\mathsf{d}},\underline{\boldsymbol{H}}}(\boldsymbol{y}^{\mathsf{d}}|\boldsymbol{x}^{\mathsf{d}}, \hat{\boldsymbol{h}}) \tag{7.24}$$

$$= \operatorname*{argmin}_{\boldsymbol{x}^{\mathsf{d}}\in\mathcal{C}} \sum_{i=1}^{B} 2\Re\{\langle \boldsymbol{y}_i^{\mathsf{d}}, \hat{h}_i\boldsymbol{x}_i^{\mathsf{d}}\rangle\} - \left|\hat{h}_i\right|^2 \|\boldsymbol{x}_i^{\mathsf{d}}\|^2. \tag{7.25}$$

For QPSK, this reduces to

$$\hat{\boldsymbol{x}}^{\mathsf{d}} = \operatorname*{argmax}_{\boldsymbol{x}^{\mathsf{d}}\in\mathcal{C}} \sum_{i=1}^{B} \Re\{\langle \boldsymbol{y}_i^{\mathsf{d}}, \hat{h}_i\boldsymbol{x}_i^{\mathsf{d}}\rangle\}. \tag{7.26}$$

To keep the overall rate fixed, the (N, K) code is punctured suitably so that the code length after puncturing is $N_{\text{punc}} = N - 2BN_p = 2BN_d$ with QPSK. The pilot and coded

symbols have the same energy. The rate in terms of bits per channel use (BPCU) is

$$R = \frac{K}{BN_c} \tag{7.27}$$

where K is the number of information bits encoded by $\mathcal{C}$. with QPSK modulation, the rate of the code $\mathcal{C}$ is instead denoted by

$$R_0 = \frac{K}{2B(N_c - N_p)}. \tag{7.28}$$

A traditional PAT scheme initiates a standard decoding algorithm (e.g., Viterbi decoding for convolutional codes, SCL decoding for polar codes, etc.) after estimating the channel with mismatched probabilities. In particular, note that the knowledge of the correct values of $h_1, \ldots, h_B$ allows to factorize as

$$p_{\underline{\boldsymbol{Y}}_i|\underline{\boldsymbol{X}}_i,H_i}(\boldsymbol{y}_i|\boldsymbol{x}_i, h_i) = \prod_{j=1}^{N_c} p_{Y|X,h_i}(y_{i,j}|x_{i,j}, h_i), i \in [B] \tag{7.29}$$

where $x_{i,j}$ and $y_{i,j}$ are the j-th element of $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$, respectively. This factorization is typically required by standard decoders. Even with imperfect channel estimations, we can factorize the mismatched likelihoods as

$$p_{\underline{\boldsymbol{Y}}_i|\underline{\boldsymbol{X}}_i,H_i}(\boldsymbol{y}_i|\boldsymbol{x}_i, \hat{h}_i) = \prod_{j=1}^{N_c} p_{Y|X,H_i}(y_{i,j}|x_{i,j}, \hat{h}_i), i \in [B] \tag{7.30}$$

which enables the use of standard decoders designed for memoryless channels. Note also that, for a fixed rate R and a fixed blocklength BN_c, a large number of pilots provides a relatively accurate channel estimate at the cost of an increase in the code rate R_0, and thus a reduction of the error correction capability. This yields a trade-off between resources allocated to channel estimation and error correction (see, e.g., [26]).

If the receiver knows the channel coefficients' distribution, the decoding metric can be modified to account for the optimum estimate, minimizing the BLEP, by jointly processing the $\boldsymbol{y}$ and $\boldsymbol{x}^{\mathsf{p}}$ as [182, Equation (9)]

$$\hat{\boldsymbol{x}}^{\mathsf{d}} = \underset{\boldsymbol{x}^{\mathsf{d}} \in \mathcal{C}}{\operatorname{argmax}} \prod_{i=1}^{B} \mathbb{E}_{H_i} \left[p_{\underline{\boldsymbol{Y}}_i|\underline{\boldsymbol{X}}_i,H_i}(\boldsymbol{y}_i|\boldsymbol{x}_i, H_i) \right] \tag{7.31}$$

$$= \underset{\boldsymbol{x}^{\mathsf{d}} \in \mathcal{C}}{\operatorname{argmax}} \prod_{i=1}^{B} \mathbb{E}_{H_i} \left[p_{\underline{\boldsymbol{Y}}_i^{\mathsf{d}}|\underline{\boldsymbol{X}}_i^{\mathsf{d}},H_i}(\boldsymbol{y}_i^{\mathsf{d}}|\boldsymbol{x}_i^{\mathsf{d}}, H_i) p_{\underline{\boldsymbol{Y}}_i^{\mathsf{p}}|\underline{\boldsymbol{X}}_i^{\mathsf{p}},H_i}(\boldsymbol{y}_i^{\mathsf{p}}|\boldsymbol{x}_i^{\mathsf{p}}, H_i) \right] \tag{7.32}$$

where (7.32) follows because given H_i, $\underline{\boldsymbol{X}}_i^{\mathsf{p}}$ and $\underline{\boldsymbol{X}}_i^{\mathsf{d}}$, the received random vectors $\underline{\boldsymbol{Y}}_i^{\mathsf{p}}$ and $\underline{\boldsymbol{Y}}_i^{\mathsf{d}}$ are independent. Equation (7.32) discards estimating the channel explicitly and focuses on the optimum decoding of the codeword by treating the pilots as part of the codebook.

There is an intermediate approach, which explicitly estimates the channel as in (7.23) and then uses the channel estimates to derive a conditional Gaussian distribution for $p_{\underline{\boldsymbol{Y}}_i^{\mathsf{d}}|\underline{\boldsymbol{X}}_i^{\mathsf{d}}}\left(\boldsymbol{y}_i^{\mathsf{d}}|\boldsymbol{x}_i^{\mathsf{d}};\hat{h}_i\right)$, where $\hat{h}_i$ is used as a parameter for the distribution. Then, an ML decision is

$$\hat{\boldsymbol{x}}^{\mathsf{d}} = \underset{\boldsymbol{x}^{\mathsf{d}}\in\mathcal{C}}{\operatorname{argmax}} \prod_{i=1}^{B} \mathbb{E}_{H_i|\hat{H}_i=\hat{h}_i}\left[p_{\underline{\boldsymbol{Y}}_i^{\mathsf{d}}|\underline{\boldsymbol{X}}_i^{\mathsf{d}},H_i}\left(\boldsymbol{y}_i^{\mathsf{d}}|\boldsymbol{x}_i^{\mathsf{d}},h_i;\hat{h}_i\right)\right] \tag{7.33}$$

$$= \underset{\boldsymbol{x}^{\mathsf{d}}\in\mathcal{C}}{\operatorname{argmax}} \prod_{i=1}^{B} p_{\underline{\boldsymbol{Y}}_i^{\mathsf{d}}|\underline{\boldsymbol{X}}_i^{\mathsf{d}}}\left(\boldsymbol{y}_i^{\mathsf{d}}|\boldsymbol{x}_i^{\mathsf{d}};\hat{h}_i\right). \tag{7.34}$$

Observe that this approach requires knowing the channel coefficients' distribution in order to provide the true distribution for $p_{\underline{\boldsymbol{Y}}_i^{\mathsf{d}}|\underline{\boldsymbol{X}}_i^{\mathsf{d}}}\left(\boldsymbol{y}_i^{\mathsf{d}}|\boldsymbol{x}_i^{\mathsf{d}};\hat{h}_i\right)$.

Remark 7.1. The metrics (7.32) and (7.34) are the same for Rayleigh block-fading channels. This can be checked using the results of [182] for the case where $B > 1$ (see, e.g., [27, Equation (43)] for (7.34)). If one further assumes QPSK modulation, then we again get (7.13), where in the evaluation of terms in the summation (7.13) one has to use $\boldsymbol{x}_i = (\boldsymbol{x}_i^{\mathsf{p}}, \boldsymbol{x}_i^{\mathsf{d}})$ and $\boldsymbol{y}_i = (\boldsymbol{y}_i^{\mathsf{p}}, \boldsymbol{y}_i^{\mathsf{d}})$.

7.3. Finite-Length Performance Bounds

We review converse and achievability bounds on the average BLEP for finite-blocklengths that will be used to benchmark the coding schemes introduced. The converse bound is based on the MC theorem in [143, Theorem 28] and the achievability bounds are based on the random coding union bound with s parameter (RCUs) [178, Theorem 1].

Let $q : \mathbb{C}^{N_c} \times \mathbb{C}^{N_c} \to \mathbb{R}^+$ be a block-wise decoding metric and let $(\underline{\bar{\boldsymbol{X}}}_i, \underline{\boldsymbol{X}}_i, \underline{\boldsymbol{Y}}_i) \sim p_{\underline{\boldsymbol{X}}_i}(\bar{\boldsymbol{x}}_i)p_{\underline{\boldsymbol{X}}_i}(\boldsymbol{x}_i)p_{\underline{\boldsymbol{Y}}_i|\underline{\boldsymbol{X}}_i}(\boldsymbol{y}_i|\boldsymbol{x}_i)$, $i = 1,\ldots,B$, be independent across coherence blocks. The generalized information density is defined [178, Equation (3)] as

$$i_s(\boldsymbol{x}_i,\boldsymbol{y}_i) \triangleq \log \frac{q(\boldsymbol{x}_i,\boldsymbol{y}_i)^s}{\mathbb{E}\left[q\left(\underline{\bar{\boldsymbol{X}}}_i,\boldsymbol{y}_i\right)^s\right]} \tag{7.35}$$

where $s \geq 0$. The RCUs states that, for a given rate R, the average BLEP over of a

random code can be upper-bounded as

$$P_B \leq \inf_{s\geq 0} \mathbb{E}\left[\exp\left(-\left[\sum_{i=1}^{B} i_s(\underline{\boldsymbol{X}}_i, \underline{\boldsymbol{Y}}_i) - \log(2^{RN_cB}-1)\right]^+\right)\right]. \tag{7.36}$$

The bound in (7.36) is evaluated for the following combinations of input distributions and decoding metrics:

i) Input symbols uniformly distributed on a shell in $\mathbb{C}^{N_c}$ with $\|\boldsymbol{x}_i\|^2 = N_c$, and ML decoding, i.e., $q(\boldsymbol{x}_i, \boldsymbol{y}_i) = p_{\underline{\boldsymbol{Y}}_i|\underline{\boldsymbol{X}}_i}(\boldsymbol{y}_i|\boldsymbol{x}_i)$ as in (7.15);

ii) a pilot-assisted scheme as in Section 7.2.2 with the $N_c - N_p$ data symbols uniformly distributed on a shell in $\mathbb{C}^{N_c-N_p}$ with $\|\boldsymbol{x}_i^{\mathsf{d}}\|^2 = N_d$ and ML decoding, i.e., $q(\boldsymbol{x}_i, \boldsymbol{y}_i) = p_{\underline{\boldsymbol{Y}}_i^{\mathsf{d}}|\underline{\boldsymbol{X}}_i^{\mathsf{d}}}(\boldsymbol{y}_i^{\mathsf{d}}|\boldsymbol{x}_i^{\mathsf{d}}; \hat{h}_i)$ as for (7.34);

iii) input distribution as in ii) and mismatched decoding, i.e., $q(\boldsymbol{x}_i, \boldsymbol{y}_i) = -\|\boldsymbol{y}_i^{\mathsf{d}} - \hat{h}_i\boldsymbol{x}_i^{\mathsf{d}}\|^2$ as for (7.25).

See [27, Section III.A-III.D] for details on how to evaluate (7.36) for each of these cases.

The converse bound is based on the MC theorem. For a given average BLEP P_B, the maximum code rate R^* is upper-bounded as

$$R^* \leq R^{\mathsf{MC}}(P_B) \tag{7.37}$$

$$\triangleq \inf_{\lambda\geq 0} \frac{1}{BN_c}\left(\lambda - \log\left[\Pr\left\{\sum_{i=1}^{B} i_1(\underline{\boldsymbol{X}}_i, \underline{\boldsymbol{Y}}_i) \leq \lambda\right\} - P_B\right]^+\right). \tag{7.38}$$

For a given rate R^*, a lower bound on P_B, denoted as P_B^{MC}, can be obtained from (7.38) by finding the P_B^{MC} for which $R^{\mathsf{MC}}(P_B^{\mathsf{MC}}) = R^*$. For more details on this converse bound, the reader is referred to [27, Section III.E].

Remark 7.2. The achievability bound based on i) is relevant for non-coherent schemes using (7.15), which becomes relevant for (7.13) for Rayleigh fading channels, see Remark 7.1. For PAT, any decoder using (7.34) can use the bound based on ii) as a benchmark. Furthermore, approach iii) is relevant for the mismatched decoding based on the channel estimates. Finally, the converse bound is valid for all the schemes, and is optimistic especially for PAT irrespective of the decoding metric.

7.4. Joint Channel Estimation and List Decoding of Short Codes

By inspecting (7.13), (7.20) or (7.34), we see that the decoding metric does not admit a trivial factorization, which seems to preclude efficient decoders such as Viterbi decoding over the code trellis, and on any decoding algorithm that relies on factorizing the channel likelihood (such as BP decoding for turbo/LDPC codes or SCL decoding of polar codes).

A pragmatic solution to this problem is to use a few pilots in the transmitted sequence to bootstrap iterative decoding and channel estimation algorithms [183–189]. In Section 7.4.1, we discuss how iterative decoding and channel estimation can be applied to list decoders via MRB decoding. The scheme is based on the expectation-maximization (EM) algorithm [190] that uses the probabilities from the list of the previous iteration. Section 7.4.2 shows that list decoders in general, and SCL decoding in particular, allow for an alternative approach to PAT by using a non-coherent decoding metric. This improves the error correction capability and reduces the decoding complexity as compared to iterative schemes. This method is useful if the number of diversity branches is large and is well-suited for existing systems that already have pilots in the frames [24].

7.4.1. Iterative Channel Estimation and List Decoding via EM

We next reduce the number of pilots (and hence allow using a lower-rate code) by iterating channel estimation and decoding. In the following, we combine the EM algorithm with a list decoder, e.g., MRB or SCL decoding. The algorithm works as follows:

1. Initialize $\hat{h}_i^{(0)}$ as in (7.23) for $i = 1, \ldots, B$, and construct the list $\mathcal{L}^{(0)}$ using the channel estimates.

2. At iteration j, construct the list $\mathcal{L}^{(j)}$ using the updated channel estimates $\hat{\boldsymbol{h}}^{(j)}$. We have

   a. Expectation step:

$$Q\left(h, \hat{\boldsymbol{h}}^{(j-1)}\right) = \sum_{\boldsymbol{x}^{\mathsf{d}} \in \mathcal{L}^{(j-1)}} -P_{\underline{\boldsymbol{X}}^{\mathsf{d}}|\underline{\boldsymbol{Y}}^{\mathsf{d}},\underline{\boldsymbol{H}}}\left(\boldsymbol{x}^{\mathsf{d}}|\boldsymbol{y}^{\mathsf{d}}, \hat{\boldsymbol{h}}^{(j-1)}\right) \times \|\boldsymbol{y}_i^{\mathsf{d}} - h\boldsymbol{x}_i^{\mathsf{d}}\|^2 \qquad (7.39)$$

where we approximate $P_{\underline{\boldsymbol{X}}^{\mathsf{d}}|\underline{\boldsymbol{Y}}^{\mathsf{d}},\underline{\boldsymbol{H}}}\left(\boldsymbol{x}^{\mathsf{d}}|\boldsymbol{y}^{\mathsf{d}},\hat{\boldsymbol{h}}^{(j-1)}\right)$ as

$$P_{\underline{\boldsymbol{X}}^{\mathsf{d}}|\underline{\boldsymbol{Y}}^{\mathsf{d}},\underline{\boldsymbol{H}}}\left(\boldsymbol{x}^{\mathsf{d}}|\boldsymbol{y}^{\mathsf{d}},\hat{\boldsymbol{h}}^{(j-1)}\right) \approx \frac{p_{\underline{\boldsymbol{Y}}^{\mathsf{d}}|\underline{\boldsymbol{X}}^{\mathsf{d}},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}^{\mathsf{d}}|\boldsymbol{x}^{\mathsf{d}},\hat{\boldsymbol{h}}^{(j-1)}\right)}{\sum_{\tilde{\boldsymbol{x}}^{\mathsf{d}}\in\mathcal{L}^{(j-1)}} p_{\underline{\boldsymbol{Y}}^{\mathsf{d}}|\underline{\boldsymbol{X}}^{\mathsf{d}},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}^{\mathsf{d}}|\tilde{\boldsymbol{x}}^{\mathsf{d}},\hat{\boldsymbol{h}}^{(j-1)}\right)}. \tag{7.40}$$

b. Maximization step:

$$\hat{h}_i^{(j)} = \operatorname*{argmax}_h Q\left(h,\hat{\boldsymbol{h}}^{(j-1)}\right). \tag{7.41}$$

After performing Step 2 for a predetermined number ℓ of iterations, the final decision is obtained as in (7.26) by replacing $\hat{\boldsymbol{h}}$ and $\mathcal{L}$ by $\hat{\boldsymbol{h}}^{(\ell)}$ and $\mathcal{L}^{(\ell)}$, respectively.

7.4.2. List Decoder with in-List GLRT

We use the channel estimate (7.23) to form a list $\mathcal{L}$ of codewords. Each codeword in the list is modified by re-inserting the pilot symbols, which yields a modified list $\mathcal{L}'$. The final codeword is chosen from $\mathcal{L}'$ according to the GLRT rule (7.13), i.e., we choose

$$\hat{\boldsymbol{x}} = \operatorname*{argmax}_{\boldsymbol{x}\in\mathcal{L}'} \sum_{i=1}^{B} |\langle \boldsymbol{y}_i, \boldsymbol{x}_i\rangle|^2 \tag{7.42}$$

$$= \operatorname*{argmax}_{\boldsymbol{x}\in\mathcal{L}'} \sum_{i=1}^{B} \Re\left\{\langle \boldsymbol{y}_i^{\mathsf{d}}, \hat{h}_i\boldsymbol{x}_i^{\mathsf{d}}\rangle\right\} + \frac{1}{2n_p}|\langle \boldsymbol{y}_i^{\mathsf{d}}, \boldsymbol{x}_i^{\mathsf{d}}\rangle|^2 \tag{7.43}$$

where $\hat{\boldsymbol{x}}^{\mathsf{d}}$ is $\hat{\boldsymbol{x}}$ without the pilots. Note that the decoding metric has two contributions: A first term that resembles a coherent metric based on the estimate $\hat{\boldsymbol{h}}$, and a second term that is related to the non-coherent correlation. The second term is weighted by the inverse of the number of pilots; hence it becomes negligible when n_p is large, i.e., when the channel estimate is reliable.

The metric (7.43) lends itself to an alternative interpretation. Suppose the distribution of the channel coefficient for the i-th coherence block given its estimate $\hat{h}_i$ given via (7.23) is a complex Gaussian distribution with mean $\hat{h}_i$ and variance $\frac{\sigma^2}{\|\boldsymbol{x}_i^{\mathsf{p}}\|^2}$, i.e., $p_{H_i|\hat{H}_i=\hat{h}_i} = \mathcal{CN}\left(\hat{h}_i, \frac{\sigma^2}{\|\boldsymbol{x}_i^{\mathsf{p}}\|^2}\right)$. Then, similar to (7.33), we obtain

$$\hat{\boldsymbol{x}} = \operatorname*{argmax}_{\boldsymbol{x}\in\mathcal{C}'} \prod_{i=1}^{B} \mathbb{E}_{H_i|\hat{H}_i=\hat{h}_i}[p_{\underline{\boldsymbol{Y}}^{\mathsf{d}}|\underline{\boldsymbol{X}}^{\mathsf{d}},H_i}(\boldsymbol{y}_i^{\mathsf{d}}|\boldsymbol{x}_i^{\mathsf{d}},H_i;\hat{h}_i)] \tag{7.44}$$

$$= \underset{\boldsymbol{x}\in\mathcal{C}'}{\operatorname{argmax}} \sum_{i=1}^{B} \Re\left\{\langle \boldsymbol{y}_i^{\mathrm{d}}, \hat{h}_i \boldsymbol{x}_i^{\mathrm{d}}\rangle\right\} + \frac{1}{2\|\boldsymbol{x}_i^{\mathrm{p}}\|^2}\left|\langle \boldsymbol{y}_i^{\mathrm{d}}, \boldsymbol{x}_i^{\mathrm{d}}\rangle\right|^2 + \frac{1}{2}\left|\hat{h}_i\right|^2 \|\boldsymbol{x}_i^{\mathrm{d}}\|^2 - \frac{\sigma^2}{2}\frac{\|\boldsymbol{x}_i^{\mathrm{d}}\|^2}{\|\boldsymbol{x}_i^{\mathrm{p}}\|^2}\log\left(1+\frac{\|\boldsymbol{x}_i^{\mathrm{d}}\|^2}{\|\boldsymbol{x}_i^{\mathrm{p}}\|^2}\right) \tag{7.45}$$

where $\mathcal{C}'$ is the modified channel code obtained by re-inserting the pilot symbols in each codeword. For QPSK we recover (7.43) as $\|\boldsymbol{x}_i^{\mathrm{p}}\|^2 = N_p$ and $\|\boldsymbol{x}_i^{\mathrm{d}}\|^2 = N_d$.

Numerical Results: PAT

We use Monte Carlo simulations to compute the BLER and express the SNR as E_s/N_0. The results are compared with the bounds of Section 7.3. We consider Rayleigh block-fading channels with $B = 4$ coherence blocks and different code rates. For the specified codes, a pseudo-random interleaver is applied to the codeword bits after encoding. We consider different numbers of pilot symbols ($N_p \in \{1, 2, 3\}$) per coherence block, where puncturing adapts the blocklength to the number of channel uses available after pilot insertion. The symbols are taken from a QPSK constellation.

We first compare the performance achieved by the PAT scheme proposed in Section 7.4.2 using MRB decoding to the two baseline decoders, namely mismatched MRB decoding of Section 7.2.2 and the iterative EM-based MRB decoding of Section 7.4.1. Each coherence block has 13 channel uses, which results in 52 channel uses per message. For the simulations, we considered $K = 32$ information bits within each codeword, yielding a rate $R = 32/52 \approx 0.62$ bits per channel use. A $(96, 32)$ quasi-cyclic code is used and a suitable number of codeword bits is punctured (in case $N_p > 1$) to accommodate the pilot symbols within the 52 channel uses. The code is obtained by a tail-biting termination of a rate$-1/3$ non-systematic convolutional code with a memory 17 and generators $[552137, 614671, 772233]$ [191, Table 10.14]. The minimum distance of the quasi-cyclic code is upper-bounded by the free distance of the underlying convolutional code, which is 32. The MRB decoding order t is set to 3, which provides a reasonable trade-off between performance and decoding complexity.

For the iterative EM-based MRB decoding, we set the number of iterations to $m = 1$.[1] Figures 7.2-7.4 show that the gains achieved by the proposed decoder is up to 1.2 dB as compared to mismatched MRB decoding. The performance of the iterative EM-based MRB decoding is only marginally better than the one obtained by the mismatched one. Remarkably, the proposed approach performs close to the RCUs for PAT and ML decoding except for $N_p = 1$. In the simulated setting, the proposed approach provides the best

[1] There is a diminishing return in the gain with larger number of iterations and MRB decoding has high complexity with a larg number of iterations.

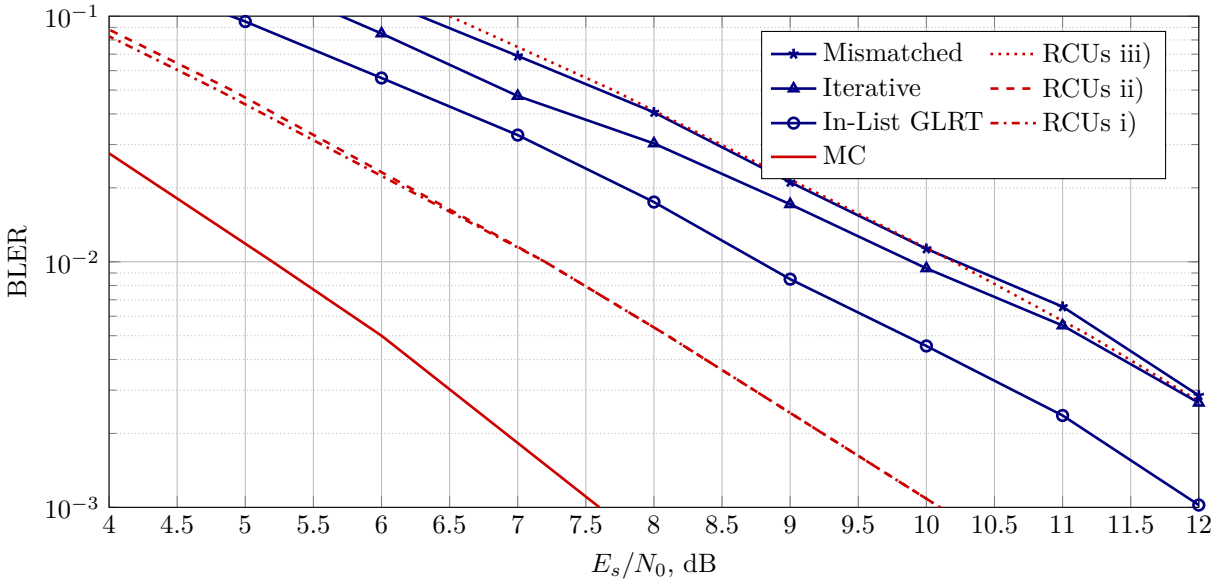

Figure 7.2.: BLER vs. SNR for the PAT schemes using MRB decoding with $N_p = 1$.

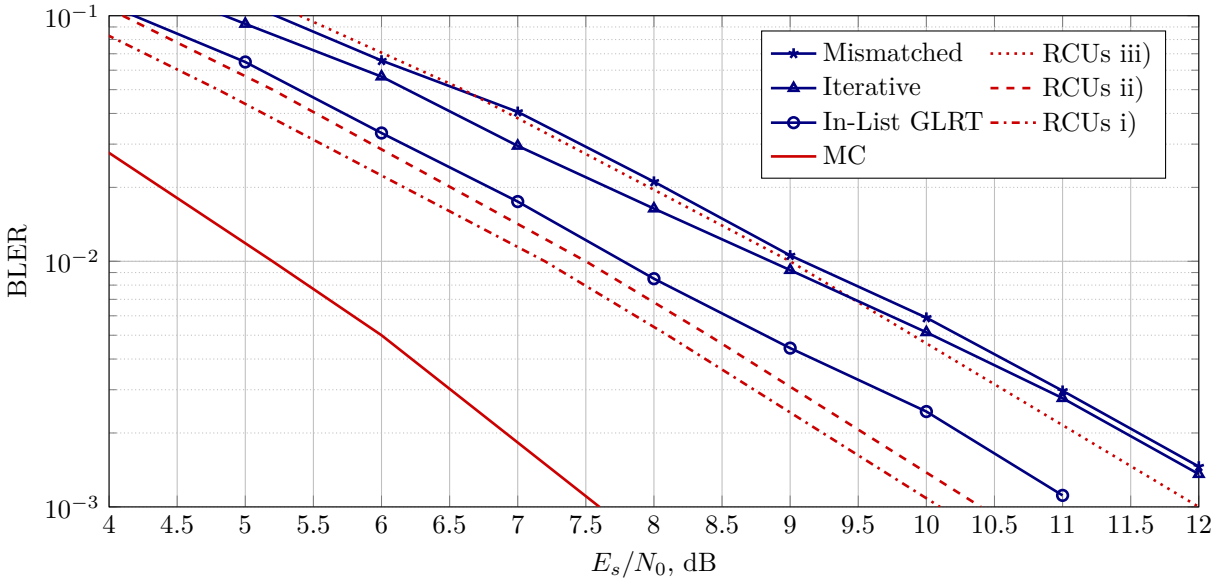

Figure 7.3.: BLER vs. SNR for the PAT schemes using MRB decoding with $N_p = 2$.

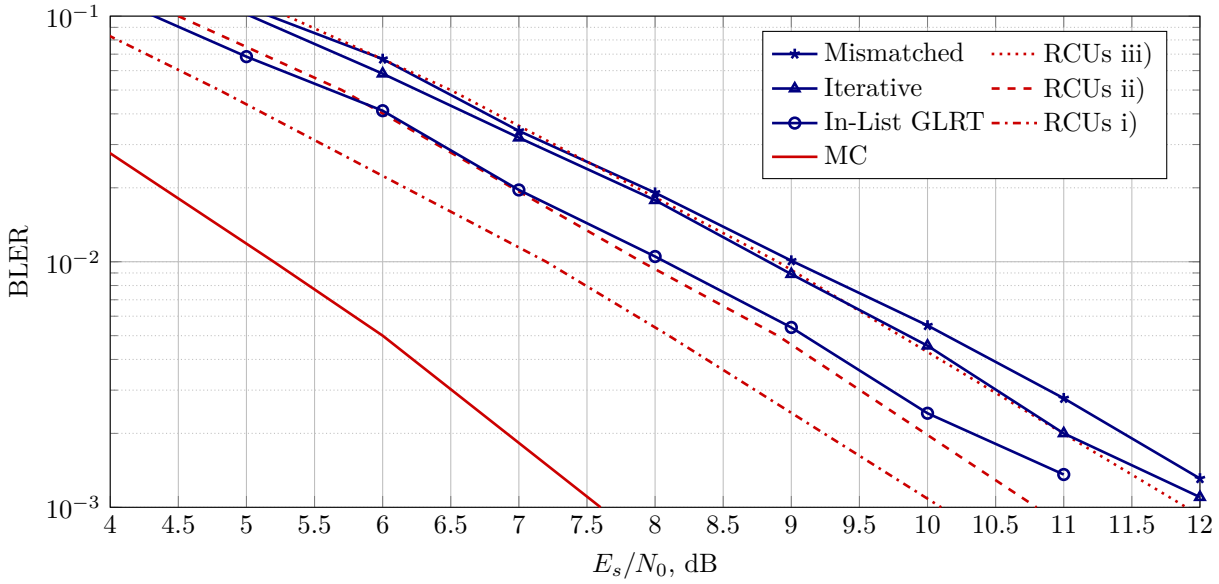

Figure 7.4.: BLER vs. SNR for the PAT schemes using MRB decoding with $N_p = 3$.

performance with $N_p = 2$, with a slight degradation visible when $N_p = 3$.

We next compare the performance of a polar code with the modified list decoders proposed in Section 7.4.2 to that of a quasi-cyclic code using MRB decoding as above. Each coherence block has 17 channel uses.[2] This results in 68 channel uses per message. For the simulations, we transmit $k = 32$ information bits within each codeword, yielding a rate $R = 32/68 \approx 0.47$ BPCU. The $(128, 32)$ quasi-cyclic code used in the simulations is obtained by a tail-biting termination of a rate$-1/4$ non-systematic convolutional code with a memory 14 and generators $[47633, 57505, 66535, 71145]$ [191, Table 10.14]. The minimum distance of the quasi-cyclic code is upper-bounded by the free distance of the underlying convolutional code, which is 37. In addition, we designed a $(128, 32)$ polar code using the Gaussian approximation of density evolution (see (3.59) and (3.60)) with a design SNR of $E_s/N_0 = 3$ dB. For the polar code, quasi-uniform puncturing (QUP) [172] is adopted while the quasi-cyclic code is punctured randomly. The MRB decoding order is set to 3 as before. With this choice, it builds a list $\mathcal{L}$ of 5489 candidate codewords. For the polar code, the list size of SCL decoding is set to 1024.

Figures 7.5-7.7 show that the gains of the proposed technique are no less than 1 dB as compared to mismatched PAT decoding at a BLER $\approx 10^{-3}$. Remarkably, the polar code under SCL decoding outperforms the quasi-cyclic code under MRB decoding despite the

[2]This choice is because the blocklengths for polar codes are suited for powers of 2 and we embed the firsts pilots to each block without puncturing.

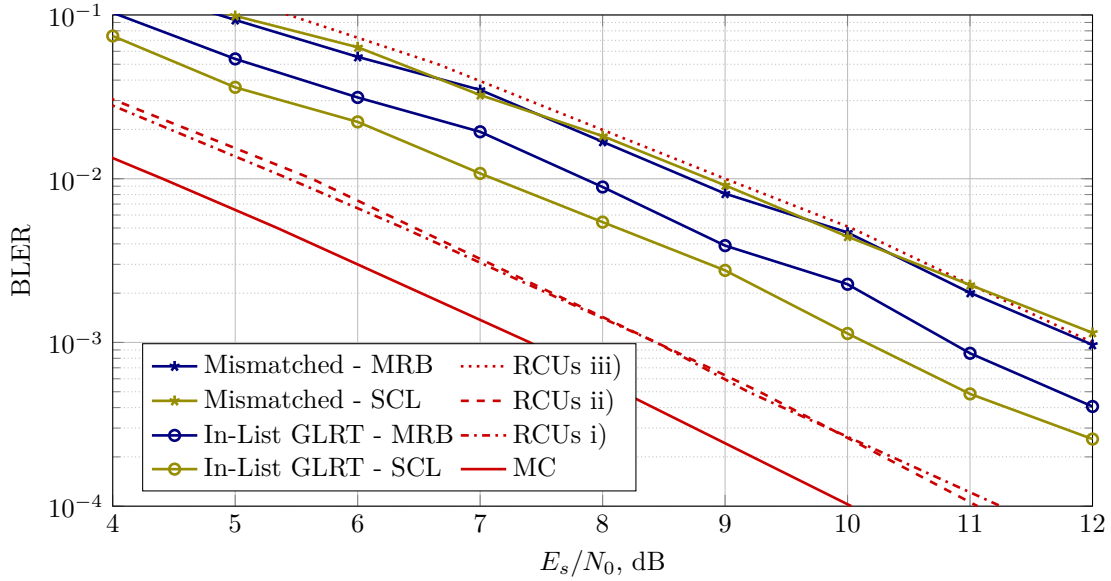

Figure 7.5.: BLER vs. SNR, where MRB decoding is used for the quasi-cyclic code and SCL decoding for the polar code in PAT with $N_p = 1$.

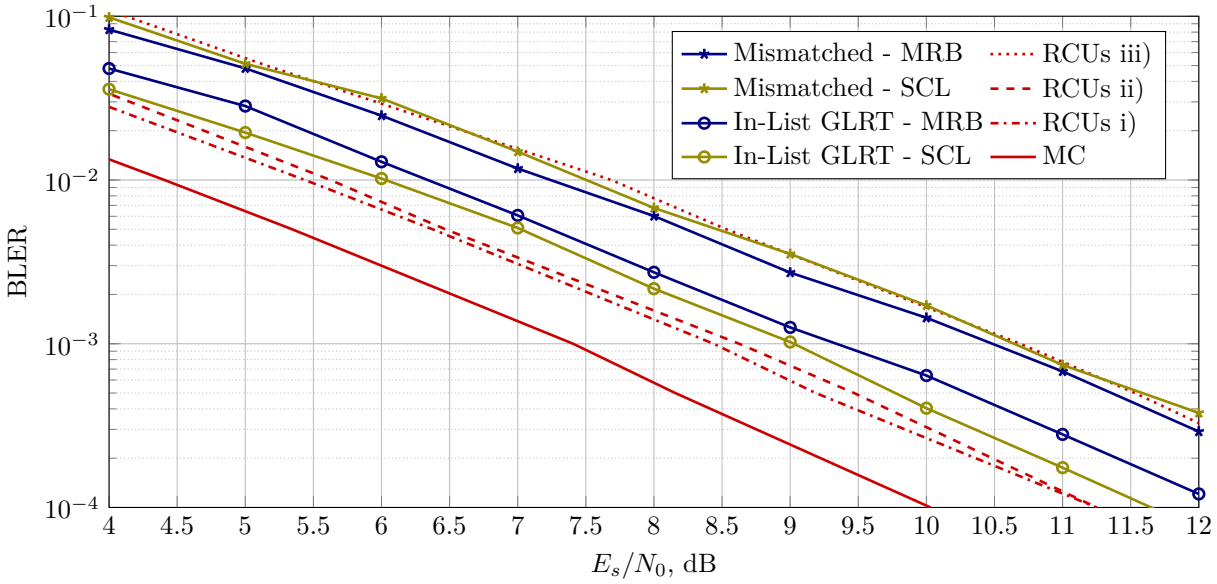

Figure 7.6.: BLER vs. SNR, where MRB decoding is used for the quasi-cyclic code and SCL decoding for the polar code in PAT with $N_p = 2$.

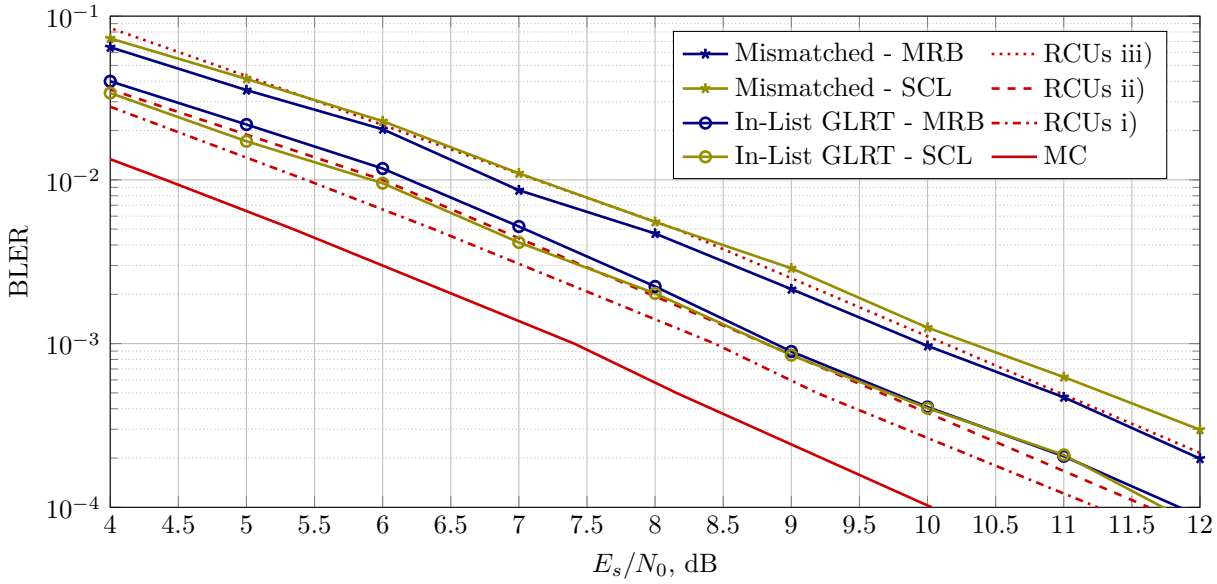

Figure 7.7.: BLER vs. SNR, where MRB decoding is used for the quasi-cyclic code and SCL decoding for the polar code in PAT with $N_p = 3$.

much smaller list size, attaining a performance close to the RCUs bound ii) for PAT under ML decoding.

7.4.3. Polar-Coded Non-Coherent Communication

The finite-length performance bounds show that there is a fundamental performance degradation due to using pilot symbols even when optimum decoding is employed [27, Figure 5]. Therefore, we next propose a pilot-free two-stage PCT scheme to jointly estimate the CSI and data with an adjustable complexity that can be made comparable to mismatched decoding in PAT. In the first stage, SCL decoding and the polar code constraints are used to estimate the CSI. In the second stage, mismatched SCL decoding proceeds with with this estimate. This scheme is of low-complexity especially when the number of diversity branches is small and is promising for future communication systems that require short frames and high efficiency in terms of rates [22].

A related method to estimate CSI uses the parity-check constraints of an LDPC code [192]. However, SCL decoding of polar codes naturally provides soft estimates of frozen bits. Moreover, polar codes are usually used with a high-rate outer code [11, 12] that can resolve CSI ambiguities, e.g., the phase ambiguity when using QPSK and Gray labeling [192]. Of course, one may consider outer codes for LDPC codes as well. Other low-

complexity methods for non-coherent channels are described in, e.g., [179, 192–195].

The Algorithm

This section presents a low-complexity joint channel estimation and decoding scheme for polar codes when there are no pilot symbols embedded in the frames, i.e., we have $N_p = 0$ and $\boldsymbol{x}_i = \boldsymbol{x}_i^{\mathrm{d}}$. A random interleaver $\boldsymbol{\Pi}$ permutes the encoded bits c_1^{2N} and is followed by the mapping (7.3). The channel model is (7.1).

Let $h_i = r_i e^{j\theta_i}$ where $r_i \in [0, \infty)$ and $\theta_i \in [0, 2\pi)$, $i \in [B]$. We begin by replacing the amplitudes $r_i = |h_i|$ via the estimates

$$\hat{r}_i = \sqrt{\frac{1}{N_c}\|\boldsymbol{y}_i\|^2 - \sigma^2}, \quad i = 1, \ldots, B. \tag{7.46}$$

Let β be a number of input bits, and recall the definitions $\mathcal{A}^{(\beta)} = \mathcal{A} \cap [\beta]$ and $\mathcal{F}^{(\beta)} = \mathcal{F} \cap [\beta]$. We use the polar code constraints to obtain an ML estimate of the phase as

$$\begin{aligned}
\left\{\hat{\theta}_1, \ldots, \hat{\theta}_B\right\} &= \underset{\{\theta_1,\ldots,\theta_B\}}{\operatorname{argmax}} \, p_{\underline{\boldsymbol{Y}}|U_{\mathcal{F}^{(\beta)}},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}\,|u_{\mathcal{F}^{(\beta)}}, \boldsymbol{h}\right) && (7.47)\\
&= \underset{\{\theta_1,\ldots,\theta_B\}}{\operatorname{argmax}} \sum_{u_{\mathcal{A}^{(\beta)}}} p_{\underline{\boldsymbol{Y}},U_{\mathcal{A}^{(\beta)}}|U_{\mathcal{F}^{(\beta)}},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}, u_{\mathcal{A}^{(\beta)}}\,|u_{\mathcal{F}^{(\beta)}}, \boldsymbol{h}\right) && (7.48)\\
&= \underset{\{\theta_1,\ldots,\theta_B\}}{\operatorname{argmax}} \sum_{u_{\mathcal{A}^{(\beta)}}} p^i_{\boldsymbol{G}^{[n+1]}}(\boldsymbol{y}, u_1^\beta; \boldsymbol{h}) && (7.49)
\end{aligned}$$

where $h_i = \hat{r}_i e^{j\theta_i}$, $i \in [B]$ and (7.49) follows from Lemma 3.5 and independent U_1^β and $\underline{\boldsymbol{H}}$, $\beta \in [2N]$. The quantities $p^i_{\boldsymbol{G}^{[n+1]}}(\boldsymbol{y}, u_1^\beta; \boldsymbol{h})$ denote the myopic probabilities obtained by SC decoding, see Lemma 3.5. Hence, the sum in (7.49) can be computed by SCL decoding up to decoding stage $|\mathcal{F}^{(\beta)}|$ with a list size $L_e = 2^{|\mathcal{A}^{(\beta)}|}$. To reduce complexity at the expense of accuracy, one can approximate the calculation with SCL decoding and L_e satisfying $1 \leq L_e < 2^{|\mathcal{A}^{(\beta)}|}$. In fact, simulations in Section 4.4.3 show that small list sizes such as $L_e = 8$ give BLER curves close to those of the coherent receiver.

Remark 7.3. Observe that if the estimator (7.49) is set to use all the frozen bits, i.e., $\beta = \gamma$, then the phase estimate is optimum in the ML sense based on the code structure.

Remark 7.4. The search space in (7.49) grows exponentially in the number of diversity branches B. There are several approaches to reduce complexity and we consider only the symmetry of the likelihood function due to the channel (7.1) and mapping (7.3) that halves the search space. We further adopt a coarse-fine search [192, 196] as an efficient optimizer.

The proofs of the following results are given in Appendix 7.5.

Lemma 7.1. Polar-coded modulations with the mapping (7.3) and the channel (7.1) have a sign ambiguity for the channel coefficients, i.e., for all $\boldsymbol{y}$, $\boldsymbol{h}$ and u_1^{2N-1}, we have

$$p_{\underline{\boldsymbol{Y}}|U_1^{2N},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}\left|(u_1^{2N-1},0),\boldsymbol{h}\right.\right) = p_{\underline{\boldsymbol{Y}}|U_1^{2N},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}\left|(u_1^{2N-1},1),-\boldsymbol{h}\right.\right). \tag{7.50}$$

Lemma 7.1 implies that if a polar code is considered for (7.1) then the decoder cannot resolve the ambiguity on bit u_N. This ambiguity occurs for any binary linear block code that has a generator matrix with an all-ones row, which is reflected in the bit u_N for polar codes.

Theorem 7.2. Polar-coded modulations with the mapping (7.3) and the channel (7.1) satisfy

$$p_{\underline{\boldsymbol{Y}}|U_1^{i},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}\left|u_1^{i},\boldsymbol{h}\right.\right) = p_{\underline{\boldsymbol{Y}}|U_1^{i},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}\left|u_1^{i},-\boldsymbol{h}\right.\right) \tag{7.51}$$

for all $\boldsymbol{y}$, $\boldsymbol{h}$ and u_1^i, $i \in [2N-1]$.

Corollary 7.3. Polar-coded modulations with the mapping (7.3) and the channel (7.1) satisfy

$$p_{\underline{\boldsymbol{Y}}|U_{\mathcal{F}^{(\beta)}},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}\left|u_{\mathcal{F}^{(\beta)}},\boldsymbol{h}\right.\right) = p_{\underline{\boldsymbol{Y}}|U_{\mathcal{F}^{(\beta)}},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}\left|u_{\mathcal{F}^{(\beta)}},-\boldsymbol{h}\right.\right) \tag{7.52}$$

for all $\boldsymbol{y}$ and $\boldsymbol{h}$.

Corollary 7.3 implies that the PCT estimator outputs two solutions for (7.49), namely $\{\hat{\theta}_1,\ldots,\hat{\theta}_B\}$ and $\{\hat{\theta}_1+\pi,\ldots,\hat{\theta}_B+\pi\}$ where addition is modulo 2π. An outer code can resolve this ambiguity by optimizing over the set $[0,2\pi)^{B-1}\times[0,\pi)$ to obtain $\{\hat{\theta}_1,\ldots,\hat{\theta}_B\}$ by using the inner code constraints. The demodulator then feeds the SCL decoder with the LLRs. Let $\mathcal{L}$ be the list of words $u_{\mathcal{A}}$ output by the decoder and define

$$\mathcal{L}' = \{(u_{\mathcal{A}^{(2N-1)}}, u_{2N}\oplus 1) : u_{\mathcal{A}} \in \mathcal{L}\}. \tag{7.53}$$

The outer code now eliminates invalid words in $\mathcal{L}\cup\mathcal{L}'$. Among the survivors, if any, the estimate $\hat{u}_1^{2N}$ is chosen to maximize $p_{\underline{\boldsymbol{Y}}|U_1^{2N},\underline{\boldsymbol{H}}}(\boldsymbol{y}|u_1^{2N},\hat{\boldsymbol{h}})$ if $u_{\mathcal{A}}\in\mathcal{L}$ or $p_{\underline{\boldsymbol{Y}}|U_1^{2N},\underline{\boldsymbol{H}}}(\boldsymbol{y}|u_1^{2N},-\hat{\boldsymbol{h}})$ if $u_{\mathcal{A}}\in\mathcal{L}'$.

Remark 7.5. An outer code with a minimum distance of at least two can resolve the phase ambiguity.

Numerical Results: PCT

This section provides Monte Carlo simulation results to compare the performance of PAT and PCT. The inner code is a $(128, 38)$ polar code and the outer code is a 6-bit CRC code with generator polynomial $x^6 + x^5 + 1$, resulting in a $(128, 32)$ code. For the QPSK modulator (7.3) we have $N = BN_c = 64$ channel uses and an overall rate of $R = 0.5$ BPCU. For PAT, the $(128, 32)$ code is punctured to obtain BN_p pilot bits in total, resulting in a $(2N - 2BN_p, 32)$ code. All curves shown in the figures below are for SCL decoding with a list size of $L = 8$ after estimating the CSI. The optimization (7.49) uses a coarse-fine search with 8 levels in both the coarse and fine search parts [196]. The performance is compared for various estimator parameters β and L_e and to the coherent receiver with perfect CSI. No puncturing is required for the coherent receiver. As discussed below, the gains of our scheme are similar for $B \in \{1, 2\}$ and with or without fading.

Consider the channel (7.1) with $B = 1$, $r_1 = 1$, and uniformly distributed phase $\Theta_1 \sim \mathcal{U}[0, 2\pi)$. Figure 7.8 compares PAT and PCT. The best PAT performance for the BLERs of interest was achieved with $N_p = 14$, i.e., 14 pilot symbols gave the lowest SNR for BLERs ranging from 10^{-2} to 10^{-4} in Figure 7.8. For smaller N_p the quality of the channel estimate limits performance, and for larger n_p the puncturing weakens the polar code and limits performance. PCT performs within 0.3 dB of the receiver with perfect CSI if the estimator is run with $L_e = 8$ and up to the last frozen bit with $\beta = 113$. It thereby outperforms PAT by about 1.5 dB at a BLER of 10^{-4}. Observe that if the estimator is run up to the last frozen bit before the first information bit, i.e., $\beta = 47$, then the performance is worse than for PAT. The parameters $\beta = 113$ and $L_e = 1$ provide a good trade-off between complexity and performance when combined with a second-stage SCL decoding with a list size $L = 8$.

Table 7.1 compares the number of visited nodes per frame in the polar decoding tree along with the BLER at $E_s/N_0 = 1$ dB. Each visited node corresponds to an input bit (including the frozen bits) visited by the algorithm [197, Remark 4]. For PCT, we state the sum of the number of nodes visited by the estimator and the number of nodes visited by the decoder. The number of visited nodes with PAT and perfect CSI is thus the same. Observe that PCT with $\beta = 113$ and $L_e = 1$ visits a similar number of nodes as PAT with a list size $L = 32$ (the difference is less than 10%) and it reduces the error probability by one order of magnitude. We remark that measuring the complexity by the number of visited nodes is pessimistic for PCT since most of the visited nodes are frozen bits. Hence, simplified SC decoders [198, 199] can significantly reduce complexity.

We next consider $B = 2$ coherence blocks. Figure 7.9 shows the BLER for $r_i = 1$

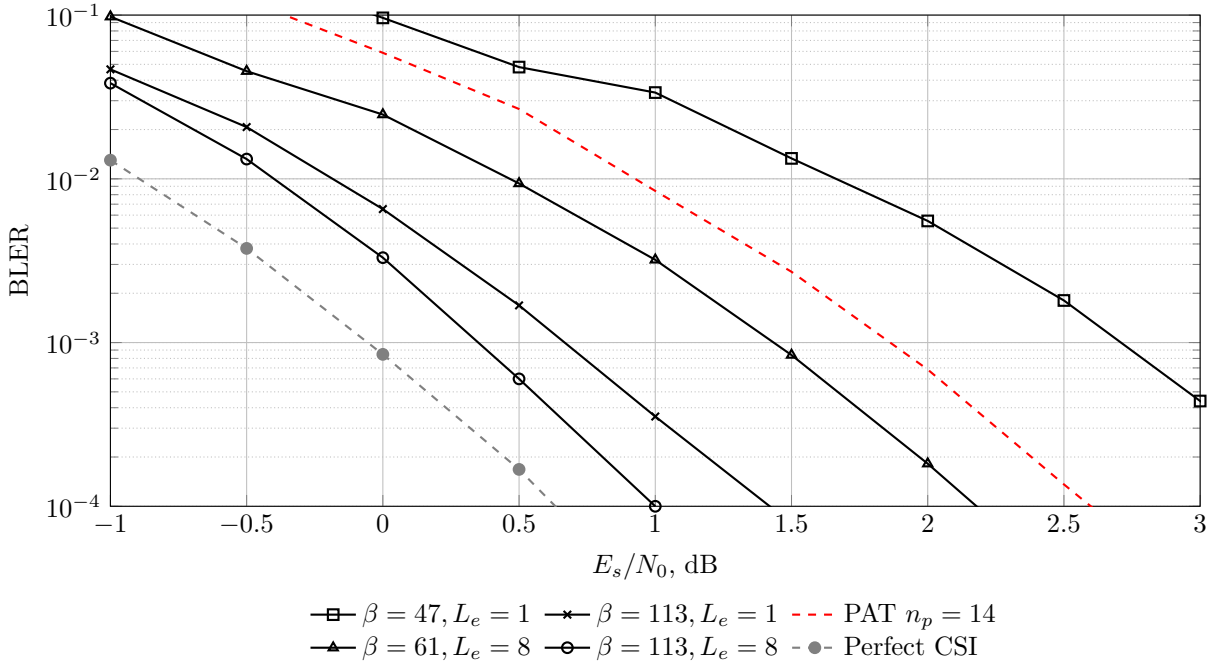

Figure 7.8.: Performance of PAT and PCT for the channel (7.1) with $B = 1$, $r_1 = 1$, and $\Theta_1 \sim [0, 2\pi)$. A $(128, 32)$ polar code was used with QPSK so that $n = n_c = 64$ and the overall rate is $R = 0.5$ BPCU. SCL decoding uses a list size of $L = 8$ for all cases.

Table 7.1.: Number of Visited Nodes per Frame at $E_s/N_0 = 1$ dB

Method	FER	Visited Nodes
PAT ($n_p = 14$, $L = 8$)	8.43×10^{-3}	631
PAT ($n_p = 14$, $L = 32$)	3.16×10^{-3}	2223
PCT ($\beta = 47$, $L_e = 1$, $L = 8$)	3.36×10^{-2}	1383
PCT ($\beta = 61$, $L_e = 8$, $L = 8$)	3.20×10^{-3}	2151
PCT ($\beta = 113$, $L_e = 1$, $L = 8$)	3.50×10^{-4}	2439
PCT ($\beta = 113$, $L_e = 8$, $L = 8$)	1.00×10^{-4}	8807
Perfect CSI ($L = 8$)	2.40×10^{-5}	631

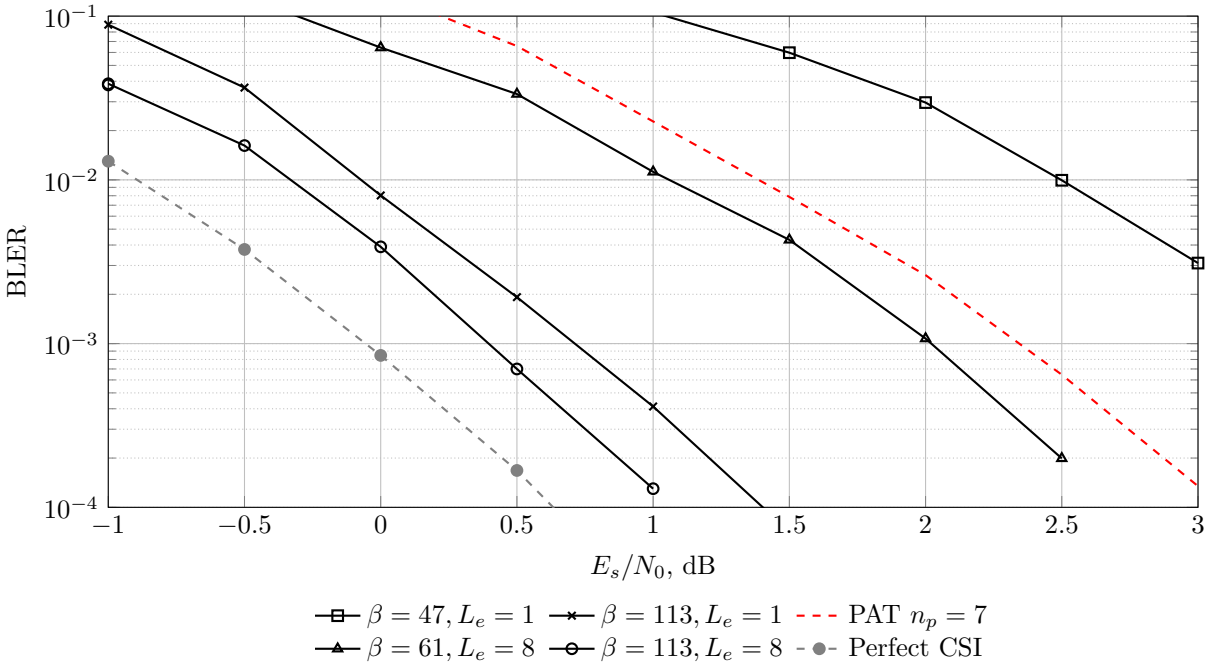

Figure 7.9.: Performance of PAT and PCT for the channel (7.1) with $B = 2$, $r_i = 1$, and $\Theta_i \sim [0, 2\pi)$ for $i \in \{1, 2\}$. A $(128, 32)$ polar code was used with QPSK so that $n = 2n_c = 64$ and the overall rate is $R = 0.5$ BPCU. SCL decoding uses a list size of $L = 8$ for all cases.

and $\Theta_i \sim \mathcal{U}[0, 2\pi)$, $i \in \{1, 2\}$. Figure 7.10 shows the BLER for a Rayleigh block-fading channel with $H_i \sim \mathcal{CN}(0, 1)$, $i \in \{1, 2\}$. The best performance for PAT was achieved with $n_p = 7$ pilot symbols per coherence block for both cases. Observe that, in both cases, PCT outperforms PAT by about 2 dB at a BLER $\approx 10^{-4}$. Moreover, PCT approaches the performance of a coherent receiver with perfect CSI.

7.5. Appendices

7.5.1. Proof of Lemma 7.1

For all $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{h}$ and $\boldsymbol{s} \in \{-1, +1\}^B$, we have

$$p_{\underline{\boldsymbol{Y}}|\underline{\boldsymbol{X}},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{h}\right) = \prod_{i=1}^{B} p_{\underline{\boldsymbol{Y}}_i|\underline{\boldsymbol{X}}_i,H_i}\left(\boldsymbol{y}_i|s_i\boldsymbol{x}_i, s_i h_i\right) \tag{7.54}$$

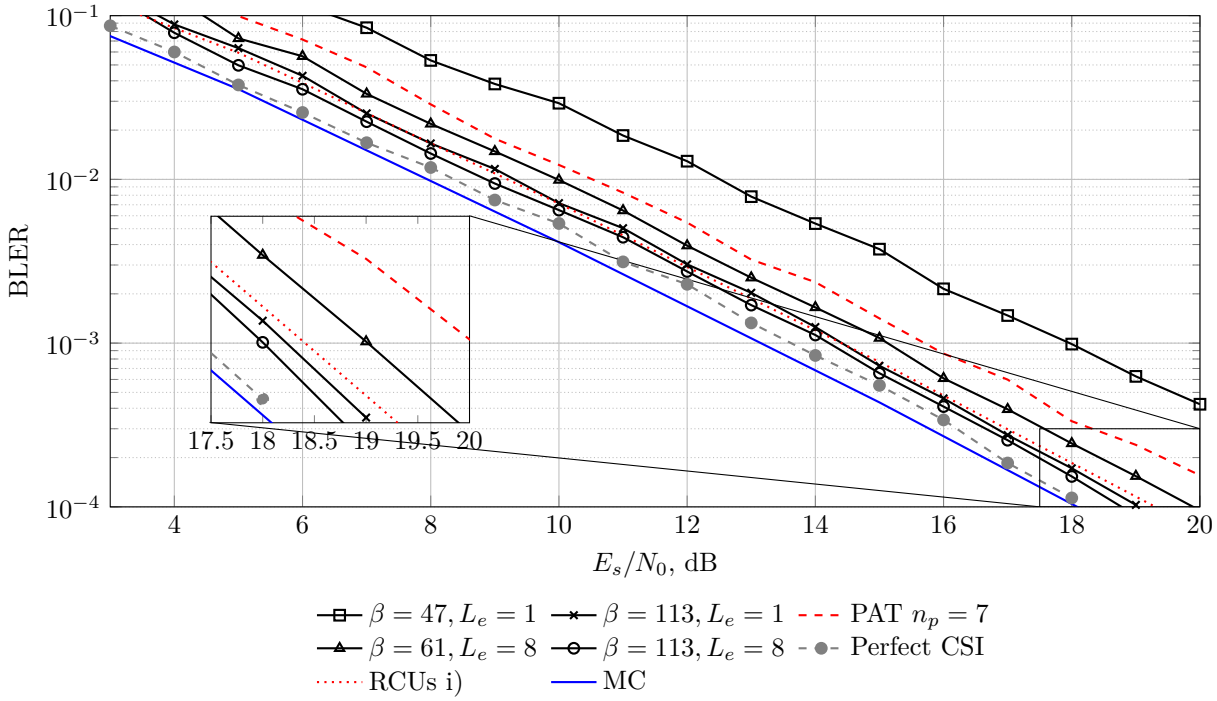

Figure 7.10.: Performance of PAT and PCT for a Rayleigh block-fading channel and $B = 2$. A $(128, 32)$ polar code was used with QPSK and the overall rate is $R = 0.5$ BPCU. SCL decoding uses a list size of $L = 8$ for all cases.

as $s_i^2 = 1$. Recall that $c_1^{2N} = \Pi^{-1}(\chi^{-1}(\boldsymbol{x}))$ so that $c_1^{2N} \oplus 1 = \Pi^{-1}(\chi^{-1}(-\boldsymbol{x}))$. By choosing $\boldsymbol{s} = -1$, we have

$$p_{\underline{\boldsymbol{Y}}|\underline{\boldsymbol{C}},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}|c_1^{2N}, \boldsymbol{h}\right) = p_{\underline{\boldsymbol{Y}}|\underline{\boldsymbol{C}},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}|c_1^{2N} \oplus 1, -\boldsymbol{h}\right). \tag{7.55}$$

Let u_1^{2N} be the vector such that $c_1^{2N} = u_1^{2N}\boldsymbol{G}^{[n+1]}$. We have $c_1^{2N} \oplus 1 = (u_1^{2N-1}, u_{2N} \oplus 1)\boldsymbol{G}^{[n+1]}$ because the last row of $\boldsymbol{G}^{[n+1]}$ is 1. ■

7.5.2. Proof of Theorem 7.2

For $i \in [2N-1]$, we have

$$\begin{aligned}
p_{\underline{\boldsymbol{Y}}|U_1^i,\underline{\boldsymbol{H}}}\left(\boldsymbol{y}|u_1^i, \boldsymbol{h}\right) &= \sum_{u_{i+1}^{2N}} P\left(u_{i+1}^{2N}\right) p_{\underline{\boldsymbol{Y}}|U_1^{2N},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}|u_1^{2N}, \boldsymbol{h}\right) && (7.56)\\
&= \sum_{u_{i+1}^{N-1}} P\left(u_{i+1}^{N-1}\right) \left[\sum_{u_N} \frac{1}{2} p_{\underline{\boldsymbol{Y}}|U_1^{2N},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}|u_1^{2N}, \boldsymbol{h}\right)\right] && (7.57)\\
&= \sum_{u_{i+1}^{N-1}} P\left(u_{i+1}^{N-1}\right) \left[\sum_{u_N} \frac{1}{2} p_{\underline{\boldsymbol{Y}}|U_1^{2N},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}|u_1^{2N}, -\boldsymbol{h}\right)\right] && (7.58)\\
&= \sum_{u_{i+1}^{2N}} P\left(u_{i+1}^{2N}\right) p_{\underline{\boldsymbol{Y}}|U_1^{2N},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}|u_1^{2N}, -\boldsymbol{h}\right) && (7.59)
\end{aligned}$$

where step (7.56) follows by the law of total probability and the mutual independence of U_1^i, U_{i+1}^{2N} and $\underline{\boldsymbol{H}}$; steps (7.57) and (7.59) follow by rearranging the sums and noting that U_N is uniform; step (7.58) follows by Lemma 7.1.

7.5.3. Proof of Corollary 7.3

We expand

$$\begin{aligned}
p_{\underline{\boldsymbol{Y}}|U_{\mathcal{F}^{(\beta)}},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}\,|\boldsymbol{0}, \boldsymbol{h}\right) &\overset{(a)}{=} \sum_{u_{\mathcal{A}^{(\beta)}}} P\left(u_{\mathcal{A}^{(\beta)}}\right) p_{\underline{\boldsymbol{Y}}|U_1^{\beta},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}|u_1^{\beta}, \boldsymbol{h}\right) && (7.60)\\
&\overset{(b)}{=} \sum_{u_{\mathcal{A}^{(\beta)}}} P\left(u_{\mathcal{A}^{(\beta)}}\right) p_{\underline{\boldsymbol{Y}}|U_1^{\beta},\underline{\boldsymbol{H}}}\left(\boldsymbol{y}|u_1^{\beta}, -\boldsymbol{h}\right) && (7.61)
\end{aligned}$$

where step (7.60) follows by the law of total probability and mutually independent $U_{\mathcal{A}^{(\beta)}}$, $U_{\mathcal{F}^{(\beta)}}$ and $\underline{\boldsymbol{H}}$; step (7.61) follows by Theorem 7.2.

8

Conclusions and Outlook

The thesis has investigated SCL decoding for the BEC, the BAWGNC and block-fading channels in the context of short-packet communications. In the following, we summarize our contributions and comment on future research directions.

Chapter 4 analyzed SCL decoding over the BEC and described an efficient modification called successive SCI decoding. For a given code, density evolution characterizes the average number of inactivations for ML decoding. Furthermore, we provided a simple and accurate approximation for the average number of unknown bits (or the logarithm of number of valid decoding paths) at each decoding stage. Numerical results over the BEC showed the potential of dynamic Reed–Muller (dRM) codes, which perform close to the Singleton bound for blocklengths $N \geq 512$. Even for short blocklengths such as $N = 128$, these codes are competitive with respect to extended Bose-Chaudhuri-Hocquengham (eBCH) codes, and with less decoding complexity under SCI decoding. We showed that the number of unknown bits normalized by the blocklength concentrates around its mean, which is supported by the numerical results. A promising direction is to extend SCI decoding for codes over q-ary erasure channels. Note that, under SCL decoding, an erasure in decoding an information bit yields a multiplication by a factor q of the number of active paths.

Chapter 5 extended the analysis of SCL decoding to general BMSCs for code design. The proposed designs outperform the polar codes adopted by the 3GPP 5G cellular standard and other state-of-art designs for lengths up to $N = 512$ bits. A future work should provide a constructive design recipe based on the combination of the proposed analysis and of the distance properties of polar codes with dynamic frozen bits, especially for

moderate blocklengths $256 \leq N \leq 1024$. Furthermore, an open (theoretical) problem is to provide tight upper bounds on the required list size for near-ML decoding over general BMSCs. To this end, a relevant question is if Theorem 4.2, which is valid for the BEC, can be extended to general BMSCs, improving an existing result [88] for high-rate codes. Numerical results support this conjecture.

Chapter 6 studied classes of product codes through the polar coding lens, where it was shown that a structured choice of frozen bits in (multi-kernel) polar code constructions yields a family of product codes. This enables applying the SCL and SCI decoding algorithms for such product codes, resulting in low-complexity close-to-ML decoding. A research problem is how to embed the random dynamic constraints for the frozen bits entailed by the product code's structure. In fact, suboptimal choices of frozen bits for SC decoding (as for the case of product codes) may significantly improve the finite-length performance under SCL decoding when such constraints are embedded. In addition, one may investigate permutation-based SCL and SCI decoding of SPC product codes to further reduce the complexity as for RM codes.

Chapter 7 studied the potential of SCL decoding in the case of block-fading channels for both pilot-assisted and pilot-free schemes. The proposed PAT scheme showed how (modified) list decoders in general, and SCL decoding in particular, can reduce the pilot overhead, resulting in gains of more than 1 dB over traditional PAT schemes. For a small number of coherence blocks, a pilot-free non-coherent transmission scheme is proposed where the channel state is estimated using the knowledge of frozen bits via SCL decoding. The imperfect CSI is then used to decode the message. The scheme performs almost as well as a receiver with the perfect knowledge of the channel state and provides gains of up to 2 dB over traditional PAT schemes. A first direction to extend the work would be to combine both methods by embedding a single pilot per coherence block and using the knowledge of the frozen bits to refine the channel estimate during decoding. This approach has the potential to be used in polar-coded schemes over block-fading channels with a very large number of coherence blocks. In addition, both schemes could be extended to operate over multiple antenna systems, as well as in combination with high-order modulation. For the pilot-free polar-coded scheme, a tailored code design should lead to interesting optimization problems since freezing reliable bit positions could improve the channel estimation, and this may be reflected in an overall performance gain.

A

Acronyms

a.s.	almost surely
APP	a-posteriori probability
AWGN	additive white Gaussian noise
AWGNC	additive white Gaussian noise channel
BAWGNC	binary-input additive white Gaussian noise channel
BEC	binary erasure channel
BLEP	block error probability
BLER	block error rate
BMSC	binary-input memoryless symmetric channel
BP	belief propagation
BPCU	bits per channel use
BRC	Berlekamp's random coding
BSC	binary symmetric channel
CDF	cumulative distribution function

CRC	cyclic redundancy check
CSI	channel state information
CWEF	complete weight enumerating function
dRM	dynamic Reed–Muller
eBCH	extended Bose–Chaudhuri–Hocquengham
EM	expectation-maximization
GLRT	generalized likelihood-ratio test
i.i.d.	independent and identically distributed
IOWE	input-output weight enumerator
IOWEF	input-output weight enumerating function
JWE	joint weight enumerator
LDPC	low-density parity-check
LHS	left-hand side
LL	log-likelihood
LLR	log-likelihood ratio
MAP	maximum a-posteriori
MC	metaconverse
MDS	maximum distance separable
ML	maximum likelihood
MRB	most reliable basis
OFDM	orthogonal frequency-division mutiplexing
QPSK	quadrature phase-shift keying
QUP	quasi-uniform puncturing

PAC	polarization-adjusted convolutional
PAT	pilot-assisted transmission
PCT	polar-coded transmission
PDF	probability density function
PM	path metric
PMF	probability mass function
PW	polarization weight
RCU	random coding union
RCUs	random coding union bound with s parameter
RHS	right-hand side
RM	Reed–Muller
RV	random variable
SC	successive cancellation
SCI	successive cancellation inactivation
SCL	successive cancellation list
SCOS	successive cancellation ordered search
SIR	symmetric information rate
SISO	soft-input soft-output
SNR	signal-to-noise ratio
SPC	single parity-check
TSB	tangential-sphere bound
TUB	truncated union bound
WE	weight enumerator
WEF	weight enumerating function

Bibliography

[1] C. E. Shannon, "A mathematical theory of communication," *Bell Sys. Techn. J.*, vol. 27, no. 3, pp. 379–423, July 1948.

[2] P. Elias, "Error-free coding," *IRE Trans. Inf. Theory*, vol. PGIT-4, pp. 29–37, Sep. 1954.

[3] ——, "Coding for noisy channels," *IRE Int. Conv. Record*, pp. 37–46, Mar. 1955.

[4] R. L. Dobrushin, "Asymptotic optimality of group and systematic codes for some channels," *Theory of Prob. & Its Applic.*, vol. 8, no. 1, pp. 47–60, 1963. [Online]. Available: https://doi.org/10.1137/1108003

[5] R. G. Gallager, *Information Theory and Reliable Communication.* John Wiley & Sons, Inc., 1968.

[6] E. Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, Jul. 2009.

[7] R. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[8] D. J. C. MacKay and R. M. Neal, "Good codes based on very sparse matrices," in *Cryptography and Coding.* Springer Berlin Heidelberg, 1995, pp. 100–111.

[9] M. Sipser and D. A. Spielman, "Expander codes," *IEEE Trans. Inf. Theory*, vol. 42, no. 6, pp. 1710–1722, 1996.

[10] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes. 1," in *IEEE Int. Conf. Commun. (ICC)*, vol. 2, May 1993, pp. 1064–1070.

[11] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.

[12] *Technical Specification Group Radio Access Network - NR - Multiplexing and channel coding*, 3GPP Technical specification TS 38.212 V16.5.0, Mar. 2021.

[13] B. Li, H. Shen, and D. Tse, "A RM-polar codes," *CoRR*, vol. abs/1407.5483, 2014. [Online]. Available: http://arxiv.org/abs/1407.5483

[14] M. Mondelli, S. H. Hassani, and R. L. Urbanke, "From polar to Reed-Muller codes: A technique to improve the finite-length performance," *IEEE Trans. Commun.*, vol. 62, no. 9, pp. 3084–3091, Sep. 2014.

[15] P. Trifonov and V. Miloslavskaya, "Polar subcodes," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 2, pp. 254–266, Feb. 2016.

[16] T. Wang, D. Qu, and T. Jiang, "Parity-check-concatenated polar codes," *IEEE Commun. Lett.*, vol. 20, no. 12, pp. 2342–2345, 2016.

[17] M. Qin, J. Guo, A. Bhatia, A. Guillén i Fàbregas, and P. H. Siegel, "Polar code constructions based on LLR evolution," *IEEE Commun. Lett.*, vol. 21, no. 6, pp. 1221–1224, Jun. 2017.

[18] P. Yuan, T. Prinz, G. Böcherer, O. İşcan, R. Böhnke, and W. Xu, "Polar code construction for list decoding," in *Int. ITG Conf. Sys. Commun. Coding*, Feb. 2019, pp. 125–130.

[19] M. Rowshan and E. Viterbo, "How to modify polar codes for list decoding," in *IEEE Int. Symp. Inf. Theory*, Jul. 2019, pp. 1772–1776.

[20] V. Miloslavskaya and B. Vucetic, "Design of short polar codes for SCL decoding," *IEEE Trans. Commun.*, vol. 68, no. 11, pp. 6657–6668, 2020.

[21] E. Biglieri, J. Proakis, and S. Shamai, "Fading channels: information-theoretic and communications aspects," *IEEE Trans. Inf. Theory*, vol. 44, no. 6, pp. 2619–2692, 1998.

[22] G. Durisi, T. Koch, and P. Popovski, "Towards massive, ultra-reliable, and low-latency wireless communications with short packets," *Proc. IEEE*, vol. 104, no. 9, pp. 1711–1726, Sep. 2016.

[23] E. Biglieri, *Coding for Wireless Channels*. Springer, 2005.

[24] L. Tong, B. M. Sadler, and M. Dong, "Pilot-assisted wireless transmissions: General model, design criteria, and signal processing," *IEEE Signal Process. Mag.*, vol. 21, no. 6, pp. 12–25, Nov. 2004.

[25] B. Hassibi and B. M. Hochwald, "How much training is needed in multiple-antenna wireless links?" *IEEE Trans. Inf. Theory*, vol. 49, no. 4, pp. 951–963, 2003.

[26] G. Liva, G. Durisi, M. Chiani, S. S. Ullah, and S. C. Liew, "Short codes with mismatched channel state information: A case study," in *IEEE Int. Workshop Signal Process. Adv. in Wireless Commun.*, Sapporo, Japan, Jul. 2017, pp. 1–5.

[27] J. Östman, G. Durisi, E. G. Ström, M. C. Coşkun, and G. Liva, "Short packets over block-memoryless fading channels: Pilot-assisted or noncoherent transmission?" *IEEE Trans. Commun.*, vol. 67, no. 2, pp. 1521–1536, Feb. 2019.

[28] M. C. Coşkun, J. Neu, and H. D. Pfister, "Successive cancellation inactivation decoding for modified Reed-Muller and eBCH codes," in *IEEE Int. Symp. Inf. Theory*, 2020, pp. 437–442.

[29] M. C. Coşkun and H. D. Pfister, "Bounds on the list size of successive cancellation list decoding," in *Int. Conf. Signal Process. Commun.*, 2020, pp. 1–5.

[30] M. C. Coşkun and H. D. Pfister, "An information-theoretic perspective on successive cancellation list decoding and polar code design," *IEEE Trans. Inf. Theory, to appear*, 2022. [Online]. Available: http://arxiv.org/abs/2103.16680

[31] M. C. Coşkun, T. Jerkovits, and G. Liva, "Successive cancellation list decoding of product codes with Reed-Muller component codes," *IEEE Commun. Lett.*, vol. 23, no. 11, pp. 1972–1976, 2019.

[32] M. C. Coşkun, G. Liva, A. Graell i Amat, M. Lentmaier, and H. D. Pfister, "Successive cancellation decoding of single parity-check product codes: Analysis and improved decoding," *IEEE Trans. Inf. Theory, submitted*, 2020. [Online]. Available: http://arxiv.org/abs/2008.06938

[33] M. C. Coşkun, G. Liva, J. Östman, and G. Durisi, "Low-complexity joint channel estimation and list decoding of short codes," in *Proc. ITG Int. Conf. Syst., Commun. and Coding*, Rostock, Germany, Feb. 2019, pp. 269–273.

[34] M. Xhemrishi, M. C. Coşkun, G. Liva, J. Östman, and G. Durisi, "List decoding of short codes for communication over unknown fading channels," in *Asilomar Conf. Signals, Systems, Computers*, 2019, pp. 810–814.

[35] P. Yuan, M. C. Coşkun, and G. Kramer, "Polar-coded non-coherent communication," *IEEE Commun. Lett.*, vol. 25, no. 6, pp. 1786–1790, 2021.

[36] M. C. Coşkun, G. Durisi, T. Jerkovits, G. Liva, W. Ryan, B. Stein, and F. Steiner, "Efficient error-correcting codes in the short blocklength regime," *Elsevier Phys. Commun.*, vol. 34, pp. 66–79, Jun. 2019.

[37] J. Neu, M. C. Coşkun, and G. Liva, "Ternary quantized polar code decoders: Analysis and design," in *Asilomar Conf. Signals, Systems, Computers*, 2019, pp. 1724–1728.

[38] P. Yuan and M. C. Coşkun, "Complexity-adaptive maximum-likelihood decoding of modified $\boldsymbol{G}_n$-coset codes," *Inf. Theory Workshop (ITW), accepted*, 2021. [Online]. Available: http://arxiv.org/abs/2105.04048

[39] ——, "Successive cancellation ordered search decoding of modified $\boldsymbol{G}_N$-coset codes," *IEEE Trans. Commun., in prep.*, 2021.

[40] G. Forney, "Exponential error bounds for erasure, list, and decision feedback schemes," *IEEE Trans. Inf. Theory*, vol. 14, no. 2, pp. 206–220, 1968.

[41] D. J. Rose, "Matrix identities of the fast fourier transform," *Linear Algebra and its Applications*, vol. 29, pp. 423–443, 1980. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0024379580902530

[42] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. USA: Cambridge University Press, 2005.

[43] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Analysis of low density codes and improved designs using irregular graphs," in *Proc. ACM Symp. on Theory of Computing*, ser. STOC '98. New York, NY, USA: Association for Computing Machinery, 1998, p. 249–258. [Online]. Available: https://doi.org/10.1145/276698.276756

[44] M. Luby, M. Mitzenmacher, M. Shokrollahi, and D. Spielman, "Efficient erasure correcting codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 569–584, 2001.

[45] I. T. I. Theory, "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 585–598, 2001.

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

[47] M. Raginsky and I. Sason, "Concentration of measure inequalities in information theory, communications, and coding," *Foundations and Trends® in Communications and Information Theory*, vol. 10, no. 1-2, pp. 1–246, 2013. [Online]. Available: http://dx.doi.org/10.1561/0100000064

[48] S. B. Korada, E. Şaşoğlu, and R. Urbanke, "Polar codes: Characterization of exponent, bounds, and constructions," *IEEE Trans. Inf. Theory*, vol. 56, no. 12, pp. 6253–6264, Dec. 2010.

[49] M. Benammar, V. Bioglio, F. Gabry, and I. Land, "Multi-kernel polar codes: Proof of polarization and error exponents," in *IEEE Inf. Theory Workshop (ITW)*, Nov 2017, pp. 101–105.

[50] T. Cover and J. Thomas, *Elements of Information Theory*. John Wiley and Sons, Inc., 2006.

[51] Y. Lomnitz and M. Feder, "A simpler derivation of the coding theorem," 2012.

[52] R. Gallager, "A simple derivation of the coding theorem and some applications," *IEEE Trans. Inf. Theory*, vol. 11, no. 1, pp. 3–18, Jan. 1965.

[53] A. Feinstein, "A new basic theorem of information theory," *Trans. IRE Professional Group on Inf. Theory*, vol. 4, no. 4, pp. 2–22, 1954.

[54] T. Richardson and R. Urbanke, *Modern Coding Theory*. New York, NY, USA: Cambridge University Press, 2008.

[55] G. Liva, "Channel codes for iterative decoding," Manuskript zur Vorlesung, Lehrstuhl für Nachrichtentechnik, Technische Universität München, 2021.

[56] F. J. Macwilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, 1st ed. Amsterdam:North-Holland, 1978, vol. 16.

[57] R. C. Singleton, "Maximum distance q-nary codes," *IEEE Trans. Inf. Theory*, vol. 10, no. 2, pp. 116–118, 1964.

[58] I. Sason and S. Shamai, "Performance analysis of linear codes under maximum-likelihood decoding: A tutorial," *Commun. Inf. Theory*, vol. 3, no. 1/2, p. 1–222, Jul. 2006. [Online]. Available: https://doi.org/10.1561/0100000009

[59] B. Dorsch, "A decoding algorithm for binary block codes and J-ary output channels (corresp.)," *IEEE Trans. Inf. Theory*, vol. 20, no. 3, pp. 391–394, 1974.

[60] M. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. Inf. Theory*, vol. 41, no. 5, pp. 1379–1396, 1995.

[61] S. Lin and D. Costello, Jr., *Error control coding.* Englewood Cliffs, NJ, USA: Prentice Hall, 2004, second edition.

[62] A. Valembois and M. Fossorier, "Box and match techniques applied to soft-decision decoding," *IEEE Trans. Inf. Theory*, vol. 50, no. 5, pp. 796–810, Dec. 2004.

[63] Y. Wu and C. Hadjicostis, "Soft-decision decoding using ordered recodings on the most reliable basis," *IEEE Trans. Inf. Theory*, vol. 53, no. 2, pp. 829–836, Feb. 2007.

[64] M. P. C. Fossorier, "Iterative reliability-based decoding of low-density parity check codes," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 5, pp. 908–917, May 2001.

[65] M. Baldi, F. Chiaraluce, N. Maturo, G. Liva, and E. Paolini, "A hybrid decoding scheme for short non-binary LDPC codes," *IEEE Commun. Lett.*, vol. 18, no. 12, pp. 2093–2096, Dec 2014.

[66] C. Yue, M. Shirvanimoghaddam, B. Vucetic, and Y. Li, "A revisit to ordered statistics decoding: Distance distribution and decoding rules," *IEEE Trans. Inf. Theory*, vol. 67, no. 7, pp. 4288–4337, 2021.

[67] H. Nickl, J. Hagenauer, and F. Burkert, "Approaching Shannon's capacity limit by 0.27 db using hamming codes in a turbo-decoding scheme," in *IEEE Int. Symp. Inf. Theory*, 1997, p. 12.

[68] R. M. Pyndiah, "Near-optimum decoding of product codes: block turbo codes," *IEEE Trans. Commun.*, vol. 46, no. 8, pp. 1003–1010, Aug. 1998.

[69] R. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vol. 27, no. 5, pp. 533–547, Sep. 1981.

[70] J. Li, K. R. Narayanan, and C. N. Georghiades, "Product accumulate codes: a class of codes with near-capacity performance and low decoding complexity," *IEEE Trans. Inf. Theory*, vol. 50, no. 1, pp. 31–46, Jan. 2004.

[71] A. J. Feltström, D. Truhachev, M. Lentmaier, and K. S. Zigangirov, "Braided block codes," *IEEE Trans. Inf. Theory*, vol. 55, no. 6, pp. 2640–2658, Jun. 2009.

[72] H. D. Pfister, S. K. Emmadi, and K. Narayanan, "Symmetric product codes," in *Proc. Inf. Theory and Applications Workshop (ITA)*, Feb. 2015, pp. 282–290.

[73] C. Berrou, R. Pyndiah, P. Adde, C. Douillard, and R. L. Bidan, "An overview of turbo codes and their applications," in *Proc. European Conference on Wireless Technology*, Oct. 2005, pp. 1–9.

[74] H. Mukhtar, A. Al-Dweik, and A. Shami, "Turbo product codes: Applications, challenges, and future directions," *IEEE Commun. Surveys Tutorials*, vol. 18, no. 4, pp. 3052–3069, 2016.

[75] C. Häger, H. D. Pfister, A. Graell i Amat, and F. Brännström, "Density evolution for deterministic generalized product codes on the binary erasure channel at high rates," *IEEE Trans. Inf. Theory*, vol. 63, no. 7, pp. 4357–4378, Jul. 2017.

[76] N. Abramson, "Cascade decoding of cyclic product codes," *IEEE Trans. Commun. Tech.*, vol. 16, no. 3, pp. 398–402, Jun. 1968.

[77] X. Tang and R. Koetter, "Performance of iterative algebraic decoding of codes defined on graphs: An initial investigation," in *IEEE Inf. Theory Workshop (ITW)*, Sep. 2007, pp. 254–259.

[78] C. Häger and H. D. Pfister, "Approaching miscorrection-free performance of product codes with anchor decoding," *IEEE Trans. Commun.*, vol. 66, no. 7, pp. 2797–2808, Jul. 2018.

[79] L. Tolhuizen and C. Baggen, "On the weight enumerator of product codes," *Discrete Mathematics*, vol. 106-107, pp. 483–488, 1992. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0012365X92905795

[80] L. M. G. M. Tolhuizen, "More results on the weight enumerator of product codes," *IEEE Trans. Inf. Theory*, vol. 48, no. 9, pp. 2573–2577, Sep. 2002.

[81] M. El-Khamy, "The average weight enumerator and the maximum likelihood performance of product codes," in *2005 Int. Conf. Wireless Networks, Commun. Mobile Computing*, vol. 2, 2005, pp. 1587–1592 vol.2.

[82] M. El-Khamy, "The average weight enumerator and the maximum likelihood performance of product codes," in *IEEE Trans. Inf. Theory, submitted*, Dec. 2005.

[83] G. Caire, G. Taricco, and G. Battail, "Weight distribution and performance of the iterated product of single-parity-check codes," in *IEEE GLOBECOM*, Nov. 1994, pp. 206–211.

[84] R. Mori and T. Tanaka, "Performance and construction of polar codes on symmetric binary-input memoryless channels," in *Proc. IEEE Int. Symp. on Inf. Theory*, Seoul, Jun. 2009, pp. 1496–1500.

[85] E. Şaşoğlu, "Polarization and polar codes," *Foundations and Trends® in Communications and Information Theory*, vol. 8, no. 4, pp. 259–381, 2012. [Online]. Available: http://dx.doi.org/10.1561/0100000041

[86] E. Arikan and E. Telatar, "On the rate of channel polarization," in *Proc. IEEE Int. Symp. Inf. Theory*, 2009, pp. 1493–1495.

[87] C. Leroux, I. Tal, A. Vardy, and W. J. Gross, "Hardware architectures for successive cancellation decoding of polar codes," in *IEEE Int. Conf. Acoustics, Speech and Signal Proc. (ICASSP)*, 2011, pp. 1665–1668.

[88] A. Fazeli, A. Vardy, and H. Yao, "List decoding of polar codes: How large should the list be to achieve ML decoding?" in *IEEE Int. Symp. Inf. Theory*, 2021.

[89] S. A. Hashemi, M. Mondelli, S. H. Hassani, C. Condo, R. L. Urbanke, and W. J. Gross, "Decoder partitioning: Towards practical list decoding of polar codes," *IEEE Trans. Commun.*, vol. 66, no. 9, pp. 3749–3759, 2018.

[90] M. Plotkin, "Binary codes with specified minimum distance," *IRE Trans. Inf. Theory*, vol. 6, no. 4, pp. 445–450, 1960.

[91] D. E. Muller, "Application of boolean algebra to switching circuit design and to error detection," *Trans. IRE Prof. Group on Electronic Computers*, vol. EC-3, no. 3, pp. 6–12, Sep. 1954.

[92] I. Reed, "A class of multiple-error-correcting codes and the decoding scheme," *Trans. IRE Prof. Group on Inf. Theory*, vol. 4, no. 4, pp. 38–49, Sep. 1954.

[93] M. Bardet, V. Dragoi, A. Otmani, and J.-P. Tillich, "Algebraic properties of polar codes from a new polynomial formalism," in *IEEE Int. Symp. Inf. Theory*, 2016, pp. 230–234.

[94] S. Kudekar, S. Kumar, M. Mondelli, H. D. Pfister, E. Şaşoğlu, and R. L. Urbanke, "Reed-Muller codes achieve capacity on erasure channels," *IEEE Trans. Inf. Theory*, vol. 63, no. 7, pp. 4298–4316, Jul. 2017.

[95] S. A. Hashemi, N. Doan, M. Mondelli, and W. J. Gross, "Decoding Reed-Muller and polar codes by successive factor graph permutations," in *IEEE Int. Symp. Turbo Codes & Iterative Inf. Process.*, 2018, pp. 1–5.

[96] M. Ye and E. Abbe, "Recursive projection-aggregation decoding of Reed-Muller codes," in *Proc. 2019 IEEE Int. Symp. Inf. Theory*, Jul. 2019, pp. 2064–2068.

[97] K. Ivanov and R. Urbanke, "Permutation-based decoding of Reed-Muller codes in binary erasure channel," in *Proc. 2019 IEEE Int. Symp. Inf. Theory*, Jul. 2019, pp. 21–25.

[98] M. Geiselhart, A. Elkelesh, M. Ebada, S. Cammerer, and S. ten Brink, "Automorphism ensemble decoding of Reed-Muller codes," *CoRR*, vol. abs/2012.07635, 2020. [Online]. Available: http://arxiv.org/abs/2012.07635

[99] N. Stolte, "Rekursive Codes mit der Plotkin-Konstruktion und ihre Decodierung," Ph.D. dissertation, TU Darmstadt, 2002.

[100] I. Dumer, "Recursive decoding and its performance for low-rate Reed-Muller codes," *IEEE Trans. Inf. Theory*, vol. 50, no. 5, pp. 811–823, May 2004.

[101] ——, "Soft-decision decoding of Reed-Muller codes: a simplified algorithm," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 954–963, Mar. 2006.

[102] I. Dumer and K. Shabunov, "Soft-decision decoding of Reed-Muller codes: recursive lists," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 1260–1266, Mar. 2006.

[103] R. Mori and T. Tanaka, "Performance and construction of polar codes on symmetric binary-input memoryless channels," in *Proc. of IEEE Int. Symp. on Inf. Theory*, Jun. 2009, pp. 1496–1500.

[104] S.-Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 db of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, no. 2, pp. 58–60, Feb. 2001.

[105] I. Tal and A. Vardy, "How to construct polar codes," *IEEE Trans. Inf. Theory*, vol. 59, no. 10, pp. 6562–6582, 2013.

[106] P. Trifonov, "Efficient design and decoding of polar codes," *IEEE Trans. Commun.*, vol. 60, no. 11, pp. 3221–3227, Nov. 2012.

[107] S.-Y. Chung, T. Richardson, and R. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a gaussian approximation," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 657–670, 2001.

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

[109] F. Br"annstr"om, L. Rasmussen, and A. Grant, "Convergence analysis and optimal scheduling for multiple concatenated codes," *IEEE Trans. Inf. Theory*, vol. 51, no. 9, pp. 3354–3364, 2005.

[110] C. Schürch, "A partial order for the synthesized channels of a polar code," in *IEEE Int. Symp. Inf. Theory*, 2016, pp. 220–224.

[111] G. He, J. Belfiore, I. Land, G. Yang, X. Liu, Y. Chen, R. Li, J. Wang, Y. Ge, R. Zhang, and W. Tong, "Beta-expansion: A theoretical framework for fast and recursive construction of polar codes," in *IEEE Global Commun. Conf.*, 2017, pp. 1–6.

[112] M. Mondelli, S. H. Hassani, and R. L. Urbanke, "Construction of polar codes with sublinear complexity," *IEEE Trans. Inf. Theory*, vol. 65, no. 5, pp. 2782–2791, 2019.

[113] W. Wu and P. H. Siegel, "Generalized partial orders for polar code bit-channels," *IEEE Trans. Inf. Theory*, vol. 65, no. 11, pp. 7114–7130, 2019.

[114] V. Bioglio, C. Condo, and I. Land, "Design of polar codes in 5G new radio," *IEEE Commun. Surveys Tutorials*, vol. 23, no. 1, pp. 29–40, 2021.

[115] N. Hussami, S. B. Korada, and R. Urbanke, "Performance of polar codes for channel and source coding," in *IEEE Int. Symp. Inf. Theory*, 2009, pp. 1488–1492.

[116] G. Ricciutelli, M. Baldi, and F. Chiaraluce, "Interleaver design for short concatenated codes," *IEEE Commun. Lett.*, vol. 22, no. 9, pp. 1762–1765, 2018.

[117] G. Ricciutelli, T. Jerkovits, M. Baldi, F. Chiaraluce, and G. Liva, "Analysis of the block error probability of concatenated polar code ensembles," *IEEE Trans. Commun.*, vol. 67, no. 9, pp. 5953–5962, 2019.

[118] P. Trifonov and V. Miloslavskaya, "Polar codes with dynamic frozen symbols and their decoding by directed search," in *IEEE Inf. Theory Workshop*, 2013, pp. 1–5.

[119] E. Arıkan, "From sequential decoding to channel polarization and back again," *CoRR*, vol. abs/1908.09594, 2019. [Online]. Available: http://arxiv.org/abs/1908.09594

[120] T. Tonnellier and W. J. Gross, "On systematic polarization-adjusted convolutional (PAC) codes," *IEEE Commun. Lett.*, pp. 1–1, 2021.

[121] B. Li, J. Gu, and H. Zhang, "Performance of CRC concatenated pre-transformed RM-polar codes," *CoRR*, vol. abs/2104.07486, 2021. [Online]. Available: http://arxiv.org/abs/2104.07486

[122] H. Yao, A. Fazeli, and A. Vardy, "List decoding of Arıkan's PAC codes," *Entropy*, vol. 23, no. 7, 2021. [Online]. Available: https://www.mdpi.com/1099-4300/23/7/841

[123] N. Presman, O. Shapira, and S. Litsyn, "Mixed-kernels constructions of polar codes," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 2, pp. 239–253, 2016.

[124] V. Bioglio, F. Gabry, I. Land, and J.-C. Belfiore, "Multi-kernel polar codes: Concept and design principles," *IEEE Trans. Commun.*, vol. 68, no. 9, pp. 5350–5362, 2020.

[125] G. Trofimiuk and P. Trifonov, "Efficient decoding of polar codes with some 16×16 kernels," in *IEEE Inf. Theory Workshop*, 2018, pp. 1–5.

[126] V. Miloslavskaya and P. Trifonov, "Design of binary polar codes with arbitrary kernel," in *IEEE Inf. Theory Workshop*, 2012, pp. 119–123.

[127] L. Zhang, Z. Zhang, and X. Wang, "Polar code with block-length $n = 3^n$," in *2012 Int. Conf. Wireless Commun. Signal Proc.*, 2012, pp. 1–6.

[128] N. Presman, O. Shapira, S. Litsyn, T. Etzion, and A. Vardy, "Binary polarization kernels from code decompositions," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2227–2239, 2015.

[129] H.-P. Lin, S. Lin, and K. A. S. Abdel-Ghaffar, "Linear and nonlinear binary kernels of polar codes of small dimensions with maximum exponents," *IEEE Trans. Inf. Theory*, vol. 61, no. 10, pp. 5253–5270, 2015.

[130] P. Trifonov, "On construction of polar subcodes with large kernels," in *IEEE Int. Symp. Inf. Theory*, 2019, pp. 1932–1936.

[131] A. Balatsoukas-Stimming, M. B. Parizi, and A. Burg, "LLR-based successive cancellation list decoding of polar codes," *IEEE Trans. Signal Proc.*, vol. 63, no. 19, pp. 5165–5179, 2015.

[132] A. Balatsoukas-Stimming, A. J. Raymond, W. J. Gross, and A. Burg, "Hardware architecture for list successive cancellation decoding of polar codes," *IEEE Trans. Circuits Sys. II: Express Briefs*, vol. 61, no. 8, pp. 609–613, 2014.

[133] A. Wyner and J. Ziv, "A theorem on the entropy of certain binary sequences and applications: Part I," *IEEE Trans. Inf. Theory*, vol. 19, no. 6, pp. 769–772, 1973.

[134] G. Strang, *Linear Algebra and Its Applications*, 3rd ed. New York, NY, USA: Academic, 1988.

[135] E. R. Berlekamp, "The technology of error-correcting codes," *Proc. IEEE*, vol. 68, no. 5, pp. 564–593, May 1980.

[136] H. Yao, A. Fazeli, and A. Vardy, "List decoding of Arıkan's PAC codes," in *Proc. IEEE Int. Symp. Inf. Theory*, 2020, pp. 443–448.

[137] M. Rowshan, A. Burg, and E. Viterbo, "Polarization-adjusted convolutional (PAC) codes: Sequential decoding vs list decoding," *IEEE Trans. Veh. Technol.*, vol. 70, no. 2, pp. 1434–1447, 2021.

[138] M. Mondelli, S. H. Hassani, and R. L. Urbanke, "Scaling exponent of list decoders with applications to polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 9, pp. 4838–4851, Sep. 2015.

[139] J. Neu, "Quantized polar code decoders: Analysis and design," *CoRR*, vol. abs/1902.10395, 2019. [Online]. Available: http://arxiv.org/abs/1902.10395

[140] C. Measson, A. Montanari, and R. Urbanke, "Maxwell construction: The hidden bridge between iterative and maximum a posteriori decoding," *IEEE Trans. Inf. Theory*, vol. 54, no. 12, pp. 5277–5307, Dec. 2008.

[141] G. Landsberg, "Über eine anzahlbestimmung und eine damit zusammenhängende Reihe," *Journal für die reine und angewandte Mathematik*, vol. 111, pp. 87–88, 1893. [Online]. Available: http://eudml.org/doc/148874

[142] G. Liva, E. Paolini, and M. Chiani, "Performance versus overhead for fountain codes over $\mathbb{F}_q$," *IEEE Commun. Lett.*, vol. 14, no. 2, pp. 178–180, 2010.

[143] Y. Polyanskiy, V. Poor, and S. Verdù, "Channel coding rate in the finite blocklength regime," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2307–235, May 2010.

[144] O. İşcan, D. Lentner, and W. Xu, "A comparison of channel coding schemes for 5G short message transmission," in *IEEE Global Commun. Conf. Workshops*, 2016, pp. 1–6.

[145] G. Liva and F. Steiner, "pretty-good-codes.org: Online library of good channel codes," http://pretty-good-codes.org, Mar. 2021.

[146] T. Baicheva, P. Kazakov, and M. Dimitrov, "Some comments about crc selection for the 5g nr specification," 2021.

[147] F. Chiaraluce and R. Garello, "Extended Hamming product codes analytical performance evaluation for low error rate applications," *IEEE Trans. Wireless Commun.*, vol. 3, no. 6, pp. 2353–2361, Nov. 2004.

[148] M. C. Valenti, "Channel coding for IEEE 802.16 e mobile WiMAX," in *Tutorial, IEEE Int. Conf. Commun. (ICC)*, 2009.

[149] "IEEE standard for air interface for broadband wireless access systems," *IEEE Std 802.16-2017 (Revision of IEEE Std 802.16-2012)*, pp. 1–2726, Mar. 2018.

[150] D. M. Rankin and T. A. Gulliver, "Single parity check product codes," *IEEE Trans. Commun.*, vol. 49, no. 8, pp. 1354–1362, Aug. 2001.

[151] D. M. Rankin, T. A. Gulliver, and D. P. Taylor, "Asymptotic performance of single parity-check product codes," *IEEE Trans. Inf. Theory*, vol. 49, no. 9, pp. 2230–2235, Sep. 2003.

[152] A. Fazeli, A. Vardy, and H. Yao, "Hardness of successive-cancellation decoding of linear codes," in *IEEE Int. Symp. Inf. Theory*, 2020, pp. 455–460.

[153] A. J. Salomon and O. Amrani, "Augmented product codes and lattices: Reed-Muller codes and Barnes-Wall lattices," *IEEE Trans. Inf. Theory*, vol. 51, no. 11, pp. 3918–3930, Nov. 2005.

[154] A. Pamuk and E. Arıkan, "A two phase successive cancellation decoder architecture for polar codes," in *IEEE Int. Symp. Inf. Theory*, 2013, pp. 957–961.

[155] C. Condo, V. Bioglio, H. Hafermann, and I. Land, "Practical product code construction of polar codes," *IEEE Trans. Signal Proc.*, vol. 68, pp. 2004–2014, 2020.

[156] S. Benedetto and G. Montorsi, "Unveiling turbo codes: some results on parallel concatenated coding schemes," *IEEE Trans. Inf. Theory*, vol. 42, no. 2, pp. 409–428, Mar. 1996.

[157] G. Battail, "Construction explicite de bons codes longs," *Ann. Télécommun.*, vol. 16, pp. 392–404, 1989.

[158] ——, "A conceptual framework for understanding turbo codes," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 2, pp. 245–254, 1998.

[159] E. Biglieri and V. Volski, "Approximately Gaussian weight distribution of the iterated product of single-parity-check codes," *Electronics Letters*, vol. 30, pp. 923–924(1), June 1994. [Online]. Available: https://digital-library.theiet.org/content/journals/10.1049/el_19940659

[160] M. Kousa, "A novel approach for evaluating the performance of spc product codes under erasure decoding," *IEEE Trans. Commun.*, vol. 50, no. 1, pp. 7–11, 2002.

[161] J. Tee, D. Taylor, and P. Martin, "Multiple serial and parallel concatenated single parity-check codes," *IEEE Trans. Commun.*, vol. 51, no. 10, pp. 1666–1675, 2003.

[162] J. Li, K. Narayanan, and C. Georghiades, "Product accumulate codes: a class of codes with near-capacity performance and low decoding complexity," *IEEE Trans. Inf. Theory*, vol. 50, no. 1, pp. 31–46, 2004.

[163] R. Garello and G. Verardo, "A simplified application of ordered statistics decoding to single parity check product codes," in *Int. Annu. Conf. (AEIT)*, 2019, pp. 1–6.

[164] H. Pfister, "Capacity via symmetry: Extensions and practical consequences," *MIT LIDS Seminar Series*, Apr. 2017. [Online]. Available: http://pfister.ee.duke.edu/talks/mit17.pdf

[165] M. C. Coşkun, "Successive cancellation decoding of single parity-check product codes," Master Thesis, Technical University of Munich, 2017.

[166] F. MacWilliams, C. Mallows, and N. Sloane, "Generalizations of Gleason's theorem on weight enumerators of self-dual codes," *IEEE Trans. Inf. Theory*, vol. 18, no. 6, pp. 794–805, Nov. 1972.

[167] D. M. Rankin, T. A. Gulliver, and D. P. Taylor, "Parallel and serial concatenated single parity check product codes," *EURASIP J. Adv. Signal Process*, vol. 2005, p. 775–783, Jan. 2005. [Online]. Available: https://doi.org/10.1155/ASP.2005.775

[168] C. Di, D. Proietti, I. E. Telatar, T. J. Richardson, and R. L. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *IEEE Trans. Inf. Theory*, vol. 48, no. 6, pp. 1570–1579, Jun 2002.

[169] G. Poltyrev, "Bounds on the decoding error probability of binary linear codes via their spectra," *IEEE Trans. Inf. Theory*, vol. 40, no. 4, pp. 1284–1292, Jul 1994.

[170] F. Gabry, V. Bioglio, I. Land, and J. C. Belfiore, "Multi-kernel construction of polar codes," in *IEEE Int. Conf. Commun. (ICC) Workshops*, May 2017, pp. 761–765.

[171] S. Kumar, R. Calderbank, and H. D. Pfister, "Beyond double transitivity: Capacity-achieving cyclic codes on erasure channels," in *IEEE Inf. Theory Workshop (ITW)*, Sep. 2016, pp. 241–245.

[172] R. Wang and R. Liu, "A novel puncturing scheme for polar codes," *IEEE Commun. Lett.*, vol. 18, no. 12, pp. 2081–2084, 2014.

[173] M. Geiselhart, A. Elkelesh, M. Ebada, S. Cammerer, and S. ten Brink, "CRC-aided belief propagation list decoding of polar codes," *CoRR*, vol. abs/2001.05303, 2020. [Online]. Available: https://arxiv.org/abs/2001.05303

[174] W. Yang, G. Durisi, T. Koch, and Y. Polyanskiy, "Quasi-static multiple-antenna fading channels at finite blocklength," *IEEE Trans. Commun.*, vol. 60, no. 7, pp. 4232–4265, Jul. 2014.

[175] G. Durisi, T. Koch, J. Östman, Y. Polyanskiy, and W. Yang, "Short-packet communications over multiple-antenna Rayleigh-fading channels," *IEEE Trans. Commun.*, vol. 64, no. 2, pp. 618–629, Feb. 2016.

[176] J. Östman, G. Durisi, E. G. Ström, J. Li, H. Sahlin, and G. Liva, "Low-latency ultra-reliable 5G communications: Finite block-length bounds and coding schemes," in *Int. ITG Conf. Sys. Commun. Coding (SCC)*, Hamburg, Germany, Feb. 2017.

[177] A. Lancho, J. Östman, G. Durisi, T. Koch, and G. Vazquez-Vilar, "Saddlepoint approximations for short-packet wireless communications," *IEEE Trans. Wireless Commun.*, vol. 19, no. 7, pp. 4831–4846, 2020.

[178] A. Martinez and A. Guillén i Fàbregas, "Saddlepoint approximation of random–coding bounds," in *Inf. Theory Applic. Workshop (ITA)*, San Diego, CA, U.S.A., Feb. 2011.

[179] D. Warrier and U. Madhow, "Spectrally efficient noncoherent communication," *IEEE Trans. Inf. Theory*, vol. 48, no. 3, pp. 651–668, 2002.

[180] J. Ding and A. Zhou, "Eigenvalues of rank-one updated matrices with some applications," *Applied Math. Lett.*, vol. 20, no. 12, pp. 1223–1226, 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893965907000614

[181] M. S. Bartlett, "An inverse matrix adjustment arising in discriminant analysis," *Annals Math. Statistics*, vol. 22, no. 1, pp. 107 – 111, 1951. [Online]. Available: https://doi.org/10.1214/aoms/1177729698

[182] G. Taricco and E. Biglieri, "Space-time decoding with imperfect channel estimation," *IEEE Trans. Wireless Commun.*, vol. 4, no. 4, pp. 1874–1888, 2005.

[183] H. Meyr, M. Moeneclaey, and S. Fechtel, *Digital Communication Receivers: Synchronization, Channel Estimation, and Signal Processing.* Wiley, 1997.

[184] H. Wymeersch, *Iterative Receiver Design.* Cambridge, 2007.

[185] C. Herzet, N. Noels, V. Lottici, H. Wymeersch, M. Luise, M. Moeneclaey, and L. Vandendorpe, "Code-aided turbo synchronization," *Proc. IEEE*, vol. 95, no. 6, pp. 1255–1271, 2007.

[186] N. Noels, C. Herzet, A. Dejonghe, V. Lottici, H. Steendam, M. Moeneclaey, M. Luise, and L. Vandendorpe, "Turbo synchronization: an EM algorithm interpretation," in *IEEE Int. Conf. Commun.*, vol. 4, 2003, pp. 2933–2937 vol.4.

[187] J. Dauwels and H. A. Loeliger, "Phase estimation by message passing," in *IEEE Int. Conf. Commun.*, vol. 1, 2004, pp. 523–527 Vol.1.

[188] C. Herzet, V. Ramon, and L. Vandendorpe, "A theoretical framework for iterative synchronization based on the sum–product and the expectation-maximization algorithms," *IEEE Trans. Signal Process.*, vol. 55, no. 5, pp. 1644–1658, 2007.

[189] M. Khalighi and J. J. Boutros, "Semi-blind channel estimation using the EM algorithm in iterative MIMO APP detectors," *IEEE Trans. Wireless Commun.*, vol. 5, no. 11, pp. 3165–3173, Nov. 2006.

[190] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Royal Stat. Soc., series B*, vol. 39, no. 1, pp. 1–38, 1977.

[191] R. Johannesson and K. S. Zigangirov, *Fundamentals of Convolutional Coding*, 2nd ed. Piscataway, NJ, USA: Wiley-IEEE Press, 2015.

[192] R. Imad, S. Houcke, and M. Ghogho, "Blind estimation of the phase and carrier frequency offsets for LDPC-coded systems," *EURASIP J. Adv. Signal Process.*, vol. 2010, no. 1, pp. 1–13, 2010.

[193] Rong-Rong Chen, R. Koetter, U. Madhow, and D. Agrawal, "Joint noncoherent demodulation and decoding for the block fading channel: a practical framework for approaching Shannon capacity," *IEEE Trans. Commun.*, vol. 51, no. 10, pp. 1676–1689, 2003.

[194] G. Coluccia and G. Taricco, "An optimum blind receiver for correlated Rician fading MIMO channels," *IEEE Commun. Lett.*, vol. 11, no. 9, pp. 738–739, 2007.

[195] B. Matuz, G. Liva, E. Paolini, M. Chiani, and G. Bauch, "Low-rate non-binary LDPC codes for coherent and blockwise non-coherent AWGN channels," *IEEE Trans. Commun.*, vol. 61, no. 10, pp. 4096–4107, 2013.

[196] D. Rife and R. Boorstyn, "Single tone parameter estimation from discrete-time observations," *IEEE Trans. Inf. Theory*, vol. 20, no. 5, pp. 591–598, 1974.

[197] M. Jeong and S. Hong, "SC-Fano decoding of polar codes," *IEEE Access*, vol. 7, pp. 81 682–81 690, 2019.

[198] A. Alamdar-Yazdi and F. R. Kschischang, "A simplified successive-cancellation decoder for polar codes," *IEEE Commun. Lett.*, vol. 15, no. 12, pp. 1378–1380, 2011.

[199] G. Sarkis, P. Giard, A. Vardy, C. Thibeault, and W. J. Gross, "Fast polar decoders: Algorithm and implementation," *IEEE J. Sel.Areas Commun.*, vol. 32, no. 5, pp. 946–957, 2014.