Polar Codes: Finite Length Implementation, Error Correlations and Multilevel Modulation

Mani Bastani Parizi

Project Supervisor: Professor Emre Telatar

Assistant: Mohammad Karzand

EPFL - Ecole Polytechnique Fédérale de Lausanne

January 2012

Revised: February 2012

Thesis presented to the faculty of computer and communication sciences for obtaining the degree of Master of Science.

Ecole Polytechnique Fédérale de Lausanne, 2012

Abstract

Shannon, in his seminal work, formalized the transmission of data over a communication channel and determined its fundamental limits. He characterized the relation between communication rate and error probability and showed that as long as the communication rate is below the capacity of the channel, error probability can be made as small as desirable by using appropriate coding over the communication channel and letting the codeword length approach infinity. He provided the formula for capacity of discrete memoryless channel. However, his proposed coding scheme was too complex to be practical in communication systems.

Polar codes, recently introduced by Arıkan, are the first practical codes that are known to achieve the capacity for a large class of channel and have low encoding and decoding complexity.

The original polar codes of Arıkan achieve a block error probability decaying exponentially in the square root of the block length as it goes to infinity. However, it is interesting to investigate their performance in finite length as this is the case in all practical communication schemes.

In this dissertation, after a brief overview on polar codes, we introduce a practical framework for simulation of error correcting codes in general. We introduce the importance sampling concept to efficiently evaluate the performance of polar codes with finite bock length.

Next, based on simulation results, we investigate the performance of different genie-aided decoders to mitigate the poor performance of polar codes in low to moderate block length and propose single-error correction methods to improve the performance dramatically in expense of complexity of decoder. In this context, we also study the correlation between error events in a successive cancellation decoder.

Finally, we investigate the performance of polar codes in non-binary channels. We compare the code construction of Şaşoğlu for Q-ary channels and classical multilevel codes. We construct multilevel polar codes for Q-ary channels and provide a thorough comparison of complexity and performance of two methods in finite length.

Acknowledgments

It has been a great pleasure for me to work on this thesis under supervision of Professor Emre Telatar. I would like to thank him for his guidance, support and contributions that made this work possible. I am extremely grateful to his assistant Mohammad Karzand for his kind helps and valuable advices.

I thank Professor Christophe Vignat who agreed to be the expert on oral exam of the thesis. I also thank Dr. Eren Şaşoğlu and Seyed Hamed Hassani for beneficial discussions and sharing their experience on polar codes with me.

I thank Damir Laurenzi for providing the cluster of computers with which I could run massive simulations that constitute an important part of this work.

Thanks to all my friends in Switzerland who made my life, thousands of kilometers far from home, pleasant.

Last but not the least, I thank my parents Dordaneh Dorreh and Hamid Bastani Parizi for their endless love and support.

Contents

Introduction to Polar Codes 1

"Channel polarization" was initially discovered by Arıkan in [1] as a means for constructing capacity achieving codes for symmetric binary-input memoryless channels (B-DMCs). Polar codes are the only explicitly described codes known so far to provably achieve the capacity of all symmetric B-DMCs. The polarization idea, later on, was studied in more detail by researchers in the information theory community and it turned out that it is actually a more general phenomena that can be exploited in a broader range of problems such as source coding, multiple access channel, etc.

In this chapter, we shall briefly review this phenomena which is the basis of the following chapters. We will restate the results in [1] and for the sake of brevity, we will omit some of the proofs.

1.1 Notations

Throughout this thesis, we use uppercase letters (like X) to indicate a random variable, and its lowercase version (x) for a realization of that random variable.

We denote the sets by script-style uppercase letters like $\mathcal{S}$ except for the standard fields like $\mathbb{R}$ or $\mathbb{F}_Q$. By superscript C we mean the complementary of a set (e.g., $\mathcal{S}^C$). The complementary set, however, should be defined with respect to a reference set which will be explicitly described whenever it is not clear from context. Finally, by $|\mathcal{S}|$ we mean the cardinality of $\mathcal{S}$.

By boldface letters like $\mathbf{x}$ we mean either a column vector or a matrix (which will be clear from context). We simply denote the elements of a vector/matrix by its normal name with subscripts that indicate the index of that element (for example x_i or $x_{i,j}$ for vector $\mathbf{x}$ or matrix $\mathbf{x}$ respectively). We denote the row vectors with a small arrow on top of them like $\vec{x}$. We denote the $n \times n$ identity matrix by $\mathbf{I}_n$.

$\mathbf{x}_i^j$ (similarly $\vec{x}_i^j$) indicates the sub-vector $[x_i, \cdots, x_j]^T$ (similarly $[x_i, \cdots, x_j]$) whenever $i \leqslant j$ and a null vector otherwise. We also use the shorthand notations of $\mathbf{x}_{i,e}^j$ and $\mathbf{x}_{i,o}^j$ to denote the sub-vectors of $\mathbf{x}$ formed by even and odd indices from i to j respectively.

An alternative way of indicating the sub vectors is by notation $\mathbf{x}_{\mathcal{I}}$ (where $\mathcal{I}$ is a set of indices) which means the sub-vector made up of the elements of $\mathbf{x}$ with indices in $\mathcal{I}$.

The Kronecker product of two matrices $\mathbf{A}$, $m \times n$ and $\mathbf{B}$, $s \times r$ is defined as:

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} A_{0,0}\mathbf{B} & \cdots & A_{0,n-1}\mathbf{B} \\ \vdots & \cdots & \vdots \\ A_{m-1,0}\mathbf{B} & \cdots & A_{m-1,n-1}\mathbf{B} \end{bmatrix} \tag{1.1}$$

whose dimensions are $ms \times nr$. The Kronecker power $\mathbf{A}^{\otimes n}$ is defined recursively as $\mathbf{A} \otimes \mathbf{A}^{\otimes(n-1)}, \quad \forall n \geqslant 1$ and conventionally $\mathbf{A}^{\otimes 0} = 1$.

The *indicator function* $\mathbb{1}_{\mathcal{S}}(x)$ is defined as

$$\mathbb{1}_{\mathcal{S}}(x) = \begin{cases} 1 & \text{if } x \in \mathcal{S} \\ 0 & \text{otherwise} \end{cases}. \tag{1.2}$$

Moreover, whenever we use the indicator symbol with a boolean condition (for example $\mathbb{1}_{x \geqslant 2}$) we mean the variable which takes value 1 when the condition is true and 0 otherwise.

We also define the *bar* notation as follows:

$$\bar{x} = 1 - x, \tag{1.3}$$

where the subtraction is defined depending on the field/group from which x is chosen.

In general, we denote a “memoryless” channel with $W : \mathcal{X} \longrightarrow \mathcal{Y}$ where $\mathcal{X}$ is the input alphabet of the channel, $\mathcal{Y}$ its output alphabet, and the transition probability $W(y|x), x \in \mathcal{X}, y \in \mathcal{Y}$. We write $W^N : \mathcal{X}^N \longrightarrow \mathcal{Y}^N$ to denote the “vector” channel corresponding to N independent uses of W:

$$W^N(\mathbf{y}|\mathbf{x}) = \prod_{i=0}^{N-1} W(y_i|x_i)$$

where $\mathbf{x}$ and $\mathbf{y}$ denote the vectors of N channel inputs and outputs respectively.

1.2 Preliminaries

Definition 1.1 (Binary Discrete Memoryless Channel (B-DMC))**.** If the input alphabet of a memoryless channel $W : \mathcal{X} \longrightarrow \mathcal{Y}$ is $\mathcal{X} = \mathbb{F}_2 = \{0, 1\}$, the channel will be called *Binary Discrete Memoryless Channel* which we abbreviate as B-DMC.

Definition 1.2 (Symmetric Capacity of a B-DMC)**.** For a B-DMC W, the symmetric capacity is defined as:

$$I(W) \triangleq \sum_{y \in \mathcal{Y}} \sum_{x \in \mathcal{X}} \frac{1}{2} W(y|x) \log \frac{W(y|x)}{\frac{1}{2}W(y|0) + \frac{1}{2}W(y|1)}. \tag{1.4}$$

Definition 1.3 (Bhattacharyya Parameter of a B-DMC)**.** For a B-DMC W, the Bhattacharyya parameter is defined as:

$$Z(W) \triangleq \sum_{y \in \mathcal{Y}} \sqrt{W(y|0)W(y|1)}. \tag{1.5}$$

We didn't put any restrictions on the output alphabet. Whenever the output alphabet is continuous, the summations (over $y \in \mathcal{Y}$) will be replaced by integrations.

Symmetric Capacity and Bhattacharyya parameter, measure the maximum possible rate of reliable information transmission (using uniformly distributed inputs) and the reliability of transmission respectively. Recall that the Bhattacharyya parameter is an upper bound on probability of maximum-likelihood decision when W is used once to transmit a bit.

It is easy to check $0 \leqslant Z(W) \leqslant 1$ and if the base of logarithm is 2, $I(W)$ will also lie in $[0, 1]$.

Lemma 1.1. *For any B-DMC W we have the following bounds:*

$$I(W) \geqslant \log \frac{2}{1 + Z(W)}, \tag{1.6a}$$

$$I(W) \leqslant \sqrt{1 - Z(W)^2}, \tag{1.6b}$$

$$I(W) \geqslant 1 - Z(W). \tag{1.6c}$$

Remark. Although the lower bound of (1.6c) is stronger than the bound of (1.6a) we have mentioned both of them since the stronger lower bound will be obtained using the statistical analysis on Polar Codes. See [1, Proposition 1] and [1, Proposition 11].

As it is expected by intuition, Lemma 1.1 suggests that whenever the transmission rate is close to one, the reliability is high ($Z(W) \approx 0$) which means a situation that the B-DMC is in fact very "good" channel. In contrast, when the maximum possible rate is close to zero, the channel is very "bad" which in turns means the reliability is low ($Z(W) \approx 1$). As depicted in Figure 1.1 except those extremal cases, most of the B-DMCs have moderate symmetric capacity and reliability. Shortly we will see that polarization is nothing but a transform that moves a bunch of (identical) B-DMCs that lie in the middle of the $I - Z$ plane to the extremal edges of the plane (useless channels or perfect noiseless channels).

Definition 1.4 (Symmetric B-DMC)**.** A B-DMC W is said to be *symmetric* if there exits a permutation $\pi_1(y)$ such that:

1. $\pi_1^{-1} = \pi_1$ and
2. $W(y|1) = W(\pi_1(y)|0), \quad \forall y \in \mathcal{Y}$

The symmetric capacity of a symmetric B-DMC is equal to its Shannon capacity.

Lemma 1.2. *Let's denote the identity permutation by π_0. Then for a symmetric B-DMC W,*

$$W(y|x + a) = W(\pi_a(y)|x) = W(\pi_x(y)|a), \qquad \forall a, x \in \mathbb{F}_2. \tag{1.7}$$

Likewise, for the vector B-DMC $W^N : \mathcal{X}^N \longrightarrow \mathcal{Y}^N$ we have:

$$W^N(\mathbf{y}|\mathbf{x} + \mathbf{a}) = W(\pi_{\mathbf{a}}(\mathbf{y})|\mathbf{x}) = W(\pi_{\mathbf{x}}(\mathbf{y})|\mathbf{a}), \qquad \forall \mathbf{a}, \mathbf{x} \in \mathbb{F}_2^N \tag{1.8}$$

where $\pi_{\mathbf{a}}(\mathbf{y}) = \left[\pi_{a_0}(y_0), \pi_{a_1}(y_1), \cdots, \pi_{a_{N-1}}(y_{N-1})\right]^T$.

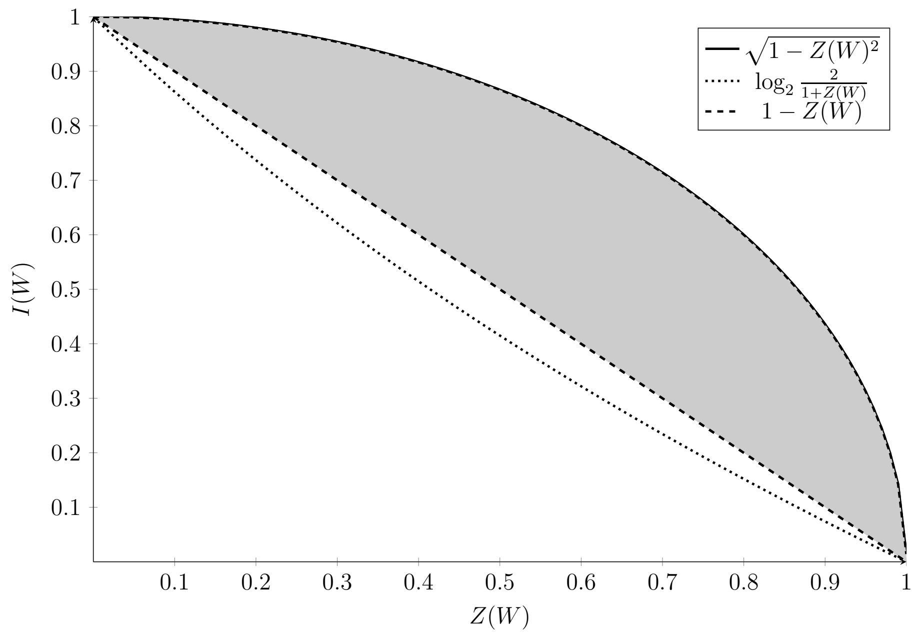


Figure 1.1: As Lemma 1.1 suggests, for any B-DMC the $(Z(W), I(W))$ pair lies in the shaded region

The proof is immediate and we omit it.

Definition 1.5 (Binary Symmetric Channel (BSC))**.** A BSC is B-DMC W with $\mathcal{Y} = \{0,1\}$ where:

$$W(0|0) = W(1|1), \text{ and } W(1|0) = W(0|1).$$

A BSC can be entirely described by a single parmeter $p \triangleq W(1|0)$ which is called its "crossover" probability. We denote such a BSC with $BSC(p)$.

It is easy to verify that the symmetric capacity and Bhattacharyya parameter of a $BSC(p)$ are equal to:

$$I(W) = C_{BSC} = 1 - h_2(p)$$
$$Z(W) = Z_{BSC} = 2\sqrt{p(1-p)}$$

respectively, where $h_2(p)$ is the binary entropy function defined as:

$$h_2(p) = p \log_2 \frac{1}{p} + (1-p) \log_2 \frac{1}{1-p}. \tag{1.9}$$

Definition 1.6 (Binary Erasure Channel (BEC))**.** A BEC is a B-DMC W such that for each $y \in \mathcal{Y}$ either:

1. $W(y|0)W(y|1) = 0$ or
2. $W(y|0) = W(y|1)$.

In the latter case y is called an *erasure symbol.* A BEC can be described with its *erasure probability* which is defined as:

$$\epsilon \triangleq \sum_{\forall y \in \mathcal{Y}: W(y|0) = W(y|1)} W(y|0)$$

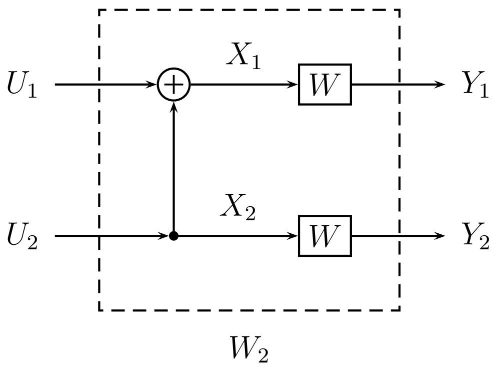


Figure 1.2: Single step polar transform

and will be denoted as $BEC(\epsilon)$.

For a $BEC(\epsilon)$, the symmetric capacity and Bhattacharyya parameters are:

$$I(W) = C_{BEC} = 1 - \epsilon$$
$$Z(W) = Z_{BEC} = \epsilon.$$

This shows BECs are the channels that form the lower bound plotted in Figure 1.1

1.3 Channel Polarization

Having established the basics and notations, we are ready to study the polarization phenomena proposed by Arıkan. Assume we have a B-DMC W. We take two copies of this channel and combine them as depicted in Figure 1.2. Simply, instead of feeding the two inputs to two channels directly, we add them up and feed the result to the first channel and feed the second channel with the second input.

Hence, we have defined a *super channel* $W_2 : \mathcal{X}^2 \longrightarrow \mathcal{Y}^2$ as:

$$W_2(y_1, y_2|u_1, u_2) = W(y_1|u_1 + u_2)W(y_2|u_2). \tag{1.10}$$

It is easy to verify that if U_1 and U_2 are chosen independently and uniformly from $\mathcal{X} = \mathbb{F}_2$, X_1 and X_2 will also have uniform distribution in $\mathbb{F}_2$ and they will be independent as well. Hence, we will have the following equalities:

$$I(W_2) = I(U_1U_2; Y_1Y_2) = I(X_1X_2; Y_1Y_2) = 2I(W) \tag{1.11}$$

which shows the transformation is preserving the mutual information.

On the other side, we can expand $I(U_1U_2; Y_1Y_2)$ by chain rule as:

$$I(U_1U_2; Y_1Y_2) = I(U_1; Y_1Y_2) + I(U_2; Y_1Y_2|U_1) = I(U_1; Y_1Y_2) + I(U_2; Y_1Y_2U_1) \tag{1.12}$$

where the last equality follows from independence of U_1 and U_2.

Next, we look at the second term, $I(U_2; Y_1Y_2U_1)$:

$$\begin{aligned} I(U_2; Y_1Y_2U_1) &= H(U_2) - H(U_2|Y_1Y_2U_1) \\ &\geqslant H(U_2) - H(U_2|Y_2) = I(W) \end{aligned} \tag{1.13}$$

Combining (1.13) with the chain rule expansion (1.12) and the information preserving property (1.11) we can conclude that:

$$I(U_1;Y_1Y_2) \leqslant I(W) \leqslant I(U_2;Y_1Y_2U_1) \tag{1.14}$$

Last inequalities suggest that if we consider the channel seen from U_1 to (Y_1, Y_2) pair we see a channel "worse" than the original channel (in terms of symmetric mutual information) while the channel seen from U_2 to (U_1, Y_1, Y_2) is a "better" channel[1]. Hence, we could *split* the super channel into two channels and define the worse and better B-DMCs as:

$$\begin{aligned} W^-(y_1, y_2|u_1) &= \sum_{u_2 \in \mathcal{X}} \frac{1}{2} W_2(y_1, y_2|u_1, u_2) \\ W^+(y_1, y_2, u_1|u_2) &= W_2(y_1, y_2|u_1, u_2) \end{aligned}$$

with the property that:

$$I(W^-) \leqslant I(W) \leqslant I(W^+)$$

The transformation we just described (including channel combining and channel splitting) is the basis for channel polarization. Indeed, if we apply the transform once again on W^+ and W^- (which is equivalent to using the main channel W four times), we get channels that are "better than better" or "worse than worse". In the following sense:

$$\begin{aligned} I(W^{+-}) \leqslant& I(W^+) \leqslant I(W^{++}) \\ I(W^{--}) \leqslant& I(W^-) \leqslant I(W^{-+}) \end{aligned}$$

Even though it is not obvious yet, but we would expect after repeating this transform several times, we end up with a set of "extremal" channels which are either perfect (noiseless) or useless.

Let us formalize what we have seen so far:

Definition 1.7 (Single Step Polar Transform)**.** Having a B-DMC W, the single step polar transform $(W, W) \mapsto (W^-, W^+)$ transforms two identical copies of W to two B-DMCs, $W^- : \mathcal{X} \longrightarrow \mathcal{Y}^2$ and $W^+ : \mathcal{X} \longrightarrow \mathcal{Y}^2 \times \mathcal{X}$ as:

$$W^-(y_1, y_2|u_1) = \sum_{u_2 \in \mathcal{X}} \frac{1}{2} W(y_1|u_1 + u_2)W(y_2|u_2) \tag{1.15a}$$

$$W^+(y_1, y_2, u_1|u_2) = Wy_1|u_1 + u_2)W(y_2|u_2) \tag{1.15b}$$

[1]At this point the reader may ask while U_1 is an input of the physical channel, how it can be considered as the output of the forged channel. In the other words, how the receiver can have access to U_1. This question will be answered shortly when we introduce the successive cancellation decoder.

Lemma 1.3 (Local Transformation of Rate). *Suppose we apply the local polar transform on a B-DMC W, $(W, W) \mapsto (W^-, W^+)$, then:*

$$I(W^+) + I(W^-) = 2I(W) \tag{1.16a}$$
$$I(W^-) \leqslant I(W^+) \tag{1.16b}$$

with equality iff $I(W) = 1$ or $I(W) = 0$ (i.e., W is an "extremal" channel itself).

Lemma 1.4 (Local Transformation of Reliabilty, [1, Proposition 5]). *Suppose we apply the local polar transform on a B-DMC W, $(W, W) \mapsto (W^-, W^+)$, then:*

$$Z(W^+) = Z(W)^2 \tag{1.17a}$$
$$Z(W^-) \leqslant 2Z(W) - Z(W)^2 \tag{1.17b}$$
$$Z(W^-) \geqslant Z(W) \geqslant Z(W^+). \tag{1.17c}$$

with equality in (1.17b) *iff W is a BEC.*

Note that (1.17a) and (1.17b) imply that equality in (1.17c) happens iff W is an extremal channel ($Z(W) = 1$ or $Z(W) = 0$). Moreover since $Z(W^-) + Z(W^+) \leqslant 2Z(W)$ we conclude that the single step polar transform improves the reliability of channels.

Lemma 1.5 (Polar Transform of BEC [1, Proposition 6]). *If W is a BEC with erasure probability ϵ, applying the single step transform $(W, W) \mapsto (W^-, W^+)$, produces two BECs W^+ with erasure probability ϵ^2 and W^- with erasure probability $2\epsilon - \epsilon^2$. Conversely, if W^- or W^+ is a BEC, then W is BEC. Moreover, (y_1, y_2) is an erasure symbol for W^- iff either y_1* or *y_2 is an erasure symbol of W. Similarly, (y_1, y_2, u_1) is an erasure symbol for W^+ iff both y_1* and *y_2 are are erasure symbols of W.*

The second step of channel combining, is shown in Figure 1.3; Two independent copies of W_2 are combined exactly in the same fashion that we combined two copies of W previously and the "super channel" W_4 is obtained. One can write the mapping $(U_1, U_2, U_3, U_4) \mapsto (X_1, X_2, X_3, X_4)$ as:

$$\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} = \mathbf{G}_4 \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix}.$$

where

$$\mathbf{G}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

After channel combining one can forge four binary-input channels out of the super channel W_4 which is equivalent to applying the single step polar transform (defined in Definition 1.7) to each of W^- and W^+ we had obtained from single step transform as follows: we have two "bad channels" W^- (one from V_1 to Y_1, Y_2 and the other one from V_3 to Y_3, Y_4), then we apply the single step transform to these two channels and obtain W^{--}

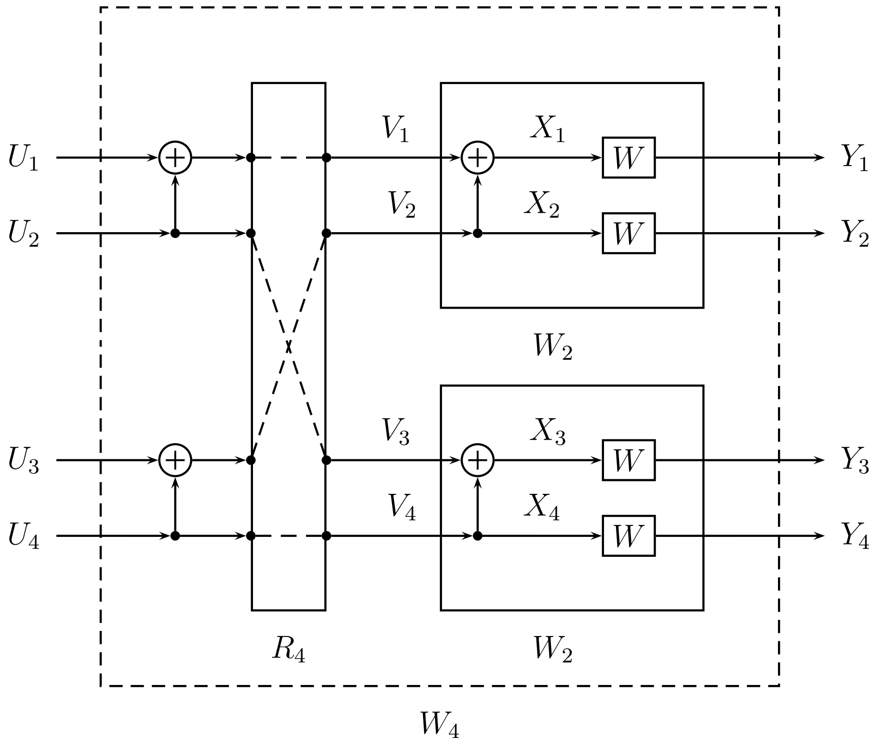


Figure 1.3: The second step of polar transform

(the channel from U_1 to Y_1, Y_2, Y_3, Y_4) and W^{-+} (from U_2 to Y_1, Y_2, Y_3, Y_4, U_1). Similarly, we have applied the same transform on good channels (W^{+}s) and obtained W^{+-} (from U_3 to $Y_1, Y_2, Y_3, Y_4, U_1, U_2$) and W^{++} (from U_4 to $Y_1, Y_2, Y_3, Y_4, U_1, U_2, U_3$).

In general, the nth step of channel combining, corresponding to $N = 2^n$ uses of the original B-DMC W is shown in Figure 1.4. $\mathbf{R}_N$ is the reverse shuffle permutation, with the input output relationship as follows:

$$\begin{aligned} V_i &= S_{2i} \\ V_{i+N/2} &= S_{2i+1}, \qquad i = 0, 1, \cdots, N/2 - 1 \end{aligned}$$

It is easy to verify that the mapping $\mathbf{U} \mapsto \mathbf{V}$ is linear and hence by induction the overall mapping from $\mathbf{U}$ to the input of raw vector channel $W^N : \mathcal{X}^N \longrightarrow \mathcal{Y}^N$ is linear. For the time being, we just denote this transform by $\mathbf{G}_N$ and the input output relationship by $\mathbf{x} = \mathbf{G}_N \mathbf{u}$. Later, we will derive explicit formula for $\mathbf{G}_N$.

We defined the n level channel combining by a linear transform on the vector of channel input and derived the super channel $W_N : \mathcal{X}^N \longrightarrow \mathcal{Y}^N$ as

$$W_N(\mathbf{y}|\mathbf{u}) = W^N(\mathbf{y}|\mathbf{G}_N\mathbf{u}). \tag{1.18}$$

Extending the single step procedure, we can forge $N = 2^n$ binary-input channels $W_N^{(i)} : \mathcal{X} \longrightarrow \mathcal{Y}^N \times \mathcal{X}^i, \quad i = 0, 1, \cdots, N-1$ out of the super channel W_N as follows:

$$W_N^{(i)}\left(\mathbf{y}, \mathbf{u}_0^{i-1}|u_i\right) \triangleq \sum_{\mathbf{u}_{i+1}^{N-1} \in \mathcal{X}^{N-i}} \frac{1}{2^{N-i}} W_N(\mathbf{y}|\mathbf{u}). \tag{1.19}$$

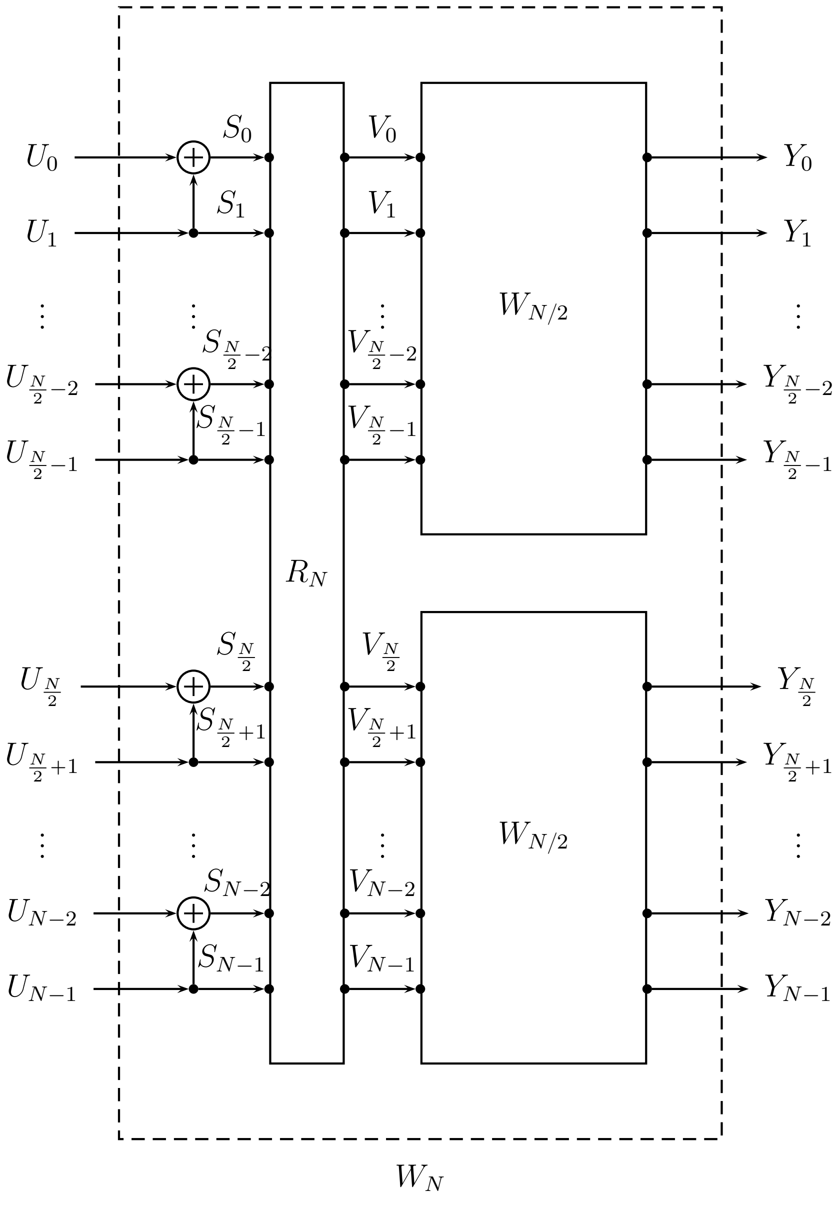


Figure 1.4: n steps of channel combining corresponding to $N = 2^n$ uses of W

Each pair of those binary-input channels are obtained by applying the single step polar transform (Definition 1.7) to one of the forged binary-inputs of the previous step. Namely:

$$\left(W_{N/2}^{(i)}, W_{N/2}^{(i)}\right) \mapsto \left(W_N^{(2i)}, W_N^{(2i+1)}\right) \qquad i = 0, 1, \cdots, N/2 - 1.$$

Due to this recursive construction, the results of lemmas 1.3 and 1.4 are true for every step of polarization. Moreover, by induction we can conclude that the n-level polar transform will preserve the rate and improve the reliability in the following sense:

$$\sum_{i=0}^{N-1} I(W_N^{(i)}) = NI(W), \tag{1.20}$$

$$\sum_{i=0}^{N-1} Z(W_N^{(i)}) \leqslant NZ(W) \tag{1.21}$$

with equality in (1.21) iff W is a BEC.

Note that since Lemma 1.5 is valid for every single step of polar transform, all $W_N^{(i)}$s will be BEC iff W is a BEC.

The more interesting property of polar transform is that the forged channels converge to extremal channels (i.e., either perfect noiseless or useless channels) for sufficiently large n:

Theorem 1.1 (Channel Polarization, [1, Theorem 1]). *For any B-DMC, the channels $W_N^{(i)}$ (obtained through channel combining and splitting procedure described by* (1.18) *and* (1.19) *respectively)* polarize *in the sense that for any $\delta > 0$,*

$$\lim_{n\to\infty} \frac{1}{N} \left|\left\{W_N^{(i)} : I(W_N^{(i)}) > 1 - \delta\right\}\right| = I(W),$$
$$\lim_{n\to\infty} \frac{1}{N} \left|\left\{W_N^{(i)} : I(W_N^{(i)}) < \delta\right\}\right| = 1 - I(W)$$

Moreover, the speed of convergence of the channels is exponentially fast:

Theorem 1.2 (Rate of Polarization, [2]). *For any B-DMC W and any $\beta < \frac{1}{2}$*

$$\lim_{n\to\infty} \frac{1}{N} \left|\left\{W_N^{(i)} : Z(W_N^{(i)}) \leqslant 2^{-N^\beta}\right\}\right| = I(W).$$

Corollary 1.1. *For any B-DMC W with $I(W) > 0$, any fixed $R < I(W)$, and fixed $\beta < 1/2$, there exits a sequence of sets $\mathcal{A}_N \subset \{0, 1, \cdots, N-1\}$, $N = 1, 2, 4, \cdots, 2^n, \cdots$ such that $I(W) \geqslant \frac{1}{N}|\mathcal{A}_N| \geqslant R$ and $Z(W_N^{(i)}) \leqslant 2^{-N^\beta} \quad \forall i \in \mathcal{A}_N$.*

Lemma 1.6 ([1, Proposition 13]). *If the B-DMC W is symmetric, then the super channel W_N and forged channels $W_N^{(i)}$ are symmetric in the following sense:*

$$W_N(\mathbf{y}|\mathbf{u}) = W_N(\pi_{\mathbf{G}_N\mathbf{a}}(\mathbf{y})|\mathbf{u} + \mathbf{a}), \tag{1.22}$$

$$W_N^{(i)}(\mathbf{y}, \mathbf{u}_0^{i-1}|u_i) = W_N^{(i)}(\pi_{\mathbf{G}_N\mathbf{a}}(\mathbf{y}), \mathbf{u}_0^{i-1} + \mathbf{a}_0^{i-1}|u_i + a_i) \tag{1.23}$$

for all $\mathbf{a}, \mathbf{u} \in \mathcal{X}^N$, $\mathbf{y} \in \mathcal{Y}^N$, $N = 2^n$, $0 \leqslant i \leqslant N - 1$.

1.4 Polar Coding

In the view of the results we obtained in Section 1.3 it is predictable that polar coding is nothing but polarizing a B-DMC and then sending the information over the "good" forged channels. In this section we formalize this idea.

1.4.1 G_N-Coset Codes

We start by studying a broader class of linear block codes that contain polar codes. For these codes the block length is a power of two ($N = 2^n$) and the encoding ($\mathbf{u} \mapsto \mathbf{x}$) is done as:

$$\mathbf{x} = \mathbf{G}_N \mathbf{u} \tag{1.24}$$

where $\mathbf{G}_N$ is the linear transform relating the input-output of n-level polar transform as we saw in Section 1.3.

Let $\mathcal{A}$ be an arbitrary subset of $\{0, 1, \cdots, N-1\}$ and define the "complementary" set $\mathcal{A}^C \triangleq \{0, 1, \cdots, N-1\} \backslash \mathcal{A}$. (1.24) can then be rewritten as:

$$\mathbf{x} = \mathbf{G}_N[\mathcal{A}]\mathbf{u}_{\mathcal{A}} + \mathbf{G}_N[\mathcal{A}^C]\mathbf{u}_{\mathcal{A}^C} \tag{1.25}$$

where by $\mathbf{G}_N[\mathcal{A}]$ we mean the sub-matrix formed by keeping the columns of $\mathbf{G}_N$ whose indices are in $\mathcal{A}$.

By fixing $\mathcal{A}$ and $\mathbf{u}_{\mathcal{A}^c}$ we obtain a mapping from the binary vectors $\mathbf{u}_{\mathcal{A}}$ of length $K \triangleq |\mathcal{A}|$ to codewords of length N ($\mathbf{x}$). This mapping is called $\mathbf{G}_N$*-coset-code*.

Clearly the rate of this code is $R = \frac{K}{N}$. $\mathcal{A}$ is referred to as *information set* and the bits $\mathbf{u}_{\mathcal{A}^C} \in \mathbb{F}_2^{N-K}$ are called *frozen bits.*

A particular $\mathbf{G}_N$-coset-code is identified by set of parameters $(N, K, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C})$.

1.4.2 Successive Cancellation Decoding

Given a $\mathbf{G}_N$-coset code with parameters $(N, K, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C})$, a message (corresponding to the binary vector $\mathbf{u}_{\mathcal{A}}$) can be encoded to the vector $\mathbf{x}$ as we just described.

This codeword is transmitted through the channel and the receiver sees the corresponding channel output vector $\mathbf{y}$ which is related to the sent codeword $\mathbf{x}$ by channel transition probability $W^N(\mathbf{y}|\mathbf{x})$. The task of decoder is now to reproduce the sent codeword $\mathbf{x}$ given the code parameters. Due to the one-to-one correspondence between $\mathbf{x}$ and $\mathbf{u}$, the task is equivalent to reproducing a correct estimation of $\mathbf{u}$, which we denote by $\hat{\mathbf{u}}$. Moreover, since the decoder knows $\mathbf{u}_{\mathcal{A}^C}$, the problem will be reduced to estimating $\mathbf{u}_{\mathcal{A}}$.

In this regard, a *Successive Cancellation* (SC) decoder can be proposed as in Algorithm 1.

The decoder can be mathematically formalized in the following equation:

$$\hat{u}_i(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}) = \begin{cases} u_i & \text{if } i \in \mathcal{A}^C, \\ h_i(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}) & \text{otherwise} \end{cases} \tag{1.26}$$

where $h_i : \mathcal{Y}^N \times \mathcal{X}^i \longrightarrow \mathcal{X}$ is defined as:

$$h_i(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}) = \begin{cases} 0 & \text{if } \frac{W_N^{(i)}(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}|0)}{W_N^{(i)}(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}|1)} \geqslant 1, \\ 1 & \text{otherwise.} \end{cases} \tag{1.27}$$

Algorithm 1 Successive Cancellation Decoding

```
Input: Channel output $\mathbf{y}$, $(N, K, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C})$.
Output: $\hat{\mathbf{u}}$ an estimation of $\mathbf{u}$.
  for all $i \in \{0, 1, \cdots, N-1\}$ do
    if $i \in \mathcal{A}^C$ then
      $\hat{u}_i \leftarrow u_i$
    else
      if $\frac{W_N^{(i)}(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}|0)}{W_N^{(i)}(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}|1)} \geqslant 1$ then
        $\hat{u}_i \leftarrow 0$
      else
        $\hat{u}_i \leftarrow 1$
      end if
    end if
  end for
  return $\hat{\mathbf{u}}$
```

The SC decoder, resembles the maximum likelihood decoding with the difference that in the decisions (1.27) *estimations* of channel-output are used. Indeed, the ith decision is ML decision iff all decisions before that are correct (which includes the trivial case that all bits before i are frozen).

Theorem 1.3 (Performance of SC Decoder)**.** *Let's denote $P_w(N, K, \mathcal{A})$ the average word-error probability of SC decoding averaged over the ensemble of the 2^{N-K} different $\mathbf{G}_N$-coset codes generated by all possible choices of $\mathbf{u}_{\mathcal{A}^C}$ and over all possible sent codewords with uniform priors on both frozen bits and sent codeword. Then*

$$P_w(N, K, \mathcal{A}) \leqslant \sum_{i \in \mathcal{A}} Z\left(W_N^{(i)}\right) \tag{1.28}$$

Proof. Note that $P_w(N, K, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C})$, the word error probability of the specific $\mathbf{G}_N$-coset code with parameters $(N, K, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C})$ is defined as:

$$P_w(N, K, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C}) = \sum_{\mathbf{u}_{\mathcal{A}} \in \mathcal{X}^K} \frac{1}{2^K} \sum_{\mathbf{y} \in \mathcal{Y}^N : \hat{\mathbf{u}} \neq \mathbf{u}} W_N(\mathbf{y}|\mathbf{u}) \tag{1.29}$$

and hence $P_w(N, K, \mathcal{A})$ is:

$$P_w(N, K, \mathcal{A}) = \sum_{\mathbf{u}_{\mathcal{A}^C} \in \mathcal{X}^{N-K}} \frac{1}{2^{N-K}} P_w(N, K, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C}). \tag{1.30}$$

If both $\mathbf{u}_{\mathcal{A}}$ and $\mathbf{u}_{\mathcal{A}^C}$ are chosen uniformly in prior, the probability assignment $\mathbb{P}[\cdot]$ on the probability space $\left(\mathcal{X}^N \times \mathcal{Y}^N, \mathbb{P}\right)$ will be:

$$\mathbb{P}[\{(\mathbf{u}, \mathbf{y})\}] = 2^{-N} W_N(\mathbf{y}|\mathbf{u}). \tag{1.31}$$

Hence, combining (1.29) and (1.30) we can deduce that:

$$P_w(N, K, \mathcal{A}) = \mathbb{P}[\mathcal{E}] \tag{1.32}$$

where

$$\mathcal{E} = \left\{(\mathbf{u}, \mathbf{y}) \in \mathcal{X}^N \times \mathcal{Y}^N : \hat{\mathbf{u}}_{\mathcal{A}} \neq \mathbf{u}_{\mathcal{A}}\right\} \tag{1.33}$$

Now if we define $\mathcal{B}_i$ as the event that the first decision error occurs at bit i, namely:

$$\mathcal{B}_i = \left\{(\mathbf{u}, \mathbf{y}) \in \mathcal{X}^N \times \mathcal{Y}^N : \mathbf{u}_0^{i-1} = \hat{\mathbf{u}}_0^{i-1}, u_i \neq \hat{u}_i(\mathbf{y}, \mathbf{u}_0^{i-1})\right\}, \tag{1.34}$$

we can see that $\mathcal{E} = \bigcup_{i \in \mathcal{A}} \mathcal{B}_i$.

Next, we observe that:

$$\begin{aligned}\mathcal{B}_i &\subset \left\{(\mathbf{u}, \mathbf{y}) \in \mathcal{X}^N \times \mathcal{Y}^N : u_i \neq h_i(\mathbf{y}, \mathbf{u}_0^{i-1})\right\} \\ &\subset \left\{(\mathbf{u}, \mathbf{y}) \in \mathcal{X}^N \times \mathcal{Y}^N : W_N^{(i)}(\mathbf{y}, \mathbf{u}_0^{i-1}|u_i) \leqslant W_N^{(i)}(\mathbf{y}, \mathbf{u}_0^{i-1}|\bar{u}_i)\right\} \triangleq \mathcal{E}_i.\end{aligned} \tag{1.35}$$

$\mathcal{E}_i$ is the event that ML decoding of a single bit sent through channel $W_N^{(i)}$ is decoded wrongly (assuming the ties are broken in favor of $\bar{u}_i$ in the decision function of (1.27)). Note that the second inclusion follows from the fact that we don't know in which direction the ties are broken in the decision function. The probability of $\mathcal{E}_i$ is then upper bounded by the Bhattacharyya parameter of that channel:

$$\mathbb{P}\left[\mathcal{E}_i\right] = \sum_{(\mathbf{u}, \mathbf{y}) \in \mathcal{X}^N \times \mathcal{Y}^N} \frac{1}{2^N} W_N(\mathbf{y}|\mathbf{u}) \mathbb{1}_{\mathcal{E}_i}(\mathbf{u}, \mathbf{y}) \tag{1.36}$$

$$\leqslant \sum_{(\mathbf{u}, \mathbf{y}) \in \mathcal{X}^N \mathcal{Y}^N} \frac{1}{2^N} W_N(\mathbf{y}|\mathbf{u}) \sqrt{\frac{W_N^{(i)}(\mathbf{y}, \mathbf{u}_0^{i-1}|\bar{u}_i)}{W_N^{(i)}(\mathbf{y}, \mathbf{u}_0^{i-1}|u_i)}} \leqslant Z(W_N^{(i)}). \tag{1.37}$$

Note that in a special case of a BEC, the event $\mathcal{E}_i$ only includes the erasures (i.e. the cases that $W_N^{(i)}(\mathbf{y}, \mathbf{u}_0^{i-1}|u_i) = W_N^{(i)}(\mathbf{y}, \mathbf{u}_0^{i-1}|\bar{u}_i)$) and the last inequality reduces to equality.

The claim follows considering the definition of $\mathcal{E}$ as the union of $\mathcal{B}_i$s, inclusion of (1.35), and the bound of (1.37). □

Corollary 1.2. *For each given $(N, K, \mathcal{A})$ there exists at least one frozen-bit vector $\mathbf{u}_{\mathcal{A}^C}$ such that the word-error probability of the $\mathbf{G}_N$-coset code defined by parameters $(N, K, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C})$ satisfies:*

$$P_w(N, K, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C}) \leqslant \sum_{i \in \mathcal{A}} Z\left(W_N^{(i)}\right) \tag{1.38}$$

Lemma 1.7. *For a symmetric B-DMC W the events $\mathcal{E}_i$ defined in* (1.35) *has the property that:*

$$(\mathbf{u}, \mathbf{y}) \in \mathcal{E}_i \iff (\mathbf{a} + \mathbf{u}, \pi_{\mathbf{G}_N \mathbf{a}}(\mathbf{y})) \in \mathcal{E}_i \tag{1.39}$$

for any $i = 0, 1, \cdots, N-1$.

Proof. The proof follows from the symmetry of channels $W_N^{(i)}$ in the sense of (1.23) and the definition of $\mathcal{E}_i$. □

Corollary 1.3. *For a symmetric B-DMC W, the events $\mathcal{E}_i$ are independent of the choice of encoder input vector $\mathbf{u}$. In the sense that $\mathbb{P}\left[\mathcal{E}_i|\{\mathbf{U} = \mathbf{u}\}\right] = \mathbb{P}\left[\mathcal{E}_i\right]$*

Proof. Let $\mathbf{x} = \mathbf{G}_n\mathbf{x}$,

$$\begin{aligned}\mathbb{P}\left[\mathcal{E}_i|\{\mathbf{U} = \mathbf{u}\}\right] &= \sum_{\mathbf{y}\in\mathcal{Y}^N} W_N(\mathbf{y}|\mathbf{u})\mathbb{1}_{\mathcal{E}_i}(\mathbf{u},\mathbf{y}) \\ &= \sum_{\mathbf{y}\in\mathcal{Y}^N} W_N(\pi_{\mathbf{x}}(\mathbf{y})|\mathbf{0})\mathbb{1}_{\mathcal{E}_i}(\mathbf{0},\pi_{\mathbf{x}}(\mathbf{y})) = \mathbb{P}\left[\mathcal{E}_i|\{\mathbf{U} = \mathbf{0}\}\right]\end{aligned}$$

□

Corollary 1.4. *For a symmetric B-DMC W, every $\mathbf{G}_N$-coset code with parameters $(N, K, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C})$ satisfies:*

$$P_w(N, K, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C}) \leqslant \sum_{i\in\mathcal{A}} Z\left(W_N^{(i)}\right) \tag{1.40}$$

Corollary 1.4 implies the explicit construction of G_N-coset-codes in general and polar codes in particular for symmetric B-DMCs. Indeed, we see that all we need is to have an appropriate choice of information bits (about which we will discuss shortly) and for any choice of frozen bits (for example the simple all-zero sequence) the desired performance is achievable.

1.4.3 Polar Code Design Rule

As we mentioned previously, polar codes are specific members of the class of $\mathbf{G}_N$-coset codes. The result of Theorem 1.3 and Corollary 1.2 suggest that the optimal performance of $\mathbf{G}_N$-coset codes under SC decoding can be achieved if:

1. $\mathcal{A}$ is chosen such that the upper bound of (1.28) is minimized and
2. $\mathbf{u}_{\mathcal{A}}$ is found such that the individual code performance satisfies the upper bound (in Corollary 1.2 it is suggested that such a frozen-bit vector exists).

Choosing $\mathcal{A}$ such that the first condition is satisfied is called *polar coding rule.*

Definition 1.8 (Ensemble of Polar Codes)**.** For a fixed B-DMC W and a rate $1 \geqslant R \geqslant 0$, *Ensemble of Polar Codes* are the sub class of $\mathbf{G}_N$-coset codes with parameters $(N, \lfloor NR \rfloor, \mathcal{A})$ where $\mathcal{A}$ is chosen according to the polar coding rule (such that (1.28) is minimized).

Throughout the rest of this chapter we denote the word-error probability of these codes under SC decoding as $P_w(N, R)$ which is the average of word-error probability of individual polar codes over the choice of frozen-bits $\mathbf{u}_{\mathcal{A}^C}$.

1.5 Performance And Complexity of Polar Coding

So far, it should be clear that how polar coding can be exploited to design capacity achieving codes. Indeed, Corollary 1.1 suggests that for any rate (below the symmetric capacity of the B-DMC W) there exits a subset of $\{0, 1, \cdots, N-1\}$ that contains "good" channels. This is the set that is chosen by polar coding rule and yields the capacity achieving codes.

Theorem 1.4 (Performance of Polar Codes)**.** *For any given B-DMC W and fixed $R < I(W)$, the word-error probability of the ensemble of polar codes (averaged over the choice of frozen bits) satisfies:*

$$P_w(N, R) = o\left(2^{-N^{\beta}}\right), \tag{1.41}$$

for any $\beta < \frac{1}{2}$.

Proof. The proof follows by combining the results of Corollary 1.1 (and observing that the set suggested there is the set chosen by polar coding rule as the information bits set) and Theorem 1.3. □

Remark. Observe that polar coding rule and consequently the achieved performance are essentially based on the structure of decoder . It is possible to have different decoders and alternative code design rules accordingly. For example in [3] the authors have shown that for MAP decoding, the Reed-Muller codes (other members of the $\mathbf{G}_N$-coset-codes) are optimal codes.

Theorem 1.5 (Polar Codes on Symmetric Channels)**.** *For any* symmetric *B-DMC W, and fixed $R < I(W)$, the word-error probability of any $\mathbf{G}_N$-coset code for which the information bits set $\mathcal{A}$ is chosen according to polar coding rule satisfies:*

$$P_w(N, \lceil NR \rceil, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C}) = o(2^{-N^{\beta}}) \tag{1.42}$$

for any fixed $\beta < \frac{1}{2}$ and $\mathbf{u}_{\mathcal{A}^C}$ fixed arbitrarily.

Proof. The proof follows from Corollary 1.4 which states that the bound of Theorem 1.3 is independent of the choice of frozen bits. So the proof of Theorem 1.4 is valid independent of the choice of frozen bits and the claim follows. □

Remark. Even though the mentioned theorems suggest that word-error rate of polar codes decrease exponentially with block length for large enough block lengths, in practice, they show relatively poor performance at rates very close to the capacity for low to moderate block lengths. This phenomena is due to existence of some unpolarized channels that will be included in the information bits set as the rate increases. In [4] the authors have analyzed the behavior of unpolarized channels. We will see in the following chapters that there exists methods to mitigate the effect of unpolarized channels and improve the performance of SC decoder.

So far we have seen that polar codes can achieve the symmetric capacity of any B-DMC under successive cancellation decoding. For the special class of symmetric B-DMCs the symmetric capacity is equal to the Shannon capacity and moreover the capacity-achieving property will be independent of the choice of frozen bits. Now we would like to show that the complexity of both encoding and decoding is relatively low.

1.5.1 Encoding Complexity

To analyze the complexity of encoding we shall start by deriving formulae for the generator matrix $\mathbf{G}_N$. From Figure 1.4 we can see that $\mathbf{G}_N$ can be written as:

$$\mathbf{G}_N = \left(\mathbf{I}_2 \otimes \mathbf{G}_{N/2}\right) \mathbf{R}_N \left(\mathbf{I}_{N/2} \otimes \mathbf{F}\right) \tag{1.43}$$

where $\mathbf{F} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$ and $\mathbf{R}_N$ is the reverse shuffle operator we described earlier.

In [1] it has been shown that by some simple manipulations $\mathbf{G}_N$ can be written also as:

$$\mathbf{G}_N = \mathbf{B}_N \mathbf{F}^{\otimes n} \tag{1.44}$$

where $\mathbf{B}_N$ is the bit-reversal permutation defined as follows: Assume $\mathbf{v} = \mathbf{B}_N \mathbf{u}$. Then $v_i = u_{\mathtt{bit-reversal}_N(i)}$ for all $i = 0, 1, \cdots, N-1$, where the bit-reversal operation is defined below.

Definition 1.9 (Bit-Reversal Operation)**.** For $N = 2^n$ and $0 \leqslant i \leqslant N-1$, let $\overline{b_{n-1}, \cdots, b_1, b_0}$ be the binary expansion of i, namely:

$$i = \sum_{j=0}^{n-1} b_j 2^j. \tag{1.45}$$

Then the size-N bit-reversal operation, denoted by $\mathtt{bit} - \mathtt{reversal}_N\,(i)$ defined as:

$$\mathtt{bit} - \mathtt{reversal}_N\,(i) = \sum_{j=0}^{n-1} b_{n-1-j} 2^j \tag{1.46}$$

Note that $\mathtt{bit} - \mathtt{reversal}_N\,(\mathtt{bit} - \mathtt{reversal}_N\,(i)) = i$ and $\mathbf{B}_N^{-1} = \mathbf{B}_N$.

Using the notion of bit-reversal permutation, we can derive the recursive formulae for $\mathbf{G}_N$ as follows:

$$\mathbf{G}_N = \mathbf{B}_N \mathbf{F}_N, \tag{1.47}$$

$$\mathbf{F}_N = \mathbf{F}_2 \otimes \mathbf{F}_{N/2}, \tag{1.48}$$

$$\mathbf{F}_2 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \tag{1.49}$$

Theorem 1.6 (Encoding Complexity of $\mathbf{G}_N$-coset codes)**.** *For any $\mathbf{G}_N$-coset block code the time complexity of encoding is*

$$\chi_{T,E}(N) = O(N \log N). \tag{1.50}$$

Moreover the encoding can be done using either

$$\chi_{S,E}(N) = O(N \log N) \tag{1.51}$$

or

$$\chi_{S,E}(N) = O(N) \tag{1.52}$$

memory elements.

Proof. Let's ignore the bit-reversal operation for the moment and compute the complexity of $\mathbf{F}_N \mathbf{u}$ which we denote by $\tilde{\chi}_T(N)$. Computation of $\mathbf{F}_N \mathbf{u}$ reduces to $\frac{N}{2}$ binary additions and two-times computing $\mathbf{F}_{N/2}$. Assuming computation of binary addition takes unit time:

$$\tilde{\chi}_T(N) = \frac{N}{2} + 2\tilde{\chi}_T\left(\frac{N}{2}\right)$$

Starting with $\tilde{\chi}_T(2) = 2$, we end up with $\tilde{\chi}_T(N) = O(N \log N)$. Now, observe that the bit-reversal permutation has the worst case complexity of $O(N)$ operations. Hence the claim on the time complexity follows.

As for the space complexity, observe that for computing $\mathbf{F}_N\mathbf{u}$ we need to compute two length $N/2$ vectors $\mathbf{F}_{N/2}\mathbf{u}_0^{N/2-1}$ and $\mathbf{F}_{N/2}\mathbf{u}_{N/2}^{N-1}$, each of which in turn requires 2 vectors of length $N/4$ and so on. Hence we need to store $N \log N$ elements in total. This yields the first claim on space complexity.

However, it is notable that by the time of computing $\mathbf{F}_N\mathbf{u}$ we *only* need the results of computation from the previous step, namely two length $N/2$ vectors and the results of computations by $\mathbf{F}_{N/4}$ are unnecessary. Hence, in practice at each time instant only two length N vectors must be stored. So, we can implement the encoder using only $2N$ memory elements which proves the second claim. □

1.5.2 Decoding Complexity

For any binary hypothesis, the likelihood ratios (defined below) are sufficient statistics of channel output.

$$\Lambda(y) \triangleq \frac{W(y|0)}{W(y|1)} \tag{1.53}$$

In the SC decoding, the decision function of (1.27) will be based on the value of

$$\Lambda_N^{(i)}(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}) \triangleq \frac{W_N^{(i)}(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}|0)}{W_N^{(i)}(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}|1)} \tag{1.54}$$

Since each pair of polarized channels at level n are obtained by applying the single step polar transform on two copies of a channel at level $n-1$ (namely, $\left(W_{N/2}^{(i)}, W_{N/2}^{(i)}\right) \mapsto \left(W_N^{(2i)}, W_N^{(2i+1)}\right)$) each pair of likelihood ratios at level n can be computed recursively from the likelihood ratios at level $n-1$. Considering the channel combination formulae of (1.15a) and (1.15b) we can obtain the recursive formulae for computing the likelihood ratios needed for the SC decoder as follows:

$$\Lambda_N^{(2i)}\left(\mathbf{y}, \hat{\mathbf{u}}_0^{2i-1}\right) = \frac{\Lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1}, \hat{\mathbf{u}}_{0,o}^{2i-1} + \hat{\mathbf{u}}_{0,e}^{2i-1}\right)\Lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1}, \hat{\mathbf{u}}_{0,o}^{2i-1}\right) + 1}{\Lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1}, \hat{\mathbf{u}}_{0,o}^{2i-1} + \hat{\mathbf{u}}_{0,e}^{2i-1}\right) + \Lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1}, \hat{\mathbf{u}}_{0,o}^{2i-1}\right)} \tag{1.55a}$$

$$\Lambda_N^{(2i+1)}\left(\mathbf{y}, \hat{\mathbf{u}}_0^{2i}\right) = \begin{cases} \Lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1}, \hat{\mathbf{u}}_{0,o}^{2i-1} + \hat{\mathbf{u}}_{0,e}^{2i-1}\right)\Lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1}, \hat{\mathbf{u}}_{0,o}^{2i-1}\right) & \text{if } \hat{u}_{2i} = 0, \\ \dfrac{\Lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1}, \hat{\mathbf{u}}_{0,o}^{2i-1}\right)}{\Lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1}, \hat{\mathbf{u}}_{0,o}^{2i-1} + \hat{\mathbf{u}}_{0,e}^{2i-1}\right)} & \text{if } \hat{u}_{2i} = 1 \end{cases} \tag{1.55b}$$

The recursive relationships, break each likelihood ratio calculation at length N (which are called *decision-level likelihood ratios*) to two likelihood ratio calculations at length $N/2$ each of which broken into two likelihood ratio calculations at length $N/4$. This process continues until we reach the length 1 calculations which are *channel-level likelihood ratios*:

$$\Lambda_1^{(0)}(y) = \frac{W(y|0)}{W(y|1)} \tag{1.56}$$

The first approach to implement the SC decoder is to compute the likelihood ratios for each information bit using the recursive formulae we just obtained separately. This is feasible but as shown in [1] the complexity of this approach for likelihood ratios calculation will be $O(N^2)$.

However, looking carefully at (1.55a) and (1.55b) we can see that for computing each pair of likelihood ratios at level n,

$$\left(\Lambda_N^{(2i)}\left(\mathbf{y}, \hat{\mathbf{u}}_0^{2i-1}\right), \Lambda_N^{(2i+1)}\left(\mathbf{y}, \hat{\mathbf{u}}_0^{2i}\right)\right),$$

two identical likelihood ratios

$$\left(\Lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1}, \hat{\mathbf{u}}_{0,o}^{2i-1} + \hat{\mathbf{u}}_{0,e}^{2i-1}\right), \Lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1}, \hat{\mathbf{u}}_{0,o}^{2i-1}\right)\right)$$

at level $n-1$ (corresponding to length $N/2$) are required.

Consequently, to compute *all* N likelihood ratios at level n,

$$\left(\Lambda_N^{(2i)}\left(\mathbf{y}, \hat{\mathbf{u}}_0^{2i-1}\right), \Lambda_N^{(2i+1)}\left(\mathbf{y}, \hat{\mathbf{u}}_0^{2i}\right)\right), \qquad i = 0, 1, \cdots, N/2-1,$$

N likelihood ratios at level $n-1$ should be computed:

$$\begin{aligned} \Lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1}, \hat{\mathbf{u}}_{0,o}^{2i-1} + \hat{\mathbf{u}}_{0,e}^{2i-1}\right), &\qquad i = 0, 1, \cdots, N/2-1. \\ \Lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1}, \hat{\mathbf{u}}_{0,o}^{2i-1}\right), &\qquad i = 0, 1, \cdots, N/2-1, \end{aligned}$$

Continuing the argument in the similar fashion, we conclude that in total $N(1+\log N)$ likelihood ratio calculations should be carried out in order to have *all* of the likelihood ratios at level n. Clearly some of them that correspond to frozen bits are not used for decisions. This argument shows that sharing the intermediate results of likelihood ratios between the decision functions can reduce the complexity of the decoder.

Based on what we have seen, the SC decoder will have a scratch-pad of likelihood ratios that are computed progressively as the decoding algorithm proceeds in the following way: The decoder starts from the first bit of $\mathbf{u}$, $\hat{u}_0$. Regardless of the fact that this bit is frozen or not, it attempts to compute $\Lambda_N^{(0)}(\mathbf{y})$ which in turn breaks into computation of $\Lambda_{N/2}^{(0)}(\mathbf{y}_0^{N/2-1})$ and $\Lambda_{N/2}^{(0)}(\mathbf{y}_{N/2}^{N-1})$. Each of those computations break into two computations at length $N/4$ and this continues until it reaches into $\Lambda_1^{(0)}(y_i), \quad i = 0, 1, \cdots, N-1$ which are fed into the decoder as sufficient statistics of the channel outputs. Now the decoder can decide on the value of the first bit or set it from the known frozen bit vector (if the bit is frozen) and proceed to the second bit. For the second bit, the decoder has to compute $\Lambda_N^{(1)}(\mathbf{y}, \hat{u}_0)$ which is again a function of $\Lambda_{N/2}^{(0)}(\mathbf{y}_0^{N/2-1})$, $\Lambda_{N/2}^{(0)}(\mathbf{y}_{N/2}^{N-1})$, and $\hat{u}_0$. However since the two likelihood ratios exist in the scratch-pad, only one computation using the (1.55b) is required to compute the likelihood ratio of the second bit. The decoder similarly continues to compute the likelihood ratios of the third bit, for which part of the required likelihood ratios at lower levels are already computed.

Theorem 1.7 (Complexity of SC Decoding)**.** *The time complexity of SC decoding is*

$$\chi_{T,D}(N) = O(N \log N). \tag{1.57}$$

Moreover, the decoder can be implemented using

$$\chi_{S,D}(N) = O(N \log N) \tag{1.58}$$

or

$$\chi_{S,D}(N) = O(N) \tag{1.59}$$

memory elements.

Proof. The discussion we had so far shows the claim about time complexity (assuming computation of each likelihood ratio using either (1.55a) or (1.55b) takes unit time). Similarly we need $N \log(N) + N$ memory elements to store the likelihood ratio scratch-pad which proves the first claim on space complexity.

The second claim on the space complexity follows from the "space-efficient" implementation of SC decoder proposed in [5]. $\square$

Technical Notes on the Implementation of SC Decoder

In an operational system, some subtle details should be considered in implementing the decoder due to the quantization of computation results and finite precision of calculations. Moreover, it is always desirable to simplify the arithmetic operations required in the algorithm as much as possible.

In practice, it is easier to compute the log-likelihood ratios given the channel output which is defined as:

$$\lambda(y) = \log \Lambda(y) = \log \frac{W(y|0)}{W(y|1)} \tag{1.60}$$

Additionally, working with log-likelihood ratios has the advantage that it takes values in $\mathbb{R}$ (unlike likelihood ratio that takes only positive real values) which exploits the entire range of the values that can be stored in a machine.

Using the log-likelihood ratios, we can rewrite the combination formulae (1.55a) and (1.55b) as:

$$\lambda_N^{(2i)}\left(\mathbf{y}, \hat{\mathbf{u}}_0^{2i-1}\right) = \log \frac{\exp\left[\lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1}, \hat{\mathbf{u}}_{0,o}^{2i-1} + \hat{\mathbf{u}}_{0,e}^{2i-1}\right) + \lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1}, \hat{\mathbf{u}}_{0,o}^{2i-1}\right)\right] + 1}{\exp\left[\lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1}, \hat{\mathbf{u}}_{0,o}^{2i-1} + \hat{\mathbf{u}}_{0,e}^{2i-1}\right)\right] + \exp\left[\lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1}, \hat{\mathbf{u}}_{0,o}^{2i-1}\right)\right]} \tag{1.61a}$$

$$\lambda_N^{(2i+1)}\left(\mathbf{y}, \hat{\mathbf{u}}_0^{2i}\right) = \begin{cases} \lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1}, \hat{\mathbf{u}}_{0,o}^{2i-1}\right) + \lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1}, \hat{\mathbf{u}}_{0,o}^{2i-1} + \hat{\mathbf{u}}_{0,e}^{2i-1}\right) & \text{if } \hat{u}_{2i} = 0, \\ \lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1}, \hat{\mathbf{u}}_{0,o}^{2i-1}\right) - \lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1}, \hat{\mathbf{u}}_{0,o}^{2i-1} + \hat{\mathbf{u}}_{0,e}^{2i-1}\right) & \text{if } \hat{u}_{2i} = 1 \end{cases} \tag{1.61b}$$

Equation (1.55a) is known as arc-tangent addition. Since if we define

$$\alpha = \tan^{-1} \Lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1}, \hat{\mathbf{u}}_{0,o}^{2i-1} + \hat{\mathbf{u}}_{0,e}^{2i-1}\right)$$

and

$$\beta = \tan^{-1} \Lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1}, \hat{\mathbf{u}}_{0,o}^{2i-1}\right)$$

the expression is indeed computing $\tan(\alpha+\beta)$. This formula is encountered in other decoding algorithms (for example the Belief Propagation algorithms) and as we see its log-domain version (1.61a) will require very complicated arithmetic operations which are never affordable in a real system. Hence, its is usually approximated by the following lemma:

Lemma 1.8. *The results of combination formula* (1.61a) *can be approximated as:*

$$\begin{aligned}\lambda_N^{(2i)}\left(\mathbf{y},\hat{\mathbf{u}}_0^{2i-1}\right) \approx \mathit{sign}\left(\lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1},\hat{\mathbf{u}}_{0,o}^{2i-1}+\hat{\mathbf{u}}_{0,e}^{2i-1}\right)\lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1},\hat{\mathbf{u}}_{0,o}^{2i-1}\right)\right)&\\ \min\left\{\left|\lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1},\hat{\mathbf{u}}_{0,o}^{2i-1}+\hat{\mathbf{u}}_{0,e}^{2i-1}\right)\right|,\left|\lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1},\hat{\mathbf{u}}_{0,o}^{2i-1}\right)\right|\right\}& \quad (1.62)\end{aligned}$$

Proof. Let $\lambda=\lambda_N^{(2i)}\left(\mathbf{y},\hat{\mathbf{u}}_0^{2i-1}\right)$, $\lambda_1=\lambda_{N/2}^{(i)}\left(\mathbf{y}_0^{N/2-1},\hat{\mathbf{u}}_{0,o}^{2i-1}+\hat{\mathbf{u}}_{0,e}^{2i-1}\right)$ and $\lambda_2=\lambda_{N/2}^{(i)}\left(\mathbf{y}_{N/2}^{N-1},\hat{\mathbf{u}}_{0,o}^{2i-1}\right)$. We have

$$\lambda=\log\frac{e^{\lambda_1+\lambda_2}+1}{e^{\lambda_1}+e^{\lambda_2}}.$$

Now observe that changing the sign of either λ_1 or λ_2 changes the sign of λ hence:

$$\lambda=\operatorname{sign}\left(\lambda_1\lambda_2\right)\log\frac{e^{|\lambda_1|+|\lambda_2|}+1}{e^{|\lambda_1|}+e^{|\lambda_2|}}.$$

Assuming $|\lambda_1|\leqslant|\lambda_2|$ we have:

$$\begin{aligned}\lambda&=\operatorname{sign}\left(\lambda_1\lambda_2\right)\log\frac{e^{|\lambda_1|}+e^{-|\lambda_2|}}{e^{|\lambda_1|-|\lambda_2|}+1}\\&=\operatorname{sign}\left(\lambda_1\lambda_2\right)\left(|\lambda_1|+\log\frac{1+e^{-(|\lambda_2|+|\lambda_1|)}}{1+e^{-(|\lambda_2|-|\lambda_1|)}}\right)\end{aligned}$$

Defining $s\triangleq|\lambda_2|+|\lambda_1|$, $d\triangleq|\lambda_2|-|\lambda_1|$, $\varepsilon(s,d)\triangleq\log\frac{1+e^{-s}}{1+e^{-d}}$ we get:

$$\lambda=\operatorname{sign}\left(\lambda_1\lambda_2\right)\left(|\lambda_1|+\varepsilon(s,d)\right) \tag{1.63}$$

The extrema of $\varepsilon(s,d)$ in the region $s\geqslant d\geqslant 0$ happen at $(s,d)=(\infty,0)$ or $(s,d)=(\infty,\infty)$, bound the error term as:

$$-\log 2\leqslant\varepsilon(s,d)\leqslant 0.$$

Combining this with (1.63) we get:

$$\operatorname{sign}\left(\lambda_1\lambda_2\right)|\lambda_1|-\log 2\leqslant\lambda\leqslant\operatorname{sign}\left(\lambda_1\lambda_2\right)|\lambda_1|+\log 2 \tag{1.64}$$

which proves the claim. Indeed the approximation is off from the exact value at most by $\log 2$. Furthermore, it is notable that for a software implementation the form of (1.63) is more suitable than (1.61a) as it does not involve exponentiation of positive numbers or subtracting two large numbers. □

The approximation of Lemma 1.8 is used in real implementations of many decoders (for example Turbo Decoders) and the experiments have shown that it will not affect the performance of decoder in terms of word-error rate. Note that in the end sign of log-likelihood ratio is important for making correct decisions. This approximation shows that the decoder can be implemented with basic arithmetic operations of addition and comparison.

In the Belief Propagation decoders that have an iterative nature, it is not easy to analytically show the approximations will not change the final decisions. However, for the SC decoder, since a finite number of approximation errors add up in the decision-level log-likelihood ratios, it might be possible to prove that the approximations will not affect the performance of SC decoder. This can be an interesting topic for further studies.

1.6 Polar Code Construction

So far we have seen that for any B-DMC W, there exists a polar code of block length $N = 2^n$ which is characterized by the information bit set $\mathcal{A}$ such that $|\mathcal{A}| \leqslant NI(W)$ with exponentially small word-error rate under successive cancellation decoding.

However, the challenge is to find that code, or more precisely, the information bits set $\mathcal{A}$. Although Corollary 1.1 suggest that there exist such an information bits set, it does not specify which indices in $\{0, 1, \cdots, N-1\}$ belong to that set.

Unfortunately, there is no explicit method for computing the mutual informations (or Bhattacharyya parameters) of the polarized channels $W_N^{(i)}$ for a general B-DMC W except when W is a BEC. In this section we review the methods that are proposed for finding the indices of “good” and “bad” polarized channels.

1.6.1 Monte Carlo Estimations of Bhattacharyya Parameters

Arıkan proposed in [1] a Monte Carlo approach to estimate the Bhattacharyya parameters of the polarized channels for an arbitrary B-DMC W.

The Bhattacharyya parameter of the ith polarized channel can be rewritten as:

$$\begin{aligned} Z(W_N^{(i)}) &= \sum_{\mathbf{u}_0^{i-1} \in \mathcal{X}^i, \mathbf{y} \in \mathcal{Y}^N} \sqrt{W_N^{(i)}(\mathbf{y}, \mathbf{u}_0^{i-1}|0) W_N^{(i)}(\mathbf{y}, \mathbf{u}_0^{i-1}|1)} \\ &= \mathbb{E}\left[\sqrt{\frac{W_N^{(i)}(\mathbf{Y}, \mathbf{U}_0^{i-1}|\bar{U}_i)}{W_N^{(i)}(\mathbf{Y}, \mathbf{U}_0^{i-1}|U_i)}}\right] \end{aligned} \tag{1.65}$$

where the joint distribution of $\mathbf{Y}, \mathbf{U}$ is $\mathbb{P}[\mathbf{y}, \mathbf{u}] = 2^{-N} W_N(\mathbf{y}|\mathbf{u})$

The above expectation can be approximated by its empirical mean which can be computed using Monte Carlo methods. More precisely, we generate the samples of $(\mathbf{y}, \mathbf{u})$ from distribution $\mathbb{P}[\mathbf{y}, \mathbf{u}] = 2^{-N} W_N(\mathbf{y}|\mathbf{u})$ by transmitting the all-zero codeword through the channel and then using the SC decoder (with $\mathcal{A} = \varnothing$ and $\mathbf{u}_{\mathcal{A}^C} = \mathbf{0}$, which is a genie-aided decoder) to compute the value of likelihood ratio we see at the output of each channel as defined in (1.54).

The variable inside the expectation in (1.65) is the inverse of square root of the likelihood ratios for each channel whose mean can be approximated by the empirical average of its samples.

After we obtain the channel estimation results (for a specific codeword length, N) for any given rate we can choose the information bits accordingly and the code is constructed.

The main problem with the above estimation process is that we need a large number of samples to obtain accurate estimations of mean.

1.6.2 BEC as a Special Case

One exception among the binary symmetric channels is the binary erasure channel (BEC) for which the analytical formulae for calculating the mutual information or Bhattacharyya parameters of the polarized channels are known. Now we briefly describe the mentioned formulae and the way we implement them for estimating the polarized channels in case the B-DMC of interest is a BEC.

As Lemma 1.5 suggests, if the channel W is a BEC with erasure probability ϵ, the transformed channels obtained from two copies of W, W^- and W^+ are also BEC and their erasure probabilities are respectively:

$$\epsilon^- = 2\epsilon - \epsilon^2 \tag{1.66a}$$
$$\epsilon^+ = \epsilon^2. \tag{1.66b}$$

We use the above mentioned equations to compute the erasure probabilities of the polarized channels $W_N^{(i)}$ recursively. Starting by the original BEC, one can compute the erasure probabilities of two channels $W_2^{(0)}$ and $W_2^{(1)}$. Then starting from each of the two BECs we just obtained, we create two of the BECs at level 2 ($W_4^{(i)}, \quad i = 0, 1, 2, 3$) and so on. Recall that for a BEC the Bhattacharyya parameter is equal to its erasure probability.

However, we should be careful about the precision of arithmetics. Indeed, the equations of (1.66a) and (1.66b) are not in a suitable form for being directly implemented since the erasure probability is itself a very small number whose second power can be much smaller in turn which causes precision lost if we compute directly using the mentioned equations.

Instead we may define the one-to-one map $a : [0, 1] \longrightarrow \mathbb{R}$:

$$a(\epsilon) = \log \frac{1-\epsilon}{\epsilon} \tag{1.67}$$

which maps the erasure probability from the interval of $[0, 1]$ to whole set of real numbers and uses the double precision of machine much better.

Using the mentioned map, we can easily derive the following relationships:

$$\epsilon = \frac{1}{1+e^a} \tag{1.68}$$

$$a^- = a - \log(2 + e^{-a}) \tag{1.69a}$$
$$a^+ = a + \log(2 + e^{a}) \tag{1.69b}$$

Hence, we implement the recursions using the new parameter a instead of ϵ. A very delicate point to consider is that in the equations (1.69a) and (1.69b) we might encounter overflows since we are computing the exponential of a potentially positive number. So we can carefully rewrite the mentioned equations as:

$$a^- = \begin{cases} a - \log(2 + e^{-a}) & \text{if } a > 0 \\ 2a - \log(1 + 2e^{a}) & \text{if } a < 0 \end{cases} \tag{1.70a}$$

$$a^+ = \begin{cases} a + \log(2 + e^{a}) & \text{if } a < 0 \\ 2a + \log(1 + 2e^{-a}) & \text{if } a > 0 \end{cases} \tag{1.70b}$$

Using the recursions we just described, the Bhattacharyya parameter of each forged BEC $W_N^{(i)}, \quad i = 0, 1, \cdots, N-1$ can be computed using $\log N$ operations. However, for computing the Bhattacharyya parameter of all forged channels at level $n = \log_2 N$ the results of computations can be shared (for example computation of $Z(W_N^{(0)})$ and $Z(W_N^{(1)})$ only differ in last combinations). Consequently, the total time complexity of code construction for BEC will be $\chi_{T,C} = O(N)$ and the required space scales as $\chi_{S,C} = O(\log N)$.

As shown in Lemma 1.4, BEC has an extremal behavior in the sense that if we start by an arbitrary channel W with a certain Bhattacharyya parameter (denoted by Z) and a BEC (denoted by $\tilde{W}$) with the same Bhattacharyya parameter (and hence erasure probability) and polarize them, for any N and i we have:

$$Z\left(W_N^{(i)}\right) \leqslant Z\left(\tilde{W}_N^{(i)}\right) \tag{1.71}$$

with equality iff W is also a BEC.

We can take advantage of (1.71) for having a conservative estimate of the Bhattacharyya parameters of the polarized versions of any channel by approximating it with a BEC with the same Bhattacharyya parameter.

This approach will never misidentify the good indices because we know that if the BEC approximation of channel $\tilde{W}_N^{(i)}$ is good (which means $Z\left(\tilde{W}_N^{(i)}\right)$ is small) then the channel itself $W_N^{(i)}$ is good (has a low Bhattacharyya parameter). We are choosing the frozen bits (bad channels) conservatively in the sense that a channel $W_N^{(i)}$ can indeed be good whereas its corresponding BEC approximation $\tilde{W}_N^{(i)}$ is bad.

Asymptotically as $N \to \infty$ the fraction of the good indices for $\tilde{W}$ will be $I\left(\tilde{W}\right) = 1 - Z$ while that of the original channel W is $I\left(W\right)$ and we know for a fixed reliability (Bhattacharyya parameter) BEC has the lowest symmetric capacity ($I(\tilde{W})$) as shown in Lemma 1.1. Indeed, with this approximation, we will commit a rate-loss equal to $I(W) + Z(W) - 1$.

Nevertheless, if we don't want to employ the code at a rate very close to the capacity of channel, one can quickly determine the frozen bits using this approximation instead of waiting for the lengthy Monte Carlo simulations to estimate the correct Bhattacharyya parameters of the channel.

1.6.3 Vardy and Tal Approximation Method

The main issue with *computing* the Bhattacharyya parameter (or symmetric capacity) of the polarized channels $W_N^{(i)}$ is that their output alphabet size is huge. Recall that the output alphabet size of $W_N^{(i)}$ is generally $|\mathcal{Y}|^N|\mathcal{X}|^i$ (let the output alphabet $\mathcal{Y}$ be discrete for the moment). Consequently, one needs to compute $2|\mathcal{Y}|^N|\mathcal{X}|^i$ transition probabilities per channel in order to characterize its rate/reliability measures. This is almost impossible although (1.15a) and (1.15b) give us the recursive relationships to compute those transition probabilities.

However, as proposed in [6] it is possible to find *approximations* of the polarized channels in such a way that the output alphabet size of the approximated channels are finite. For the approximated channels any parameter of interest can be computed efficiently and as shown by the authors the approximations are indeed very accurate and enable us to construct polar codes for any underlying B-DMC W. We will briefly explain this method here.

Channel Degradation and Upgradation

The approximations we just talked about are formally obtained through *degradation* and *upgradation* defined here:

Definition 1.10 (Degraded Channel)**.** A channel $\overset{\smile}{W} : \mathcal{X} \longrightarrow \check{\mathcal{Y}}$ is *degraded* with respect to channel $W : \mathcal{X} \longrightarrow \mathcal{Y}$ if there exist a channel $Q : \mathcal{Y} \longrightarrow \check{\mathcal{Y}}$ such that for all $y \in \check{\mathcal{Y}}$ and $x \in \mathcal{X}$,

$$\overset{\smile}{W}(y|x) = \sum_{y' \in \mathcal{Y}} W(y'|x)Q(y|y'). \tag{1.72}$$

We write $\overset{\smile}{W} \preceq W$ to denote $\overset{\smile}{W}$ is degraded with respect to W.

Definition 1.11 (Upgraded Channel)**.** A channel $\widehat{W} : \mathcal{X} \longrightarrow \hat{\mathcal{Y}}$ is *upgraded* with respect to channel $W : \mathcal{X} \longrightarrow \mathcal{Y}$ if there exist a channel $Q : \hat{\mathcal{Y}} \longrightarrow \mathcal{Y}$ such that for all $y \in \hat{\mathcal{Y}}$ and $x \in \mathcal{X}$,

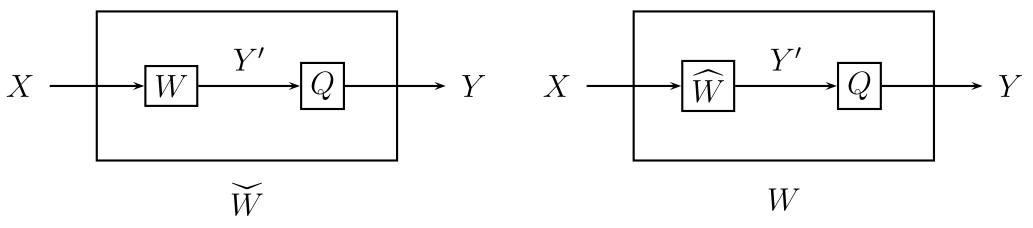
$$W(y|x) = \sum_{y' \in \hat{\mathcal{Y}}} \widehat{W}(y'|x)Q(y|y'). \tag{1.73}$$

We write $\widehat{W} \succeq W$ to denote $\widehat{W}$ is upgraded with respect to W.

By definition (and as shown in Figure 1.5):

$$\overset{\smile}{W} \preceq W \iff W \succeq \overset{\smile}{W}. \tag{1.74}$$

Moreover, a channel is both degraded and upgraded with respect to itself (taking Q as the identity function in both definitions 1.10 and 1.11). If a channel W' is both degraded and upgraded with respect to W (i.e. $W' \preceq W$ and $W \succeq W'$) then we say W is *equivalent* to W' and we write $W \equiv W'$.



(a) Degrading (b) Upgrading

Figure 1.5: Degradation and Upgradation processes

Example 1.1. Assume both W_1 and W_2 are AWGN with noise powers N_1 and N_2 respectively, if $N_1 \leqslant N_2$ then $W_1 \preceq W_2$. The reason as as follows:

$$W_1(y|x) = \frac{1}{\sqrt{2\pi N_1}} e^{-\frac{(y-x)^2}{2N_1}}$$
$$W_2(y|x) = \frac{1}{\sqrt{2\pi N_2}} e^{-\frac{(y-x)^2}{2N_2}}$$

Taking Q as another AWGN with noise power $N_2 - N_1$ proves the claim. This can be verified either by looking at Figure 1.5 and considering the fact that the overall signal path from X to Y sees a Gaussian noise with variance N_2 or the following:

$$\begin{aligned}
\int_{y' \in \mathbb{R}} Q(y|y')W_1(y'|x)dy' &= \int_{y'} \frac{1}{\sqrt{2\pi(N_2 - N_1)}} e^{-\frac{(y-y')^2}{2(N_2-N_1)}} \frac{1}{\sqrt{2\pi N_1}} e^{-\frac{(y'-x)^2}{2N_1}} dy' \\
&= \int_{y'} \frac{1}{\sqrt{2\pi N_1}} \frac{1}{\sqrt{2\pi(N_2 - N_1)}} e^{-\left[\frac{(y-x)^2}{2N_2} + \frac{(N_2 y' - (N_1 y + (N_2 - N_1)x))^2}{2(N_2-N_1)N_1 N_2}\right]} dy' \\
&= \frac{1}{\sqrt{2\pi N_2}} e^{-\frac{(y-x)^2}{2N_2}} \int_{y' \in \mathbb{R}} \frac{1}{\sqrt{2\pi \frac{N_1(N_2-N_1)}{N_1}}} e^{-\frac{\left(y' - \frac{N_1 y + (N_2-N_1)x}{N_2}\right)^2}{2\frac{N_1(N_2-N_1)}{N_2}}} dy' \\
&= \frac{1}{\sqrt{2\pi N_2}} e^{-\frac{(y-x)^2}{2N_2}} = W_2(y|x)
\end{aligned}$$

Lemma 1.9 ([6, Lemma 1]). *If $\breve{W} : \mathcal{X} \longrightarrow \breve{\mathcal{Y}}$ is degraded with respect to symmetric B-DMC $W : \mathcal{X} \longrightarrow \mathcal{Y}$, that is $\breve{W} \preceq W$, then:*

$$Z(\breve{W}) \geqslant Z(W) \text{ and } \tag{1.75}$$
$$I(\breve{W}) \leqslant I(W). \tag{1.76}$$

Similarly, if $\widehat{W} \succeq W$,

$$Z(\widehat{W}) \leqslant Z(W) \text{ and } \tag{1.77}$$
$$I(\widehat{W}) \geqslant I(W). \tag{1.78}$$

Lemma 1.9 shows that degradation (upgradation) transforms a symmetric B-DMC to a worse (better) channel in terms of rate and reliability.

Lemma 1.10 ([6, Lemma 3]). *Let $W : \mathcal{X} \longrightarrow \mathcal{Y}$ be a B-DMC and $\overset{\smile}{W}$ and $\widehat{W}$ be its degraded and upgraded versions respectively, i.e., $\overset{\smile}{W} \preceq W \preceq \widehat{W}$, then:*

$$\overset{\smile}{W}^{-} \preceq W^{-} \preceq \widehat{W}^{-} \tag{1.79a}$$

$$\overset{\smile}{W}^{+} \preceq W^{+} \preceq \widehat{W}^{+} \tag{1.79b}$$

where "plus" and "minus" versions of the channels are obtained by single-step polar transform $(W, W) \mapsto (W^{-}, W^{+})$ according to Definition 1.7.

Corollary 1.5. *Let W, $\overset{\smile}{W}$ and $\widehat{W}$ be defined as in Lemma 1.10, then the polarized channels forged from $N = 2^n$ independent copies of each of them (applying* (1.18) *and* (1.19) *to each of them separately) will satisfy:*

$$\overset{\smile}{W}_N^{(i)} \preceq W_N^{(i)} \preceq \widehat{W}_N^{(i)}, \qquad i = 0, 1, \cdots, N-1 \tag{1.80}$$

Code Construction Algorithms

The code construction method proposed in [6] is based on approximating each of the polarized channels $W_N^{(i)}$ by their upgraded and degraded versions in such a way that the upgraded/degraded versions are "easy to deal" channels. More precisely, the idea is to find the upgraded/degraded versions of each polarized channel such that the output alphabet size of them are limited to a parameter μ. Such channel can be described with a limited number of parameters and its properties (Bhattacharyya parameter, symmetric capacity, etc.) can be computed easily.

Next, as deduced from Lemma 1.9 the parameter of interest (e.g., Bhattacharyya) for the actual channel $W_N^{(i)}$ is bounded by the corresponding parameter of the degraded channel from one side and that of the upgraded channel from the other side. If this bound is tight we have good approximation for the parameters of interest and we can find the indices of "good" and "bad" polarized channels obtained from the original channel W.

Let's assume we have access to two "merging" functions as follows:

$$\overset{\smile}{W} = \texttt{degrading} - \texttt{merge}\,(W, \mu)$$

returns a degraded version of W such that the output alphabet size of $\overset{\smile}{W}$ $(\overset{\smile}{\mathcal{Y}})$ is at most μ. Similarly:

$$\widehat{W} = \texttt{upgrading} - \texttt{merge}\,(W, \mu)$$

creates an upgraded version of W such that that the output alphabet size of $\widehat{W}$ $(\widehat{\mathcal{Y}})$ is at most μ. We will describe how to obtain such merging functions later.

The approximation procedure described in Algorithm 2.

Using Lemma 1.10 at each step of the algorithm shows that the outputs of the algorithm are upgraded and degraded versions of $W_N^{(i)}$. Additionally, at each step the polar transform is applied to channels with output alphabet size at most μ. The result will be

Algorithm 2 Approximation Algorithm

```
Input: The underlying B-DMC $W$, parameter $\mu$, and an index $i \in \{0, 1, \cdots, N-1\}$.
Output: $\widetilde{W_N^{(i)}}$ and $\widehat{W_N^{(i)}}$ such that $\widetilde{W_N^{(i)}} \preceq W_N^{(i)} \preceq \widehat{W_N^{(i)}}$
  Compute $b_{n-1}, \cdots, b_1, b_0$, the $n$-bit binary expansion of $i$.
  $\breve{W} \leftarrow \texttt{degrading} - \texttt{merge}\,(W, \mu)$
  $\widehat{W} \leftarrow \texttt{upgrading} - \texttt{merge}\,(W, \mu)$
  for $j = 0, 1, 2, \cdots, n-1$ do
    if $b_j = 0$ then
      $\tilde{W}_d \leftarrow \breve{W}^-$
      $\tilde{W}_u \leftarrow \widehat{W}^-$
    else
      $\tilde{W}_d \leftarrow \breve{W}^+$
      $\tilde{W}_u \leftarrow \widehat{W}^+$
    end if
    $\breve{W} \leftarrow \texttt{degrading} - \texttt{merge}\left(\tilde{W}_d, \mu\right)$
    $\widehat{W} \leftarrow \texttt{upgrading} - \texttt{merge}\left(\tilde{W}_u, \mu\right)$
  end for
  return $(\breve{W}, \widehat{W})$
```

channels with larger output alphabet size potentially. However, the upgrading/degrading merges approximate the transformed channels with channels with alphabet size limited to μ.

Note that the approximations of this algorithm are *not* the channels we considered in Corollary 1.5. In that corollary we considered the "polarized versions of upgraded/degraded channels" while here we obtain the "upgraded/degraded versions of the polarized channels".

Having the approximations from Algorithm 2 we can construct the polar code with the rate of interest easily. For example if for some i, $\widetilde{W_N^{(i)}}$ is a "good" channel, we know that i should be in the information bits set since $W_N^{(i)}$ is indeed a better channel. Likewise, if $\widehat{W_N^{(i)}}$ is a "bad" channel, we conclude i should not be in the information bits set since the original channel $W_N^{(i)}$ is worse. In summary, if we denote $\check{\mathcal{A}}$ and $\hat{\mathcal{A}}$ the information bits sets (for a given rate R) constructed from the upgraded and degraded approximations of the polarized channels respectively we have the following inclusions:

$$\check{\mathcal{A}} \subseteq \mathcal{A} \subseteq \hat{\mathcal{A}} \tag{1.81}$$

Merging Functions

Now we shall specify the merging methods proposed in [6]. In fact, the problem of finding "optimal" merging functions is still open. Even it is not clear what is the correct definition of "optimality" in this context. Hence, we just restate the functions proposed in [6], keeping this in mind that it might be possible to find better merging functions although the proposed functions provide accurate enough code construction algorithms.

The channels of interest are Symmetric B-DMCs $W : \mathcal{X} \longrightarrow \mathcal{Y}, \mathcal{X} = \mathbb{F}_2$ and for the time being we will put two restrictions on channel output which are:

1. Channel output alphabet size $|\mathcal{Y}|$ is finite.
2. For each $y \in \mathcal{Y}$, y and $\pi_1(y)$ (described in Definition 1.4) are distinct. We will use the shorthand notation of $\pi_1(y) = \bar{y}$.

Consequently, we can assume the output alphabet of channel is $\mathcal{Y} = \{y_0, y_1, \cdots, y_{L-1}, \bar{y_0}, \bar{y_1}, \cdots, \bar{y_{L-1}}\}$ for some L. Later we will see how we can relax the mentioned conditions.

Recall that by (1.53), $\Lambda(y) = \frac{1}{\Lambda(\bar{y})}$ and we assume we have labeled the output symbols such that:

$$1 \leqslant \Lambda(y_0) \leqslant \Lambda(y_1) \leqslant \cdots \leqslant \Lambda(y_{L-1}). \tag{1.82}$$

The following lemma shows how a channel can be degraded to the one whose output alphabet size is reduced by a 2:

Lemma 1.11. *Let $W : \mathcal{X} \longrightarrow \mathcal{Y}$ a Symmetric B-DMC and y_i and y_j two arbitrary symbols in $\mathcal{Y}$. Define*

$$\breve{\mathcal{Y}} = \mathcal{Y} \backslash \{y_i, \bar{y_i}, y_j, \bar{y_j}\} \cup \{y_*, \bar{y_*}\}$$

and

$$\breve{W}(y|x) = \begin{cases} W(y|x) & \text{if } y \notin \{y_*, \bar{y_*}\}, \\ W(y_i|x) + W(y_j|x) & \text{if } y = y_*, \\ W(\bar{y_i}|x) + W(\bar{y_j}|x) & \text{if } y = \bar{y_*}. \end{cases}$$

Then $\breve{W} \preceq W$.

Proof. Take the intermediate channel Q as the channel that maps y_i and y_j to y_* (with probability 1), $\bar{y_i}$ and $\bar{y_j}$ to $\bar{y_*}$ and the rest of input symbols to themselves. □

The degrading merge function can now be constructed by successively applying the Lemma 1.11 to a channel until the resulting degraded channel has the input alphabet-size lower than the parameter μ. However, this can be done in many ways as the two alphabets y_i and y_j can be chosen arbitrarily.

If we defined "optimality" in terms of the amount of loss in symmetric capacity, we would like to find the degrading merge in such a way that the result has maximum symmetric capacity. It is shown ([6, Theorem 6]) that to construct such a degrading merge function, the alphabets we merge should be consecutive (when the symbols are ordered as in (1.82)). Hence the degrading merge function will first need to find and index i such that the degraded channel obtained by applying Lemma 1.11 has the highest possible mutual information and then apply the mentioned lemma to reduce the alphabet size by two. This procedure is repeated until the resulting degraded channel has output alphabet size less than (or equal to) μ.

Assuming the input alphabet size of W in the input of $\texttt{degrading} - \texttt{merge}\,(W, \mu)$ has alphabet size at most $2\mu^2$ (obtained by "plus" channel operation) it has been shown that the proposed degrading merge function can be implemented to run in $O(\mu^2 \log \mu)$ time.

Finding an upgrading merge function seems less natural. Merging the output symbols of a channel to get a worse one sounds logical while doing the same in order to obtain a better channel seems odd at first glance. However, it turns out to be possible as proposed in the following lemmas. Since the proofs are similar to the degrading situation and can be found in [6] we omit them.

Lemma 1.12 ([6, Lemma 7])**.** *Let $W : \mathcal{X} \longrightarrow \mathcal{Y}$ a Symmetric B-DMC and y_i and y_j two arbitrary symbols in $\mathcal{Y}$. Let $\Lambda_i = \Lambda(y_i)$ and $\Lambda_j = \Lambda(y_j)$ and assume $1 \leqslant \Lambda_i \leqslant \Lambda_j$. Let $a_i = W(y_i|0)$ and $b_i = W(\bar{y_i}|0)$ and α_j and β_j as:*

$$\begin{aligned} \alpha_j &= \Lambda_j \frac{a_i + b_i}{\Lambda_j + 1}, \\ \beta_j &= \frac{a_i + b_i}{\Lambda_j + 1}. \end{aligned}$$

Define

$$t(\alpha, \beta|x) \triangleq \begin{cases} \alpha & \text{if } x = 0 \\ \beta & \text{if } x = 1 \end{cases}$$

for $\alpha, \beta \in \mathbb{R}$ and $x \in \mathcal{X}$. Next, define:

$$\widehat{\mathcal{Y}} = \mathcal{Y} \backslash \{y_i, \bar{y_i}, y_j, \bar{y_j}\} \cup \{y_*, \bar{y_*}\}$$

and

$$\widehat{W}(y|x) = \begin{cases} W(y|x) & \text{if } y \notin \{y_*, \bar{y_*}\}, \\ W(y_i|x) + t(\alpha_j, \beta_j|x) & \text{if } y = y_*, \\ W(\bar{y_i}|x) + t(\beta_j, \alpha_j|x) & \text{if } y = \bar{y_*}. \end{cases}$$

Then $\widehat{W} \succeq W$.

It is notable that if we choose a pair of output symbols y_i, y_j such that $\Lambda(y_i) = \Lambda(y_j)$ (if there exists such a pair), both lemmas 1.12 and 1.11 result in the same channel which is indeed equivalent to the initial channel ($\widecheck{W} \equiv W \equiv \widehat{W}$) with two less output symbols. Indeed since likelihood ratios are sufficient statistics for the output of any binary channel, we don't need to have different output symbols with the same likelihood ratio.

Lemma 1.13 ([6, Lemma 9])**.** *Let $W : \mathcal{X} \longrightarrow \mathcal{Y}$ a Symmetric B-DMC and y_i, y_j and y_k three arbitrary symbols in $\mathcal{Y}$. Let $\Lambda_i = \Lambda(y_i)$, $\Lambda_j = \Lambda(y_j)$ $\Lambda_k = \Lambda(y_k)$ and assume $1 \leqslant \Lambda_i < \Lambda_j < \Lambda_k$. Let $a_j = W(y_j|0)$ and $b_j = W(\bar{y_j}|0)$ and α_i, β_i, α_k and β_k as:*

$$\begin{aligned} \alpha_i &= \Lambda_i \frac{\Lambda_k b_j - a_j}{\Lambda_k - \Lambda_i}, \\ \beta_i &= \frac{\Lambda_k b_j - a_j}{\Lambda_k - \Lambda_i}, \\ \alpha_k &= \Lambda_k \frac{a_j - \Lambda_i b_j}{\Lambda_k - \Lambda_i}, \\ \beta_k &= \frac{a_j - \Lambda_i b_j}{\Lambda_k - \Lambda_i}. \end{aligned}$$

Let $t(\alpha, \beta|x)$ be defined as in Lemma 1.12 and define:

$$\widehat{\mathcal{Y}} = \mathcal{Y} \backslash \{y_i, \bar{y_i}, y_j, \bar{y_j}, y_k, \bar{y_k}\} \cup \{y_*, \bar{y_*}, y_\star, \bar{y_\star}\}$$

and

$$\widehat{W}(y|x) = \begin{cases} W(y|x) & \text{if } y \notin \{y_*, \bar{y_*}, y_\star, \bar{y_\star}\}, \\ W(y_i|x) + t(\alpha_i, \beta_i|x) & \text{if } y = y_*, \\ W(\bar{y_i}|x) + t(\beta_i, \alpha_i|x) & \text{if } y = \bar{y_*}, \\ W(y_k|x) + t(\alpha_k, \beta_k|x) & \text{if } y = y_\star, \\ W(\bar{y_k}|x) + t(\beta_k, \alpha_k|x) & \text{if } y = \bar{y_\star}. \end{cases}$$

Then $\widehat{W} \succeq W$.

It has been shown that the channel obtained by Lemma 1.13 is degraded with respect to the one obtained from Lemma 1.12, hence it is a better upgraded version of W. However, the problem with Lemma 1.13 is that if two symbols y_i and y_k are too close in terms of their likelihood ratio ($\Lambda_i \approx \Lambda_k$) the computations will become unstable in Lemma 1.13.

Consequently, the upgrading merge function will be defined in a fashion similar to the degrading merge function as follows: First one has to check if there exists consecutive output symbols y_i and y_{i+1} such that their likelihood ratios are "too close" (for example $\frac{\Lambda(y_{i+1})}{\Lambda(y_i)} \leqslant 1 + \delta$ for some small δ). If such cases exist, we use Lemma 1.12 to merge them. This initial step is repeated until no close output symbol pairs are found. The second phase of algorithm starts by finding the index i such that merging y_i, y_{i+1}, y_{i+2} using Lemma 1.13 results in the channel with highest mutual information. Those symbols are merged and the second phase is repeated until the resulting upgraded channel has input alphabet size less than μ. This method requires $O(\mu^2 \log \mu)$ running time.

Complexity of Code Construction

As we see in Algorithm 2, each run of the algorithm, invokes $\texttt{upgrading} - \texttt{merge}\,(\cdot, \mu)$ and $\texttt{degrading} - \texttt{merge}\,(\cdot, \mu)$ $n = \log N$ times. Hence, the time complexity of each run of the algorithm is $O(\log N \mu^2 \log \mu)$.

However, for a complete code construction we need to run Algorithm 2 N times (for each of the polarized channels). We can see that the intermediate results can be shared for different channels (for example approximations of $W_N^{(0)}$ and $W_N^{(1)}$ only differ in the last round of the "for" loop in Algorithm 2). Hence, we can conclude that the time and space complexity of code construction will be:

$$\chi_{T,C} = O(\mu^2 \log \mu N), \tag{1.83}$$

$$\chi_{S,C} = O(\mu \log N) \tag{1.84}$$

which is much more efficient than time-consuming Monte Carlo simulations.

Representing B-DMCs as a Collection of BSCs

In this part, we provide a general method of representing any B-DMC as a collection of BSCs which is a very suitable form to use in the merging functions. Moreover, with this method we can maintain general code construction algorithms and reduce the problem of code construction for an arbitrary B-DMC to the problem of representing it as a collection of BSCs (each of which called a *sub-BSC*).

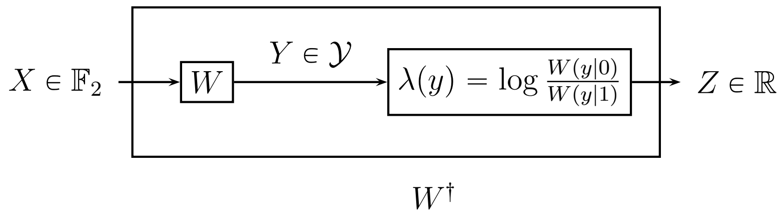


Figure 1.6: Equivalent channel obtained by considering the likelihood ratio as the output

For any B-DMC, the log-likelihood ratio of the output is a sufficient statistic for decoding. Hence, for any B-DMC $W : \mathcal{X} \longrightarrow \mathcal{Y}$ we can define an *equivalent* channel $W^\dagger : \mathcal{X} \longrightarrow \mathbb{R}$ whose output is the log-likelihood ratio of the original channel (see Figure 1.6)

However, the transition probabilities of the equivalent channel must have specific property:

Lemma 1.14. *Let $W : \mathcal{X} \longrightarrow \mathcal{Y}$ be an arbitrary B-DMC and $W^\dagger : \mathcal{X} \longrightarrow \mathbb{R}$ its equivalent as depicted in Figure 1.6. Then:*

$$W^\dagger(z|0) = e^z W^\dagger(z|1) \tag{1.85}$$

Moreover, if W is symmetric, $W^\dagger$ is also symmetric in the following sense:

$$W^\dagger(-z|1) = W^\dagger(z|0) \tag{1.86}$$

Proof. The claim is obtained by observing that the log-likelihood ratio of z is z itself. Furthermore, observe that in a symmetric channel, substituting y and $\pi_1(y)$ is equivalent to substituting z by $-z$. □

Now, lets define $\rho \triangleq |Z|$ and $S \triangleq \text{sign}(Z)$. Observe that ρ measures the reliability of the decision that the ML decoder takes. The decision boundary of ML decision is $z = 0$.

Lemma 1.15. *Let $W : \mathcal{X} \longrightarrow \mathcal{Y}$ be a symmetric B-DMC and $W^\dagger$ as in Lemma 1.14 and $\rho = |Z|$. ρ is independent of the channel input x.*

Proof.

$$\begin{aligned}\mathbb{P}[\rho = r|X = 0] &= \mathbb{P}[|Z| = r|X = 0]\\ &= \mathbb{P}[Z = r|X = 0] + \mathbb{P}[Z = -r|X = 0]\\ &= W^\dagger(r|0) + W^\dagger(-r|0)\end{aligned}$$

On the other side,

$$\begin{aligned}\mathbb{P}[\rho = r|X = 1] &= \mathbb{P}[|Z| = r|X = 1]\\ &= \mathbb{P}[Z = r|X = 1] + \mathbb{P}[Z = -r|X = 1]\\ &= W^\dagger(r|1) + W^\dagger(-r|1)\\ &= W^\dagger(-r|0) + W^\dagger(r|0)\end{aligned}$$

□

Lemma 1.16. *Let W and $W^\dagger$ be as defined in Lemma 1.15. Given $\rho = r$, the channel seen from X to S is a BSC with parameter $p(r) = \frac{1}{1+e^r}$, i.e.,*

$$\mathbb{P}[S = -1|\rho = r, X = 1] = \mathbb{P}[S = 1|\rho = r, X = 0] = 1 - p(r), \tag{1.87}$$
$$\mathbb{P}[S = 1|\rho = r, X = 1] = \mathbb{P}[S = -1|\rho = r, X = 0] = p(r). \tag{1.88}$$

Proof.

$$\begin{aligned}\mathbb{P}[S = 1|\rho = r, X = 0] &= \frac{\mathbb{P}[S = 1, \rho = r|X = 0]}{\mathbb{P}[\rho = r]} \\ &= \frac{W^\dagger(r|0)}{W^\dagger(r|0) + W^\dagger(-r|0)} \\ &= \frac{W^\dagger(r|0)}{W^\dagger(r|0) + W^\dagger(r|1)} \\ &= \frac{W^\dagger(r|0)}{W^\dagger(r|0) + e^{-r}W^\dagger(r|0)} = \frac{e^r}{1+e^r}\end{aligned}$$

The rest of transition probabilities can be obtained in a same fashion. □

Note that for a *BSC* $\left(\frac{1}{1+e^r}\right)$ the log-likelihood ratio is $\pm r$.

Proposition 1.1 (Decomposition of Symmetric B-DMCs into sub-BSCs)**.** *Every symmetric B-DMC can be described by a set of possible "reliabilities" $\mathcal{R} \subset \mathbb{R}_+$ and the distribution of reliability $P_\rho : \mathcal{R} \longrightarrow [0, 1]$.*

Proof. In the view of the results of lemmas 1.15 and 1.16 we can describe the internal structure of the equivalent channel $W^\dagger : \mathcal{X} \longrightarrow \mathbb{R}$ as follows: Corresponding to each pair of log-likelihood ratio $(\pm r, r \geqslant 0)$ there exist a BSC with parameter $p(r) = \frac{1}{1+e^r}$. Without looking at the input, one of these BSCs are selected at each time according to the distribution $P_\rho(r) = \mathbb{P}[\rho = r]$ then the input is fed to that BSC. The output of $W^\dagger$ is the log-likelihood ratio of that BSC. □

Proposition 1.2. *Assume $W : \mathcal{X} \longrightarrow \mathcal{Y}$ is a Symmetric B-DMC that has the following properties:*

1. *Channel output alphabet size $|\mathcal{Y}|$ is finite.*

2. *For each $y \in \mathcal{Y}$, y and $\pi_1(y)$ (described in Definition 1.4) are distinct.*

Hence $|\mathcal{Y}| = 2L$. This B-DMC can be decomposed into at most L sub-BSCs with the approach described in Proposition 1.1, namely $|\mathcal{R}| \leqslant L$. Moreover all elements of $\mathcal{R}$ are strictly positive.

Proof. We will use the shorthand notation of $\pi_1(y) = \bar{y}$. Observe that we can assume the output alphabet is $\mathcal{Y} = \{y_0, y_1, \cdots, y_{L-1}, \bar{y_0}, \bar{y_1}, \cdots, \bar{y_{L-1}}\}$. Each pair of output $(y, \bar{y})$ alphabet correspond to a pair of log-likelihood ratio $(\lambda(y), -\lambda(y))$ which means $\mathcal{R}$ has at most L distinct elements. Observe that since $\lambda(y)$ and $-\lambda(y)$ are distinct for all $y \in \mathcal{Y}$ non of them can be zero. Note that, if there exits identical log-likelihood ratios for different output pairs (let's say for distinct pairs $(y_i, \bar{y_i})$ and $(y_j, \bar{y_j})$, $r_i = |\lambda(y_i)| = |\lambda(y_j)| = r_j$) the corresponding sub-BSCs can be merged into a new sub-BSC which is chosen with probability $P_\rho(r_i) + P_\rho(r_j)$. This is exactly the equivalent channel obtained by applying Lemma 1.11 and Lemma 1.12. $\square$

Proposition 1.3. *Assume $W : \mathcal{X} \longrightarrow \mathcal{Y}$ is a Symmetric B-DMC that has the following properties:*

1. *Channel output alphabet size $|\mathcal{Y}|$ is finite.*
2. *For each $y \in \mathcal{Y}$, y and $\pi_1(y)$ (described in Definition 1.4) are distinct. Except one element $y_? \in \mathcal{Y}$ such that $y_? = \pi_1(y_?)$.*

Hence $|\mathcal{Y}| = 2L + 1$. This B-DMC can be decomposed into at most $L + 1$ sub-BSCs with the approach described in Proposition 1.1, namely $|\mathcal{R}| \leqslant L + 1$.

Proof. Except for $y_?$ the arguments in Proposition 1.2 apply to all elements of $\mathcal{Y}$. For $y_?$ we can easily verify that $\lambda(y_?) = 0$ which adds 0 to the elements of $\mathcal{R}$ when decomposing W. $\square$

Remark. By repeatedly applying Proposition 1.3 to any B-DMC with finite output alphabet size we can decompose it to finite number of sub-BSCs. Moreover, observe that the elements $y_?$ does not increase size of $\mathcal{R}$. Essentially size of $\mathcal{R}$ is determined by the number of elements $y \in \mathcal{Y}$ such that y and $\pi_1 y$ are distinct.

Example 1.2. a $BEC(\epsilon)$, can be decomposed into 2 BSCs: $\mathcal{R} = \{0, +\infty\}$, $P_\rho(0) = \epsilon$, $P_\rho(+\infty) = 1 - \epsilon$.

The results we obtained, shows that every symmetric B-DMC with discrete output alphabet can be described in a homogeneous fashion which is suitable for degrade/upgrade procedures as well. Moreover, any parameter of the channel of interest can be computed by computing the corresponding parameters of the sub-BSCs and then averaging them with respect to the distribution P_ρ.

Continuous Output Channels

The method suggested in Proposition 1.1 does not limit the channel output alphabet to be discrete. However, in order to apply the approximation methods, one has to quantize the output of a continuous output channel and degrade it to a channel with finite discrete outputs.

This quantization can be carried out in a straightforward manner using the decomposition method in the following way: The continuous set $\mathcal{R}$ is partitioned into L disjoint subsets $\mathcal{R}_0, \mathcal{R}_1, \cdots, \mathcal{R}_{L-1}$ and then the whole bunch of sub-BSCs corresponding to each subset will be merged into a single BSC. The probability of choosing each of the new sub-BSCs will then be $\check{P}_\rho(r_j) = \int_{r \in \mathcal{R}_j} P_\rho(r)$.

The problem of choosing an optimal partitioning strategy is still open. For example in [6] the authors have proposed a strategy as follows: Each sub-BSC (corresponding to

$\rho = r$) has a capacity which is a function of r. Denoting this capacity (with a slight abuse of notation) by $C[r]$, the subsets are defined as:

$$\mathcal{R}_j = \left\{ r \in \mathcal{R} : \frac{j}{L} \leqslant C[r] \leqslant \frac{j+1}{L} \right\} \tag{1.89}$$

Then it has been shown ([6, Lemma 13]) that the loss of mutual information by this quantization strategy is bounded as:

$$0 \leqslant I(W) - I(\breve{W}) \leqslant \frac{1}{L} \tag{1.90}$$

2 Efficient and Accurate Simulations of Error Correction Codes

Perhaps numerical simulations are the most important tool for both verifying the results of the analysis as well as obtaining some rough idea about what the results of an specific analysis could be. However, the results of numerically inaccurate computations can be easily deceptive.

Consequently, having a correct simulation framework is of crucial importance and in this section we will see how we can write accurate simulation programs.

Basically, a simulation of any error correction code (even a communication system from a more general point of view) consists of generating the samples of noise and source information in a certain number of trials and measuring the probability of an event of interest (e.g. bit error probability) by counting the number of times that event occurred and dividing it by the total number of trails.

Usually, "the event of interest" is a rare event. Hence, we must have sufficiently large number of trials in order to *observe* a certain number of its occurrences throughout the trials. This implies very long simulations especially when the event of interest is very rare.

Let's consider the following example: assume the bit error rate for a certain code is in the order of 10^{-6}. This means if we simulate transmission of roughly 1 million bits, we *might* see one wrongly decoded bit. Even with this number of trials, the results we obtain are not accurate since it is very probable that the event does not happen.

Fortunately, there exits a method called *Importance Sampling* that helps us get more accurate results in shorter times for simulating the rare events. We will discuss this method in this chapter.

2.1 Monte Carlo Simulations

The most common method for computing the probability of the events of interest is to use the Monte Carlo simulations. More formally, we are interested in computing the probability of an event $\mathcal{E}$ under a given probability distribution $P(\cdot)$ on the sample space Ω.

In this approach, one generates M independent samples from sample space $(x_i \in \Omega, \quad i = 1, 2, \dots M)$ according to distribution $P(\cdot)$ and for each sample, checks if the event of interest is occurred or not (more formally if $x_i \in \mathcal{E}$ or not). Finally the number of occurrences is divided by total number of samples (M) and announced as an estimation of $\mathbb{P}[\mathcal{E}]$.

The mathematical explanation for correctness of Monte Carlo estimations is the following: The output of the estimator can be written as:

$$\hat{P}_{MC} = \frac{1}{M} \sum_{i=1}^{M} \mathbb{1}_{\mathcal{E}}(x_i). \tag{2.1}$$

If the number of samples is sufficiently large, based on the law of large numbers we can write:

$$\hat{P}_{MC} \approx \mathbb{E}_P[\mathbf{1}_{\mathcal{E}}] = \mathbb{P}_P[\mathcal{E}] \tag{2.2}$$

Note that we used the subscript P to indicate we are calculating the probability or expectation assuming the distribution P on the sample space.

Lemma 2.1. *The Monte Carlo estimator is unbiased and consistent with variance:*

$$var_{MC} = \frac{1}{M}(P - P^2) \tag{2.3}$$

where we defined $P \triangleq \mathbb{P}_P[\mathcal{E}]$ for ease of notation.

In order for the estimations to be reliable their variance should be low enough. A natural measure for the reliability of the results is the ratio between the standard deviation of the estimations and the correct value which we denote by δ. For the Monte Carlo estimator this ratio is:

$$\begin{aligned} \delta_{MC} &= \frac{\sqrt{\frac{1}{M}(P - P^2)}}{P} \\ &= \frac{1}{\sqrt{M}}\sqrt{\frac{1}{P} - 1} \end{aligned} \tag{2.4}$$

Now if the estimator is written in such a way that it produces a fixed number of samples (M) regardless of rareness of the event of interest, we can see that when the event becomes more rare $(P \to 0)$, the reliability of the estimations will deteriorate $(\delta \to \infty)$.

However, one can easily verify that the correct way of writing the simulator is to continue generating the samples at least a certain number of *events* are seen. Let's denote this number by E. In order for the event $\mathcal{E}$ with probability P is seen E times we must roughly generate $M \approx \frac{E}{P}$ samples. Hence one can rewrite (2.4) as:

$$\delta_{MC} = \frac{1}{\sqrt{E}}\sqrt{1 - P} < \frac{1}{\sqrt{E}} \tag{2.5}$$

Note that in practice we would like to have a certain accuracy for the estimations and hence we can determine E from (2.5) using the desired accuracy (for example, if we want the results to be within 1% of the real value, we can easily see that we need to generate at least 10000 occurrences of $\mathcal{E}$).

These conditions suggest Algorithm 3 as a good way to estimate the probability of rare events.

Algorithm 3 Accurate Monte Carlo Simulation

```
Input: The probability space $(\Omega, P(\cdot))$, the event of interest $\mathcal{E}$ and the requested accuracy ($\delta$).
Output: $\hat{P}$ an estimation of $\mathbb{P}[\mathcal{E}]$ with property $1-\delta \leqslant \frac{\hat{P}}{\mathbb{P}[\mathcal{E}]} \leqslant 1+\delta$ with very high probability.
  $E \leftarrow \frac{1}{\delta^2}$
  count $\leftarrow 0$
  trials $\leftarrow 0$
  while count $< E$ do
    Draw $x \in \Omega$ according to $P(x)$
    if $x \in \mathcal{E}$ then
      count $\leftarrow$ count $+ 1$
    end if
    trials $\leftarrow$ trials $+ 1$
  end while
  $\hat{P} \leftarrow \frac{\texttt{count}}{\texttt{trials}}$
  return $\hat{P}$
```

The method mentioned in Algorithm 3 has the advantage that its result is accurate enough and the simulation time is just as long as needed for reaching the required accuracy. In other words, if we terminate the Monte Carlo simulations based on the number of trials we might generate too many samples when the event of interest is not of very low probability or too few samples when the event of interest is too rare.

In practice, however, it is also necessary to put some termination conditions based on the number of trials in order to avoid writing the simulation codes that will never halt (in case the event of interest has a very low probability). It is clear that in this case, we can compute the accuracy of results based on the number of events we have generated (that can be less than the desired number E) using (2.5) (see Algorithm 4).

2.2 Importance Sampling

We just saw that if the event of interest $\mathcal{E}$ is rare, in order to compute its probability using Monte Carlo approach one needs to run very long simulations and since in practice the number of samples we generate is limited (let's denote it by M), it would be very probable that the output of MC Estimator is zero (particularly if $P < \frac{1}{M}$). Even if the output is non-zero, the number of *events* that are generated can be very low and hence (according to (2.5)) the accuracy of the results is very low.

Algorithm 4 Practical Monte Carlo Simulations

```
Input: The probability space $(\Omega, P(\cdot))$, the event of interest $\mathcal{E}$ and the requested accuracy
    $(\delta)$, maximum number of trials $M$.
Output: $\hat{P}$ an estimation of $\mathbb{P}[\mathcal{E}]$ and the achieved accuracy $\tilde{\delta}$ with property $1 - \tilde{\delta} \leqslant \frac{\hat{P}}{\mathbb{P}[\mathcal{E}]} \leqslant 1 + \tilde{\delta}$ with very high probability.
  $E \leftarrow \frac{1}{\delta^2}$
  count $\leftarrow 0$
  trials $\leftarrow 0$
  while count $< E$ and trials $< M$ do
    Draw $x \in \Omega$ according to $P(x)$
    if $x \in \mathcal{E}$ then
      count $\leftarrow$ count $+ 1$
    end if
    trials $\leftarrow$ trials $+ 1$
  end while
  $\hat{P} \leftarrow \frac{\texttt{count}}{\texttt{trials}}$
  $\tilde{\delta} \leftarrow \frac{1}{\sqrt{\texttt{count}}}$ {Ideally should be equal to $\delta$}
  return $\hat{P}, \tilde{\delta}$
```

For example, consider an event $\mathcal{E}$ whose probability is roughly $\frac{1}{M}$. We can see that the Monte Carlo simulator of Algorithm 4 will probably only generate one event ($\texttt{count} = 1$) and the results can have 100% error.

In this situation, another method is used which is called "Importance Sampling". In Importance Sampling estimation we draw the samples x_i according to a *fake* distribution $Q(\cdot)$ on the probability space instead of $P(\cdot)$ and we estimate the probability of interest, $\mathbb{P}_P[\mathcal{E}]$ as:

$$\hat{P}_{IS} = \frac{1}{M} \sum_{i=1}^{M} \mathbb{1}_{\mathcal{E}}(x_i) w(x_i) \tag{2.6}$$

where

$$w(x) \triangleq \frac{P(x)}{Q(x)}. \tag{2.7}$$

Intuitively, we see that it is better to choose $Q(\cdot)$ such that $\mathcal{E}$ is more probable under that. This is roughly true and we will formally show the optimal choice of $Q(\cdot)$ shortly. First we see the properties of the mentioned estimator:

Lemma 2.2. *IS estimator is unbiased and consistent with variance:*

$$var_{IS} = \frac{1}{M} \left(\sum_{x \in \mathcal{E}} w(x) P(x) - P^2 \right) \tag{2.8}$$

where $P \triangleq \mathbb{P}_P[\mathcal{E}]$.

Proof.

$$\begin{aligned}\mathbb{E}_Q[\hat{P}_{IS}] &= \frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_Q[\mathbb{1}_{\mathcal{E}}[x_i]w(x_i)] \\ &= \mathbb{E}_Q[\mathbb{1}_{\mathcal{E}}[x]w(x)]\end{aligned}$$

where the last equality follows from the fact that x_is are identically distributed. Now observe that:

$$\begin{aligned}\mathbb{E}_Q[\mathbb{1}_{\mathcal{E}}(x)w(x)] &= \sum_{x\in\mathcal{E}}\mathbb{1}_{\mathcal{E}}(x)w(x)Q(x) \\ &= \sum_{x\in\mathcal{E}}\mathbb{1}_{\mathcal{E}}(x)P(x) = \mathbb{P}_P[\mathcal{E}]\end{aligned}$$

As for the variance of estimator:

$$\begin{aligned}\mathbb{E}_Q\left[\hat{P}_{IS}^2\right] &= \mathbb{E}_Q\left[\frac{1}{M^2}\sum_{i,j}\mathbb{1}_{\mathcal{E}}(x_i)w(x_i)\mathbb{1}_{\mathcal{E}}(x_j)w(x_j)\right] \\ &= \frac{1}{M^2}\sum_{i,j}\mathbb{E}_Q\left[\mathbb{1}_{\mathcal{E}}(x_i)w(x_i)\mathbb{1}_{\mathcal{E}}(x_j)w(x_j)\right] \\ &= \frac{1}{M^2}\left(\sum_{i=j}\mathbb{E}_Q\left[\mathbb{1}_{\mathcal{E}}(x_i)w(x_i)\mathbb{1}_{\mathcal{E}}(x_j)w(x_j)\right] + \sum_{i\neq j}\mathbb{E}_Q\left[\mathbb{1}_{\mathcal{E}}(x_i)w(x_i)\mathbb{1}_{\mathcal{E}}(x_j)w(x_j)\right]\right) \\ &= \frac{1}{M^2}\left(M\mathbb{E}_Q\left[\mathbb{1}_{\mathcal{E}}(x)w(x)^2\right] + M(M-1)\mathbb{E}_Q\left[\mathbb{1}_{\mathcal{E}}(x)w(x)\right]^2\right) \\ &= \frac{1}{M}\left(\sum_{x\in\mathcal{E}}w(x)P(x) + (M-1)P^2\right)\end{aligned}$$

Since:

$$var_{MC} = \mathbb{E}_Q\left[\hat{P}_{IS}^2\right] - \mathbb{E}_Q\left[\hat{P}_{IS}\right]^2$$

the claim will be obtained. □

Equation (2.8) shows that if we choose

$$Q(x) = \frac{P(x)\mathbb{1}_{\mathcal{E}}(x)}{P} \tag{2.9}$$

we will obtain a zero-variance estimator. But this is impossible in practice since we actually need to estimate P. However, we can get the hint that a good choice of $Q(\cdot)$ is the one that is similar to the original distribution $P(\cdot)$ in the region of error event and zero elsewhere.

Actually the name "Importance Sampling" follows from the fact that the optimal distribution is the one that puts all of the weight on the important (error producing) samples of the sample space.

There exist different methods for finding optimal distribution for performance evaluation of error correcting codes in Gaussian channel, such as [7] and [8]. However, we obtain some results for case of BSC and BEC in the sequel.

2.2.1 Importance Sampling for Forward Error Correction Codes

In this part, we will see how we can exploit Importance Sampling to efficiently evaluate the performance of error correction codes. We restrict out analysis to two special B-DMCs, namely BSC and BEC. However, we believe the results can be generalized to other symmetric discrete output B-DMCs as well. We also note that there is nothing specific to polar codes in our analysis and the whole results can be used for performance evaluation of any other error correcting code whose minimum distance is known.

In the problems related to error correcting codes, the "rare events of interest" are decoding errors such as word-error or bit-error. For example the word-error event of a code is defined as :

$$\mathcal{E} = \left\{ \exists i : X_i \neq \hat{X}_i \right\}$$

where $\mathbf{X}$ is the vector of sent codeword and $\hat{\mathbf{X}}$ will be the vector of decoded codeword.

It can be easily seen that the sample space of interest is the set of all possible codewords and all possible channel outputs $\Omega = \mathcal{C} \times \mathcal{Y}^N$ where $\mathcal{C} \subset \mathcal{X}^N$ is the codebook.The probability distribution is denoted by $P_{\mathbf{X},\mathbf{Y}}(\mathbf{x}, \mathbf{y})$. In coding arguments we usually assume a uniform distribution on choice of codewords hence:

$$P_{\mathbf{X},\mathbf{Y}}(\mathbf{x}, \mathbf{y}) = 2^{-NR} W^N(\mathbf{y}|\mathbf{x}), \qquad \forall \mathbf{x}, \mathbf{y} \in \mathcal{C} \times \mathcal{Y}^N \tag{2.10}$$

where $W^N : \mathcal{X}^N \longrightarrow \mathcal{Y}^N$ is the channel corresponding to N uses of the initial binary channel W which in case of a memoryless channel reduces to:

$$W^N(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{N} W(y_i|x_i)$$

and R is the code rate (hence we have $|\mathcal{C}| = K = NR$ codewords).

Now the question will be what is the correct choice of the fake distribution $Q_{\mathbf{U},\mathbf{Y}}$ on the sample space? This distribution depends on the distribution of codewords and the channel transition probability.

While we have the freedom of choosing the fake distribution arbitrarily, in order to keep the problem simple, we consider the following restrictions further:

- The distribution of sent codeword is not altered and they are always chosen uniformly.
- We maintain the memorylessness of channel.

Hence, the problem will reduce to finding the transition probability of the fake B-DMC $\tilde{W} : \mathcal{X} \longrightarrow \mathcal{Y}$ such that the variance of IS estimator is minimized.

Using (2.7) we can see the weight of each event $(\mathbf{x}, \mathbf{y}) \in \Omega$ is

$$w(\mathbf{x}, \mathbf{y}) = \prod_{i=0}^{N-1} \frac{W(y_i|x_i)}{\tilde{W}(y_i|x_i)} \tag{2.11}$$

We should mention that only the channel output sample should be drawn according to the fake distribution, however, the log-likelihood ratios should remain unchanged.

Recall that for any B-DMC W, there exists an equivalent channel $W^{\dagger}$ whose output is the log-likelihood ratio of the original channel (see Figure 1.6). The sample space using the equivalent channel will be the space of codewords and possible likelihood ratios. In Importance Sampling we only change the distribution according to which the elements of sample space are drawn not the elements themselves.

No we restrict the channels of interest to BSC and BEC. We start by analyzing the BSC and then use the same arguments for BEC.

BSC

If the B-DMC of interest is a BSC (which is described with its crossover probability parameter p). Naturally, the fake channel $\tilde{W}$ is also a BSC with another crossover probability q. As a result:

$$\frac{W(y_i|x_i)}{\tilde{W}(y_i|x_i)} = \begin{cases} \frac{1-p}{1-q} & \text{if } y_i = x_i \\ \frac{p}{q} & \text{if } y_i \neq x_i. \end{cases} \tag{2.12}$$

If we define n_f as the number of channel flips occurred in a one codeword, namely:

$$n_f = |\{y_i, \quad i = 0, 1, \cdots, N-1 : y_i \neq x_i\}| \tag{2.13}$$

we can write the weight corresponding to a channel output and codeword pair as:

$$w(\mathbf{x}, \mathbf{y}) = \left(\frac{p}{q}\right)^{n_f} \left(\frac{1-p}{1-q}\right)^{N-n_f} \tag{2.14}$$

From basic coding theorems, we know that for a given error correcting code, if the number of flipped bits are less than $\frac{d_{min}}{2}$ the error event does not happen[1]. So in order to *see* an error event, the channel should at least flip $t = \frac{d_{min}}{2} + 1$ of its inputs. Rewriting equation (2.8) we see:

$$var_{IS} \leqslant \frac{1}{M}\left(\sum_{n_f=t}^{N} P(N_f = n_f)\left(\frac{p}{q}\right)^{n_f}\left(\frac{1-p}{1-q}\right)^{N-n_f} - P^2\right) \tag{2.15}$$

Note that $P(N_f = n_f)$ is probability of channel flipping n_f bits under the *original* channel distribution. Moreover, the inequality comes from the fact that making $n_f \geqslant t$ flips does not necessarily produces a decoding error since the sent codeword can still be the closest codeword to the received sequence if the errors does not move the sent codeword toward another codeword (so we can say $\{N_f \geqslant t\} \supseteq \mathcal{E}$). Nevertheless, we have:

$$P(N_f = n_f) = \binom{N}{n_f} p^{n_f}(1-p)^{N-n_f}. \tag{2.16}$$

Now the problem is to find an optimal q such that (2.15) is minimized. Since in (2.15), P is the actual probability of error event and it is independent of q and M is a positive constant indicating number of trials, the problem reduces to minimizing:

$$c(q) = \sum_{n_f=t}^{N} \binom{N}{n_f} p^{n_f}(1-p)^{N-n_f}\left(\frac{p}{q}\right)^{n_f}\left(\frac{1-p}{1-q}\right)^{N-n_f} \tag{2.17}$$

with respect to $0 \leqslant q \leqslant 1$.

[1]This is indeed true for a ML decoder.

Lemma 2.3. *The minimum of function*

$$c(q) = \sum_{i=t}^{N} \binom{N}{i} p^i (1-p)^{N-i} \left(\frac{p}{q}\right)^i \left(\frac{1-p}{1-q}\right)^{N-i} \qquad 0 \leqslant p \leqslant \frac{1}{2}$$

with respect to $0 \leqslant q \leqslant 1$ *is obtained at*

$$q = \begin{cases} p & \text{if } t < Np \\ \frac{t}{N} & \text{if } t \geqslant Np \end{cases} \tag{2.18}$$

Proof. Let's define

$$p' \triangleq \frac{\frac{p^2}{q}}{\frac{p^2}{q} + \frac{\bar{p}^2}{\bar{q}}}.$$

It can be verified that:

$$c(q) = \left(\frac{p^2}{q} + \frac{\bar{p}^2}{\bar{q}}\right)^N \sum_{i=t}^{N} \binom{N}{i} p'^i (1-p')^{N-i}.$$

Now observe that the sum above is the probability of a binomial random variable with parameter p' to be greater than t which is well approximated by $\exp\left[-ND\left(\frac{t}{N} \| p'\right)\right]$ if $\frac{t}{N} \geqslant p'$ where $D(\cdot\|\cdot)$ is the Kullback-Leibler divergence defined as:

$$D(x\|y) = x \log\left(\frac{x}{y}\right) + (1-x)\log\left(\frac{1-x}{1-y}\right).$$

We also define

$$\lambda \triangleq \frac{t}{N}$$

hence:

$$c(q) \approx \exp\left[N\left(\log\left(\frac{p^2}{q} + \frac{\bar{p}^2}{\bar{q}}\right) + \lambda \log\frac{p'}{\lambda} + \bar{\lambda}\log\frac{\bar{p'}}{\bar{\lambda}}\right)\right].$$

So, minimizing $c(q)$ with respect to q is equivalent to minimizing

$$e(q) \triangleq \left(\log\left(\frac{p^2}{q} + \frac{\bar{p}^2}{\bar{q}}\right) + \lambda \log\frac{p'}{\lambda} + \bar{\lambda}\log\frac{\bar{p'}}{\bar{\lambda}}\right)$$

with respect to q.

Taking derivative of $e(q)$ with respect to q we will obtain:

$$\begin{aligned}
\frac{\partial e}{\partial q} &= \frac{\left(-\frac{p^2}{q^2} + \frac{\bar{p}^2}{\bar{q}^2}\right)}{\left(\frac{p^2}{q} + \frac{\bar{p}^2}{\bar{q}}\right)} + \left(\frac{\partial p'}{\partial q}\right)\left(\frac{\lambda}{p'} - \frac{\bar{\lambda}}{\bar{p'}}\right) \\
&= \frac{\left(-\frac{p^2}{q^2} + \frac{\bar{p}^2}{\bar{q}^2}\right)}{\left(\frac{p^2}{q} + \frac{\bar{p}^2}{\bar{q}}\right)} + \frac{\frac{p^2\bar{p}^2}{q^2\bar{q}^2}}{\left(\frac{p^2}{q} + \frac{\bar{p}^2}{\bar{q}}\right)^2}\left(\frac{\lambda}{p'} - \frac{\bar{\lambda}}{\bar{p'}}\right) \\
&= \frac{\left(-\frac{p^2}{q^2} + \frac{\bar{p}^2}{\bar{q}^2}\right)}{\left(\frac{p^2}{q} + \frac{\bar{p}^2}{\bar{q}}\right)} + \frac{\frac{p^2\bar{p}^2}{q^2\bar{q}^2}}{\left(\frac{p^2}{q} + \frac{\bar{p}^2}{\bar{q}}\right)^2}\left(\frac{\lambda\left(\frac{p^2}{q} + \frac{\bar{p}^2}{\bar{q}}\right)}{\frac{p^2}{q}} - \frac{\bar{\lambda}\left(\frac{p^2}{q} + \frac{\bar{p}^2}{\bar{q}}\right)}{\frac{\bar{p}^2}{\bar{q}}}\right) \\
&= \frac{1}{\frac{p^2}{q} + \frac{\bar{p}^2}{\bar{q}}}\left[-\frac{p^2}{q^2} + \frac{\bar{p}^2}{\bar{q}^2} + \lambda\frac{\bar{p}^2}{\bar{q}^2 q} - \bar{\lambda}\frac{p^2}{q^2\bar{q}}\right].
\end{aligned}$$

Equating the derivative to zero we obtain:

$$\begin{aligned} &\frac{\bar{p}^2 q^2 - p^2 \bar{q}^2 + \bar{p}^2 q \lambda - p^2 \bar{q}^2 \bar{\lambda}}{q^2 \bar{q}^2} = 0 \\ \Rightarrow \quad &\left(p^2 + q - 2pq\right)(q - \lambda) = 0. \end{aligned}$$

Solving the above equation for q we obtain:

$$q = \begin{cases} \lambda \\ \frac{p^2}{2p-1} \end{cases}$$

The second solution is not in domain of $0 \leqslant q \leqslant 1$ (particularly when $p < 1/2$ it is negative) hence the solution is

$$q = \lambda = \frac{t}{n} \tag{2.19}$$

Now we should check if our assumption for approximating the binomial distribution with an exponential term is valid with this choice of q or not. Setting $q = \lambda$ we obtain:

$$p' = \frac{p^2 \bar{\lambda}}{p^2 \bar{\lambda} + \bar{p}^2 \lambda}.$$

For the approximation to be valid, we must have:

$$\begin{aligned} &\lambda \geqslant p' \\ \Rightarrow \quad &p^2 \lambda \bar{\lambda} + \bar{p}^2 \lambda^2 \geqslant p^2 \bar{\lambda} \\ \Rightarrow \quad &(1 - 2p)\lambda^2 + 2p^2 \lambda - p^2 \geqslant 0 \end{aligned}$$

The last expression is quadratic in λ and with the assumption of $p \leqslant \frac{1}{2}$ it is a convex parabola with one root at $\lambda = p$ and another negative root. Hence it is positive if $\lambda \geqslant p$.

For the case that $\lambda \leqslant p$ we obtain the same approximations except we must replace λ by p. This proves the lemma. □

Replacing $t = \frac{d_{min}}{2} + 1$ in the Lemma 2.3 we obtain:

$$q_{BSC} \approx \begin{cases} p & \text{if } \frac{d_{min}}{2} + 1 \leqslant Np \\ \frac{\frac{1}{2} d_{min} + 1}{N} & \text{otherwise} \end{cases} \tag{2.20}$$

BEC

A BEC is well described with its erasure probability which is normally noted by ϵ. However, in this subsection we note the erasure probability of a BEC with p to easily see the analogy with the analysis of BSC.

Again the fake channel is a BEC with erasure probability of q and we can write:

$$\frac{W(y_i|x_i)}{\tilde{W}(y_i|x_i)} = \begin{cases} \frac{1-p}{1-q} & \text{if } y_i = x_i \\ \frac{p}{q} & \text{if } y_i = E, \end{cases} \tag{2.21}$$

where E stand for the erasure symbol of the channel.

Similar to BSC, if we define n_e as the number of erasures happened among all N bits of a codeword we can write

$$w(\mathbf{x}, \mathbf{y}) = \left(\frac{p}{q}\right)^{n_e} \left(\frac{1-p}{1-q}\right)^{N-n_e} \tag{2.22}$$

We know an error correcting code with minimum distance d_{min} can correct up to d_{min} erasures. Hence, in order for a decoding error event to occur, the channel should at least erase $t = d_{min} + 1$ bits of the input codeword and probability of having n_e erasures in channel is:

$$P(N_e = n_e) = \binom{n}{n_e} p^{n_e} (1-p)^{N-n_e}. \tag{2.23}$$

Plugging the derived expressions into (2.8) we see that the variance of IS estimations in case of a BEC channel is exactly the same as that of BSC and can be obtained just by substituting the dummy variable n_f in (2.15) with n_e. Consequently, we can use the results of Lemma 2.3 and obtain the optimal fake erasure probability as:

$$q_{BEC} \approx \begin{cases} p & \text{if } d_{min} + 1 \leqslant Np \\ \frac{d_{min}+1}{N} & \text{otherwise} \end{cases} \tag{2.24}$$

2.2.2 Results for Polar Codes

So far, our discussions were not specific to polar codes and the arguments are true for any error correcting code. We saw that the only property of the code we need to know is its minimum distance.

In [3, Lemma 4] the minimum distance of a polar code of codeword length $N = 2^n$ is upper-bounded by $2^{\frac{n}{2}+c\sqrt{n}}$. However this bound is not helpful since it is valid for very large block lengths.

Even if we do the simulations for large block lengths, we need to find the minimum distance as a function of rate since in low rates the error events become more rare (as the minimum distance increases) while the mentioned result does not depend on the rate.

Nevertheless, since in the simulations we exactly know the position of information bits we know the columns of generator matrix (recall that the generator matrix of the code is $\mathbf{G}_N[\mathcal{A}]$). In general, the minimum distance of a code is less than the minimum weight of its generator matrix columns, however in [3, Lemma 3] it is proved that for polar codes, the minimum distance is exactly minimum weight of the columns of the generator matrix.

We use Proposition 17 in [1] to find the hamming weight of generator matrix columns which states that if the binary expansion of $0 \leqslant i \leqslant N-1$ is $b_0 b_1 \dots b_{n-1}$ then Hamming weight of ith column of $\mathbf{G}_N$ is:

$$2^{w_H(b_0, b_1, \cdots, b_{n-1})} \tag{2.25}$$

where

$$w_H(b_0, b_1, \cdots, b_{n-1}) \triangleq \sum_{i=0}^{n-1} b_i \tag{2.26}$$

One critical point in all of the arguments we had so far is the assumption that the decoding errors happen at minimum distance of the code. This assumption might not be always true. For example, Gallager in [9, Section 3.6] argues that most of the decoding errors of random codes happen at a distance of $N\alpha(s)$ which is greater than the minimum distance of the code[2]. Even the general coding-theory arguments on the distance at which the decoding errors occur are true for ML/MAP decoder and might not be valid for the SC decoder.

Moreover, we can see that as the block length increases, the ratio of d_{min}/N decreases toward zero and at some point, the equations (2.20) and (2.24) will give us the original channel (namely the crossover/erasure probability of p).

Hence, it still remains an important problem to see at which distance (or more generally due to which phenomena) most of the decoding errors happen for polar codes and then choose the fake channel sampling parameter accordingly.

The results we obtained here are, to some extents, *too conservative.* However, when they are applicable they will still give us better numerical results.

2.3 Taking Advantage of Channel's Symmetry

As we saw in Chapter 1 symmetric B-DMCs have specific properties. Indeed, polar codes are good candidates for those class of channels since their Shannon capacity can be achieved using these codes.

In terms of error events, as we saw in Corollary 1.3 the errors in SC decoder are independent of the source information vector. Consequently, for a symmetric B-DMC, we can always assume the sent codeword is all zero codeword and any conclusions we draw on this basis is valid for the code. In practice we use this fact and we eliminate the complexity of generating random codewords and encoding them from our simulations and initiate the computations from the channel output samples assuming all zero codeword is sent.

However, we should pay attention to decoding process when ties occur in decision. A tie in SC decoding happens when the log-likelihood ratio of a certain bit is zero. If we break the ties in favor of zeros the results could be easily misleading.

An example of this case is when we simulate a BEC in a rate very close to capacity or even at rates above capacity where we expect to see some decoding errors (at least for short block lengths). However, even if we use no polarization, namely we just use the raw channels for transmitting bits, the log-likelihood ratios will be either infinity or zero (meaning erasure) which are all decoded into zeros. Therefore we see no decoding errors while this is not true in practice.

The safest way to implement the decoder is to use random decisions when ties occur. Which is, if log-likelihood ratio is zero, the decoder flips a fair coin and based on the result announces 0 or 1 as the value of the decoded bit.

2.4 A Numerical Example

In this section, we see a numerical example that justifies the results we recently obtained.

Let's consider the polar code with block length of $N = 64$. This length is too low for practical use, however, the ratio between the minimum distance of the code and the

[2]Nevertheless we know that polar codes does not behave as random codes, hence we cannot use that result for simulating polar codes.

block length is such that the fake distribution will be something different from original channel transition probability in importance sampling.

We simulated the code both on $BEC(0.1)$ whose capacity is 0.9 bits per channel use and $BSC(0.1)$ whose capacity is approximately 0.53 bits per channel use to obtain the word-error rate of the code.

The results of estimations are shown in tables 2.1 and 2.2. Estimations are done in $M = 10^6$ iterations and then each estimation is repeated 100 times. We have intentionally put the termination condition of the simulations on the number of trials so that we can have a fair comparison between two methods in different rates.

Rate	d_{min}	q	$\sum_{i\in\mathcal{A}} Z\left(W_N^{(i)}\right)$	$\hat{P}_{wMC}$	$\hat{P}_{wIS}$	$\frac{\sigma_{MC}}{\hat{P}_{wMC}}$	$\frac{\sigma_{IS}}{\hat{P}_{wIS}}$
0.13	16	0.266	$4 \cdot 10^{-16}$	–	$6.1 \cdot 10^{-18}$	–	9.9
0.25	16	0.266	$5.8 \cdot 10^{-13}$	–	$3.9 \cdot 10^{-13}$	–	6.5
0.38	8	0.141	$2.8 \cdot 10^{-7}$	$6 \cdot 10^{-8}$	$9.8 \cdot 10^{-8}$	4	$8.9 \cdot 10^{-1}$
0.50	8	0.141	$2.7 \cdot 10^{-5}$	$1.4 \cdot 10^{-5}$	$1.3 \cdot 10^{-5}$	$2.7 \cdot 10^{-1}$	$9.4 \cdot 10^{-2}$
0.63	8	0.141	$1.8 \cdot 10^{-3}$	$1 \cdot 10^{-3}$	$8.9 \cdot 10^{-4}$	$3 \cdot 10^{-2}$	$1.5 \cdot 10^{-2}$
0.75	4	0.100	$6.9 \cdot 10^{-2}$	$3.3 \cdot 10^{-2}$	$3.3 \cdot 10^{-2}$	$5.7 \cdot 10^{-3}$	$4.7 \cdot 10^{-3}$
0.88	2	0.100	$9.8 \cdot 10^{-1}$	$3.5 \cdot 10^{-1}$	$3.5 \cdot 10^{-1}$	$1.4 \cdot 10^{-3}$	$1.4 \cdot 10^{-3}$

Table 2.1: Estimation of P_w on $BEC(0.1)$ using Monte Carlo and Importance Sampling methods

Rate	d_{min}	q	$\sum_{i\in\mathcal{A}} Z\left(W_N^{(i)}\right)$	$\hat{P}_{wMC}$	$\hat{P}_{wIS}$	$\frac{\sigma_{MC}}{\hat{P}_{wMC}}$	$\frac{\sigma_{IS}}{\hat{P}_{wIS}}$
0.09	32	0.266	$5 \cdot 10^{-5}$	$8.6 \cdot 10^{-6}$	$5.3 \cdot 10^{-6}$	$3.2 \cdot 10^{-1}$	$1.5 \cdot 10^{-2}$
0.17	16	0.141	$1.3 \cdot 10^{-2}$	$1.7 \cdot 10^{-3}$	$1.5 \cdot 10^{-3}$	$2.6 \cdot 10^{-2}$	$1.2 \cdot 10^{-2}$
0.25	16	0.141	$1 \cdot 10^{-1}$	$1.4 \cdot 10^{-2}$	$1.3 \cdot 10^{-2}$	$8.9 \cdot 10^{-3}$	$3.6 \cdot 10^{-3}$
0.33	8	0.100	$4.7 \cdot 10^{-1}$	$6.8 \cdot 10^{-2}$	$6.7 \cdot 10^{-2}$	$3.5 \cdot 10^{-3}$	$3.8 \cdot 10^{-3}$
0.41	8	0.100	$1.5 \cdot 10^{0}$	$2 \cdot 10^{-1}$	$2 \cdot 10^{-1}$	$2 \cdot 10^{-3}$	$2 \cdot 10^{-3}$
0.48	8	0.100	$3.4 \cdot 10^{0}$	$4 \cdot 10^{-1}$	$4 \cdot 10^{-1}$	$1.2 \cdot 10^{-3}$	$1.4 \cdot 10^{-3}$

Table 2.2: Estimation of P_w on $BSC(0.1)$ using Monte Carlo and Importance Sampling methods

For the BEC as we see, we are unable to obtain any results about the error events for two first low rates using Monte Carlo approach, since in fact the events were so rare that they never occur in totally 100 million runs. This is actually natural, since by Corollary 1.4, P_w is upper bounded by sum of the Bhattacharyya parameters of the channels in the information set which is shown in the table as well. However, the importance sampling makes the error events of interest happen and the estimator will be successful, despite its relatively high variations (see the ratio between the standard deviation of the estimations and their average).

For the rate of 0.38, the Monte Carlo estimator has produced non-zero output in some of the runs, however, we see that compared to the IS estimator, the estimations are very inaccurate (the result of MC has 400% error while that of the IS has 89%).

In higher rates, we see that the results of IS and MC estimations are approximately equal, however, generally the IS estimator has lower variance, so its results are more reliable.

We have the same observations for $BSC(0.1)$. We see that for low rates, the Monte Carlo estimator has high variations. As we increase the rate, the results of two estimators become closer, while the variations of the IS estimator is lower than that of the MC estimator.

Genie Aided Decoding 3

Perhaps the main disadvantage of polar codes is their poor performance in low to moderate block lengths. This can be either an intrinsic weakness of the codes or the sub-optimality of the successive cancellation decoder.

Due to the structure of SC decoding, errors propagate throughout the decoding process; if the decoder takes a wrong decision on ith bit, it can potentially decide incorrectly on any of the following bits, despite the probable good likelihood ratio for those bits. Stating differently, if the likelihood ratio for decision on ith bit is not good enough, the ML decoder can potentially decode all other bits except that one correctly while the SC decoder will probably decode more wrong bits. Note that in terms of word-error rate there would be no difference between ML decoder or SC decoder in the mentioned scenario since both of them would decode a wrong codeword. However, as we see in this chapter, most of the error events in the SC decoder in certain rates of communication stem from a single wrong decision which will cause the decoding path to deviate from that of the correct codeword in potentially many bits which results in a codeword much different than the sent codeword while the ML decoder's output can only differ with the sent codeword in a single bit.

This observation has several important implications. For example, if one wants to improve the performance of the decoders by converting them to list decoders, typically much larger list sizes would be required for the SC decoder compared to ML decoder.

We will also see that it might be possible to improve the performance of SC decoder by only detecting and correcting the first wrong decision throughout the decoding process.

One method for analyzing the errors in SC decoding is to consider a *genie aided* decoder in which a genie corrects the decisions of the decoder after it decides on every bit and prevents it from deviating from the correct decoding path. The behavior of this genie can give us useful information about error events in the ordinary SC decoding. For example the event that genie does not help at all is equivalent to correct decoding of the codeword.

We partially focus the analysis of genie aided decoding on BEC. When the B-DMC is BEC all forged channels will also be BECs and this simplifies the analysis. However, some of the results can be generalized to other channels as well. We should also point out that as we saw in Chapter 1 BEC is somehow a extremal case of all binary symmetric channels, hence the results for BEC can at least provide some bounds for other binary channels.

3.1 Analysis of Genie Behavior

The genie-aided decoder is formally defined as follows: the "genie" knows the sent codeword and at each decoding step of the SC decoder if the decision is wrong it corrects the decision. Hence the SC decoding algorithm (Algorithm 1) will be modified to the genie-aided SC decoder in Algorithm 5

Algorithm 5 Genie-Aided Successive Cancellation Decoding

```
Input: Channel output $\mathbf{y}$, $(N, K, \mathcal{A}, \mathbf{u}_{\mathcal{A}^C})$ and information bits $\mathbf{u}_{\mathcal{A}}$.
Output: $\mathbf{g}$, a binary vector exhibiting genie's behavior.
  $\mathbf{g} \leftarrow \mathbf{0}$
  for all $i \in \{0, 1, \cdots, N-1\}$ do
    if $i \in \mathcal{A}^C$ then
      $\hat{u}_i \leftarrow u_i$
    else
      if $\frac{W_N^{(i)}(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}|0)}{W_N^{(i)}(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}|1)} \geqslant 1$ then
        $\hat{u}_i \leftarrow 0$
      else
        $\hat{u}_i \leftarrow 1$
      end if
      if $\hat{u}_i \neq u_i$ then
        $\hat{u}_i \leftarrow u_i$
        $g_i \leftarrow 1$
      end if
    end if
  end for
  return $\mathbf{g}$
```

Let's define the indicator random variable $G_N^{(i)}, \quad i = 0, 1, \cdots, N-1$ as the random variable which takes the value of 1 if the genie helps at ith bit and 0 otherwise. Hence $\mathbf{g}$, the output of Algorithm 5, will be a realization of the random vector $\mathbf{G}_N = \left[G_N^{(0)}, G_N^{(1)}, \cdots, G_N^{(N-1)}\right]^T$.

In genie-aided decoding, since by the time of deciding on bit i all previous bits are correct (either they are decided correctly or the genie has corrected their wrongly decided value), the decisions of (1.26) can be rewritten in the following way:

$$\hat{u}_i\left(\mathbf{y}, \mathbf{u}_0^{i-1}\right) = \begin{cases} u_i & \text{if } i \in \mathcal{A}^c \\ h_i\left(\mathbf{y}, \mathbf{u}_0^{i-1}\right) & \text{if } i \in \mathcal{A}. \end{cases} \tag{3.1}$$

Proposition 3.1. *For any binary discrete memoryless channel W, and $i \in \mathcal{A}$*

$$\mathbb{P}\left[G_N^{(i)} = 1\right] \leqslant Z\left(W_N^{(i)}\right). \tag{3.2}$$

For $i \in \mathcal{A}^c$, trivially $\mathbb{P}\left[G_N^{(i)} = 1\right] = 0$.

Proof. The second part of the claim is trivial. Proof of the first part is quite similar to the proof of Theorem 1.3. Observe that the genie acts if $\hat{u}_i \neq u_i$, hence, for $i \in \mathcal{A}$,

$$\begin{aligned}\mathbb{P}\left[G_N^{(i)} = 1\right] &= \mathbb{P}\left[h_i\left(\hat{u}_0^{i-1}, \mathbf{y}\right) \neq u_i\right] \\ &\leqslant \mathbb{P}\left[\mathcal{E}_i\right] \leqslant Z\left(W_N^{(i)}\right)\end{aligned}$$

where $\mathcal{E}_i$ is defined in (1.35). □

Corollary 3.1. *If the B-DMC W is BEC and the decision function* (1.27) *is randomized such that the ties are broken in favor of* 0 *or* 1 *randomly with equal probability, for $i \in \mathcal{A}$*

$$\mathbb{P}\left[G_N^{(i)} = 1\right] = \frac{1}{2} Z\left(W_N^{(i)}\right) \tag{3.3}$$

Proof. If W is a BEC, all forged channels $(W_N^{(i)})$ are BEC with erasure probability $Z\left(W_N^{(i)}\right)$. Moreover, the events $\mathcal{E}_i$ only include the erasures, namely:

$$\mathcal{E}_i = \left\{(\mathbf{u}, \mathbf{y}) \in \mathcal{X}^N \mathcal{Y}^N : W_N^{(i)}(\mathbf{y}, \mathbf{u}_0^{i-1} | u_i) = W_N^{(i)}(\mathbf{y}, \mathbf{u}_0^{i-1})\right\},$$

and $\mathbb{P}\left[\mathcal{E}_i\right] = Z\left(W_N^{(i)}\right)$.

In case of erasure the decoded bit is correct with probability $\frac{1}{2}$, hence:

$$\mathbb{P}\left[G_N^{(i)} = 1\right] = \frac{1}{2} \mathbb{P}\left[\mathcal{E}_i\right] \tag{3.4}$$

□

We are interested in total number of genie helps, which we define by

$$S_I \triangleq \sum_{i \in \mathcal{A}} G_N^{(i)}. \tag{3.5}$$

This quantity gives us useful information about the performance of ordinary SC decoder. For example probability of correct decoding is equal to the probability of the event $\mathcal{E}^C = \{S_I = 0\}$. Hence we attempt to characterize the distribution of this random variable.

S_I is sum of non identical and correlated binary random variables and it could be difficult to obtain its distribution. Difference in the distributions of binary random variables $G_N^{(i)}$ is obvious from Proposition 3.1. However, the dependence between them might not be clear.

To see the dependence, we can simply consider the following example: Assume the B-DMC is a BEC and for the code of block length $N = 2^n$, both indices $i = N - 2$ and $j = N - 1$ are in the information bits set. Note that those channels are obtained by "minus" and "plus" operations on the same erasure channel at level $n-1$. We know from the Lemma 1.5 that if the plus channel erases the minus channel should have erased. Hence, if $G_N^{(j)} = 1$, then $G_N^{(i)}$ will be 1 with probability $\frac{1}{2}$.

Proposition 3.2. *In genie-aided decoding, average number of genie helps is upper bounded as:*

$$\mathbb{E}\left[S_I\right] \leqslant \sum_{i \in \mathcal{A}} Z\left(W_N^{(i)}\right). \tag{3.6}$$

When W is a BEC we have

$$\mathbb{E}\left[S_I\right] = \frac{1}{2} \sum_{i \in \mathcal{A}} Z\left(W_N^{(i)}\right). \tag{3.7}$$

Proof. The result is easily obtained by taking the expectation from both sides of 3.5 and applying the results of Proposition 3.1 and Corollary 3.1. □

3.1.1 Correlation between polarized BEC channels

The main difficulty in characterizing the behavior of the genie is due to the correlation between the random variables $G_N^{(i)}$. In this section we derive the second order statistics of those random variables, when the underlying B-DMC W is a BEC, recursively.

Let's start by a single BEC where the binary random variable $E_1^{(0)}$ denotes the indicator of the event that the BEC erases. The following equalities are obvious:

$$\mathbb{E}\left[E_1^{(0)}\right] = \epsilon.$$
$$\mathbb{E}\left[E_1^{(0)^2}\right] = \epsilon.$$

We take two independent copies of that channel and name the indicator variable for the second channel $E_1^{(1)'}$. Note that due to independence we have

$$\mathbb{E}\left[E_1^{(0)} E_1^{(0)'}\right] = \epsilon^2.$$

We apply the polar transform to obtain two channels at level 1 which are both BECs (denoted by $W_2^{(0)}$ and $W_2^{(1)}$). By Lemma 1.5 we know if minus channel erases if *either* of the initial channels erase while the plus one erases when *both* of them erase. Hence the indicator functions for the new forged BECs are:

$$\begin{aligned} E_2^{(0)} &= E_1^{(0)} + E_1^{(0)'} - E_1^{(0)} E_1^{(0)'} \\ E_2^{(1)} &= E_1^{(0)} E_1^{(0)'} \end{aligned}$$

Those recursions are true for each step of polarization. Assuming we have $N = 2^n$ polarized channels at level n, in order to obtain the polarized channels at level $n+1$ we take two independent copies of this N channels and apply the polar transform to each pair of them. Hence, $2N$ indicator variables at level $n+1$ will be:

$$E_{2N}^{(2i)} = E_N^{(i)} + E_N^{(i)'} - E_N^{(i)} E_N^{(i)'}, \tag{3.8a}$$

$$E_{2N}^{(2i+1)} = E_N^{(i)} E_N^{(i)'} \tag{3.8b}$$

for $i = 0, 1, \cdots, N-1$.

Having the first order and second order statistics for the indicator variables $E_N^{(i)}$ $i = 0, 1, \ldots, N-1$:

$$Z_N^{(i)} \triangleq \mathbb{E}\left[E_N^{(i)}\right], \tag{3.9}$$

$$R_N^{(i,j)} \triangleq \mathbb{E}\left[E_N^{(i)} E_N^{(j)}\right], \tag{3.10}$$

we can obtain the first and second order statistics for the indicator variables at level $l+1$ as follows:

First we easily find the first order statistics:

$$\begin{aligned}
\mathbb{E}\left[E_{2N}^{(2i)}\right] &= \mathbb{E}\left[E_N^{(i)} + E_N^{(i)'} - E_N^{(i)} E_N^{(i)'}\right] \\
&= \mathbb{E}\left[E_N^{(i)}\right] + \mathbb{E}\left[E_N^{(i)'}\right] - \mathbb{E}\left[E_N^{(i)} E_N^{(i)'}\right] \\
&= \mathbb{E}\left[E_N^{(i)}\right] + \mathbb{E}\left[E_N^{(i)'}\right] - \mathbb{E}\left[E_N^{(i)}\right] \mathbb{E}\left[E_N^{(i)'}\right] \\
&= 2Z_N^{(i)} - {Z_N^{(i)}}^2
\end{aligned}$$

where we used the independence of $E_N^{(i)}$ and $E_N^{(i)'}$. Similarly:

$$\begin{aligned}
\mathbb{E}\left[E_{2N}^{(2i+1)}\right] &= \mathbb{E}\left[E_N^{(i)} E_N^{(i)'}\right] = \mathbb{E}\left[E_N^{(i)}\right] \mathbb{E}\left[E_N^{(i)'}\right] \\
&= {Z_N^{(i)}}^2
\end{aligned}$$

Recall that the recursions we just obtained are exactly (1.17b) (with equality for BEC) and (1.17a).

For the second order statistics we have:

$$\begin{aligned}
\mathbb{E}\left[E_{2N}^{(2i)} E_{2N}^{(2j)}\right] &= \mathbb{E}\left[\left(E_N^{(i)} + E_N^{(i)'} - E_N^{(i)} E_N^{(i)'}\right)\left(E_N^{(j)} + E_N^{(j)'} - E_N^{(j)} E_N^{(j)'}\right)\right] \\
&= \mathbb{E}\Big[E_N^{(i)} E_N^{(j)} + E_N^{(i)} E_N^{(j)'} - E_N^{(i)} E_N^{(j)} E_N^{(j)'} \\
&\quad + E_N^{(i)'} E_N^{(j)} + E_N^{(i)'} E_N^{(j)'} - E_N^{(i)'} E_N^{(j)} E_N^{(j)'} \\
&\quad - E_N^{(i)} E_N^{(i)'} E_N^{(j)} - E_N^{(i)} E_N^{(i)'} E_N^{(j)'} + E_N^{(i)} E_N^{(j)} E_N^{(i)'} E_N^{(j)'}\Big] \\
&\overset{(a)}{=} R_N^{(i,j)} + Z_N^{(i)} Z_N^{(j)} - R_N^{(i,j)} \\
&\quad + Z_N^{(i)} Z_N^{(j)} + R_N^{(i,j)} - Z_N^{(j)} R_N^{(i,j)} \\
&\quad - R_N^{(i,j)} - Z_N^{(i)} R_N^{(i,j)} + R_N^{(i,j)} R_N^{(i,j)} \\
&= 2Z_N^{(i)} Z_N^{(j)} - 2(Z_N^{(i)} + Z_N^{(j)} - 1) R_N^{(i,j)} + {R_N^{(i,j)}}^2
\end{aligned}$$

where in (a) we have again used the independence between the variables with *prime* and the ones without that.

Moreover, we have:

$$\begin{aligned}\mathbb{E}\left[E_{2N}^{(2i)}E_{2N}^{(2j+1)}\right] &= \mathbb{E}\left[\left(E_N^{(i)}+E_N^{(i)\prime}-E_N^{(i)}E_N^{(i)\prime}\right)\left(E_N^{(j)}E_N^{(j)\prime}\right)\right]\\ &= \mathbb{E}\left[E_N^{(i)}E_N^{(j)}E_N^{(j)\prime}+E_N^{(i)\prime}E_N^{(j)}E_N^{(j)\prime}-E_N^{(i)}E_N^{(j)}E_N^{(i)\prime}E_N^{(i)\prime}\right]\\ &= R_N^{(i,j)}Z_N^{(j)}+Z_N^{(j)}R_N^{(i,j)}-{R_N^{(i,j)}}^2\\ &= 2Z_N^{(j)}R_N^{(i,j)}-{R_N^{(i,j)}}^2\end{aligned}$$

and by symmetry:

$$\mathbb{E}\left[E_{2N}^{(2i+1)}E_{2N}^{(2j)}\right] = 2Z_N^{(i)}R_N^{(i,j)}-{R_N^{(i,j)}}^2.$$

Finally:

$$\begin{aligned}\mathbb{E}\left[E_{2N}^{(2i+1)}E_{2N}^{(2j+1)}\right] &= \mathbb{E}\left[E_N^{(i)}E_N^{(i)\prime}E_N^{(j)}E_N^{(j)\prime}\right]\\ &= R_N^{(i,j)}R_N^{(i,j)}\\ &= {R_N^{(i,j)}}^2\end{aligned}$$

Hence, the second (and first) order statistics of $2N$ indicator variables at level $l+1$ will be:

$$Z_{2N}^{(2i)} = 2Z_N^{(i)}-{Z_N^{(i)}}^2, \tag{3.11a}$$

$$Z_{2N}^{(2i+1)} = {Z_N^{(i)}}^2. \tag{3.11b}$$

$$R_{2N}^{(2i,2j)} = 2Z_N^{(i)}Z_N^{(j)}-2(Z_N^{(i)}+Z_N^{(j)}-1)R_N^{(i,j)}+{R_N^{(i,j)}}^2, \tag{3.12a}$$

$$R_{2N}^{(2i,2j+1)} = 2Z_N^{(j)}R_N^{(i,j)}-{R_N^{(i,j)}}^2, \tag{3.12b}$$

$$R_{2N}^{(2i+1,2j)} = 2Z_N^{(i)}R_N^{(i,j)}-{R_N^{(i,j)}}^2, \tag{3.12c}$$

$$R_{2N}^{(2i+1,2j+1)} = {R_N^{(i,j)}}^2. \tag{3.12d}$$

Alternatively, instead of the second order moments $R_N^{(i,j)} \triangleq \mathbb{E}\left[E_N^{(i)}E_N^{(j)}\right]$ we can work with the correlations defined as:

$$C_N^{(i,j)} \triangleq \mathbb{E}\left[E_N^{(i)}E_N^{(j)}\right]-\mathbb{E}\left[E_N^{(i)}\right]\mathbb{E}\left[E_N^{(j)}\right]. \tag{3.13}$$

In this case, the equations (3.12a) to (3.12d) would be translated to:

$$C_{2N}^{(2i,2j)} = 2\left(1-Z_N^{(i)}\right)\left(1-Z_N^{(j)}\right)C_N^{(i,j)}+{C_N^{(i,j)}}^2, \tag{3.14a}$$

$$C_{2N}^{(2i,2j+1)} = 2Z_N^{(j)}\left(1-Z_N^{(i)}\right)C_N^{(i,j)}+{C_N^{(i,j)}}^2, \tag{3.14b}$$

$$C_{2N}^{(2i+1,2j)} = 2Z_N^{(i)}\left(1-Z_N^{(j)}\right)C_N^{(i,j)}+{C_N^{(i,j)}}^2, \tag{3.14c}$$

$$C_{2N}^{(2i+1,2j+1)} = 2Z_N^{(i)}Z_N^{(j)}C_N^{(i,j)}+{C_N^{(i,j)}}^2. \tag{3.14d}$$

Lemma 3.1. *In general the mth order moments of the random variables $E_{2N}^{(i)},\quad i = 0, 1, \cdots, 2N-1$ can be computed by knowledge of the mth order moments of random variables $E_N^{(i)},\quad i = 0, 1, \cdots, N-1$.*

Proof. By mth order moment we mean:

$$\mathbb{E}\left[E_{2N}^{(j_1)} E_{2N}^{(j_2)} \cdots E_{2N}^{(j_m)}\right] \tag{3.15}$$

for some set of indices $j_1, j_2, \cdots, j_m$ which are *not* necessarily distinct.

Let $i = \lfloor \frac{j}{2} \rfloor$, and observe that $E_{2N}^{(j)}$ is a polynomial of degree 1 in each of $E_N^{(i)}$ and $E_N^{(i)\prime}$ (see (3.8a) and (3.8b)). This means in expansion of $E_{2N}^{(j_1)} E_{2N}^{(j_2)} \cdots E_{2N}^{(j_k)}$ we will have the terms in form of $E_N^{(i_1)} E_N^{(i_2)} \cdots E_N^{(i_l)} \times E_N^{(k_1)\prime} E_N^{(k_2)\prime} \cdots E_N^{(k_p)\prime}$ for some $l \leqslant m$ and $p \leqslant m$.

By independence of the variables with prime and the one without prime we see that the expectation of such product terms will be product of two expectations each of which is at most an mth order moment of the random variables $E_N^{(i)}$. $\square$

Having the second order statistics, we can also compute the variance of the random variable S_I. First let's redefine the indicator variables $G_N^{(i)}$ that we defined at the beginning of the section in terms of the indicator variables for erasures $E_N^{(i)}$s. Due to randomized tie-breaking in the decoder we have:

$$G_N^{(i)} = \begin{cases} \mathcal{B}\left(\frac{1}{2}\right) & \text{if } E_N^{(i)} = 1, \\ 0 & \text{if } E_N^{(i)} = 0 \end{cases} \tag{3.16}$$

where $\mathcal{B}(p)$ denotes a binary random variable that is 1 with probability p. Observe that given $E_N^{(i)},\quad i \in \mathcal{I} \subset \{0, 1, \cdots, N-1\}$ corresponding $G_N^{(i)},\quad i \in \mathcal{I}$ are independent.

Consequently we have:

$$\begin{aligned}
\mathbb{E}\left[G_N^{(i)}\right] &= \mathbb{P}\left[G_N^{(i)} = 1\right] \\
&= \mathbb{P}\left[G_N^{(i)} = 1 | E_N^{(i)} = 1\right] \mathbb{P}\left[E_N^{(i)} = 1\right] + \mathbb{P}\left[G_N^{(i)} = 1 | E_N^{(i)} = 0\right] \mathbb{P}\left[E_N^{(i)} = 0\right] \\
&= \frac{1}{2} \mathbb{P}\left[E_N^{(i)} = 1\right] = \frac{1}{2} \mathbb{E}\left[E_N^{(i)}\right].
\end{aligned} \tag{3.17}$$

and

$$\begin{aligned}
\mathbb{E}\left[G_N^{(i)} G_N^{(j)}\right] &= \mathbb{P}\left[G_N^{(i)} = 1, G_N^{(j)} = 1\right] \\
&= \mathbb{P}\left[G_N^{(i)} = 1, G_N^{(j)} = 1 | E_N^{(i)} = 1, E_N^{(j)} = 1\right] \mathbb{P}\left[E_N^{(i)} = 1, E_N^{(j)} = 1\right] \\
&\quad + \mathbb{P}\left[G_N^{(i)} = 1, G_N^{(j)} = 1 | E_N^{(i)} = 0 \text{ or } E_N^{(j)} = 0\right] \mathbb{P}\left[E_N^{(i)} = 0 \text{ or } E_N^{(j)} = 0\right] \\
&= \frac{1}{2} \times \frac{1}{2} \times \mathbb{P}\left[E_N^{(i)} = 1, E_N^{(i)} = 1\right] = \frac{1}{4} \mathbb{E}\left[E_N^{(i)} E_N^{(j)}\right].
\end{aligned} \tag{3.18}$$

Note that (3.17) can be used alternatively to prove Corollary 3.1.

Proposition 3.3. *The variance of the number of genie helps S_I for BEC with $N = 2^l$ levels of polarization is:*

$$var[S_I] = \frac{1}{4} \sum_{i,j \in \mathcal{A}} C_N^{(i,j)} \tag{3.19}$$

where $C_N^{(i,j)}$ can be obtained recursively via (3.14a) *to* (3.14d)*.*

Proof. The claim is obtained easily by applying the definition of variance to (3.5)

$$\begin{aligned}
\operatorname{var}[S_I] &= \mathbb{E}\left[S_I^2\right] - \mathbb{E}[S_I]^2 \\
&= \sum_{i,j \in \mathcal{A}} \mathbb{E}\left[G_N^{(i)} G_N^{(j)}\right] - \sum_{i,j \in \mathcal{A}} \mathbb{E}\left[G_N^{(i)}\right] \mathbb{E}\left[G_N^{(j)}\right] \\
&= \sum_{i,j \in \mathcal{A}} \left(\mathbb{E}\left[G_N^{(i)} G_N^{(j)}\right] - \mathbb{E}\left[G_N^{(i)}\right] \mathbb{E}\left[G_N^{(j)}\right]\right) \\
&= \frac{1}{4} \sum_{i,j \in \mathcal{A}} \left(\mathbb{E}\left[E_N^{(i)} E_N^{(j)}\right] - \mathbb{E}\left[E_N^{(i)}\right] \mathbb{E}\left[E_N^{(j)}\right]\right) \\
&= \frac{1}{4} \sum_{i,j \in \mathcal{A}} C_N^{(i,j)}
\end{aligned}$$

where we have used (3.17) and (3.18). □

Even though the mean and variance are not sufficient to characterize a random variable completely, using the results of propositions 3.2 and 3.3 and the recursive relationships for BEC we can compute and plot the average and the variance of number of genie helps S_I for BEC at different rates to obtain some rough ideas about the behavior of S_I.

Recall that as $N \to \infty$ a fraction around $I(W)$ percent of the $Z\left(W_N^{(i)}\right)$s will go to 0 that correspond to the indices we put in the information set. This means that S_I will be sum of the random variables that are zero with probability 1 and hence its variance will also be zero.

However, in finite length setting, we see that the code does not perform very well as the rate approaches the capacity. This is due to some *non-polarized* channels that we have to include in the information bits set which contribute to S_I as some indicator variables that can be both zero and one with non-negligible probabilities which also increase the variance of S_I.

The performance of code is essentially determined by the probability of the event that $\{S_I \geqslant 1\}$ which is equivalent to the word-error event. Considering this point of view, we have the following lemma:

Lemma 3.2. *Polar coding rule minimizes an upper bound on the probability of the event $\mathcal{E} = \{S_I \geqslant 1\}$.*

Proof. Using Markov inequality:

$$\mathbb{P}[\mathcal{E}] = \mathbb{P}[S_I \geqslant 1] \leqslant \mathbb{E}[S_I] \leqslant \sum_{i \in \mathcal{A}} Z\left(W_N^{(i)}\right)$$

in general, and for the BEC:

$$\mathbb{P}[\mathcal{E}] = \mathbb{P}[S_I \geqslant 1] \leqslant \mathbb{E}[S_I] = \frac{1}{2}\sum_{i\in\mathcal{A}} Z\left(W_N^{(i)}\right).$$

Polar coding rule puts the channels with lowest Bhattacharyya parameter in the information bits set. □

In the view of the mentioned lemma, if one can characterize the exact distribution of S_I (which seems to be difficult) potentially better coding rules can be proposed in terms of the choice of information bit so as to $\mathbb{P}\{S_I \geqslant 1\}$ is minimized itself. Alternatively, it might be possible to show that the polar coding rule of Arıkan is equivalent to the optimal choice of information bits in the sense that it will minimizes the probability of error of the SC decoder.

Computations show that for the rates far below the capacity the average and variance of S_I will be very close to zero (as we expect), however as we see in Figure 3.1 as the rate approaches to the capacity of the channel the average of S_I will increase and, considering its standard deviation (plotted as vertical dotted lines), it can be non-zero with considerable probability. This is the region where the code's performance is poor. Note that we have intentionally plotted the curves for the rates above the capacity, the reason will be clear later.

To improve the performance of code at the rates close to the capacity, we shall consider the event $\mathcal{E}_1 \triangleq \{S_I = 1\}$ at the first step. We know that the error event $\mathcal{E}$ includes this event. However if this event constitutes a considerable part of the whole error events we can conclude that most of the error events are due to a single wrong decision whose detection (and correction) can be simple enough. Consequently code's performance could be improved significantly.

The curves in Figure 3.1 also confirm that considering the average and the standard deviation of S_I, at the rates close to the capacity (but not too close), its value can barely exceed one.

It seems to be very difficult to obtain further analytical results to characterize the distribution of S_I or at least the probability of the event $\{S_I = 1\}$. However, for the BEC we can do deeper analysis by defining a *hyperactive genie* and analyzing its behavior.

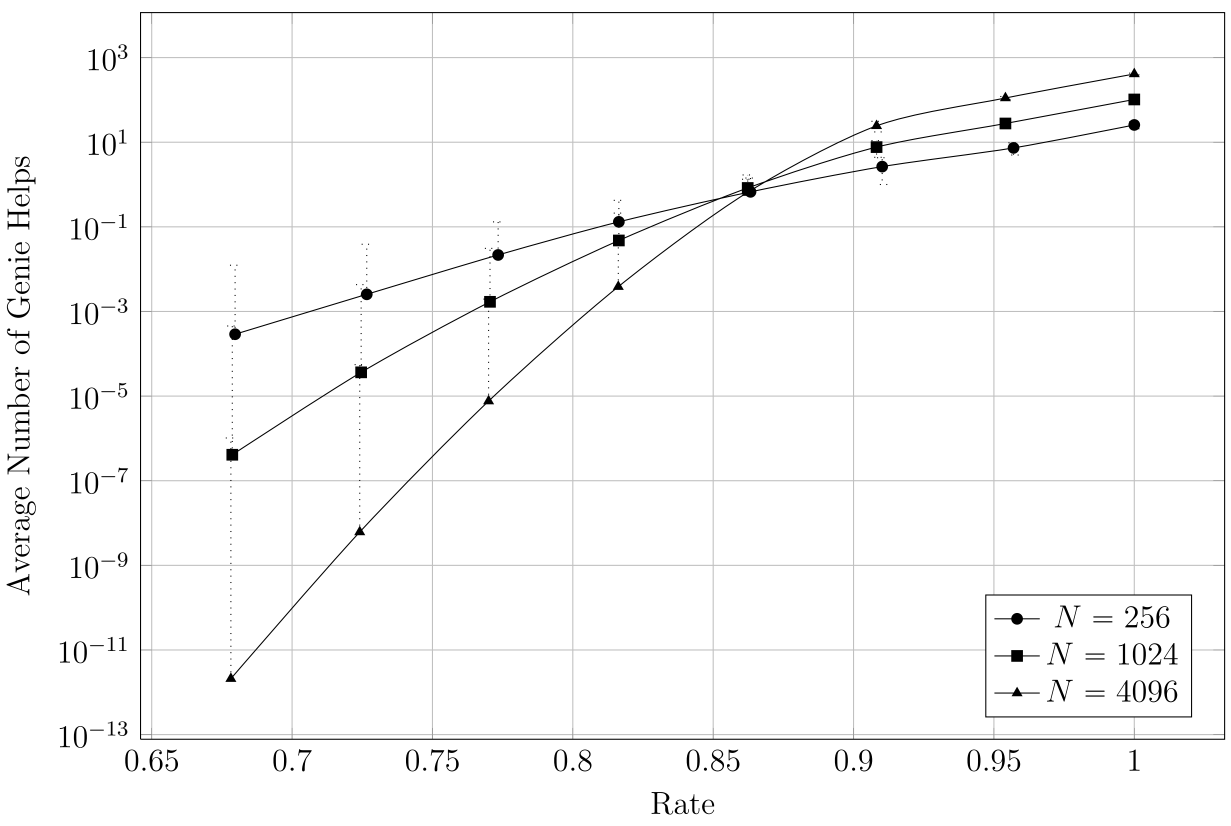
3.1.2 Hyperactive Genie

The hyperactive genie is a genie who acts on frozen bits as well as the information bits, although it is not necessary. More precisely, we saw that in an efficient implementation of the SC decoder the likelihood ratios of the frozen bits will also be calculated. In a genie-aided decoder, we can consider that decisions are also taken on the frozen bits and corrected by the hyper-active genie. The formalization in Algorithm 6 might be useful to understand the idea.

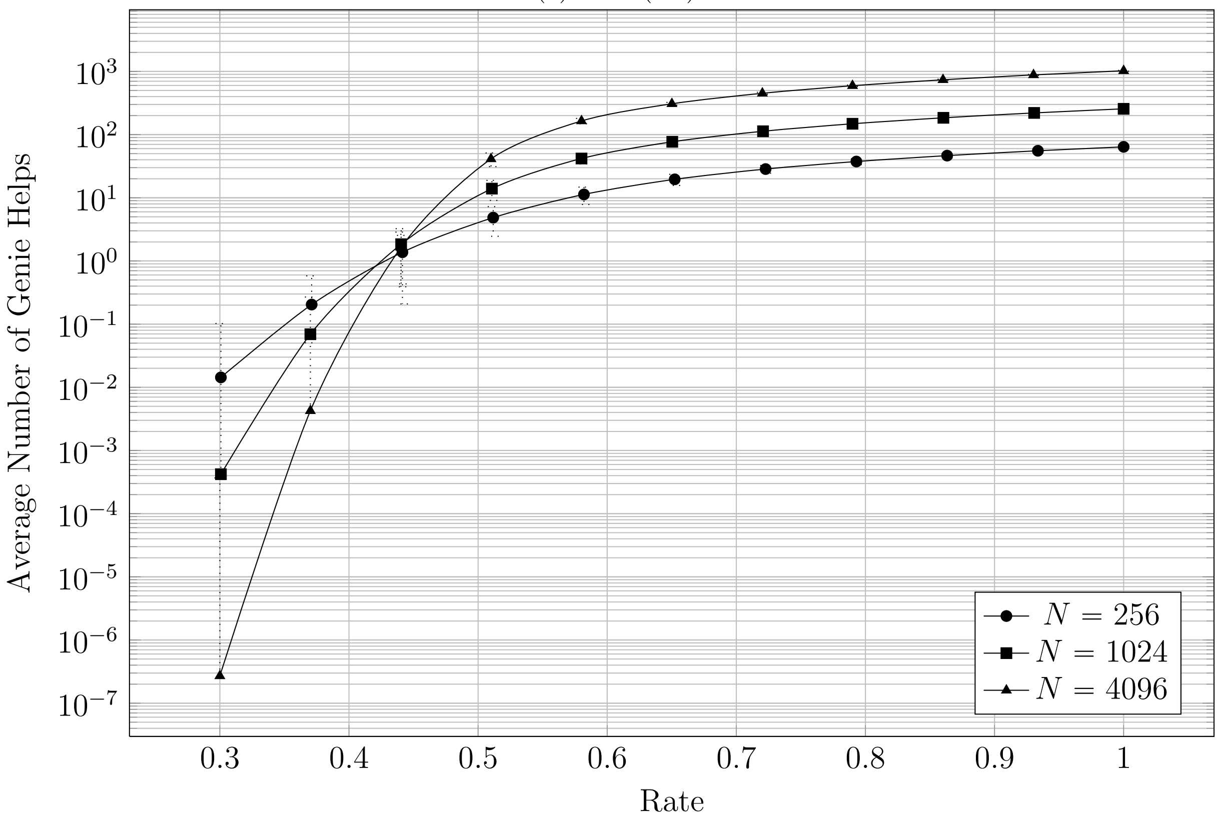
For the hyperactive genie the result of Proposition 3.1 is true without exceptions for frozen bits. Namely, (3.2) is valid for all $i = 0, 1, \cdots, N-1$.

We can similarly define the total number of the hyperactive genie helps as

$$S \triangleq \sum_{i=0}^{N-1} G_N^{(i)}. \tag{3.20}$$



(a) $BEC(0.1)$



(b) $BEC(0.5)$

Figure 3.1: Average number of Genie helps for BEC.

Algorithm 6 Successive Cancellation Decoding with Hyperactive Genie

```
Input: Channel output $\mathbf{y}$, $(N, K, \mathcal{A}, \mathbf{u}_{A^C})$ and information bits $\mathbf{u}_{\mathcal{A}}$.
Output: $\mathbf{g}$, a binary vector exhibiting genie's behavior.
  $\mathbf{g} \leftarrow \mathbf{0}$
  for all $i \in \{0, 1, \cdots, N-1\}$ do
    if $\frac{W_N^{(i)}(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}|0)}{W_N^{(i)}(\mathbf{y}, \hat{\mathbf{u}}_0^{i-1}|1)} \geqslant 1$ then
      $\hat{u}_i \leftarrow 0$
    else
      $\hat{u}_i \leftarrow 1$
    end if
    if $\hat{u}_i \neq u_i$ then
      $\hat{u}_i \leftarrow u_i$
      $g_i \leftarrow 1$
    end if
  end for
  return $\mathbf{g}$
```

We will shortly see that total number of this hyperactive genie helps follow a binomial distribution.

Lemma 3.3. *If the underlying B-DMC $W : \mathcal{X} \longrightarrow \mathcal{Y}$ is a BEC, in a genie-aided SC decoder, total number of erasures in polarized channels is equal to the total number of erasures at channel outputs.*

Proof. Assume that all-zero codeword is sent. The likelihood ratios on which the decoder decides are either 1 or ∞. Recall that the likelihood ratios are computed through recursive relationships (1.55a) and (1.55b).

First consider the simplest case of $N = 2$, which means we have two channel likelihood ratios ($\Lambda_c^{(0)}$ and $\Lambda_c^{(1)}$) based on which we compute:

$$\Lambda_2^{(0)} = \frac{\Lambda_c^{(0)}\Lambda_c^{(1)} + 1}{\Lambda_c^{(0)} + \Lambda_c^{(1)}}$$

and

$$\Lambda_2^{(1)} = \Lambda_c^{(0)}\Lambda_c^{(1)}$$

For computing the second likelihood ratio we used the assumption that all-zero codeword is sent and even if the first decision is not correct, the second likelihood ratio is calculated based on correct decision on the first bit (due to presence of the genie).

If both channel likelihood ratios are $+\infty$ (means no erasure) both of the obtained likelihood ratios will be $+\infty$ (i.e. non of the forged channels will erase). If either of them is 1 (means erasure on one of the channels) first likelihood ratio will be 1 (and hence one erasure at the decision level) and if both of them are one, the second likelihood ratio will also be one (which means two erasures at decision level).

Furthermore, observer that after one level of polarization the likelihood ratios are still either one or infinity.

For any level of polarization, the decision likelihood ratios are all obtained by recursive application of the equations we just mentioned.

Since the single step combination preserves the number of erasures and possible likelihood ratios we can conclude that the whole combination process preserves the number of erasures at the channel output. This proves our claim. □

Corollary 3.2. *Let the erasure indicator RVs $E_N^{(i)}$ be as defined in Section 3.1.1. The hamming weight of the random vector*

$$\mathbf{E}_N \triangleq \left[E_N^{(0)}, E_N^{(1)}, \cdots, E_N^{(N-1)}\right]^T$$

follows a binomial distribution with parameters N and ϵ. Namely:

$$\mathbb{P}\left[w_H\left(\mathbf{E}_N\right) = w\right] = \binom{N}{w} \epsilon^w \left(1-\epsilon\right)^{N-w}. \tag{3.21}$$

Proposition 3.4. *Total number of helps of the hyperactive genie follows a binomial distribution with parameters N and $\epsilon/2$ if the underlying B-DMC, W is a BEC. Namely:*

$$\mathbb{P}[S=j] = \binom{N}{j} \left(\frac{\epsilon}{2}\right)^j \left(1-\frac{\epsilon}{2}\right)^{N-j}. \tag{3.22}$$

Proof. First let's define the random vector

$$\mathbf{G}_N \triangleq \left[G_N^{(0)}, G_N^{(1)}, \ldots, G_N^{(N-1)}\right]^T$$

and note that $S = w_H\left(\mathbf{G}_N\right)$.

Observe that (according to (3.16)) given the random vector $\mathbf{E}_N$, $\mathbf{G}_N$ will be a random vector of i.i.d $\mathcal{B}\left(\frac{1}{2}\right)$ random variables on the positions where $E_N^{(i)}$ is one and zero elsewhere. Hence:

$$\mathbb{P}\left[w_H\left(\mathbf{G}_N\right) = j | \mathbf{E}_N = \mathbf{e}_N\right] = \binom{w_H\left(\mathbf{e}_N\right)}{j} \left(\frac{1}{2}\right)^{w_H\left(\mathbf{e}_N\right)}$$

So:

$$\begin{aligned}
\mathbb{P}\left[w_H\left(\mathbf{G}_N\right)=j\right] &= \sum_{\forall \mathbf{e}_N} \mathbb{P}\left[w_H\left(\mathbf{G}_N\right)=j \middle| \mathbf{E}_N=\mathbf{e}_N\right] \mathbb{P}\left[\mathbf{E}_N=\mathbf{e}_N\right] \\
&= \sum_{\forall \mathbf{e}_N} \binom{w_H\left(\mathbf{e}_N\right)}{j} \left(\frac{1}{2}\right)^{w_H\left(\mathbf{e}_N\right)} \mathbb{P}\left[\mathbf{E}_N=\mathbf{e}_N\right] \\
&= \sum_{w=0}^{N} \sum_{\forall \mathbf{e}_N : w_H\left(\mathbf{e}_N\right)=w} \binom{w_H\left(\mathbf{e}_N\right)}{j} \left(\frac{1}{2}\right)^{w_H\left(\mathbf{e}_N\right)} \mathbb{P}\left[\mathbf{E}_N=\mathbf{e}_N\right] \\
&= \sum_{w=0}^{N} \binom{w}{j} \left(\frac{1}{2}\right)^{w} \sum_{\forall \mathbf{e}_N : w_H\left(\mathbf{e}_N\right)=w} \mathbb{P}\left[\mathbf{E}_N=\mathbf{e}_N\right] \\
&= \sum_{w=0}^{N} \binom{w}{j} \left(\frac{1}{2}\right)^{w} \mathbb{P}\left[w_H\left(\mathbf{E}_N\right)=w\right] \\
&\overset{(a)}{=} \sum_{w=0}^{N} \binom{w}{j} \left(\frac{1}{2}\right)^{w} \binom{N}{w} \epsilon^{w} \left(1-\epsilon\right)^{N-w} \\
&\overset{(b)}{=} \sum_{w=j}^{N} \binom{w}{j} \binom{N}{w} \left(\frac{1}{2}\right)^{w} \epsilon^{w} \left(1-\epsilon\right)^{N-w}
\end{aligned}$$

where in (a) we used the results of Corollary 3.2 and in (b) the fact that the summand is zero for $w < j$. Now we can use the combinatorial identity $\binom{w}{j}\binom{N}{w} = \binom{N}{j}\binom{N-j}{w-j}$ and write:

$$\begin{aligned}
\mathbb{P}\left[w_H\left(\mathbf{G}_N\right)=j\right] &= \binom{N}{j} \sum_{w=j}^{N} \binom{N-j}{w-j} \left(\frac{1}{2}\right)^{w} \epsilon^{w} \left(1-\epsilon\right)^{N-w} \\
&= \binom{N}{j} \sum_{l=0}^{N-j} \binom{N-j}{l} \left(\frac{\epsilon}{2}\right)^{l+j} \left(1-\epsilon\right)^{N-j-l} \\
&= \binom{N}{j} \left(\frac{\epsilon}{2}\right)^{j} \sum_{l=0}^{N-j} \binom{N-j}{l} \left(\frac{\epsilon}{2}\right)^{l} \left(1-\epsilon\right)^{N-j-l} \\
&\overset{(c)}{=} \binom{N}{j} \left(\frac{\epsilon}{2}\right)^{j} \left(1-\frac{\epsilon}{2}\right)^{N-j}
\end{aligned}$$

where (c) is due to the fact that $\sum_{l=0}^{N-j} \binom{N-j}{l} \left(\frac{\epsilon}{2}\right)^{l} \left(1-\epsilon\right)^{N-j-l}$ is the binomial expansion of $\left(\frac{\epsilon}{2}+\left(1-\epsilon\right)\right)^{N-j}$. $\square$

Looking carefully at the curves of Figure 3.1 we see that at rate $R = 1$, average number of genie helps is exactly equal to $N\frac{\epsilon}{2}$ and its standard deviation is $\sqrt{N\frac{\epsilon}{2}\left(1-\frac{\epsilon}{2}\right)}$.

The hyperactive genie might simplify the analysis of the genie aided decoder. If we define $S_F \triangleq \sum_{i\in\mathcal{A}^C} G_N^{(i)}$, we see the random variable that we are interested in, S_I is:

$$S_I = S - S_F \tag{3.23}$$

Asymptotically, S_F is sum of the indicator variables $G_N^{(i)}$ over the indices in the frozen bits set. Since for $i \in \mathcal{A}^C$ around $N\epsilon$ of the erasure indicator variables $E_N^{(i)}$ will be almost always one and the rest zero, S_F will follow a binomial distribution with parameters $N\epsilon$ and $\frac{1}{2}$. We see that this variable is almost always concentrated around $N\frac{\epsilon}{2}$ (note that as $N \to \infty$, the standard deviation of S_F scales with $\sqrt{N}$ hence the ratio between its standard deviation and average goes to zero).

So, intuitively we can see that S_I's distribution should be the distribution of deviation of S around its mean. Looking from another point of view, if we ask to have m erasures in the polarized channels, due to polarization, first the $N(1-R)$ channels in $\mathcal{A}^C$ should erase and as a result S_F is more probable to take non-zero values. Then, the remaining $N(1-R)-m$ erasures must occur at the channels in $\mathcal{A}$ that contribute to the value of S_I. The challenge is to analytically show that S_F's value is around $N(1-R)$ with high probability.

Unfortunately we couldn't find such an analytical proof. One result that we obtained regarding the random variables S_F and S_I is the following:

Proposition 3.5. *Assume we partition the set of channels into the following three disjoint subsets:*

$$\begin{aligned}\mathcal{G}_\delta &= \{i : Z(W_N^{(i)}) \leqslant \delta\},\\ \mathcal{B}_\delta &= \{i : Z(W_N^{(i)}) \geqslant 1-\delta\},\\ \mathcal{M}_\delta &= \{i : \delta < Z(W_N^{(i)}) < 1-\delta\}.\end{aligned}$$

(Note that normally, $\mathcal{A} \supseteq \mathcal{G}_\delta$). We have the following bounds:

$$\mathbb{P}[S_F \geqslant |\mathcal{B}_\delta|] \geqslant \left(\frac{1-\delta}{2}\right)^{|\mathcal{B}_\delta|} \tag{3.24a}$$

$$\mathbb{P}[S_I \leqslant |\mathcal{M}_\delta|] \geqslant (1-\delta)^{|\mathcal{G}_\delta|} \tag{3.24b}$$

Proof. For the sake of brevity, we drop the subscript δ when we denote the subsets we defined in the argument.

$$\begin{aligned}\mathbb{P}[S_F \geqslant |\mathcal{B}|] &\geqslant \mathbb{P}\left[G_N^{(i)} = 1 : \forall i \in \mathcal{B}\right]\\ &= \mathbb{P}\left[G_N^{(i)} = 1 : \forall i \in \mathcal{B} | E_N^{(i)} = 1 : \forall i \in \mathcal{B}\right] \mathbb{P}\left[E_N^{(i)} = 1 : \forall i \in \mathcal{B}\right]\\ &= \left(\frac{1}{2}\right)^{|\mathcal{B}|} \prod_{i \in \mathcal{B}} \mathbb{P}\left[E_N^{(i)} = 1 | E_N^{(j)} = 1 : j < i, j \in \mathcal{B}\right]\\ &\overset{(a)}{\geqslant} \left(\frac{1}{2}\right)^{|\mathcal{B}|} \prod_{i \in \mathcal{B}} \mathbb{P}\left[E_N^{(i)} = 1\right]\\ &\geqslant \left(\frac{1}{2}\right)^{|\mathcal{B}|} (1-\delta)^{|\mathcal{B}|}\end{aligned}$$

where in (a) we have used the positive correlation between the channels: If channel j erases, the erasure probability of channel i can only be increased. Similarly we have:

$$\begin{aligned}
\mathbb{P}\left[S_I \leqslant |\mathcal{M}|\right] &\geqslant \mathbb{P}\left[G_N^{(i)} = 0 : \forall i \in \mathcal{G}\right] \\
&\geqslant \mathbb{P}\left[G_N^{(i)} = 0 : \forall i \in \mathcal{G} | E_N^{(i)} = 0 : \forall i \in \mathcal{G}\right] \mathbb{P}\left[E_N^{(i)} = 0 : \forall i \in \mathcal{G}\right] \\
&= \prod_{i \in \mathcal{G}} \mathbb{P}\left[E_N^{(i)} = 0 | E_N^{(j)} = 0 : j < i, j \in \mathcal{G}\right] \\
&\overset{(b)}{\geqslant} \prod_{i \in \mathcal{G}} \mathbb{P}\left[E_N^{(i)} = 0\right] \\
&\geqslant (1-\delta)^{|\mathcal{G}|}.
\end{aligned}$$

Again note that if channel j does not erase, this can only increase the probability that channel i does not erase and this yields (b). □

Results of Proposition 3.5 can be combined with the results on the rate of channel polarization and escape rate (in [2] and [4]) to see the asymptotic behavior of the genie although we are chiefly interested in its behavior in finite length regime in the present work.

3.1.3 Numerical Evidences

Since we couldn't analytically derive the distribution of number of genie-helps, we can estimate its empirical distribution from simulations.

In figures 3.2a to 3.2c we see (as we expected) for a range of the rates close to the channel's capacity, the distribution of genie helps is concentrated around 1. We have also plotted the probability of the following three events in Figure 3.3:

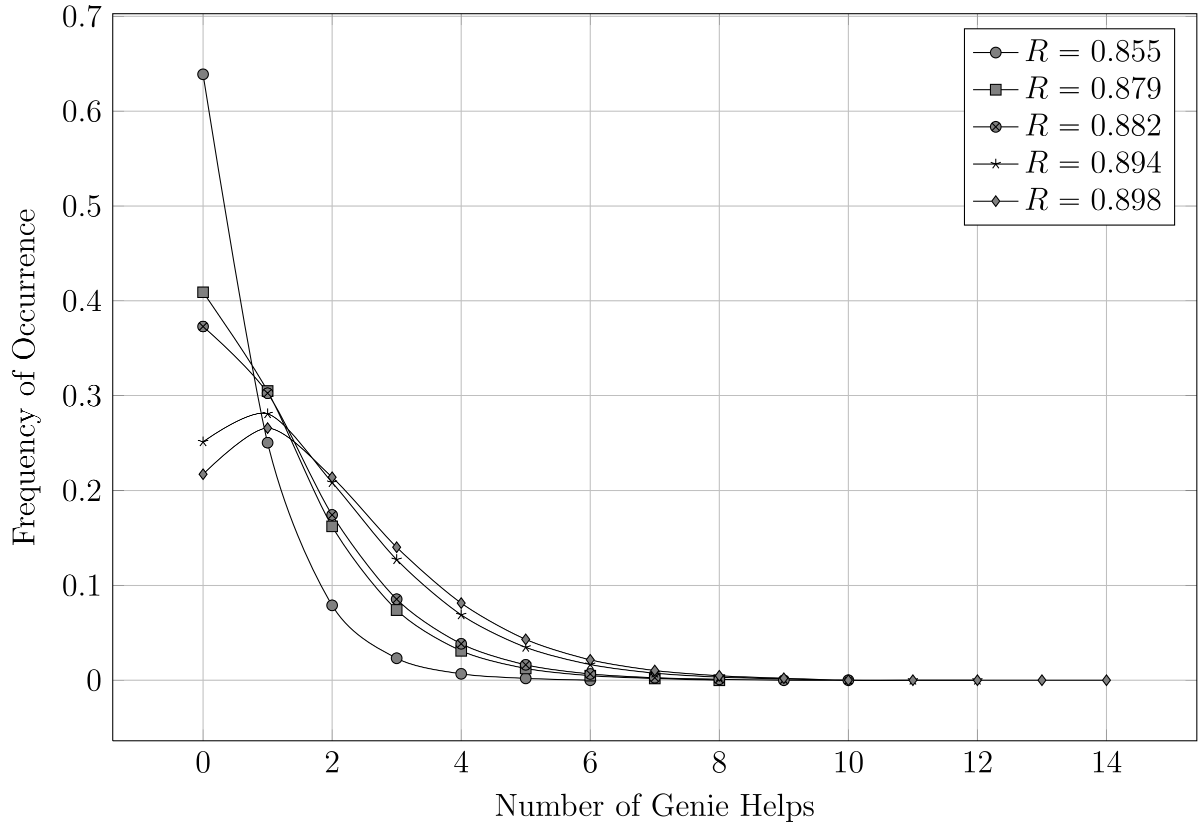
$$\begin{aligned}
\mathcal{E}^C &= \{S_I = 0\} \\
\mathcal{E}_1 &= \{S_I = 1\} \\
\mathcal{E}_{2^+} &= \{S_I \geqslant 2\}
\end{aligned}$$

These evidences confirm our guess that at the rates close to the channel's capacity, most of the decoding errors in SC decoder stem from a single deviation of the decoding path from the correct path.

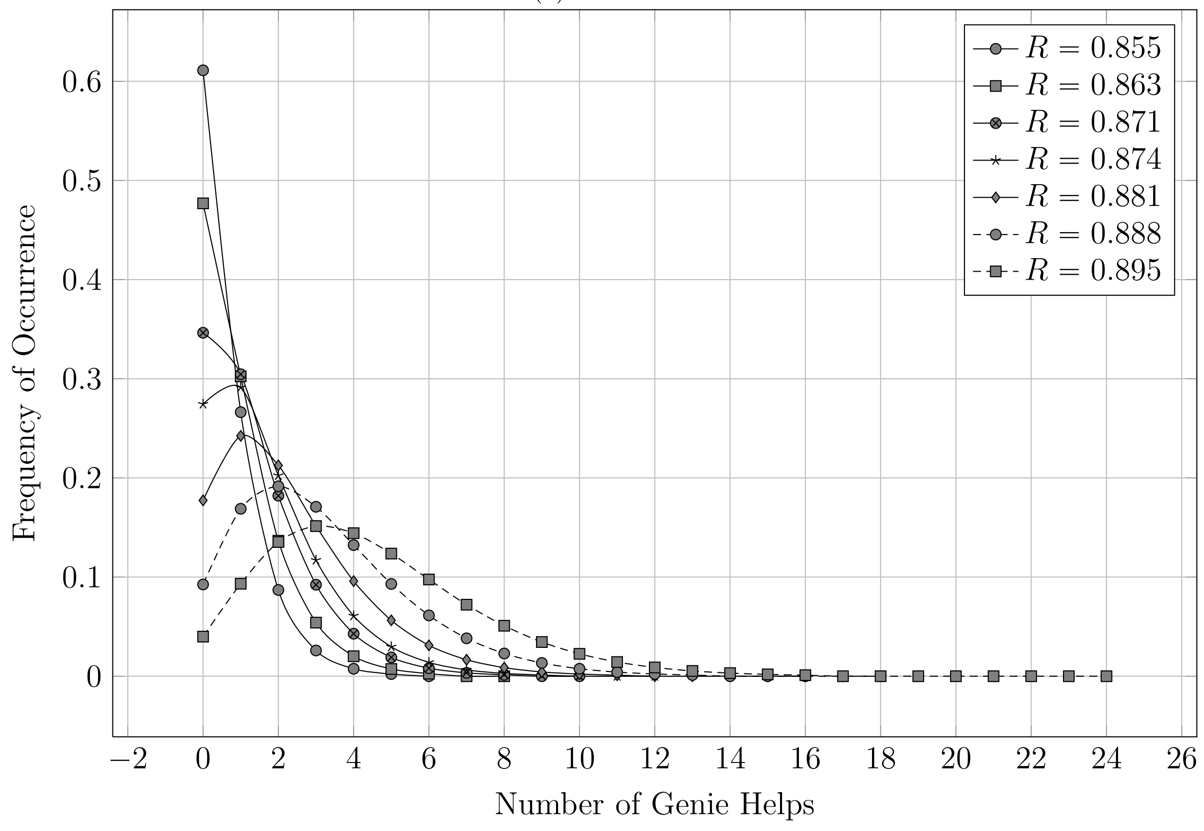
Furthermore, it is predictable that S_I should follow a bell-shaped distribution. This is consistent with our analytical results on $S = S_F + S_I$ whose distribution is binomial and the fact that we expect S_I to model the variations of S around its mean value.

3.2 Performance Improvement by Single Error Correction

The numerical results we obtained in the previous section suggest that for a range of rates not very close to the capacity the main source of SC decoding error events is only one wrong decision that causes the decoder to decode the wrong codeword. Note that

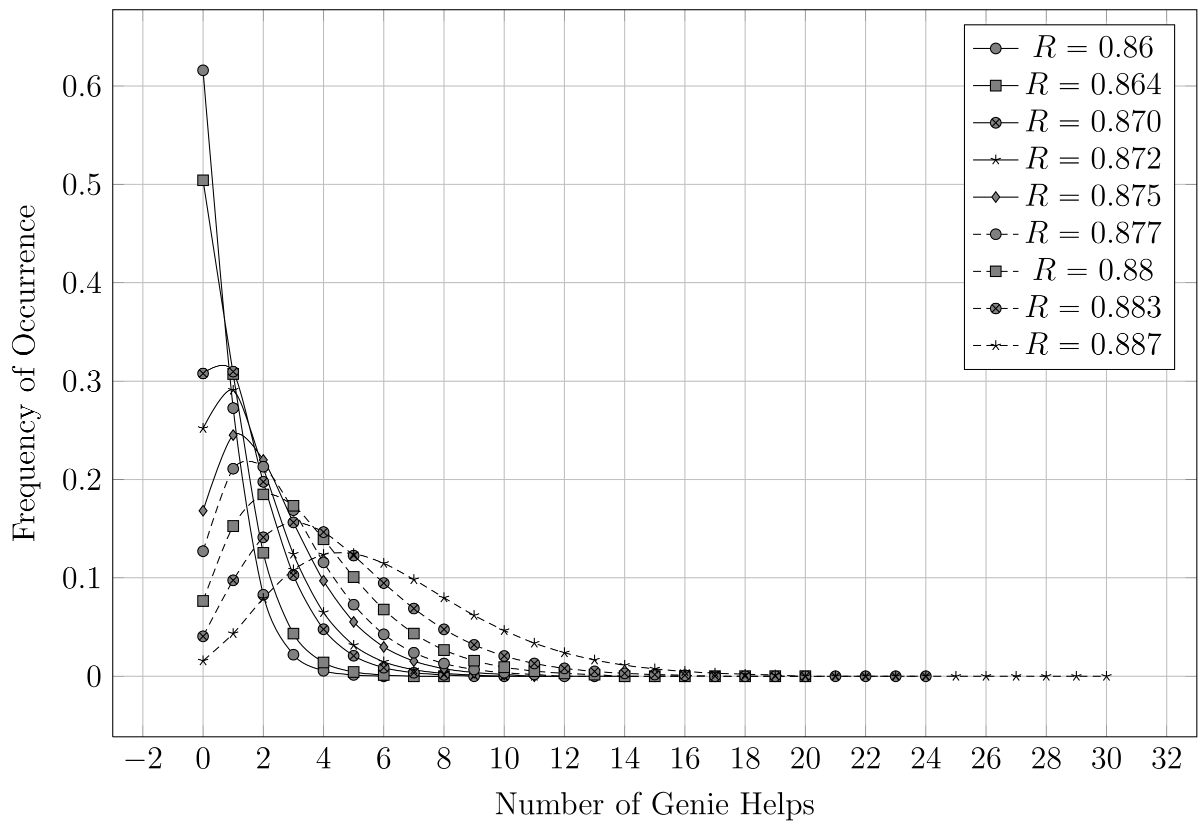


(a) $N = 256$



(b) $N = 1024$

Figure 3.2: Histogram of Number of Genie Helps for $BEC(0.1)$



(c) $N = 4096$

Figure 3.2: Histogram of Number of Genie Helps for $BEC(0.1)$.

this does not mean that most of wrongly decoded codewords differ only in one bit with the correct one since in the ordinary SC decoding the errors propagate and potentially all information bits following the bit which is correctly decoded in genie-aided decoder can be decoded wrongly.

However, we expect to be able to improve the performance of SC decoding by detecting and correcting that single decoding error and decrease the probability of error of decoding from $\mathbb{P}[\mathcal{E}_1] + \mathbb{P}[\mathcal{E}_{2^+}]$ to $\mathbb{P}[\mathcal{E}_{2^+}]$.

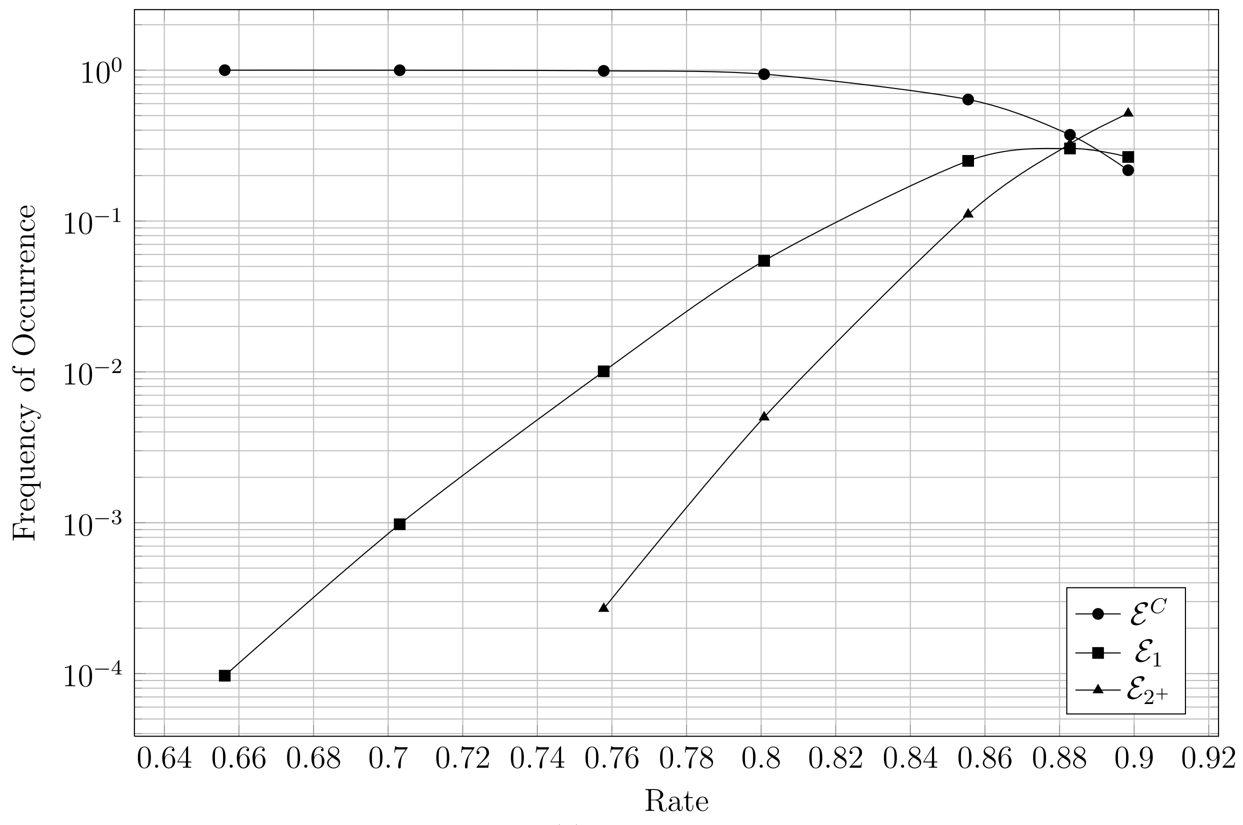
This modification improves the performance at the rates where $\mathcal{E}_1$ constitutes a considerable part of the whole error event $\mathcal{E} = \{S_I \geqslant 1\}$. Stating differently, improvements will take place at the rates that $\mathbb{P}[\mathcal{E}_1]$ is considerably larger than $\mathbb{P}[\mathcal{E}_{2^+}]$. The curves in Figure 3.3 show that such a region exists.

One can define an *improvement factor* as

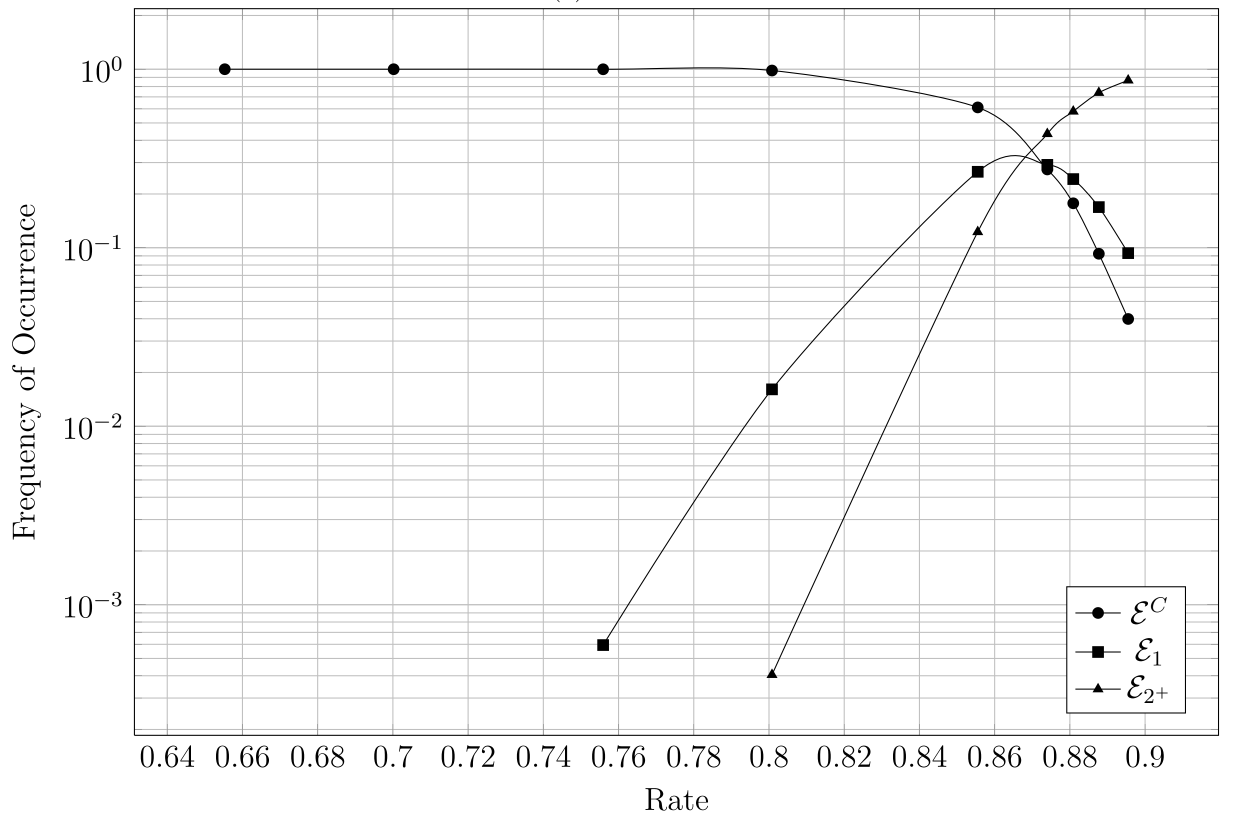
$$\alpha(R) \triangleq \frac{\mathbb{P}[\mathcal{E}_1]}{\mathbb{P}[\mathcal{E}_{2^+}]}. \tag{3.25}$$

in order to have a mathematical figure on the performance improvement we can obtain by single error correction.

In Figure 3.4 see the plot of α calculated from our numerical results. Figure 3.5 shows that the performance can be improved significantly in moderate rates if we could have a genie that helps us only once, which is equivalent to correcting the decoding errors that

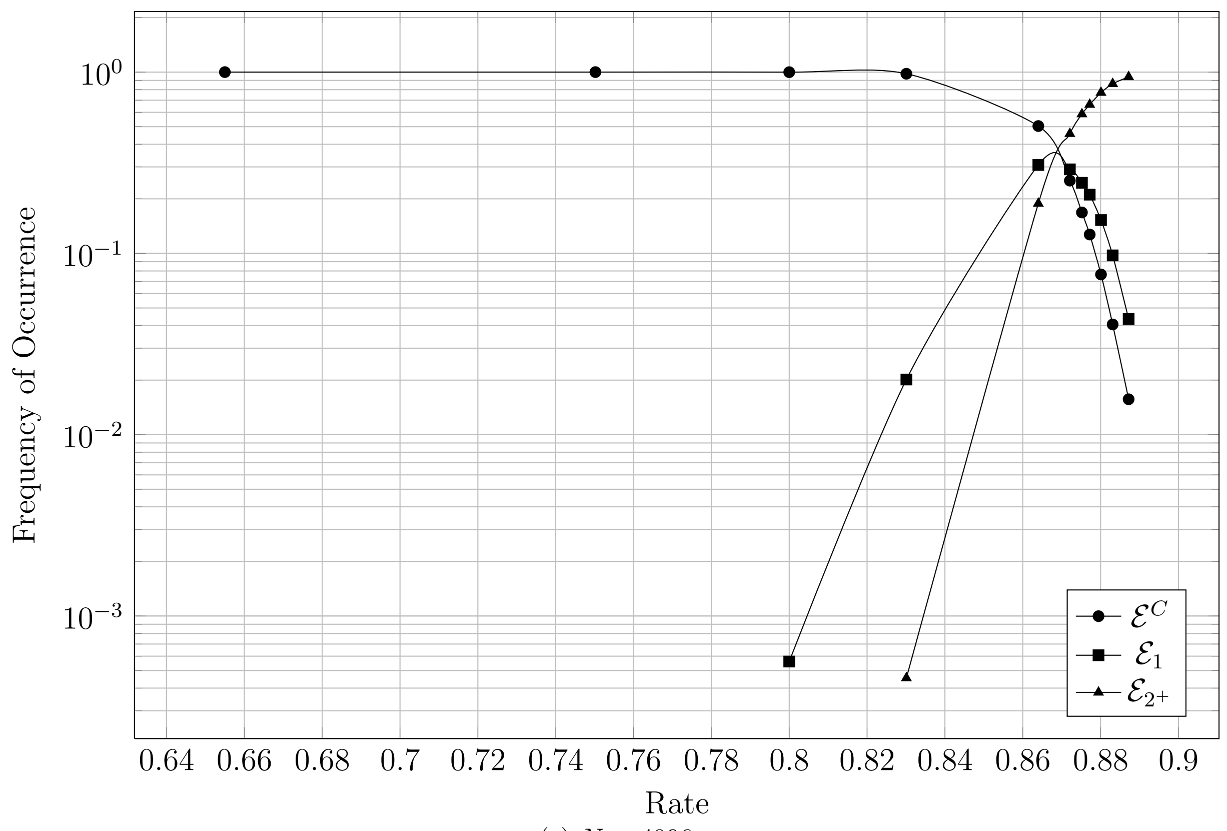


(a) $N = 256$



(b) $N = 1024$

Figure 3.3: Probability of events $\mathcal{E}^C$, $\mathcal{E}_1$ and $\mathcal{E}_{2^+}$ for $BEC(0.1)$.



(c) $N = 4096$

Figure 3.3: Probability of events $\mathcal{E}^C$, $\mathcal{E}_1$ and $\mathcal{E}_{2^+}$ for $BEC(0.1)$.

stem from a single wrong decision (note that these curves are simply obtained by plotting $\mathbb{P}[\mathcal{E}_1] + \mathbb{P}[\mathcal{E}_{2^+}]$ and $\mathbb{P}[\mathcal{E}_{2^+}]$ in different rates).

Now the challenge is to see how we can detect the index where the genie should help the decoder to improve the performance. The following methods can be proposed:

Likelihood Test A single wrong decision will propagate to potentially all of the following bits. Hence, the resulting codeword would most probably be an atypical sequence. The decoder can re-encode the decoded codeword and compute the likelihood of the result which will be usually very low in case of a decoding error.

The decoder can use this evidence as a sign of decoding error and then try to re-decode the codeword and flip the value of first information bit (assuming the wrong decision was on the first bit) and pass the result through the likelihood test again. Then, this process can be repeated until the resulting codeword passes the likelihood test.

This method increases the worst case complexity of SC decoder from $O(N \log N)$ to $O(N^2 \log N)$ (since if the wrong decision is on the last bit, the decoder should roughly repeat the above process N times to find the correct result).

Moreover, it is possible that this method fails which is the case that more than one wrong decisions are made in the decoding process (whose equivalent genie-aided decoder event is $\{S_I \geqslant 2\}$).

However, this method can only improve the performance of SC decoding up to the ML decoding. Since the output of the mentioned algorithm can be a codeword which

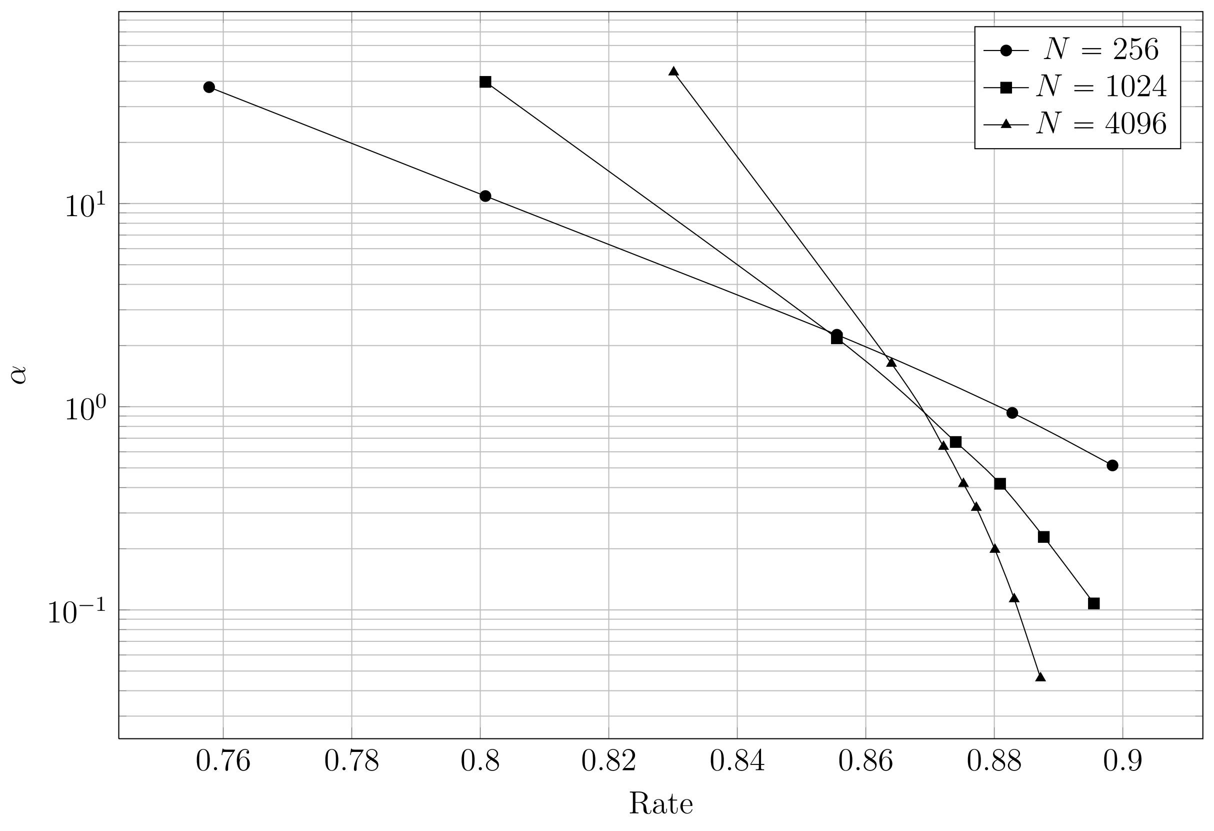


Figure 3.4: Single error correction improvement factor for $BEC(0.1)$ as defined in (3.25).

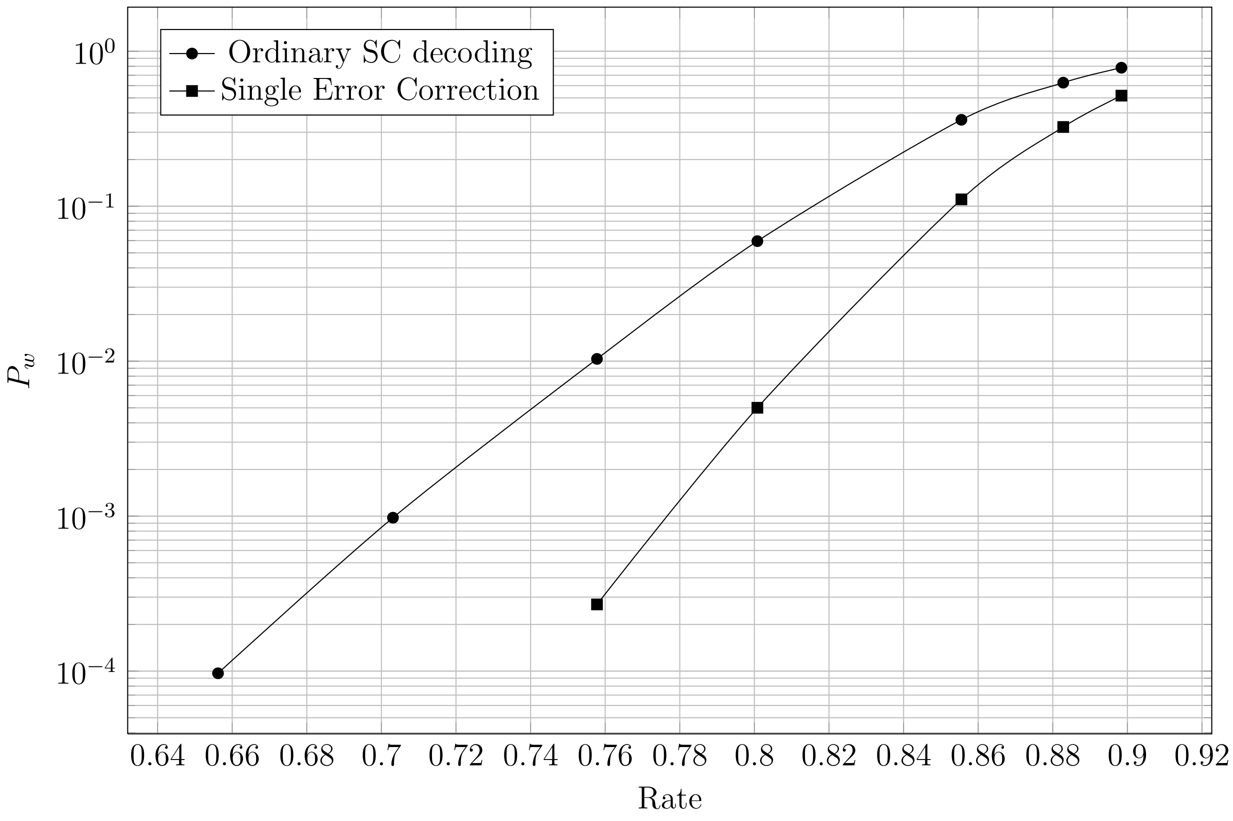
passes the likelihood test (is a likely codeword) but is not the sent codeword. In the other words, we hope the result of this method to be equivalent to asking a genie to correct the first wrong decision of an ordinary SC decoder while the result can be flipping a correct decision and keeping the wrong decision that all yield a wrong codeword that passes the likelihood test.

Frozen bits for BEC We saw previously that BEC is a special case in the sense that all forged channels will be BEC.

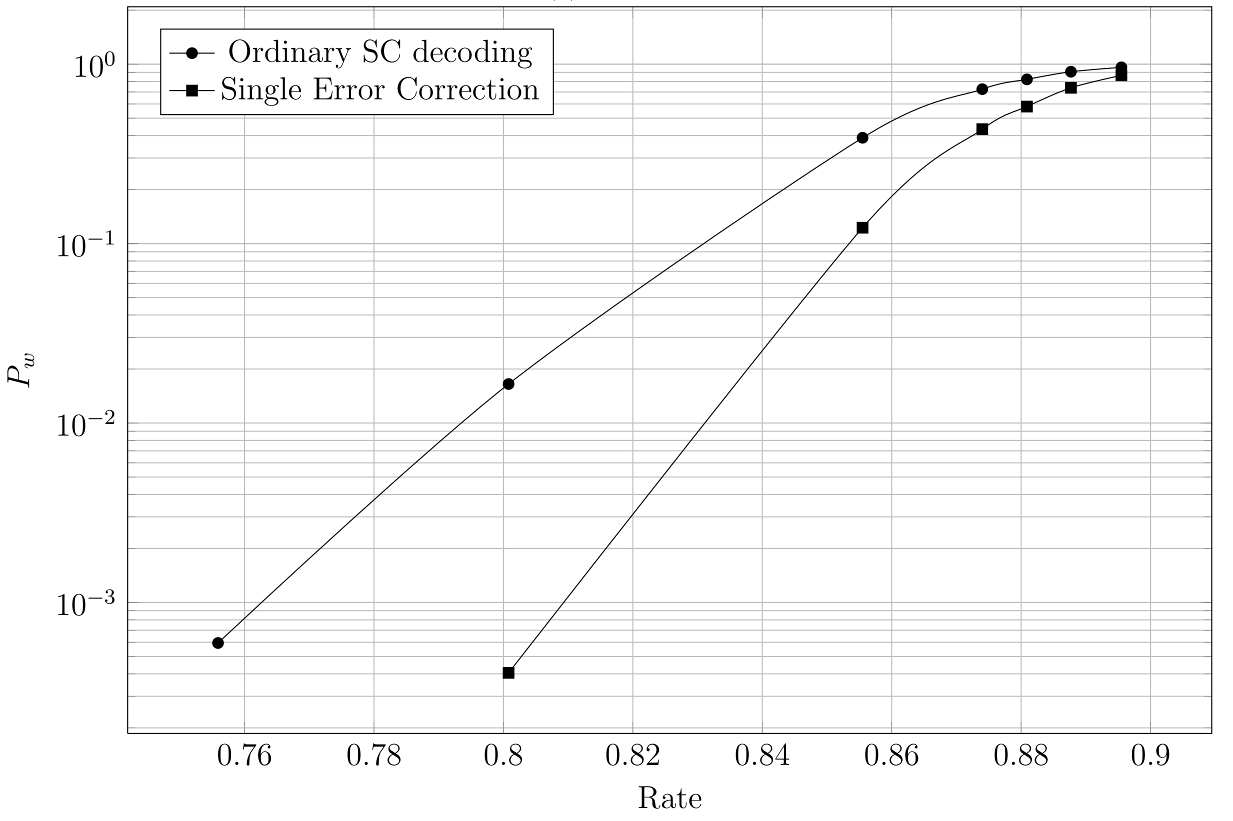
Assume the frozen bits are all set to zero at the encoder. The decoder starts by computing the likelihood ratio of the first bit which is most probably frozen and since the channel $W_N^{(1)}$ is a BEC, the result can only be 1 or $+\infty$. Then the decoder proceeds to calculate the likelihood ratios of the following bits until it reaches the next frozen bit. Let's denote its index by i. We know that if all information bits before i are decoded correctly, the likelihood ratio $\Lambda_N^{(i)}$ should be again either 1 or $+\infty$, hence if this value is calculated as 0 we will deduce that one of the information bits before bit number i must be decoded wrongly.

Then the decoder can come back and flip the value of the first information bit and re-decode the following bits and check if $\Lambda_N^{(i)}$ is again 0 or not?

This method can reduce the average complexity of the decoding compared to the likelihood test, but the worst case complexity will be the same. It is not clear if this method will be equivalent to asking the genie to correct the first wrong decision or it can result in flipping some correct decisions.

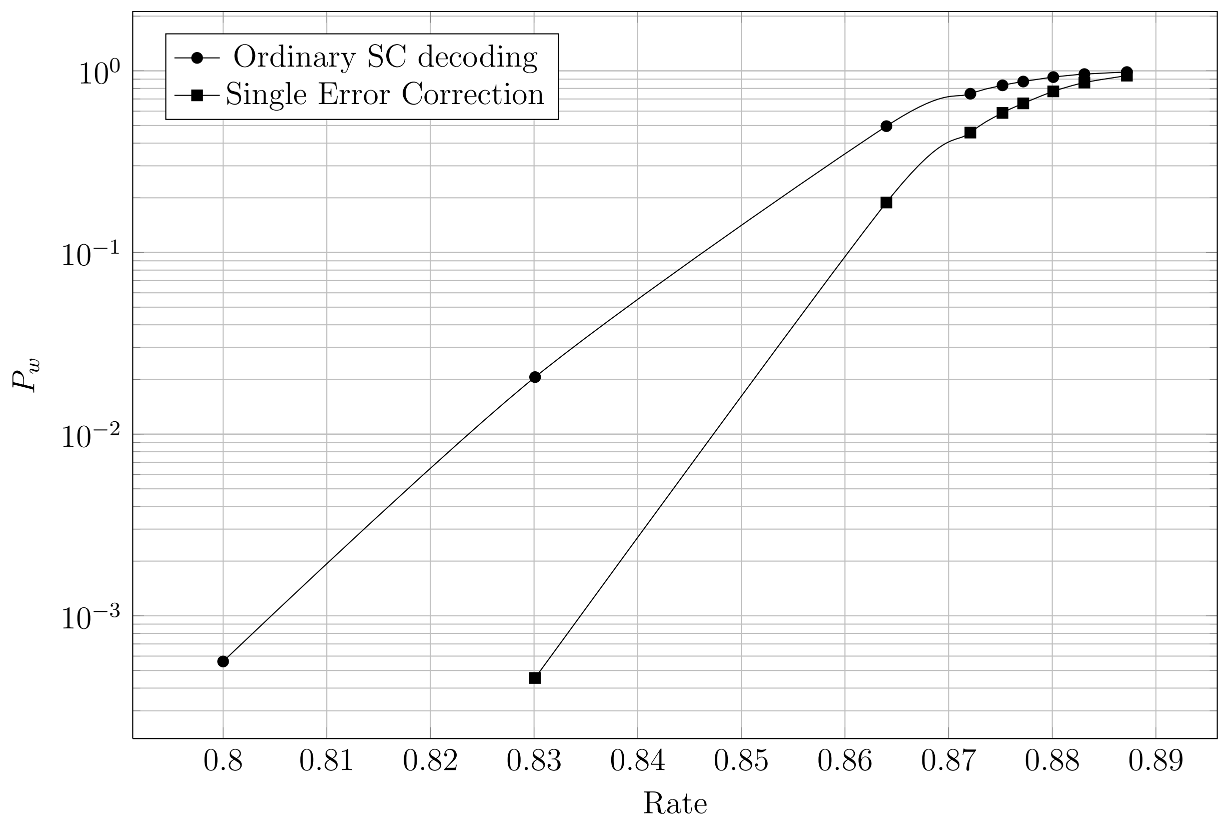


(a) $N = 256$



(b) $N = 1024$

Figure 3.5: Single error correction significantly improves the finite-length performance of polar codes.



(c) $N = 4096$

Figure 3.5: Single error correction significantly improves the finite-length performance of polar codes.

Discovering other possible methods for detecting the single wrong decision is an interesting question to be studied in future. One class of approaches can be based on adding some structured redundancies (like parity checks) to information bits so that the decoder can use them as a hint for correctness of decisions. These methods naturally decrease the effective rate of the code. However, if this rate loss is not significant they can still be useful to improve the performance of polar codes.

Another approach might be to statistically analyze the position of wrong decision in the genie-aided decoder and try to relate it to Bhattacharyya parameter of the channel on which the wrong decision is taken and the absolute value of the log-likelihood ratio (which measures the reliability of decisions as we discussed in Chapter 1) that caused the wrong decision. These are natural measures as one expects the decision errors to be more likely to happen on less polarized channels and as a result of marginal likelihood ratios. Based on these analysis one might be able to come up with an scheme that detects the position of wrong decision with high probability.

4 Non-binary Channel Coding

In many communication systems, it is always desirable to use high-order modulation schemes in order to improve on bandwidth efficiently. In these situations, one *symbol* is obtained by combining several *bit*s and sent over the communication channel. From an information theoretic point of view, we see that as the communication SNR increases the (Shannon) capacity of the channel increases beyond one bit per channel use. So, in order to reach the capacity it is essential to send more than one *binary digit*s per channel use.

Error correcting codes can also be used here to achieve the capacity of this high SNR channel. This channel is not a binary channel anymore, hence the binary error correcting codes are no longer suitable for it.

In order to deal with this non-binary channel, one has two alternatives: to design and employ the error correcting codes for channels with more than two input alphabets or the classical multilevel coding scheme where a non-binary input symbol is constructed by combining bits of different codewords (from different codes).

In this chapter we focus on employing polar codes for channels with non-binary input using both of the mentioned approaches and compare them.

It is worthwhile to note that in any communication system, one can always make hard decisions on channel output and decode it into discrete symbols and then use an error correcting code for the equivalent discrete output channel or alternatively see the signal level channel as a channel with continuous output alphabet and design a code for that channel which is technically known as feeding the decoder with soft decisions.

Usually the latter is preferable in terms of the rates one can achieve. In fact, data processing inequality implies one can never improve the capacity by making hard decisions and converting the continuous output channel to a discrete output one.

For example consider binary antipodal signaling over an AWGN channel with $\frac{E_s}{N_0} =$ 5dB. Note that the Shannon capacity of this channel is

$$C = \frac{1}{2}\log_2\left(1 + 2\frac{E_s}{N_0}\right) = 1.4359 \text{ bits per channel use}$$

which is clearly not achievable by binary signaling. Nevertheless, the equivalent binary-AWGN channel has a capacity equal to 0.9762 bits per channel use. In contrast, if one first uses hard decisions to decode the sent bit, the bit-error probability would be:

$$P_b = Q\left(\sqrt{\frac{2E_s}{N_0}}\right) = 6 \times 10^{-3}.$$

The capacity of a BSC with cross-over probability of 6×10^{-3} can be calculated to be 0.9471 which is lower than that of the BAWGN.

4.1 Definitions

As in the binary case, we denote the (non-binary) channel by its transition probability $W(y|x) : \mathcal{X} \longrightarrow \mathcal{Y}$ where $y \in \mathcal{Y}$ is its output symbol (possibly continuous) and $x \in \mathcal{X}$ is one of Q discrete possible inputs of the channel (hence $|\mathcal{X}| = Q$). We can always assume that the input symbol is obtained through a one-to-one mapping from the binary strings of length $L = \log_2 Q$.

Definition 4.1 (Bit-addressing map)**.** A one-to-one mapping $\mathcal{M} : \mathbb{F}_2^L \longrightarrow \mathcal{X}$ that maps the binary vectors of length $L = \log_2 Q$ to channel input alphabets is called "bit-addressing" map:

$$\mathcal{M} : \left(x^{(0)}, x^{(1)}, \dots x^{(L-1)}\right) \mapsto x.$$

Remark. In the sequel we use the shorthand notation of $\vec{x} \triangleq \left(x^{(0)}, x^{(1)}, \dots x^{(L-1)}\right)$.

In a practical communication system, usually the "bit-addressing" map is the modulator that maps the individual information bits to signal points.

Lemma 4.1. *If the elements of the random bit-string*

$$\vec{X} = \left(X^{(0)}, X^{(1)}, \dots, X^{(L-1)}\right)$$

are chosen i.i.d. and uniformly distributed from $\mathbb{F}_2$, and $\mathcal{M}(\vec{X})$ is a one-to-one map, $X = \mathcal{M}(\vec{X})$ will also be uniformly distributed in $\mathcal{X}$.

We would like to use polar codes in order to achieve the symmetric capacity of W which is defined as:

$$I(W) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} \frac{1}{Q} W(y|x) \log \frac{W(y|x)}{\sum_{x' \in \mathcal{X}} \frac{1}{Q} W(y|x')}. \tag{4.1}$$

In the special case of a Gaussian channel, as proposed in [10], if we choose the mapping $\mathcal{M}$ such that the distribution of $X = \mathcal{M}(\vec{X})$ approaches a normal distribution, the Shannon capacity of the channel can be achieved which is:

$$C = \frac{1}{2} \log\left(1 + \frac{\mathbb{E}\left[|X|^2\right]}{\sigma^2}\right) \tag{4.2}$$

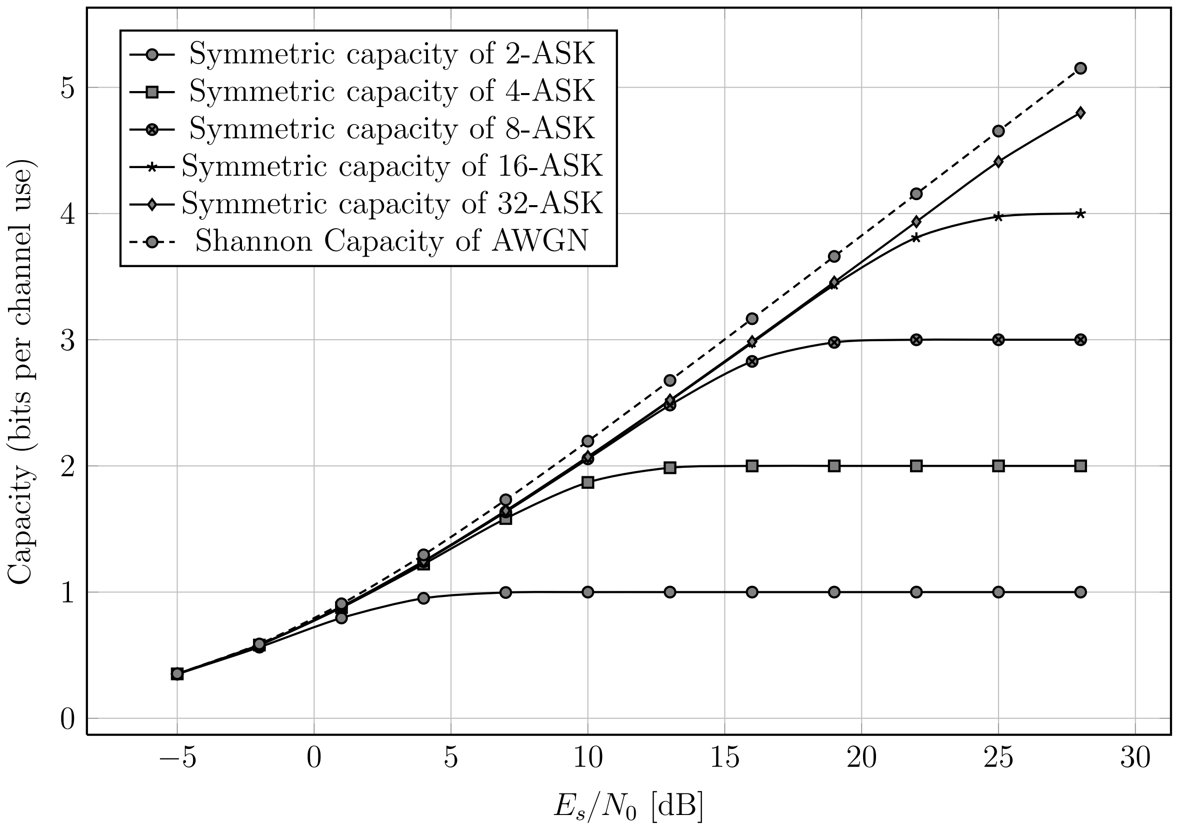


Figure 4.1: At high SNRs, it is essential to increase the input alphabet size to approach the Shannon capacity of the channel.

It is clear that the symmetric capacity $I(W)$ cannot exceed $L = \log_2 Q$ bits per channel use. However, as mentioned before, by increasing the alphabet size, one can approach the Shannon capacity of the channel.

For example in Figure 4.1 we have plotted the Shannon capacity of a Gaussian channel, (4.2), and the symmetric capacity of Q-ASK signaling over that channel versus SNR. Note that even for the binary signaling (B-AWGN channel) there is no closed form to compute the capacity (4.1) and the results should be obtained using numerical integration methods.

4.2 Multilevel Coding

We start by the classical method of non-binary channel coding. In multilevel coding, the idea is to perform L levels of binary channel coding on each bit that constructs the Q-ary input symbol of the channel. We first introduce the general definitions and theorems of multilevel coding briefly and proceed with designing multilevel polar codes.

4.2.1 Preliminaries

As we mentioned before, for multilevel coding, one basically defines L binary-input sub-channels as follows:

Definition 4.2 (Multilevel Subchannels)**.** Having the Q-ary input channel $W : \mathcal{X} \longrightarrow \mathcal{Y}$ and the bit-addressing map $\mathcal{M} : \mathbb{F}_2^L \longrightarrow \mathcal{X}$, $L = \log_2 Q$ binary sub-channels $W^\ell : \mathbb{F}_2 \longrightarrow \mathcal{Y} \times \mathbb{F}_2^\ell$ will be defined as:

$$W^\ell\left(y, \vec{x}_0^{\ell-1} | x^{(\ell)}\right) \triangleq \sum_{\vec{x}_{\ell+1}^{L-1} \in \mathbb{F}_2^{L-(\ell+1)}} \frac{1}{2^{L-(\ell+1)}} W\left(y | \mathcal{M}\left(\vec{x}\right)\right) \tag{4.3}$$

for $\ell = 0, 1, \cdots, L-1$.

Simply, the first channel (W^0) is assumed the binary channel whose input is $X^{(0)}$ and its output is Y (considering the contribution of all other bits to Y and as and additional noise). The second one is the channel whose input is $X^{(1)}$ and its output is the pair of $Y, X^{(0)}$ (again seeing the effects of $X^{(2)}$, $X^{(3)}$, ... as noise terms) and so on.

Remark. It is possible to define the sub channels using a different ordering. For example we can define the first channel as the channel whose input is $X^{(L-1)}$ and its output is Y, considering the contribution of all other terms as noise. Then the second channel as the one whose input is $X^{(L-2)}$ with the output pair $Y, X^{(L-1)}$ alternatively. In fact there exists $L!$ possibilities to define the channels. In practice this choice could be important as we see later on, although this does not affect the theoretical results we provide here. Nevertheless, for the sake of consistency of notation we always use the ordering we just defined. It is always possible to consider any changes in the decoding order by modifying the definition of bit-addressing map $\mathcal{M}$.

Theorem 4.1. *As far as the elements of bit-string $\vec{X}$ are chosen independently uniformly distributed total capacity of the binary sub-channels defined in Definition 4.2 will be equal to the symmetric capacity of the Q-ary channel W, $I(W)$ which is defined in* (4.1)*.*

Proof. $I(W)$ as defined in (4.1) is equal to $I(X;Y)$ whenever X is uniformly distributed, now we have the following equalities:

$$\begin{aligned} I(X;Y) \overset{(a)}{=} I\left(\mathcal{M}(\vec{X});Y\right) &\overset{(b)}{=} I\left(\vec{X};Y\right) \\ &\overset{(c)}{=} \sum_{\ell=0}^{L-1} I\left(X^{(\ell)}; Y | \vec{X}_0^{\ell-1}\right) \\ &\overset{(d)}{=} \sum_{\ell=0}^{L-1} I\left(X^{(\ell)}; Y, \vec{X}_0^{\ell-1}\right) = \sum_{\ell=0}^{L-1} I(W) \end{aligned}$$

where (a) is due to Lemma 4.1, (b) is since the bit-addressing map $\mathcal{M}$ is bijective, (c) is the chain-rule for mutual information and (d) is due to independence of $X^{(\ell)}$ and $\vec{X}_0^{\ell-1}$. □

In the view of Theorem 4.1, one can use a separate binary code of block length N for each of the L mentioned binary sub-channels. Let's denote these codes by $\mathcal{C}^\ell, \quad \ell = 0, 1, \cdots, L-1$. Each code has its own rate defined as

$$R^\ell \triangleq \frac{\log_2 |\mathcal{C}^\ell|}{N}, \quad \ell = 0, 1, \cdots, L-1,$$

and clearly the elements of each codebook are binary vectors of length N which we denote by $\mathbf{X}^{(\ell)}, \quad \ell = 0, 1, \cdots, L-1$

In the encoder $K = NR$ bits of source information[1], are first divided into L different groups, each containing $K^{\ell} = NR^{\ell}$ bits such that:

$$K = \sum_{\ell=0}^{L-1} K^{\ell}. \tag{4.4}$$

Next, each group is encoded using the corresponding code to a codeword $\mathbf{X}^{(\ell)}$.

The ith element of the length N channel input is then generated by mapping the ith bits of the all L codewords with $\mathcal{M}$ to the channel input symbol. Namely, the length N Q-ary vector of channel input will be:

$$\mathbf{X} = \begin{bmatrix} \mathcal{M}\left(X_0^{(0)}, X_0^{(1)}, \ldots, X_0^{(L-1)}\right) \\ \mathcal{M}\left(X_1^{(0)}, X_1^{(1)}, \ldots, X_1^{(L-1)}\right) \\ \vdots \\ \mathcal{M}\left(X_{N-1}^{(0)}, X_{N-1}^{(1)}, \ldots, X_{N-1}^{(L-1)}\right) \end{bmatrix} \tag{4.5}$$

Complexity of encoding scheme depends on the complexity of individual binary encoders, and clearly that of the bit-addressing map $\mathcal{M}$. However, the bit-addressing map is always a very simple operation since (as the name suggests) it only requires selecting one of $Q = 2^L$ elements of channel input alphabet $\mathcal{X}$.

The decoder then, starts by decoding the first codeword. Provided that the first code is designed properly, the result of decoding $\hat{\mathbf{X}}^{(0)}$ is exactly the first sent codeword $\mathbf{X}^{(0)}$. Hence, the decoder can decode the second sent codeword $\mathbf{X}^{(1)}$ for which the knowledge of channel output and $\mathbf{X}^{(0)}$ is essential (recall that the second channel was the one whose output is the pair of $Y, X^{(0)}$). This process is continued until all L codewords are decoded.

Observe that the complexity of decoding essentially depends on the structure of the bit-addressing map $\mathcal{M}$. Generally, at the time of decoding the ℓth level codeword, the equivalent channel's output alphabet is $\mathcal{Y} \times \mathbb{F}_2^{\ell}$. Even though only likelihood ratios are sufficient statistics for decoding, combining all outputs and computing the likelihood ratios can be complex depending on the structure of $\mathcal{M}$. For example, in terms of space complexity, initially the decoder needs only to store N channel outputs, while for decoding the ℓth level codeword, it can generally be necessary to store all $(\ell - 1) \times N$ bits of the previously decoded codewords. Nevertheless, in practice, a cleverly designed map can simplify the whole decoding process.

The decoding method we just described is known in as *multistage decoding.* It might be predictable that due to the nature of the decoding process error propagation can be an important flaw in the scheme.

The usefulness of multistage decoding is its relative low-complexity. It can be easily seen that one only needs to use L binary decoders successively to implement such a decoder. Provided that the codes are designed properly, the multistage decoder can achieve the capacity of the Q-ary channel without need for the complexity of maximum-likelihood

[1]Note that R can now be an arbitrary ratio

decoding as formalized in the following Theorem which is somehow paraphrasing Theorem 2 of [11]:

Theorem 4.2. *The symmetric capacity $I(W)$ of the Q-ary channel W can be achieved by multilevel encoding and* overall maximum-likelihood decoding *if and only if the rates R^ℓ satisfy the following conditions:*

(i) $\sum_{\ell=0}^{L-1} R^\ell \leqslant I\left(Y; X^{(0)}, X^{(1)}, \cdots, X^{(L-1)}\right) = I(W)$

(ii) $\sum_{\ell\in\mathcal{S}} R^\ell \leqslant I\left(Y; \left\{X^{(i)} : i \in \mathcal{S}\right\} \mid \left\{X^{(j)} : j \in \mathcal{S}^C\right\}\right)$*, for all possible subsets* $\mathcal{S} \subset \{0, 1, \cdots, L-1\}$ *where* $\mathcal{S}^C = \{0, 1, \cdots, L-1\} \backslash \mathcal{S}$.

Moreover, if the individual rates satisfy:

$$R^\ell \leqslant I\left(W^\ell\right) \tag{4.6}$$

where W^ℓs are the binary sub channels defined in Definition 4.2, multistage decoding *will be sufficient to achieve the capacity.*

Proof. We don't go into deep technical details of the proof since our aim is to mainly show the relationship between the first part and the second part of the theorem.

The problem of multilevel coding can be seen as a multiple-access channel with inputs $X^{(0)}$, $X^{(1)}$,..., $X^{(L-1)}$ and output Y. Then the conditions in the first part of the claim follow from basic multiple-access channel coding results (see [12] for example).

The second part is simply application of the channel coding theorem to individual binary channels.

It is instructive to consider a simple case of two-level coding where we have to consider two rates, R^0 and R^1. As depicted in Figure 4.2, condition (i) says that sum of two rates cannot exceed the total capacity of the system, hence (R^0, R^1) should lie below the dashed line in the figure. Moreover, condition (ii) further restricts the rate to be inside the shaded pentagonal region where all rates are achievable using (a possibly complex) maximum likelihood decoder.

However, if we don't operate at the rates inside the dark triangle, only multistage decoder will be sufficient to achieve the desired rate.

Particularly, every rate pair on the tangent line connecting points A and B is achievable using the maximum-likelihood decoder, while the multistage decoder is sufficient to approach the vertices (A or B). □

While according to Theorem 4.2 it is possible to approach the capacity of the Q-ary channel by choosing the rates of individual codes equal to the capacity of the individual binary sub-channels asymptotically (namely the equality condition in (4.6)), in an operational system, due to the finite length of codewords, the achievable rate (for a given word error rate) is always lower than total capacity. Hence the question will be that how to choose the individual code rates such that total transmission rate is equal to a given rate $R < I(W)$. In the literature there exists different ad-hoc rate design rules which are summarized in [11].

Before proceeding with the special results on multilevel polar codes, we mention the following result regarding degradation/upgradation of non-binary channel and the corresponding binary sub-channels.

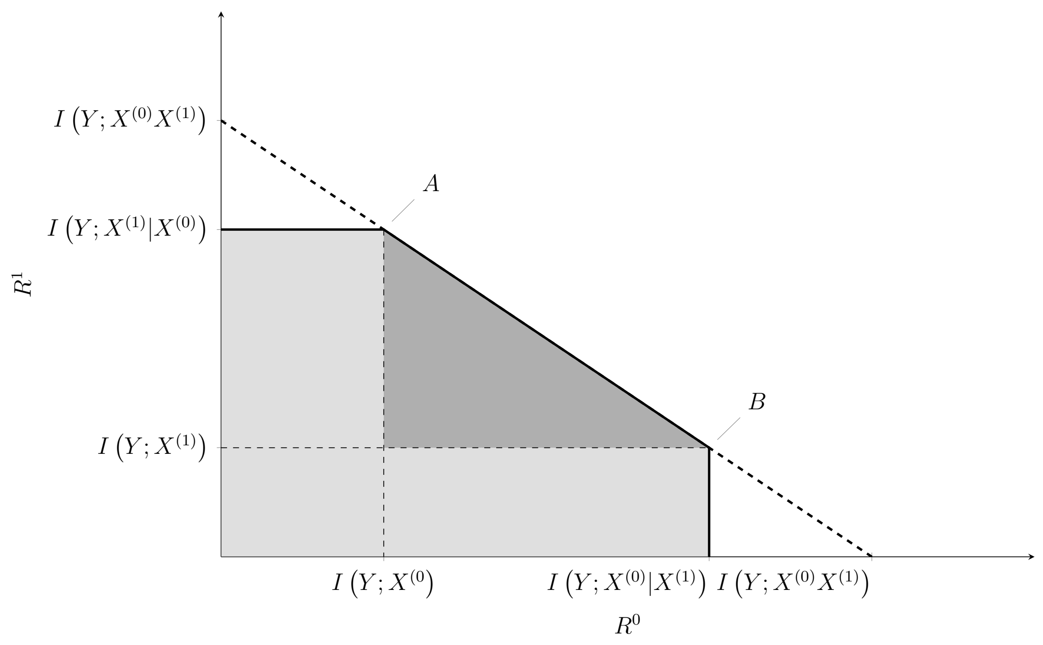


Figure 4.2: Multistage decoding is sufficient for achieving the capacity of the modulation scheme if the individual rates are chosen properly.

Theorem 4.3. *If the channel* $\breve{W} : \mathcal{X} \longrightarrow \breve{\mathcal{Y}}$ *is* degraded *with respect to channel* $W : \mathcal{X} \longrightarrow \mathcal{Y}$ *(according to Definition 1.10) then all binary sub-channels constructed from* $\breve{W}$ *(according to Definition 4.2) ,* $\breve{W}^\ell : \mathbb{F}_2 \longrightarrow \breve{\mathcal{Y}} \times \mathbb{F}_2^\ell, \quad \ell = 0, 1, \cdots, L-1$ *are degraded with respect to the corresponding binary sub-channels constructed from* W*. Namely:*

$$\breve{W}^\ell \preceq W^\ell, \qquad \forall \ell = 0, 1 \cdots, L-1. \tag{4.7}$$

Moreover, the same results are true if we replace degradation *with* upgradation *(according to Definition 1.11).*

Proof. By definition $\breve{W} \preceq W$ means there exists a channel $Q : \mathcal{Y} \longrightarrow \breve{\mathcal{Y}}$ such that:

$$\breve{W}(y|x) = \sum_{y' \in \mathcal{Y}} W(y'|x) Q(y|y').$$

Hence:

$$\begin{aligned}
\breve{W}^\ell\left(y, \vec{x}_0^{\ell-1}|x^{(\ell)}\right) &= \sum_{\vec{x}_{\ell+1}^{L-1} \in \mathbb{F}_2^{L-(\ell+1)}} \frac{1}{2^{L-(\ell+1)}} \breve{W}\left(y|\mathcal{M}\left(\vec{x}\right)\right) \\
&= \sum_{\vec{x}_{\ell+1}^{L-1} \in \mathbb{F}_2^{L-(\ell+1)}} \frac{1}{2^{L-(\ell+1)}} \sum_{y' \in \mathcal{Y}} Q(y|y') W\left(y'|\mathcal{M}\left(\vec{x}\right)\right) \\
&= \sum_{y' \in \mathcal{Y}} Q(y|y') \sum_{\vec{x}_{\ell+1}^{L-1} \in \mathbb{F}_2^{L-(\ell+1)}} \frac{1}{2^{L-(\ell+1)}} W\left(y|\mathcal{M}\left(\vec{x}\right)\right) \\
&= \sum_{y' \in \mathcal{Y}} Q(y|y') W^\ell\left(y, \vec{x}_0^{\ell-1}|x^\ell\right)
\end{aligned}$$

which shows $\widetilde{W}^\ell \preceq W, \quad \forall \ell$. The proof for upgradation follows simply by observing that $\widehat{W} \succeq W$ implies $W \preceq \widehat{W}$. Hence $W^\ell \preceq \widehat{W}^\ell$ which in turn imples $\widehat{W}^\ell \succeq W^\ell$. □

4.2.2 Multilevel Polar Codes

So far we have briefly seen the principal multilevel coding theorems. The main idea, as explained before is to use L binary codes in parallel. Note that the only assumption on the underlying binary codes is that all of them are designed at appropriate rates such that they can be decoded without errors.

While there is no obligation in using the same type of code for all levels, we would like to focus on the case of using L parallel polar codes in a multilevel code for non-binary channel coding. As we will see in the sequel, polar codes have a number of nice properties that qualifies them as a good candidate for multilevel coding.

Recall that in the multilevel setting, each individual code is working on the corresponding binary sub-channel W^ℓ which is a binary discrete memoryless channel by definition. Hence all of the results on the performance of conventional binary polar codes (we saw in Chapter 1 are valid for the performance of individual codes in the multilevel setting. So we just need to focus on the problem of combining individual codes and overall performance of the scheme here.

First, let's review and fix some notation: We define the "word-error" event for individual codes of a multilevel code of block length N as :

$$\mathcal{E}^\ell \triangleq \left\{\exists \quad i \in \{0, 1, \cdots, N-1\} : \hat{X}_i^{(\ell)} \neq X_i^{(\ell)}\right\}. \tag{4.8}$$

where $\hat{\mathbf{X}}^{(\ell)}$ is the output of the ℓth stage of multistage decoding. We also denote the probability of this event (the word-error probability of the ℓth code) as: $P_w^\ell = \mathbb{P}\left[\mathcal{E}^\ell\right]$.

Furthermore, the global "word-error" event is the event that at least one of the codewords are decoded wrongly, namely:

$$\mathcal{E} \triangleq \left\{\exists \quad i \in \{0, 1, \cdots, N-1\}, \ell \in \{0, 1, \cdots, L-1\} : \hat{X}_i^{(\ell)} \neq X_i^{(\ell)}\right\}. \tag{4.9}$$

Likewise we denote the global word-error probability by $P_w = \mathbb{P}\left[\mathcal{E}\right]$. It is obvious that:

$$\mathcal{E} = \bigcup_{\ell=0}^{L-1} \mathcal{E}^\ell \tag{4.10}$$

Theorem 4.4 (Upper-bound on error probability of multilevel polar codes)**.** *In an L-level polar coding scheme with successive cancellation decoder, the global word-error probability is bounded as:*

$$P_w \leqslant \sum_{\ell=0}^{L-1} \sum_{i \in \mathcal{A}^\ell} Z\left(W^{\ell\,(i)}_{\ N}\right). \tag{4.11}$$

Where $\mathcal{A}^\ell$ is the information bits set chosen for the ℓth level binary polar-code.

Proof.

$$\mathcal{E} = \bigcup_{\ell=0}^{L-1} \mathcal{E}^\ell = \bigcup_{\ell=0}^{L-1} \left(\mathcal{E}^\ell \cap \left(\bigcup_{\ell'=0}^{\ell-1} \mathcal{E}^{\ell'}\right)^C\right).$$

Hence the word-error probability can be written as:

$$\begin{aligned} P_w = \mathbb{P}\left[\mathcal{E}\right] &= \mathbb{P}\left[\bigcup_{\ell=0}^{L-1}\left(\mathcal{E}^{\ell} \cap \left(\bigcup_{\ell'=0}^{\ell-1}\mathcal{E}^{\ell'}\right)^{C}\right)\right] \\ &\overset{(a)}{=} \sum_{\ell=0}^{L-1}\mathbb{P}\left[\mathcal{E}^{\ell} \cap \left(\bigcup_{\ell'=0}^{\ell-1}\mathcal{E}^{\ell'}\right)^{C}\right] \end{aligned} \tag{4.12}$$

where (a) is since the events $\mathcal{E}^{\ell} \cap \left(\bigcup_{\ell'=0}^{\ell-1}\mathcal{E}^{\ell'}\right)^{C}$ are disjoint. Now observe that the event $\mathcal{E}^{\ell} \cap \left(\bigcup_{\ell'=0}^{\ell-1}\mathcal{E}^{\ell'}\right)^{C}$ is the event that ℓth level code is decoded wrongly and all previous level codes are decoded correctly, which is a subset of the event that the ℓth level polar code (with the decoder possessing correct value of all previous codewords $\mathbf{X}^{(\ell')}, \quad \ell' < \ell$) is decoded wrongly. The latter is the error event of a plain binary polar code under SC decoding, which is bounded in Theorem 1.3. Hence:

$$\mathbb{P}\left[\mathcal{E}^{\ell} \cap \left(\bigcup_{\ell'=0}^{\ell-1}\mathcal{E}^{\ell'}\right)^{C}\right] \leqslant \sum_{i\in\mathcal{A}^{\ell}} Z\left(W_N^{\ell(i)}\right).$$

Plugging the above bound into (4.12) yields (4.11). □

Corollary 4.1 (Assymptotic Behavior of Multilevel Polar Codes)**.** *For large enough block length, N, word error probability of L-level polar code under SC cancellation decoding satisfies:*

$$P_w = o\left(L2^{-N^{\beta}}\right). \tag{4.13}$$

for fixed $\beta < \frac{1}{2}$ if the individual code rates are assigned appropriately, that is:

$$R^{\ell} \leqslant I(W^{\ell}),$$

Proof. The result is obtained by combining the results of Theorem 4.4 and Theorem 1.4. □

Here comes the first specific property of multilevel polar coding: Due to the explicit code construction properties of polar codes, we can have specific code design rule for multilevel polar codes. Considering the results of Theorem 4.4, we can conclude that the best strategy for assigning the individual rates to L parallel polar codes is to choose their corresponding information bits set such that the double sum in (4.11) is minimized, similar to Arıkan's method for constructing the (single level) polar codes. This idea can be formalized in Algorithm 7.

Note that we didn't provide any explanations about how to obtain the Bhattacharyya parameters and assumed that they are available since W^{ℓ}s are all binary symmetric channels for which we can estimate or approximate the Bhattacharyya parameters of the polarized channels using the methods explained in Section 1.6.

Algorithm 7 Code construction for multilevel polar codes

```
Input: Bhattacharyya parameters $Z\left(W^{\ell (i)}_{N}\right)$ for $i = 0, 1, \cdots, N-1$   $\ell = 0, 1, \cdots, L-1$
    and the target rate $R < L$
Output: Information bits sets of individual codes, $\mathcal{A}^{\ell}$   $\ell = 0, 1, \cdots, L-1$ such that
    $\sum_{\ell=0}^{L-1}|\mathcal{A}^{\ell}| = \lceil NR \rceil$.
    Initialize $\mathcal{A}^{\ell} \leftarrow \varnothing \quad \forall \ell = 0, 1, \cdots, L-1$
    while $\sum_{\ell=0}^{L-1}|\mathcal{A}^{\ell}| < NR$ do
        $(\ell, i) \leftarrow \arg\min_{\ell', i' : i' \notin \mathcal{A}^{\ell'}} Z\left(W^{\ell' (i')}_{N}\right)$;
        $\mathcal{A}^{\ell} \leftarrow \mathcal{A}^{\ell} \cup \{i\}$;
    end while
    return $\mathcal{A}^{\ell}, \quad \ell = 0, 1, \cdots, L-1$
```

Theorem 4.5 (Complexity of Multilevel Polar Coding). *In L-level polar coding (for a channel with $Q = 2^L$-ary alphabet), for the block length N, the complexity of encoding and decoding are:*

$$\chi_{T,E} = O\left(LN \log N\right) \tag{4.14a}$$

$$\chi_{T,D} = O\left(LN \log N\right) \tag{4.14b}$$

respectively.

Furthermore, the space complexity of L-level polar encoding and decoding are

$$\chi_{S,E} = O\left(N \log N\right) \tag{4.15a}$$

$$\chi_{S,D} = O\left(N \log N\right) \tag{4.15b}$$

respectively if the conventional implementation of polar encoder/decoder is used. Using the space-efficient implementations proposed in sections 1.5.1 and 1.5.2 the space complexities can be reduced into

$$\chi_{S,E} = O\left(N\right) \tag{4.16a}$$

$$\chi_{S,D} = O\left(N\right) \tag{4.16b}$$

respectively.

Proof. Simply L individual polar encoding (decoding) procedures are required for multilevel encoding (decoding) and the complexity of polar encoding (decoding) is $\chi_{T,E} = O\left(N \log N\right)$ ($\chi_{T,D} = O\left(N \log N\right)$) as shown in theorems 1.6 and 1.7.

Additionally, while one needs to deal with L different binary channels in the multilevel scheme, indeed is not necessary to implement L individual encoders/decoders. On the encoder side, the encoding process only depends on the position of information bits. Hence, one can use a single polar encoder, L times in a row and just take care of the correct position of information/frozen bits at each step.

Similarly, since the SC decoder works with the likelihood ratios of the channel output (which are sufficient statistics), after computing those quantities at each level (which depends on the channel transition probability), the same SC decoder can be used (with different information bits sets).

In conclusion, the space complexity of the encoder (decoder) will not be increased compared to the binary case.

It is notable that in practice, at the decoder side, after each decoding an encoding should probably be run in order to remove the contribution of the codeword just decoded from the channel output (consider the example of 2^L-ASK we just had). However, this only increases the constant factor of the complexity term. $\square$

According to Theorem 4.5, the code construction time complexity will also be $O\left(LN\log N\right)$ if one chooses to estimate the Bhattacharyya parameters for code construction by Monte Carlo approach (and its space complexity will not increase). The channel estimation for L-level code can also be integrated into a part of SC decoder in an operational system as suggested by Arıkan. Alternatively, the approximation methods introduced in Section 1.6 (particularly the method described in Section 1.6.3) can be directly employed to approximate the Bhattacharyya parameters of the sub-channels and construct the code with $O\left(LN\right)$ time and $O\left(\log N\right)$ space complexity.

Choice of Frozen Bits

In the conventional polar coding, it has been shown that if the channel is symmetric, the value of frozen bits does not affect the performance of code (see Corollary 1.4). While this is true for individual binary sub-channels W^ℓ in many standard modulation schemes, it is noticeable that we have assumed that the value of higher-level codewords are seen as additional noise terms by lower-level decoders. As a consequence, choosing an arbitrary (probably constant) value for frozen bits in this case will bias the noise of other levels and can affect the performance. So, essentially the frozen bits should now be chosen randomly. In practice this is not a big problem. One solution is to define a pseudo-random generator in both transmitter and received and ask them to agree on the initial seed during the handshaking phase of the session. Afterwards, the transmitter chooses the value of frozen bits using the output of this pseudo-random generator whose value is known to the receiver which does not introduce any biasing effects.

Adaptive Modulation Schemes

Adaptive modulation methods are widely used in practical digital communication systems. When the physical channel conditions are good, higher-order modulations (for example 1024-QAM) are used to exploit the entire capacity of channel while in low SNRs (or when very high level of reliability is required, for example one transmitting the frame controls) lower order modulation schemes are employed. As it might be predictable, an important advantage of multilevel coding (and in particular multilevel polar codes) is that the adaptive modulation algorithms can easily be embedded with the channel encoder/decoder. Number of code levels L can easily be increased or decreased without need for changing the encoder/decoder structure since they both employ a normal binary polar encoder/decoder consecutively L times.

It is also worthwhile to note that the code construction algorithm (Algorithm 7), an adaptive modulation scheme is implicit. When the physical channel is bad, its capacity $I(W)$ is essentially low. Since this quantity is equal to the cumulative sum of the capacity of the binary sub-channels $I\left(W^\ell\right)$ (which are positive quantities) this implies that some of these capacities $I\left(W^{\ell}\right)$ should be close to zero. This in turn implies the corresponding Bhattacharyya parameters of the polarized versions of those channels are high, hence the code construction algorithm will essentially assign the rate 0 to the codes for those very noisy sub-channels. This is indeed equivalent to using lower levels of codes for information transmission, i.e., lower order modulations.

4.3 Non-binary Polar Codes

One of the nice properties of the polarizing phenomena introduced by Arıkan is that it happens also for non-binary alphabets. This property is well studied in [13, Chapter 3] and is used to construct polar codes for channels with non-binary input. We will briefly review the results here with focus on comparing this method with multilevel polar codes. We will omit the proof of most of the arguments in this section as they are rather long and can be found in [13].

Throughout this section, without loss of generality, we assume that the channel input alphabet is $\mathcal{X} = \mathbb{F}_Q = \{0, 1, \cdots, Q-1\}$. This normally does not correspond to actual physical channel input alphabet and in practice an intermediate mapping (such as bit-addressing) would be necessary to relabel the input alphabet.

First, we have to extend the definition of Bhattacharyya parameter for binary alphabet which is defined as

$$Z\left(W\right) = \sum_{y \in \mathcal{Y}} \sqrt{W(y|0)W(y|1)}.$$

For a channel with non-binary input, the Bhattacharyya distance between two alphabets x and x' is defined as:

$$Z\left(W_{x,x'}\right) = \sum_{y \in \mathcal{Y}} \sqrt{W(y|x)W(y|x')}. \tag{4.17}$$

Indeed the notation $W_{x,x'}$ denotes the channel W whose input alphabet is restricted to the two-element subset $\{x, x'\} \subset \mathcal{X}$. The average Bhattacharyya parameter of the channel will be defined as:

$$Z\left(W\right) = \sum_{x,x' \in \mathcal{X}, x \neq x'} \frac{1}{Q(Q-1)} Z\left(W_{x,x'}\right). \tag{4.18}$$

Like the binary case, Bhattacharyya parameter bounds the error probability of the optimal (maximum likelihood) decision rule on the output of the channel $W(y|x)$ as follows:

Lemma 4.2 ([13, Proposition 3.2])**.** *The error probability of maximum-likelihood decision on the output of a Q-ary channel W is bounded as:*

$$P_e \leqslant (Q-1)Z\left(W\right) \tag{4.19}$$

(where $Z\left(W\right)$ is defined in (4.18)*)*

Moreover, similar relationships between Bhattacharyya parameter and capacity can be derived:

Lemma 4.3 ([14, Proposition 3]). *We have the following relationships between $I(W)$ and $Z(W)$:*

$$I(W) \geqslant \log \frac{Q}{1+(Q-1)Z(W)} \tag{4.20a}$$

$$I(W) \leqslant \log \frac{Q}{2} + (\log 2)\sqrt{1-Z(W)^2} \tag{4.20b}$$

$$I(W) \leqslant 2(Q-1)(\log e)\sqrt{1-Z(W)^2} \tag{4.20c}$$

4.3.1 Prime-Size Alphabet

The first result regarding the polarization of non-binary channels is restricted to the case that alphabet size Q is a prime number.

Lemma 4.4 ([13, Lemma 3.1]). *If Q, channel alphabet size, is a prime number, the transformation:*

$$W^-\left(y_1, y_2|u_1\right) \triangleq \sum_{u_2} \frac{1}{Q} W\left(y_1|u_1+u_2\right) W\left(y_2|u_2\right)$$

$$W^+\left(y_1, y_2, u_1|u_2\right) \triangleq W\left(y_1|u_1+u_2\right) W\left(y_2|u_2\right),$$

where the addition in u_1+u_2 is a modulo-Q addition, polarizes the Q-ary channel in the sense that:

$$I\left(W^-\right) \leqslant I\left(W\right) \leqslant I\left(W^+\right)$$

and

$$I\left(W^+\right) + I\left(W^-\right) = 2I\left(W\right).$$

Observe that Lemma 4.4 suggests that the polar transform in Q-ary case (when Q is prime) is exactly that of the binary case which was defined recursively as:

$$\begin{aligned} \mathbf{G}_N &= \mathbf{B}_N \mathbf{F}_N, \\ \mathbf{F}_N &= \mathbf{F}_2 \otimes \mathbf{F}_{N/2}, \\ \mathbf{F}_2 &= \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \end{aligned}$$

(where $\mathbf{B}_N$ is the bit-reversal permutation).

Hence, the same binary polar encoder can be used for channels with prime alphabet size just by replacing the modulo-2 additions with modulo-Q additions.

Similar to the binary case, for $N = 2^n$, if we define $\mathbf{x} \triangleq \mathbf{G}_N \mathbf{u}$ and the vector channel $W_N : \mathcal{X}^N \longrightarrow \mathcal{Y}^N$ as

$$W_N\left(\mathbf{y}|\mathbf{u}\right) \triangleq W^N\left(\mathbf{y}|\mathbf{x}\right) = \prod_{i=0}^{N-1} W(y_i|x_i),$$

we can have N, Q-ary input channels $W_N^{(i)} : \mathcal{X} \longrightarrow \mathcal{Y}^N \times \mathcal{X}^i, \quad i = 0, 1, \cdots, N-1$ defined as:

$$W_N^{(i)}\left(\mathbf{y}, \mathbf{u}_0^{i-1} | u_i\right) \triangleq \sum_{\mathbf{u}_{i+1}^{N-1} \in \mathcal{X}^{N-i}} \frac{1}{Q^{N-(i+1)}} W_N(\mathbf{y}|\mathbf{u}).$$

Theorem 4.6 (Polarization of Q-ary Channels, [13, Theorem 3.1]). *For all $\epsilon \geqslant 0$,*

$$\lim_{n\to\infty} \frac{1}{N} \left| \left\{ i : I\left(W_N^{(i)}\right) > 1 - \epsilon \right\} \right| = I(W)$$
$$\lim_{n\to\infty} \frac{1}{N} \left| \left\{ i : I\left(W_N^{(i)}\right) < \epsilon \right\} \right| = 1 - I(W)$$

Basically, Theorem 4.6 suggest that among $N = 2^n$ forged channels, a fraction equal to $I(W)$ of them converge to perfect noiseless channels while the rest (a fraction $1 - I(W)$ of them) converge to complete noisy (useless) channels. Considering Lemma 4.3 one may alternatively propose that polarization happens in Bhattachryya parameters exactly the same as what happens in the binary case (indeed, the relationship between $I(W)$ and $Z(W)$ is used in the first version of proof of Theorem 4.6 in [14]).

Having established the similarities between the binary and non-binary case (for prime alphabet size) it is easy to deduce the structure of polar encoder/decoder for non-binary input channels as well as the performance and complexity of the code. However, we postpone this analysis in order to maintain the generality of the results and study the case of arbitrary input channels first.

4.3.2 Arbitrary Finite Alphabet Size

Unfortunately, it turns out that polarizing transform of Arıkan does not necessarily polarize the channels with composite alphabet size (Q) due to existence of proper non-trivial subgroups in $(\mathcal{X}, +)$. (see [13, Proposition 3.1]).

Şaşoğlu has proposed two different solutions for polarizing channels with arbitrary alphabet size in [14] and [13] respectively.

First solution is based on keeping the problematic addition operation in "minus" operation and replacing the identity operation ($X_2 = U_2$) in creation of the "plus" channel by a random permutation over $\mathcal{X}$ ($X_2 = \Pi(U_2)$). Then, it has been shown that this transform, averaged over the random permutation, polarize the channel and it has been concluded that there exists at least one permutation (out of all $Q!$ possible permutations) that gives us the polarizing transform.

The main drawback of this method is that it complicates the process of code construction as basically for each "plus" operation, in the recursion tree of code construction one potentially has to examine all $Q!$ possible permutations to find the polarizing permutation. Fortunately, in encoding/decoding these known permutations will be used and no randomization is involved.

Second scheme, proposed by Şaşoğlu based on the fact that the problem arises from the addition in creating the "minus" version of channels. Hence the solution will essentially be to replace it with another operation such that the channels polarize. We will review this method in details here and use it throughout the rest of our discussion.

Definition 4.3 (Polarizing map, [13, Defintion 3.1])**.** The mapping $f : \mathcal{X}^2 \longrightarrow \mathcal{X}$ is called *polarizing* if:

1. for all $x_2 \in \mathcal{X}$, the mapping $x_1 \mapsto f(x_1, x_2)$ is invertible,
2. for all $x_1 \in \mathcal{X}$, the mapping $x_2 \mapsto f(x_1, x_2)$ is invertible, and
3. for all $2 \leqslant m \leqslant Q-1$ and distinct $a_0, a_1, \cdots, a_{m-1} \in \mathcal{X}$, the matrix
$$\mathbf{B} = [B_{ij}], \qquad B_{i,j} = f(a_i, a_j), \qquad i, j = 0, \cdots, m-1$$
has at least $m+1$ distinct elements.

Remark. Şaşoğlu has shown that the mapping $f(x_1, x_2) = x_1 + \pi(x_2)$ (with modulo-Q addition) and the permutation $\pi : \mathcal{X} \longrightarrow \mathcal{X}$ defined as:

$$\pi(x) = \begin{cases} \lfloor \frac{Q}{2} \rfloor & \text{if } x = 0 \\ x - 1 & \text{if } 1 \leqslant x \leqslant \lfloor \frac{Q}{2} \rfloor \\ x & \text{otherwise} \end{cases} \tag{4.21}$$

is polarizing.

Corollary 4.2. *For composite Q, modulo-Q addition is* not *a polarizing map.*

Proof. In the group $(\mathbb{F}_Q, +)$, at least there exist one element $a \in \mathbb{F}_Q$ whose order is less than Q (and hence the group has non-trivial proper subgroups). Choose m equal to the order of this element and $a_i = ia \quad i = 0, 1, \cdots, m-1$. Then $B_{ij} = (i+j)a$ is an element of the subgroup generated by a. Since the order of this subgroup is m, B_{ij}s take only m distinct values and condition 3 is contradicted. □

Lemma 4.5 ([13, Theorem 3.1])**.** *For any arbitrary discrete memoryless channel with finite alphabet size and a polarizing map f (according to Definition 4.3), the transformation*

$$W^-(y_1, y_2 | u_1) \triangleq \sum_{u_2} \frac{1}{Q} W(y_1 | f(u_1, u_2)) W(y_2 | u_2) \tag{4.22a}$$

$$W^+(y_1, y_2, u_1 | u_2) \triangleq W(y_1 | f(u_1, u_2)) W(y_2 | u_2), \tag{4.22b}$$

polarizes the Q-ary channel in the sense that:

$$I\left(W^-\right) \leqslant I(W) \leqslant I\left(W^+\right)$$

and

$$I\left(W^+\right) + I\left(W^-\right) = 2I(W).$$

Remark. The information preserving property is due to the fact that the mapping $(U_1, U_2) \mapsto (f(U_1, U_2), U_2)$ is a one-to-one mapping which is in turn guaranteed by condition 2 in Definition 4.3.

In the view of Lemma 4.5 we can deduce that polarization takes place for sources with arbitrary finite alphabet using an appropriate polarizing map and the corresponding polarizing transform. Due to the non-linearity of the map, it is no longer possible to describe the code as a linear transform (matrix multiplication) but this is of no importance. The recursions for encoding are still valid:

$$\begin{aligned} \mathbf{G}_N(\mathbf{u}) &= \mathbf{B}_N\mathbf{F}_N(\mathbf{u}), \\ \mathbf{F}_N(\mathbf{u}) &= \begin{bmatrix} f\left(\mathbf{F}_{N/2}\left(\mathbf{u}_0^{N/2-1}\right), \mathbf{F}_{N/2}\left(\mathbf{u}_{N/2}^{N}\right)\right) \\ \mathbf{F}_{N/2}\left(\mathbf{u}_{N/2}^{N}\right) \end{bmatrix}, \\ \mathbf{F}_1(u) &= u. \end{aligned}$$

Note that in the above notation $f(\cdot,\cdot)$ operates element-wise on its arguments.

Exactly in a similar fashion, for $N = 2^n$, we can define $\mathbf{x} = \mathbf{G}_N(\mathbf{u})$ the vector channel $W_N : \mathcal{X}^N \longrightarrow \mathcal{Y}^N$ and the polarized channels $W_N^{(i)} : \mathcal{X} \longrightarrow \mathcal{Y}^N \times \mathcal{X}^i, \quad i = 0, 1, \cdots, N-1$ and conclude that Theorem 4.6 is indeed valid for sources with any finite alphabet size as far as the polarizing map is selected correctly.

4.3.3 Implementation Concerns

We saw that polar codes for Q-ary input channels can be obtained by slight modifications to binary polar codes. Hence the implementation of encoder/decoder are indeed quite the same. However, we briefly explain some issues that should be considered in implementation of non-binary polar encoder/decoder.

Polarizing Map

The first obvious difference between binary polar codes and their non-binary counterpart is that the (modulo-2) addition in derivation of the "minus" channel should be replaced with a more complicated polarizing map.

Since the input alphabet is discrete, in a real implementation, a $Q \times Q$ lookup table of all possible values of f can be precomputed and stored in both encoder and decoder. Then computation of the polarizing map will only be reduced to reading the corresponding value from the table. An operation whose cost (time) is more or less the same as binary addition.

Hence, we conclude that replacing the addition by the non-linear polarizing map, will not introduce any cost to the system (in terms of time complexity) and just adds the space required for a small lookup table to the system.

Decoding the Q-ary channel's output

While the only difference between binary and non-binary encoder is presence of the non-linear polarizing map, the polar decoder will be more different in a non-binary setting.

In general, in a binary channel, the task of the decoder is to decide between two possible value of the input, which can be essentially accomplished by only considering a single likelihood ratio per channel output:

$$\Lambda(y) = \frac{W(y|0)}{W(y|1)}$$

In contrast, in a non-binary setting, the decoder should choose among one of Q possible channel inputs for which the knowledge of all Q a prior distributions:

$$W(y|0), W(y|1), \cdots, W(y|Q-1)$$

which increases the space complexity of the decoder by a factor of Q.[2]

In polar decoder, the binary SC decoder starts with $N = 2^n$ likelihood ratios of the channel outputs and combine them in n levels using the recursions (1.55a) and (1.55b) to obtain the decision-level likelihood ratios. As we saw, it is also possible to work with log-likelihood ratios alternatively to have simpler operations in the update relationships.

However, the non-binary case, the decoder starts with set of Q a-priori distributions per channel output and combine them using the direct relationships (4.22a) and (4.22b) at each step to finally obtain a-priori distributions at the decision level. Combining the probabilities in the "plus" direction, (4.22b), requires Q multiplications (one per each element of the set $\{W^+(y_1, y_2, u_1|u_2), \quad \forall u_2\}$) while in the "minus" direction the combination requires Q^2 multiplications (as Q multiplications per element of the set $\{W^-(y_1, y_2|u_1), \quad \forall u_1\}$ is necessary).

Apart from simplifying the combination rules (replacing multiplications by additions) by working in logarithm domain and simple calculations of log-likelihood ratios from observed channel output (compared to calculation of probabilities by direct evaluation of channel transition probability) as we saw in Section 1.5.2, log-domain calculations fit the machine precision very well since they will be real numbers that occupy the whole range of double precision numbers. Moreover, in case of hardware implementation and fixed-point calculations, since the values are well-spread, quantization errors can barely affect the decoding performance.

In comparison, in non-binary case, the inputs and outputs of computations are all values between 0 and 1 which occupy only a small portion of all double-precision numbers. For fixed-point implementations the quantization errors will be more likely to affect the decisions.

Indeed, due to polarization, we could guess that the a-priori distributions will converge to peaky distributions around the sent value (for good channels) and very flat distributions (for bad channels). In both of the cases, most of the values of $W_N^{(i)}(\cdot|u_i)$ will decrease toward zero. Our numerical experiments show that this decrement is so fast that even for small block lengths ($N = 256$) the computations will be unstable and at the decision level all a-priori probabilities will be zero (which are indeed some very tiny numbers below machine's precision).

One quick solution to this problem is to convert the a-priori probabilities to a-posteriori probabilities at each node of the decoding using the very simple normalization

[2]The keen reader will observe that indeed storing $Q-1$ values (for example the ratios $\frac{W(y|1)}{W(y|0)}, \cdots, \frac{W(y|Q-1)}{W(y|0)}$ or $Q-1$ of a-posteriori distributions) is sufficient for decision. However, in large alphabet sizes, this will not make a big difference.

(assuming the input alphabet is uniformly distributed in $\mathcal{X}$):

$$P(u|y) = \frac{W(y|u)}{\sum_{\forall u'} W(y|u')}, \qquad \forall u. \tag{4.23}$$

This way, at least we ensure that at each node sum of the values we have is one and they cannot be extremely low to fall below the machine's precision.

It can be shown with some simple calculations that in order to work with a posteriori probabilities, the combination relationships of (4.22a) and (4.22b) remain the same by just exchanging the a-priori probabilities with a-posteriori ones and multiplying a normalization factor in the following way:

$$P^{-}\left(u_1|y_1, y_2\right) = C^{-} \sum_{u_2} \frac{1}{Q} P\left(f(u_1, u_2)|y_1\right) P\left(u_2|y_2\right) \tag{4.24a}$$

$$P^{+}\left(u_2|y_1, y_2, u_1\right) = C^{+} P\left(f(u_1, u_2)|y_1\right) P\left(u_2|y_2\right) \tag{4.24b}$$

where C^- and C^+ are constant terms independent of u_1 and u_2 respectively.

Hence, in practice, the SC decoder can be initialized with a-posteriori probabilities of channel. Afterwards, at each combination step, first a scaled version of P^- and P^+ values will be computed using the old combination procedures (assuming the constant factors to be unity) and the results will be normalized such that they sum up to one. This way, the computations can be stabilized, however they are still not as good as that of binary codes in terms of accuracy and efficiency.

4.3.4 Performance and Complexity Analysis

Due to the analogy that we have shown between binary and non-binary polar codes, analysis of complexity and performance of non-binary codes is quite the same.

Lemma 4.6 ([13, Lemma 3.5])**.** *Let $f : \mathcal{X}^2 \longrightarrow \mathcal{X}$ be such that both functions $f(x_1, \cdot) : \mathcal{X} \longrightarrow \mathcal{X}$ and $f(\cdot, x_2) : \mathcal{X} \longrightarrow \mathcal{X}$ are invertible for all x_1 and x_2. Defining the "plus" and "minus" channels exactly as in Lemma 4.5, we have:*

$$Z\left(W^{-}\right) \leqslant (Q^2 - Q + 1)Z(W),$$
$$Z\left(W^{+}\right) \leqslant (Q - 1)Z(W)^2$$

where $Z(W)$ is defined in (4.18)*.*

Note that in Lemma 4.6 the mapping f should not necessarily be polarizing, only the first two conditions in Definition 4.3 are sufficient for the results to hold.

Having the bounds of Lemma 4.6, the same method in [2] can be applied to obtain the rate of polarization:

Theorem 4.7 ([13, Theorem 3.5])**.** *For all $0 < \beta < \frac{1}{2}$ and $N = 2^n$,*

$$\lim_{n \to \infty} \frac{1}{N} \left| \left\{ i : Z(W_N^{(i)}) \leqslant 2^{-N^{\beta}} \right\} \right| = I(W)$$

where $W_N^{(i)}$s are the Q-ary forged channels we defined previously.

Corollary 4.3. *For any $0 < \beta < 1/2$ the word-error probability of Q-ary polar code is bounded as:*

$$P_w = o\left(Q2^{-N^{\beta}}\right)$$

Proof. The proof follows from plugging the results of Theorem 4.7 and Lemma 4.2 into the proof of Theorem 1.4 □

Theorem 4.8 (Complexity of Q-ary Polar Coding)**.** *For a Q-ary polar code of block length N, the time complexity of encoding and decoding are:*

$$\chi_{T,E} = O\left(N \log N\right) \tag{4.25a}$$

$$\chi_{T,D} = O\left(Q^2 N \log N\right) \tag{4.25b}$$

respectively.

Moreover, the space complexity of encoding/decoding will be

$$\chi_{S,E} = O\left(N \log N\right) \tag{4.26a}$$

$$\chi_{S,D} = O\left(QN \log N\right) \tag{4.26b}$$

respectively using the conventional implementation of polar codes or

$$\chi_{S,E} = O\left(N\right) \tag{4.27a}$$

$$\chi_{S,D} = O\left(QN\right) \tag{4.27b}$$

by generalizing the space efficient implementations introduced in sections 1.5.1 and 1.5.2.

Proof. The claims generally follow from the discussion in Section 4.3.3.

The encoder, will be obtained by replacing the modulo-2 addition in binary encoder by the polarizing map. Assuming the computation of polarizing map is done in unit time the claim on its time-complexity will be quickly deduced. Considering its space complexity, we again see that polarizing map can only potentially add the space requirement of a $Q \times Q$ look-up table and this yields the claim about possible space complexities.

For the decoder, we saw that at *each node* Q values corresponding to Q different possible values of the input alphabet should be stored which multiplies the space complexity of the decoder by a factor of Q. Moreover, the "plus" and "minus" combination operations require Q^2 or Q operations respectively, hence the time complexity of the binary polar decoder will be multiplied by Q^2. □

Remark. Attention should be paid when comparing the time-complexity of binary and non-binary polar codes. In fact, the Landau notations describe how the algorithm scales with block length and alphabet size with respect to the "principal" operations that take unit time. However, these principal operations are quite different in binary and non-binary setting.

In binary encoder, the principal operation is modulo-2 addition, while in the non-binary case it is a memory access. Normally these two operations are the same in terms of complexity, hence we could expect that increasing the alphabet size from 2 to $Q > 2$ will not change the running time of a polar encoder considerably.

However, in binary decoder, as we saw in Section 1.5.2 that the decoder can be implemented using simple operations of addition and comparison. In contrast, the non-binary decoder requires double-precision multiplication which is much more complex than addition. Hence, despite the scaling factor of Q^2 in the Landau notations (which for example proposes that the decoding time should be multiplied by 4 when going from binary to 4-ary channel) we will see a much more significant difference in running time of the decoder by switching from binary to non-binary.

Choice of frozen symbols

It is worthwhile to mention that in case of non-binary channel, since we don't have a similar notion of channel symmetry we cannot arbitrarily choose the value of frozen channels. They should be chosen randomly in order for the operational performance of the code to be consistent with the theoretical results.

As we have noted before, this can be done by initializing a pseudo-random number generator at both transmitter and receiver side with a common seed. Both sides can agree on the seed during the initiation phase of communication.

4.3.5 Note on Non-binary Code Construction

It is clear that like the binary case, one should know the position of "good" and "bad" polarized Q-ary channels in order to design a code of a given rate R. For binary case we saw that the only case of *exact* and *efficient* channel probing (and hence code construction) is for the erasure channel (BEC). Apart from that, either the lengthy Monte-Carlo estimations or the approximations (all introduced in Section 1.6) should be employed.

The same situation is true for a non-binary setting with the difference that even there is no equivalent for BEC in non-binary input channels for which we know the exact evolution of Bhattacharyya parameters.

In [13] it is postulated that the approximation method we explained in Section 1.6.3 can be extended in a straightforward manner so that the non-binary codes can be constructed with $O(Q^2N)$ time and $O(Q\log N)$ space complexity. This seems to be an important necessity as we see that extension of Monte-Carlo estimation method of Arıkan will be highly inefficient for non-binary channels.

First observe that according to (4.17):

$$Z\left(W_{x,x'}\right) = \sum_{y\in\mathcal{Y}} \sqrt{\frac{W(y|x')}{W(y|x)}} W(y|x) = \mathbb{E}_{y|x}\left[\sqrt{\frac{W(y|x')}{W(y|x)}}\right]. \tag{4.28}$$

Hence, the pairwise Bhattacharyya distances ($Z(W_{x,x'})$) for each channel can be estimated by transmitting alphabet x (hence we used the subscript $y|x$ for the expectation operator above) and computing the empirical average of the value $\sqrt{\frac{W(y|x')}{W(y|x)}}$ in a genie-aided decoder (i.e., setting all bits as frozen bits) which converges to its statistical mean[3]. This method can be used for every pair of channel input alphabet and then the estimations can be plugged into (4.18) to obtain the Bhattacharrya parameter of each channel. This approach requires $Q(Q-1)$ Monte Carlo estimations which can be a huge number even for moderate alphabet size considering the increased complexity of decoder (with a factor of Q^2 compared to the binary case). For example for 4-ary channel coding, we would need 12 Monte-Carlo estimations, each of which takes (more than) 4 times the running time of a binary decoder.

However, it is clear that at each decoder run after transmission of x, much more transition probabilities other than $W(y|x')$ are also computed which are never used. Hence we can rewrite (4.18) to obtain a slightly more efficient channel estimation method as follows:

$$\begin{aligned}
Z(W) &= \frac{1}{Q(Q-1)} \sum_{x,x' \in \mathcal{X}, x \neq x'} Z(W_{x,x'}) \\
&= \frac{1}{Q(Q-1)} \sum_{x=0}^{Q-1} \sum_{x'=0}^{Q-1} \mathbb{1}_{x \neq x'} Z(W_{x,x'}) \\
&= \frac{1}{Q(Q-1)} 2 \sum_{x=0}^{Q-2} \sum_{x'=x+1}^{Q-1} Z(W_{x,x'}) \\
&= \frac{2}{Q(Q-1)} \sum_{x=0}^{Q-2} \sum_{x'=x+1}^{Q-1} \mathbb{E}_{y|x} \left[\sqrt{\frac{W(y|x')}{W(y|x)}} \right] \\
&= \frac{2}{Q(Q-1)} \sum_{x=0}^{Q-2} \mathbb{E}_{y|x} \left[\sum_{x'=x+1}^{Q-1} \sqrt{\frac{W(y|x')}{W(y|x)}} \right]
\end{aligned} \tag{4.29}$$

Using (4.29), we can do the Bhattacharyya parameters estimation by transmitting each alphabet x and computing the empirical mean of $\sum_{x'=x+1}^{Q-1} \sqrt{\frac{W(y|x')}{W(y|x)}}$. This way we will have the elements of the summand in the outer sum in (4.29) and we can sum them up to have the estimation of Bhattacharyya parameter. This way, at least we have reduced the number of Monte-Carlo estimations required from $Q(Q-1)$ to $Q-1$.

Nevertheless, due to the lengthiness of Monte-Carlo estimations, for large input alphabet sizes, extension of the methods in [6] is of crucial importance.

4.4 Comparison of Multilevel and Non-Binary Polar Codes

As we have seen in the previous sections, there exist two different methods for reliable, capacity achieving information transmission over a non-binary channel: Multilevel Coding which uses a sequence of binary codes in parallel and non-binary codes which directly

[3]We use the a-priori probabilities in our discussions. However it is obvious that since the a-priori and a-posteriori distributions differ in a constant normalization factor (independent of the input) as in (4.23), they can be interchanged without affecting the results.

encode and decode the non-binary alphabet. Moreover, we showed that polar codes are good candidates for both schemes and in both cases the capacity of non-binary channel can be achieved using large enough block length. Here, we would summarize the characteristics of each of the methods and compare them.

4.4.1 Complexity

- Comparing the results of Theorem 4.5 and Theorem 4.8 we see that the advantage of non-binary polar coding is that the time-complexity of encoding does not increase with the alphabet size, while in the multilevel case, it increases logarithmically.
- Decoding time-complexity will increase quadratically with the alphabet size in non-binary polar coding whereas in the case of multilevel coding it again increases logarithmically.
- The binary decoder can be implemented using very basic arithmetic operations (addition and comparison for "plus" and "minus" combination of log-likelihood ratios) while the non-binary decoder will demand more processing power to compute the combination rules.
- The space complexity of non-binary decoder will increase linearly with the alphabet size while for the multilevel scheme this will be independent of alphabet size.
- The binary decoder is less sensitive to quantization noise due to finite precision of computations as we mentioned in Section 4.3.3.
- The efficient methods for binary code construction can readily be applied to multilevel polar codes while for the non-binary scheme we still need to use the lengthy Monte-Carlo channel estimations.

It is, however, notable that from a system level point of view, complexity of the communication system that uses multilevel polar codes will partially depend on complexity of the bit addressing map which in turn determines the complexity of combining the results of lower level codes and computing the likelihood ratios for the current decoding level. In contrast, in a non-binary setting, this intermediate step is not required. Evaluation of the channel transition distributions at output points is sufficient to initialize the decoder.

4.4.2 Performance

Unfortunately, even for the binary case, no closed form results are available for finite length performance of polar codes.

The results of Corollary 4.1 and Corollary 4.3 both provide upper bounds on asymptotic behavior of the word-error probability of the codes. Hence we cannot use them to compare the performance of two schemes.

So we can only rely on numerical simulations as we will see later. However, we expect to obtain better performance (in terms of word-error rate) in non-binary polar codes compared to the multilevel setting in return of the higher complexity cost that we pay.

4.4.3 Scalability

As we noted before, in an operational system, the modulation order is normally varied according to the physical channel state. This means that a single system normally should be capable of dealing with channels with different alphabet sizes.

We saw that in multilevel coding, as far as Bhattacharyya parameters for different channels at different levels are available, increasing or decreasing the number of code levels (and hence the input alphabet size of channel) is easily accomplished since the encoding (decoding) is principally done by running a binary encoder (decoder) L times in a row.

In contrast, in the non-binary setting, both encoder and decoder are highly dependent on the channel input size. In the encoder side, changing the channel alphabet size will require the change in the polarizing map which is implemented as a look-up table. This can be difficult, but the difficulty can be mitigated to some extent by finding polarizing maps that are less dependent on input alphabet size. However, the decoder has much stronger dependency on the alphabet size as in every node the number of values that are stored is equal to the size of channel input alphabet. Hence in a VLSI implementation the decoding circuit should be entirely replaced in case of a change in the input alphabet size.

So we conclude that non-binary codes cannot be easily embedded in the systems with adaptive modulation capabilities since for every modulation order, a different decoder circuit (and potentially a different encoding circuit) will be required.

4.4.4 Summary

Table 4.1 summarizes the results of the comparison we just explained.

	Multilevel Coding	Q-ary coding
Encoding Complexity	$\chi_{T,E} = O(\log Q \cdot N \log N)$ $\chi_{S,E} = O(N)$	$\chi_{T,E} = O(N \log N)$ $\chi_{S,E} = O(N)$
Decoding Complexity	$\chi_{T,D} = O(\log Q \cdot N \log N)$ $\chi_{S,D} = O(N)$	$\chi_{T,D} = O(Q^2 N \log N)$ $\chi_{S,E} = O(QN)$
Asymptomatic Performance	$P_w = o\left(\log Q \cdot 2^{-N^\beta}\right), \quad \beta < 1/2$	$P_w = o\left(Q 2^{-N^\beta}\right), \quad \beta < 1/2$

Table 4.1: Comparison of Multilevel and Non-binary polar codes

4.5 Results for ASK Signaling over AWGN

In this part, we have simulated the non-binary polar coding methods we described for ASK signaling over AWGN, which is a very typical fashion in real communication systems, and we present the results.

First we derive the mathematical relationships we need for implementing the encoder/decoder and then proceed with the results of simulations.

4.5.1 Derivation of Encoder/Decoder Relationships

As it is known from basics of digital communications, in Q-ASK signaling, one of the Q signal points from the set

$$\mathcal{X} = \{-(Q-1), -(Q-3), \ldots, (Q-3), (Q-1)\} \tag{4.30}$$

is chosen and sent through the physical channel that adds white Gaussian noise $\eta \sim \mathcal{N}(0, \sigma^2)$. Hence the received channel output will be:

$$Y = X + \eta. \tag{4.31}$$

Note that normally Q is a power of 2, namely $Q = 2^L$.

It is clear that the channel has continues output ($\mathcal{Y} = \mathbb{R}$) and its transition probability is simply:

$$W(y|x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y-x)^2}{2\sigma^2}\right). \tag{4.32}$$

When we use the channel N times, the output of channel will be a vector $\mathbf{Y}$ like:

$$\mathbf{Y} = \begin{bmatrix} Y_0 \\ Y_1 \\ \vdots \\ Y_{N-1} \end{bmatrix} = \begin{bmatrix} X_0 \\ X_1 \\ \vdots \\ X_{N-1} \end{bmatrix} + \begin{bmatrix} \eta_0 \\ \eta_1 \\ \vdots \\ \eta_{N-1} \end{bmatrix} \tag{4.33}$$

where η_is are i.i.d.

Since η is the filtered and sampled version of noise with power spectrum density of $\frac{N_0}{2} = \sigma^2$, the transmission SNR will be accordingly defined as:

$$\frac{E_s}{N_0} = \frac{\mathbb{E}\left[|X^2|\right]}{2\sigma^2} = \frac{Q^2-1}{6\sigma^2} \tag{4.34}$$

where the last equality is obtained assuming the messages (and accordingly the elements of the sent signal) are uniformly chosen among Q possible messages.

Note that the Shannon capacity of the Gaussian channel (with the same SNR) is:

$$C = \frac{1}{2}\log_2\left(1 + \frac{\mathbb{E}\left[|X^2|\right]}{\mathbb{E}\left[|\eta^2|\right]}\right) = \frac{1}{2}\log_2\left(1 + \frac{E_s}{2N_0}\right). \tag{4.35}$$

The "bit-addressing" map (see Definition 4.1) is conventionally defined as:

$$\mathcal{M}\left(\vec{x}\right) = \sum_{\ell=0}^{L-1} (-1)^{x^{(\ell)}} 2^\ell \tag{4.36}$$

For Q-ary coding over this channel, the only requirement is to map the elements of $\mathbb{F}_Q$ (output alphabet of encoder) to $\mathcal{X}$ defined in (4.30) which can be done in numerous ways. For example we can use $\phi : \mathbb{F}_Q \longrightarrow \mathcal{X}$

$$\phi(x) = 2x - (Q+1).$$

Then, what remains is to evaluate the channel transition distributions $W(y|\phi(x)) \quad \forall x \in \mathbb{F}_Q$ which is defined in (4.32) for each element of the channel output vector $\mathbf{Y}$ and feed them into the Q-ary polar decoder.

For multilevel coding, however, we need a bit more complicated computations before we can start the decoders. The encoding process is rather straightforward and just involves running L binary decoders and combining their outputs using the bit addressing map and obtaining the vector of channel input (as in (4.5)).

For the sake of brevity, let's define a bipolar version of a bit as

$$\psi \triangleq (-1)^x \quad x \in \mathbb{F}_2.$$

We will use $\psi \in \{\pm 1\}$ instead of $x \in \mathbb{F}_2$ throughout the rest of discussion, as they are completely equivalent. Next, we define a generalized version of ASK bit-addressing map (4.36) as:

$$\mathcal{M}(\vec{x}) = \sum_{\ell=0}^{L-1} \psi^{(\ell)} A_\ell. \tag{4.37}$$

Note that in general this mapping might not be one-to-one depending on the choice of amplitudes A_ℓ. Nevertheless, we assume they are chosen such that the mapping is one-to-one.

The Gaussian channel output can be rewritten as:

$$\mathbf{Y} = \begin{bmatrix} \sum_{\ell=0}^{L-1} \Psi_0^{(\ell)} A_\ell \\ \sum_{\ell=0}^{L-1} \Psi_1^{(\ell)} A_\ell \\ \vdots \\ \sum_{\ell=0}^{L-1} \Psi_{N-1}^{(\ell)} A_\ell \end{bmatrix} + \begin{bmatrix} \eta_0 \\ \eta_1 \\ \vdots \\ \eta_{N-1} \end{bmatrix} \tag{4.38}$$

which shows that contribution of the elements of each levels' codeword to the channel output that the receiver observes (as well as the physical channel noise) is additive.

Lemma 4.7. *Assume Y is defined as:*

$$Y = \Psi_1 + \Psi_2 + \eta$$

where η, Ψ_1 and Ψ_2 are independent. Then $\tilde{Y} = Y - \Psi_1$ is sufficient statistic for detecting Ψ_2 using the observations (Y, Ψ_1).

Proof.

$$\begin{aligned}
\mathbb{P}[Y = y, \Psi_1 = \psi_1 | Y - \Psi_1 = \tilde{y}, \Psi_2 = \psi_2] &= \mathbb{P}[\Psi_1 + \Psi_2 + \eta = y, \Psi_1 = \psi_1 | \Psi_2 + \eta = \tilde{y}, \Psi_2 = \psi_2] \\
&= \mathbb{P}[\Psi_1 = y - \tilde{y}, \Psi_1 = \psi_1 | \Psi_2 + \eta = \tilde{y}, \Psi_2 = \psi_2] \\
&= \mathbb{P}[\Psi_1 = y - \tilde{y}, \Psi_1 = \psi_1] \\
&= \mathbb{1}_{y - \tilde{y} = \psi_1}.
\end{aligned}$$

The independence of the result from value of Ψ_2 yields the sufficiency of $\tilde{Y} = Y - \Psi_1$. □

Corollary 4.4. *In ASK signaling, where the bit-addressing map is defined as in* (4.37)*, for ℓth binary sub-channel $W^\ell : \mathcal{X} \longrightarrow \mathcal{Y} \times \mathcal{X}^\ell, \quad \ell = 0, 1, \cdots, L-1$ defined in Definition 4.2,*

$$y^{(\ell)} = y - \sum_{\ell'=0}^{\ell-1} \psi^{(\ell)} A_\ell$$

is sufficient statistics.

Proof. Observe that by definition the ℓth sub-channel's input is ψ_ℓ and its outputs are $y, \vec{\psi}_0^{\ell-1}$. Also note that by definition additive noise η and bits of codewords $\psi^{(\ell)}$s are independent. Hence we can apply Lemma 4.7 repeatedly to obtain the claim. □

As a consequence, we can define "equivalent" binary sub-channels as the channel seen from $\Psi^{(\ell)}$ to $Y^{(\ell)}$ and replace $W^{(\ell)}$s in general Definition 4.2 by these equivalent sub-channels $\tilde{W}^\ell$. The input-output relationship for $\tilde{W}^\ell$ will be:

$$Y^{(\ell)} = \Psi^{(\ell)} A_\ell + \sum_{\ell'=\ell+1}^{L-1} \Psi^{(\ell')A_{\ell'}} + \eta. \tag{4.39}$$

Hence, the channel transition probability can be derived as:

$$\tilde{W}\left(y^{(\ell)}|\psi^{(\ell)}\right) = \sum_{\vec{\psi}_{\ell+1}^{L-1} \in \{\pm 1\}^{L-(\ell+1)}} \frac{1}{2^{L-(\ell+1)}} f_\eta \left(y^{(\ell)} - \psi^{(\ell)} A_\ell - \sum_{\ell'=\ell+1}^{L-1} \psi^{(\ell')A_{\ell'}} \right) \tag{4.40}$$

where $f_\eta(\cdot)$ is the PDF of η (in our case Gaussian PDF).

The advantage of this alternative description of multilevel code is that each equivalent sub-channel has a single (real valued) output which reduces the complexity of the decoder. Moreover, the channel outputs can be updated recursively after each stage of decoding.

Simply input of the first decoder ($\ell = 0$) is vector $\mathbf{Y}^{(0)} = \mathbf{Y}$ (the physical channel output itself). After decoding the first level codeword $\mathbf{X}^{(0)}$, the receiver can simply subtract the contribution of this codeword from the channel output and form:

$$\mathbf{Y}^{(1)} = \mathbf{Y} - \begin{bmatrix} (-1)^{X_0^{(0)}} A_0 \\ (-1)^{X_1^{(0)}} A_0 \\ \vdots \\ (-1)^{X_{N-1}^{(0)}} A_0 \end{bmatrix} \tag{4.41}$$

which is the input of second level "equivalent" channel. This process can be repeated at every stage of decoding. Namely, after decoding the codeword of level ℓ, it is sufficient to compute

$$\mathbf{Y}^{(\ell+1)} = \mathbf{Y}^{(\ell)} - \begin{bmatrix} (-1)^{X_0^{(\ell)}} A_\ell \\ (-1)^{X_1^{(\ell)}} A_\ell \\ \vdots \\ (-1)^{X_{N-1}^{(\ell)}} A_\ell \end{bmatrix} \tag{4.42}$$

and use it for the next stage of decoding. Note that due to the recursive manner of the equation we just described there is no need to store all previously decoded codewords, just the channel output vector will be updated at each stage by subtracting the last decision from it.

Moreover, we have some other nice recursions that can be useful for code construction and computation of likelihood ratios (which are inputs of SC decoder):

Proposition 4.1. *The channel transition probabilities of the "equivalent" channels* $\tilde{W}^{\ell}$ *can be obtained through following recursions:*

$$\tilde{W}^{\ell}\left(y|\psi^{(\ell)}\right) = \frac{1}{2}\sum_{\psi^{(\ell+1)}\in\{\pm1\}} \tilde{W}^{\ell+1}\left(y-\psi^{(\ell)}A_{\ell}|\psi^{(\ell+1)}\right) \tag{4.43a}$$

$$\tilde{W}^{L-1}\left(y|\psi^{(L-1)}\right) = f_{\eta}\left(y-\psi^{(L-1)}A_{L-1}\right) \tag{4.43b}$$

Proof. Proof will be followed by simple manipulations of (4.40) □

Proposition 4.2. *The likelihood ratios for the "equivalent channels"* $\tilde{W}^{\ell}\left(y^{(\ell)}|\psi^{(\ell)}\right)$ *defined as:*

$$\Lambda_{\ell}(y) \triangleq \frac{\tilde{W}^{\ell}\left(y|1\right)}{\tilde{W}^{\ell}\left(y|-1\right)} \tag{4.44}$$

can be computed via the double recursions:

$$\Lambda_{\ell}(y) = F_{\ell+1}(y,A_{\ell})\frac{1+\Lambda_{\ell+1}(y-A_{\ell})}{1+\Lambda_{\ell+1}(y+A_{\ell})}, \tag{4.45a}$$

$$F_{\ell}(y,B) = F_{\ell+1}(y+A_{\ell},B)\frac{1+\Lambda_{\ell+1}(y-B+A_{\ell})}{1+\Lambda_{\ell+1}(y+B+A_{\ell})} \tag{4.45b}$$

with initial values:

$$\Lambda_{L-1}(y) = \frac{f_{\eta}\left(y-A_{L-1}\right)}{f_{\eta}\left(y+A_{L-1}\right)} \tag{4.46a}$$

$$F_{L-1}(y,B) = \frac{f_{\eta}\left(y-B-A_{L-1}\right)}{f_{\eta}\left(y+B-A_{L-1}\right)} \tag{4.46b}$$

Note that $F(\cdot,\cdot)$ *is an auxiliary value defined as:*

$$F_{\ell}(y,B) \triangleq \frac{\tilde{W}^{\ell}\left(y-B|-1\right)}{\tilde{W}^{\ell}\left(y+B|-1\right)}.$$

Proof. Considering the recursions in transition probabilities:

$$\begin{aligned}
\Lambda_{\ell}(y) &= \frac{\tilde{W}^{\ell}\left(y|1\right)}{\tilde{W}^{\ell}\left(y|-1\right)}\\
&= \frac{\tilde{W}^{\ell+1}\left(y-A_{\ell}|1\right)+\tilde{W}^{\ell+1}\left(y-A_{\ell}|-1\right)}{\tilde{W}^{\ell+1}\left(y+A_{\ell}|1\right)+\tilde{W}^{\ell+1}\left(y+A_{\ell}|-1\right)}\\
&= \frac{\tilde{W}^{\ell+1}\left(y-A_{\ell}|-1\right)}{\tilde{W}^{\ell+1}\left(y+A_{\ell}|-1\right)}\frac{1+\Lambda_{\ell+1}(y-A_{\ell})}{1+\Lambda_{\ell+1}(y+A_{\ell})}\\
&= F_{\ell+1}(y,A_{\ell})\frac{1+\Lambda_{\ell+1}(y-A_{\ell})}{1+\Lambda_{\ell+1}(y+A_{\ell})}
\end{aligned}$$

Similarly:

$$\begin{aligned}
F_\ell(y, B) &= \frac{\tilde{W}^\ell\,(y - B| - 1)}{\tilde{W}^\ell\,(y + B| - 1)} \\
&= \frac{\tilde{W}^{\ell+1}\,(y - B + A_\ell|1) + \tilde{W}^{\ell+1}\,(y - B + A_\ell| - 1)}{\tilde{W}^{\ell+1}\,(y + B + A_\ell|1) + \tilde{W}^{\ell+1}\,(y + B + A_\ell| - 1)} \\
&= \frac{\tilde{W}^{\ell+1}\,(y - B + A_\ell| - 1)}{\tilde{W}^{\ell+1}\,(y + B + A_\ell| - 1)} \frac{1 + \Lambda_{\ell+1}(y - B + A_\ell)}{1 + \Lambda_{\ell+1}(y + B + A_\ell)} \\
&= F_{\ell+1}(y + A_\ell, B)\frac{1 + \Lambda_{\ell+1}(y - B + A_\ell)}{1 + \Lambda_{\ell+1}(y + A + A_\ell)}
\end{aligned}$$

□

Remark. In practice it is also possible to work with the logarithm domain version of the recursions in Proposition 4.2. First advantage of this is that for the Gaussian channel we have the following simple initial conditions:

$$\begin{aligned}
\lambda_{L-1}(y) = \log \Lambda_{L-1}(y) &= \frac{2A_{L-1}}{\sigma^2} y, \\
\log F_{L-1}(y, B) &= \frac{2B}{\sigma^2}\,(y + A_{L-1})\,.
\end{aligned}$$

Moreover, the recursions will be very similar to the "minus" combination formulae for log-likelihood ratios in SC decoder which can be (approximately) implemented using basic addition/comparison arithmetic operations (as we saw in Chapter 1).

4.5.2 Sensitivity to Change of SNR

One particular property of AWGN channel is that increase (decrease) in SNR is equivalent to upgrading (degrading) the channel (see Example 1.1).

Furthermore, as we see in Theorem 4.3, if the underlying non-binary channel $W : \mathcal{X} \longrightarrow \mathcal{Y}$ is upgraded (degraded) all binary sub-channels forged in Definition 4.2 will be upgraded (degraded).

Combining these results with the results of Corollary 1.5 we can conclude that both of the coding schemes over AWGN are not too sensitive to change of SNR in the following sense:

Assume we have designed a code for a certain rate whose word-error probability is a certain acceptable amount. If SNR is increased, the channel will be upgraded. Even though the capacity of the new channel is higher and we can probably have a code with higher rate, the current code (designed for a channel with lower SNR) will perform at least as good as before on the new channel since the indices in the information bits set correspond to the polarized channels that are upgraded with respect to the low SNR case. Note that this code might not be exactly the polar code for the new channel since (with the same rate) we might be able to find better polarized channels to put in the information bits set.

4.5.3 Simulation Results

In this section, we provide some simulation results on performance of multilevel and non-binary polar codes for ASK signaling over AWGN.

We have simulated multilevel polar codes for the alphabet sizes of up to 32 and block lengths of 256, 1024 and 4096. At each SNR, different codes are designed for the rates from 50% to 90% of channel capacity at that SNR.

Asymptotically (as $N \to \infty$) the binary polar codes at all levels can be decoded without errors as far as the individual rates are kept below the individual capacities of binary sub-channels. Hence there would be no error-propagation issues. However, in finite block length, decoding errors in a specific code can affect the decoding result of (potentially all) following level codes.

Recall that, we conventionally defined the bit-addressing map for ASK as:

$$\mathcal{M}(\vec{x}) = \sum_{\ell=0}^{L-1} (-1)^{x^{(\ell)}} 2^{\ell}$$

which means the codeword which is decode first ($\ell = 0$) corresponds to the signal with low amplitude (which is also contaminated with all higher level, high amplitude signals as well as additive noise). In contrast, the last level codeword appears as a high-amplitude signal in channel input and by the time of decoding it should only contain the effects of additive Gaussian noise.

However, we might change this ordering. For example assume the first level codeword is mapped to the highest amplitude signal in the Gaussian channel input (which is again contaminated by the signal components of other levels that now have lower amplitudes) while the last codeword is mapped to the lowest amplitude signal, hence the bit-addressing map would be:

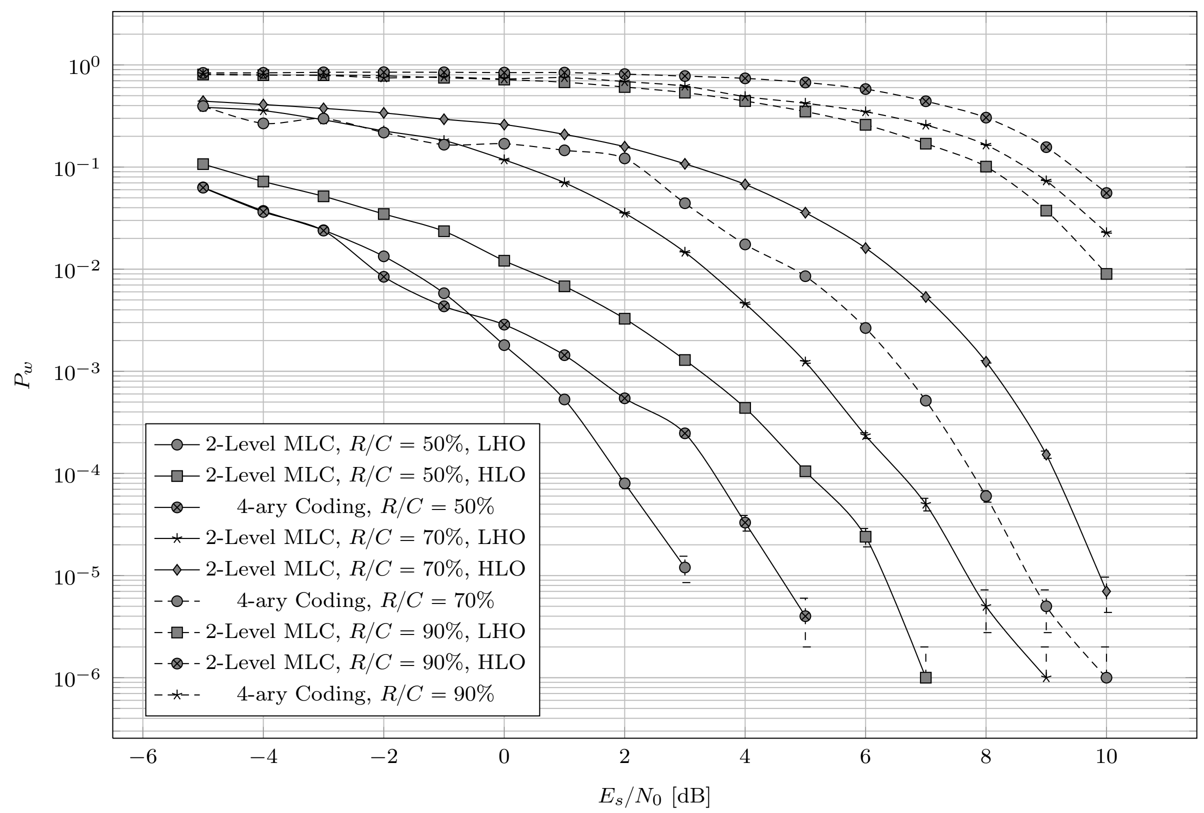
$$\mathcal{M}(\vec{x}) = \sum_{\ell=0}^{L-1} (-1)^{x^{(\ell)}} 2^{L-1-\ell}$$

There exist $L!$ different orderings for assigning the codewords to different amplitudes. Since this ordering can change the performance of code (in finite length) we are interested to see its effect by means of simulations.

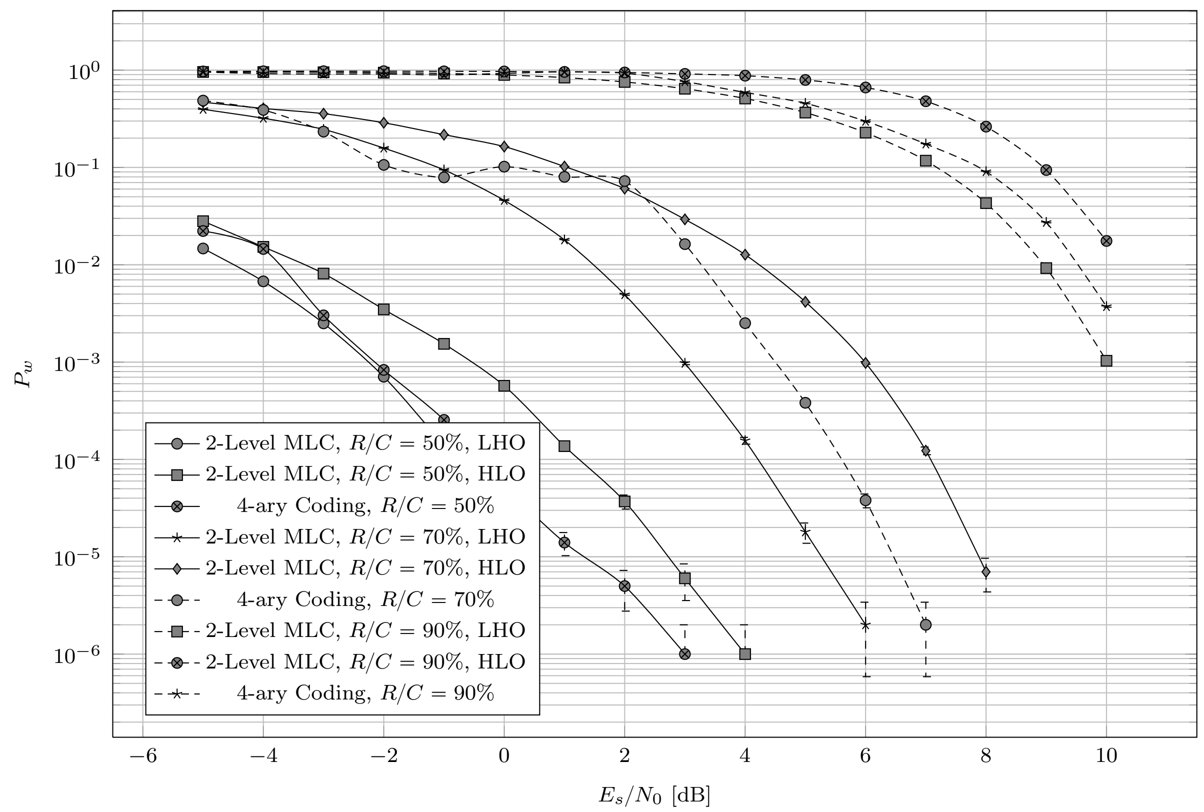
However, for the sake of simplicity, we have only considered two different orderings that we just mentioned:

- "High to Low Ordering" (abbreviated as *HLO* in the plots) which means the first decoded codeword is mapped to the highest amplitude (2^{L-1}) signal component and the last one to the lowest amplitude.
- "Low to high ordering" (abbreviated as *LTO*) which means assigning the first codeword to be decoded to the lowest amplitude component and the last codeword to the highest amplitude component.

Additionally, we have simulated the non-binary polar codes (using the polarizing map of (4.21)) at the same rates to compare their performance with that of multilevel coding. However, due to increasing time-complexity of non-binary decoder and code construction we only have their performance results for limited block lengths and alphabet sizes.

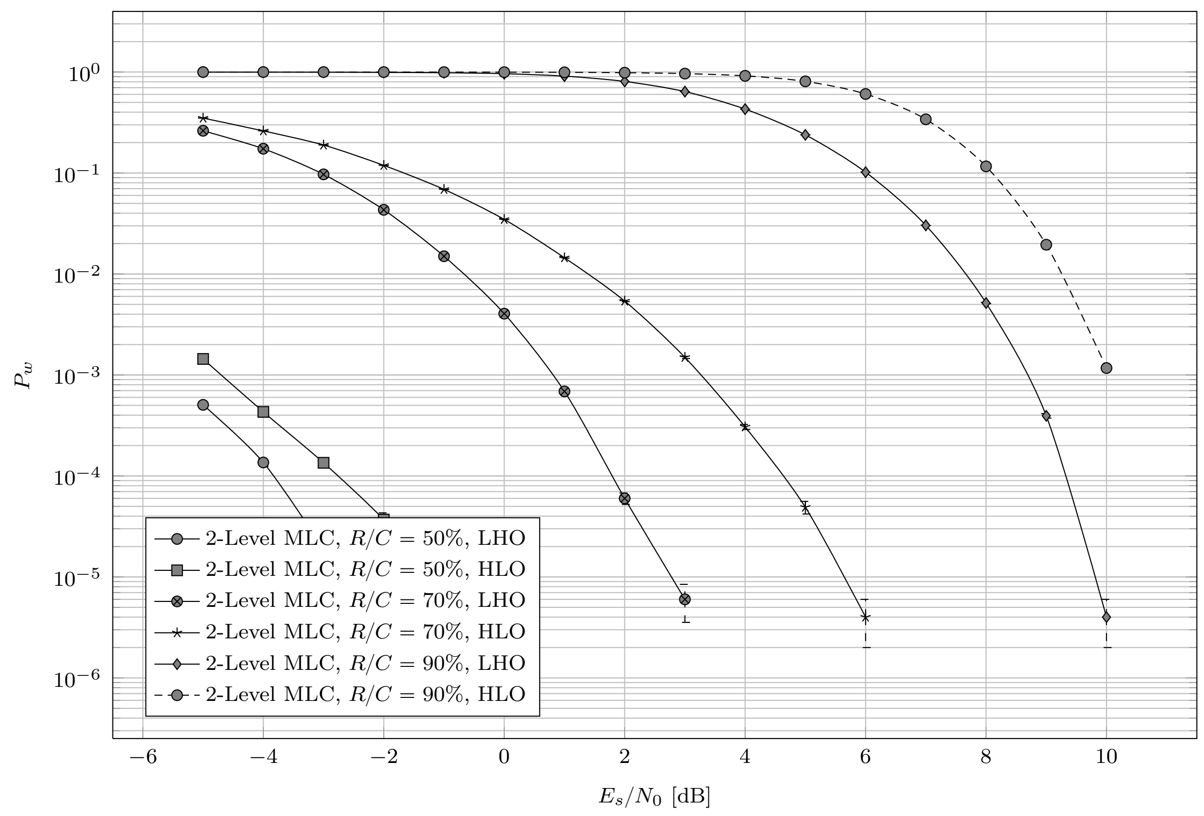


(a) $N = 256$



(b) $N = 1024$

Figure 4.3: Performance of Polar Codes for 4-ASK signaling over AWGN



(c) $N = 4096$

Figure 4.3: Performance of Polar Codes for 4-ASK signaling over AWGN

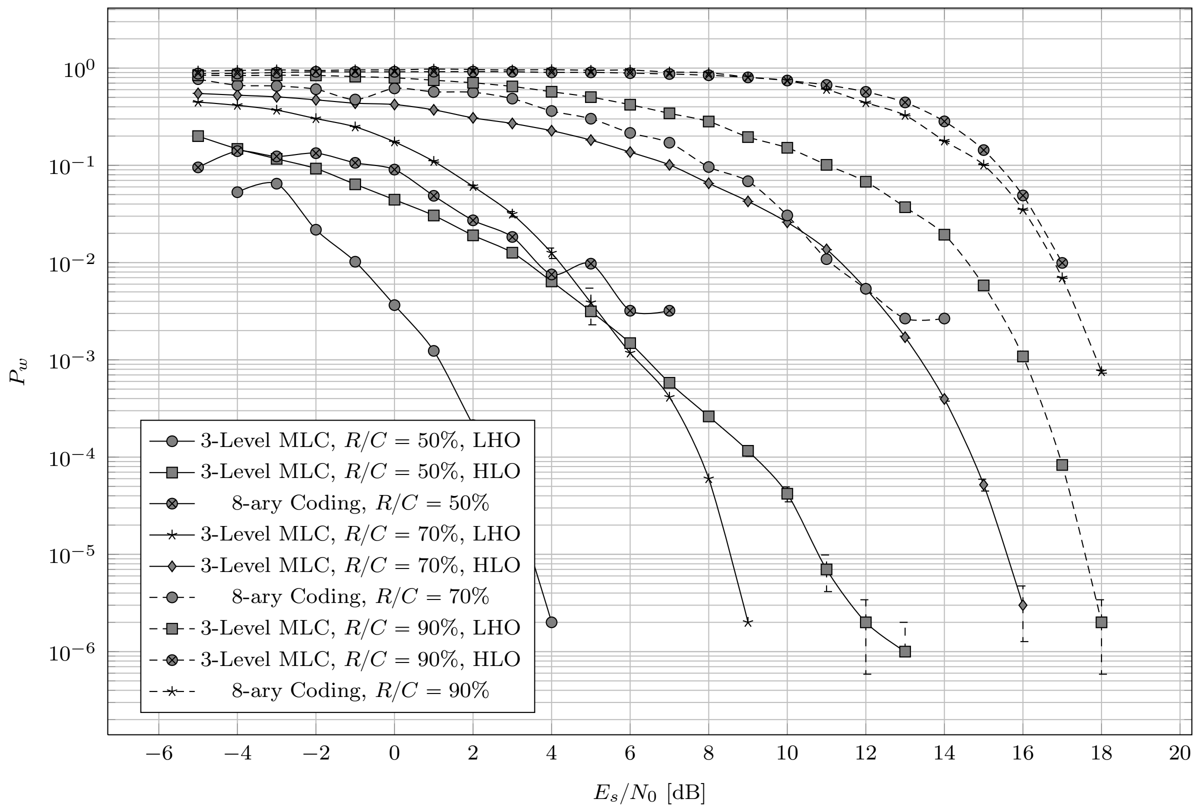
As depicted in figures 4.3 to 4.6, multilevel polar codes can be used at rates close to the symmetric capacity of signaling scheme with low word-error probability. We see that in order for the rate to be close to the capacity we need large block lengths. Typically with block length of $N = 4096$ and at operational SNRs, we can approach the rates close to 90 % of capacity by employing the multilevel polar codes.

Unlike one would expect at first glance, the order of assigning codewords to signal levels seems to be very important. More surprisingly, the simulation results suggest that decoding the weak codewords first results in much better word-error rates. We will discus in more detail about this observation further.

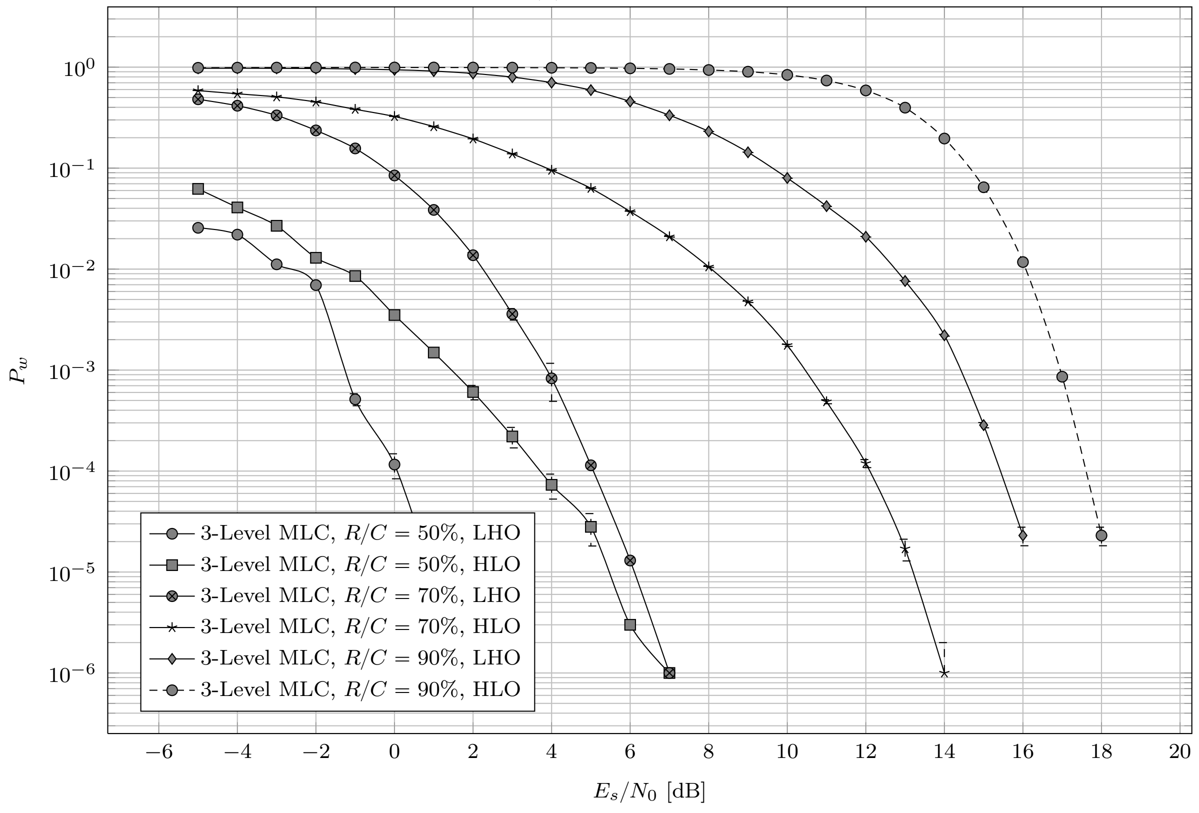
We can also see the polarization effect from the curves. Recall that at different SNRs codes with different rates are employed such that the ratio between code rate and capacity is constant. So in general there is no reason for the word-error rate to decrease when we increase the SNR. However, we see this and the reason is that as we increase the SNR the underlying B-DMCs become better channels. Namely their capacity increases and Bhattacharyya parameter decreases. Hence, at higher SNR polarization starts with a channel which is closer to the extremal points and after a fixed number of polarizing transforms we will have more number of extremal channels (and less unpolarized ones). So even with increase of rate we get better performance.

To some extents, this behavior appears in Q-ary codes well. However, we see more fluctuations in curves.

A remarkable observation is that despite more decoding complexity non-binary codes

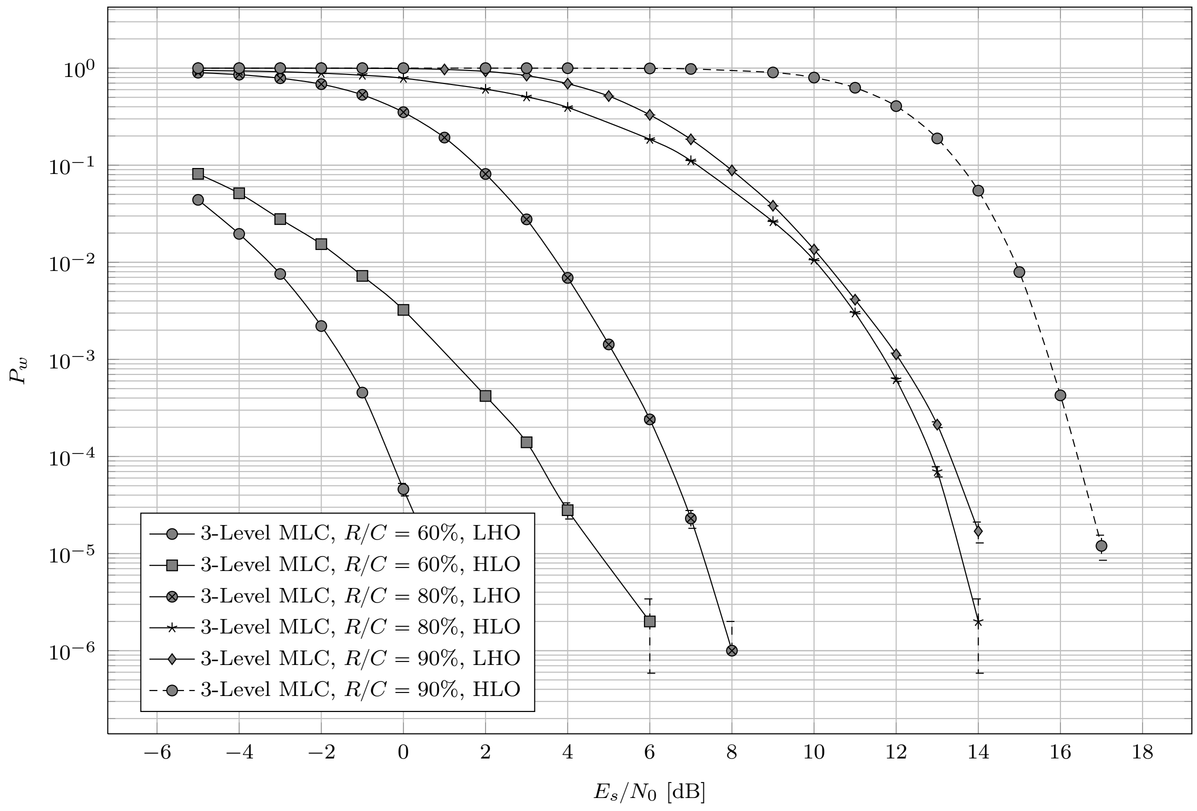


(a) $N = 256$



(b) $N = 1024$

Figure 4.4: Performance of Polar Codes for 8-ASK signaling over AWGN



(c) $N = 4096$

Figure 4.4: Performance of Polar Codes for 8-ASK signaling over AWGN

does not perform better than multilevel codes. This evidence can imply that using the proposed polarizing map of (4.21) does not polarize the non-binary channel fast enough.

4.5.4 Remarks On The Order of Decoding

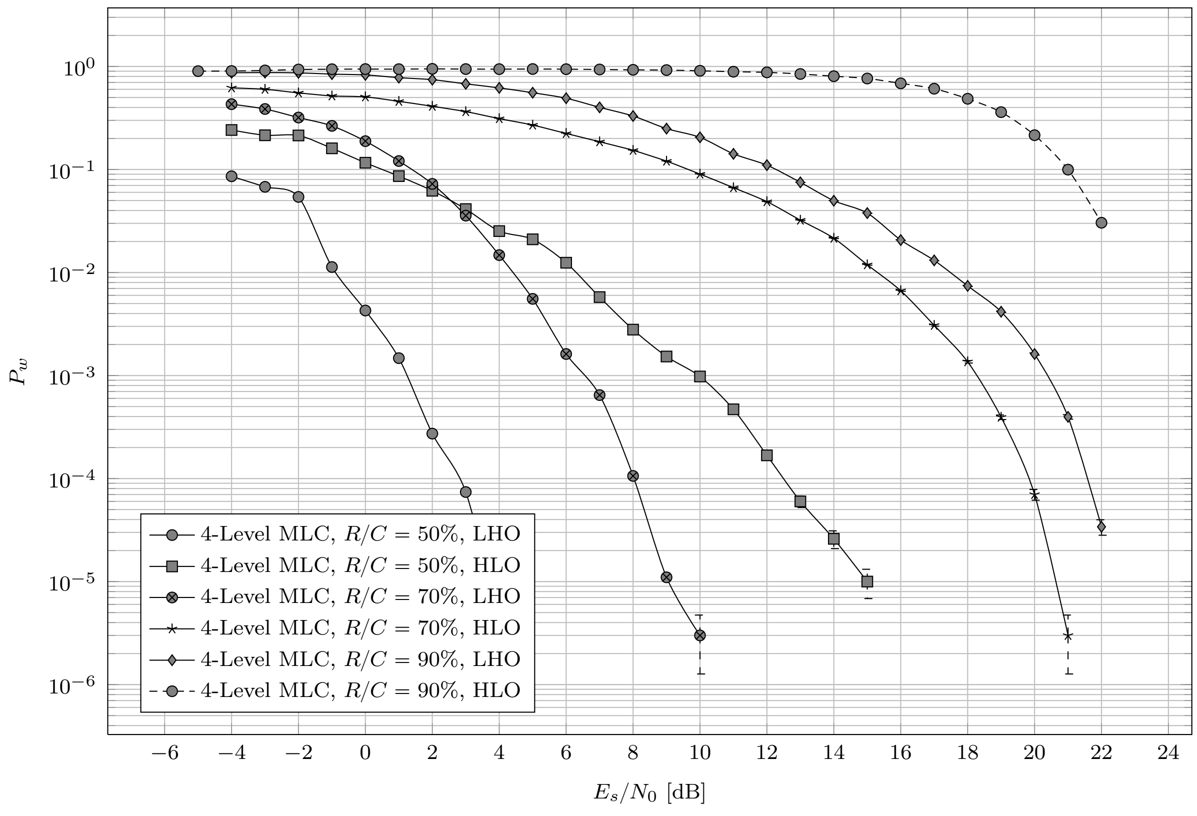
Numerical results on multilevel polar codes suggest that better performance is obtained if decoding is started from the codeword assigned to weakest amplitude and continued toward the ones mapped to higher amplitude components compared to the reverse case. Here we would like to shortly explain the potential reasons of this behavior.

Let us consider only the simple two level case. Each use of channel will produce an output in the form of:

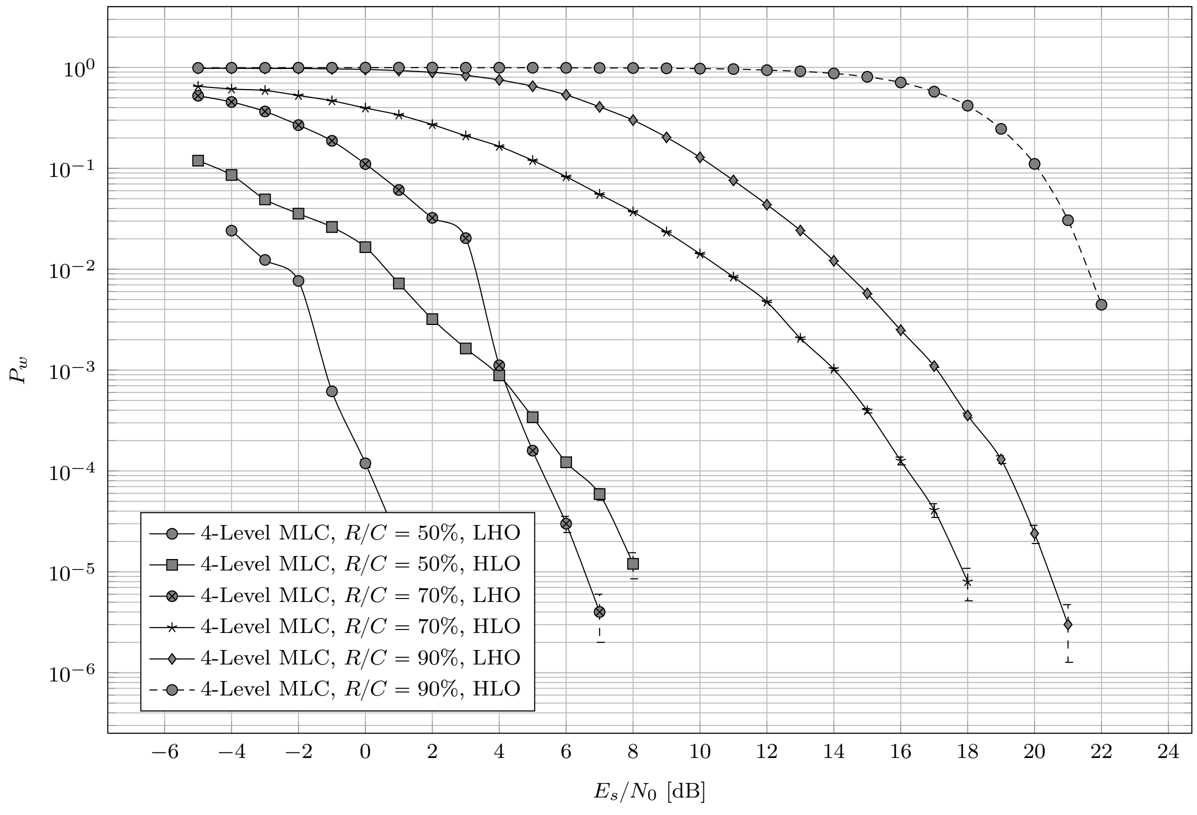
$$Y_i = \Psi_i^{(0)} A_0 + \Psi_i^{(1)} A_1 + \eta_i, \quad i = 0, 1, \cdots, N-1. \tag{4.47}$$

We can assume η is unit variance Gaussian noise without loss of generality and $A_1 \geqslant A_0$.

For the multilevel code, we essentially deal with two channels, a pure Gaussian channel (which is the channel seen by the code decoded last) and a channel whose noise is a mixture of Gaussian noise η and the bipolar signal of the other level ($\pm A_0$ or $\pm A_1$ depending on the order of decoding).

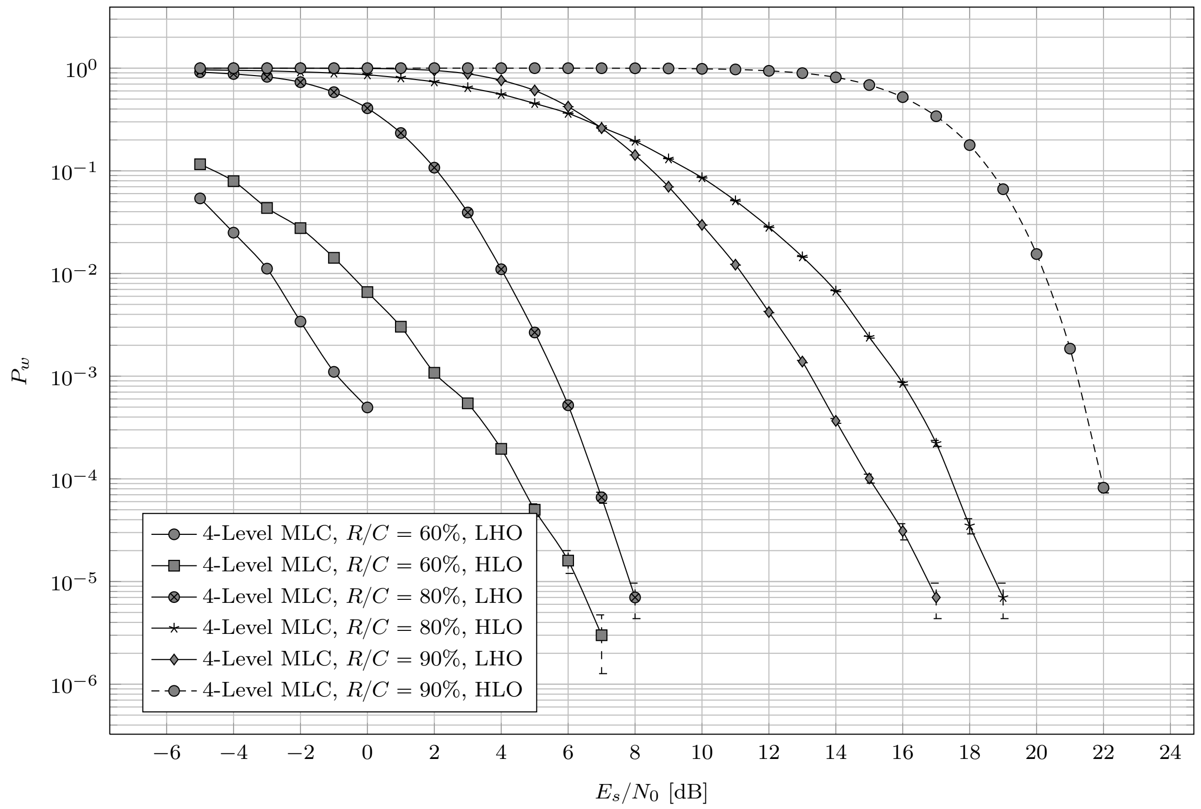


(a) $N = 256$



(b) $N = 1024$

Figure 4.5: Performance of Polar Codes for 16-ASK signaling over AWGN



(c) $N = 4096$

Figure 4.5: Performance of Polar Codes for 16-ASK signaling over AWGN

Let's denote the Gaussian channel with W_G^A where A is the amplitude of signal sent over that channel and the other channel with W_M^A (again A is the amplitude of signal sent over that channel).

Now the question is how the channel pairs $(W_G^{A_1}, W_M^{A_0})$ and $(W_G^{A_0}, W_M^{A_1})$, and more importantly multilevel polar codes on them compare. Even though there is no standard comparison figure.

Clearly:

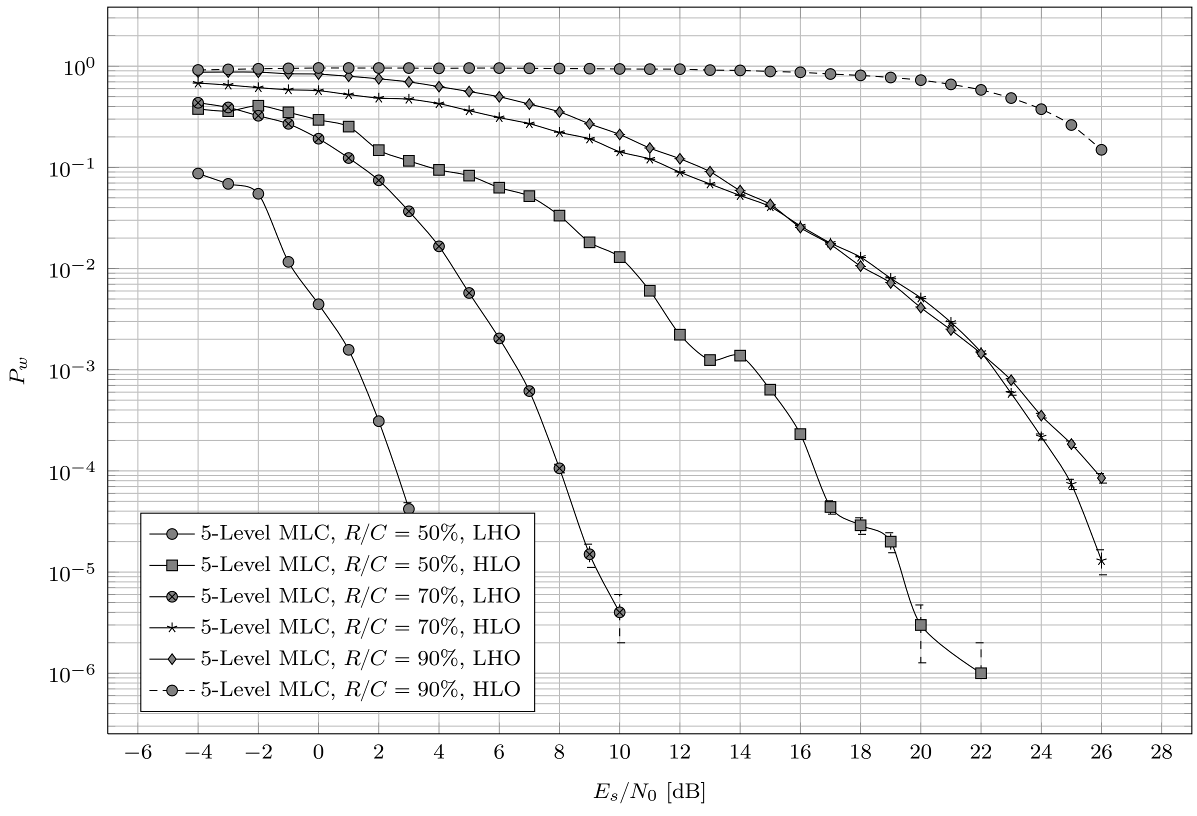
$$I\left(W_G^{A_0}\right) + I\left(W_M^{A_1}\right) = I\left(W_G^{A_1}\right) + I\left(W_M^{A_0}\right) \tag{4.48}$$

Moreover we can verify that:

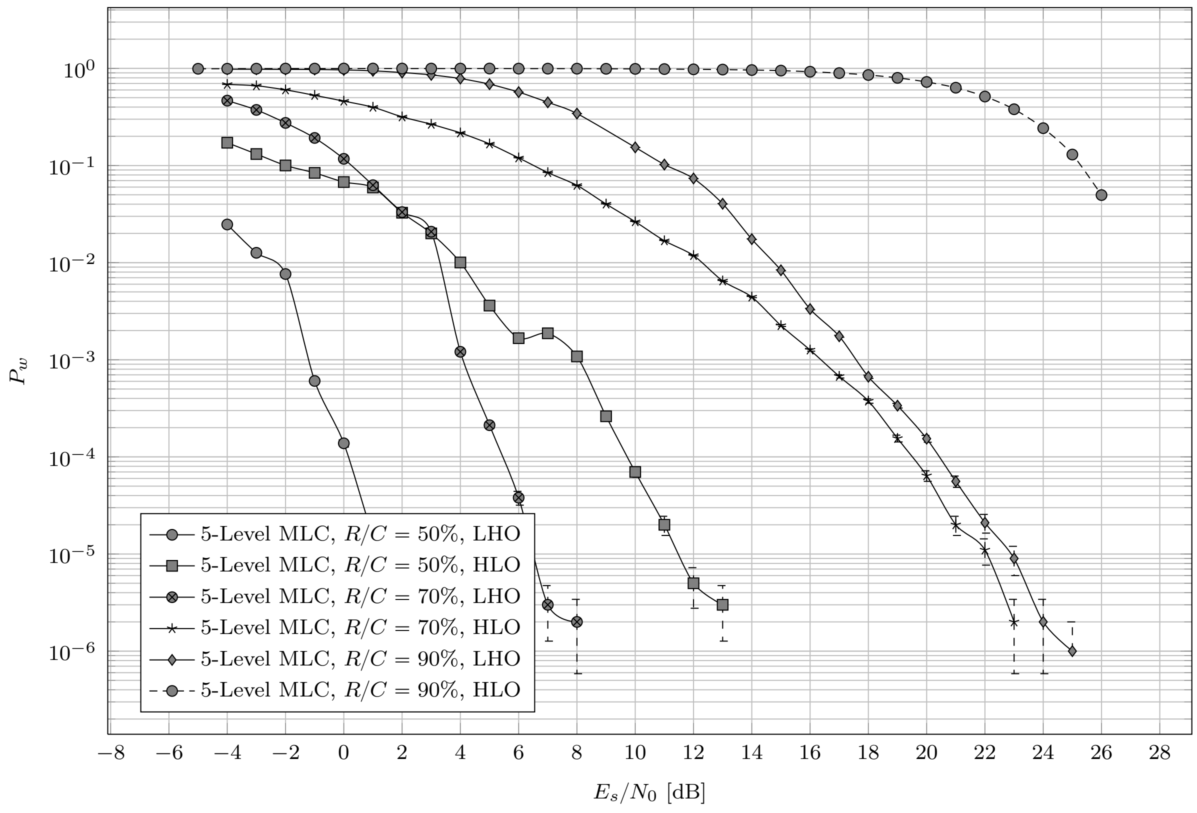
$$\begin{gathered} I\left(W_G^{A_0}\right) \leqslant I\left(W_G^{A_1}\right) \\ I\left(W_M^{A_1}\right) \geqslant I\left(W_M^{A_1}\right) \\ Z\left(W_G^{A_0}\right) \geqslant Z\left(W_G^{A_1}\right) \\ Z\left(W_M^{A_1}\right) \leqslant Z\left(W_M^{A_1}\right) \end{gathered}$$

which shows in the $Z - I$ plane (see Figure 1.1), both of the channels $(W_G^{A_1}, W_M^{A_0})$ are closer to the extremal channels compared to $(W_G^{A_0}, W_M^{A_1})$

Hence, after a certain number of polarization levels, the resulting channels from the first pair are more polarized than the results from the second pair (or there are less unpolarized channels in the set of forged B-DMCs). Consequently in order to choose a

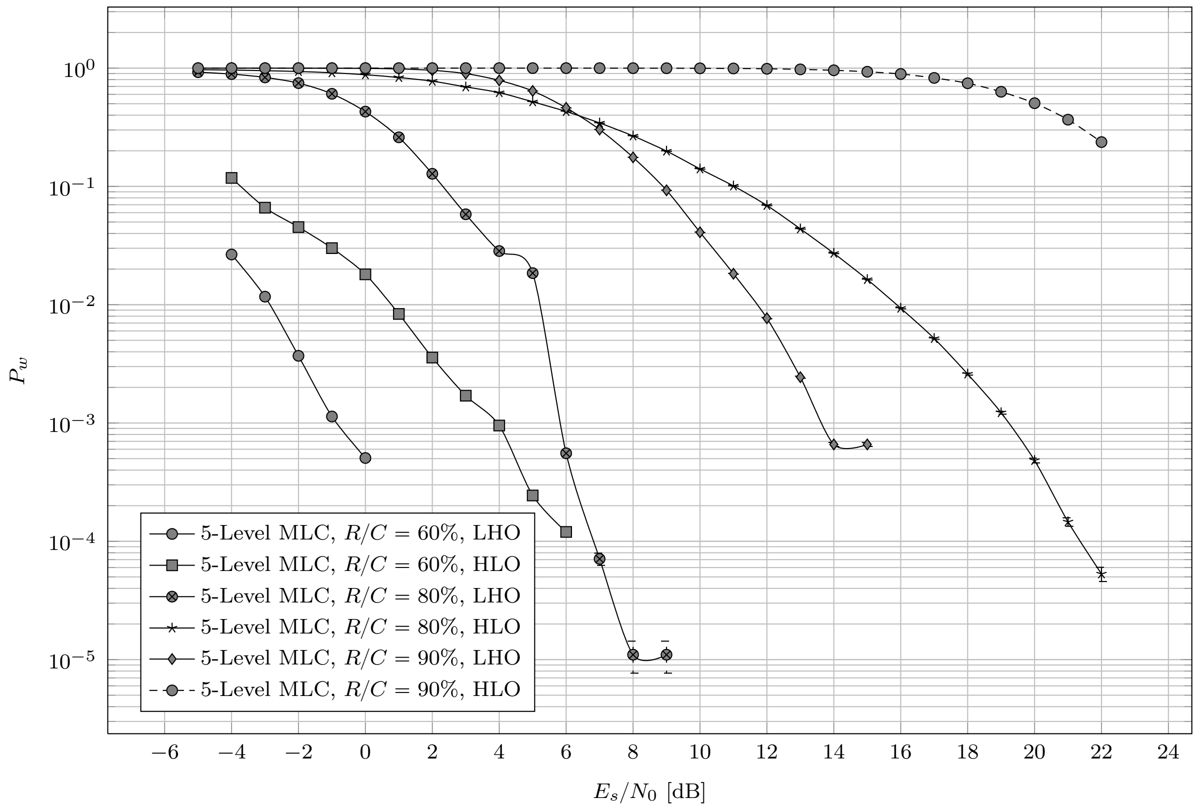


(a) $N = 256$



(b) $N = 1024$

Figure 4.6: Performance of Polar Codes for 32-ASK signaling over AWGN



(c) $N = 4096$

Figure 4.6: Performance of Polar Codes for 32-ASK signaling over AWGN

specific number of channels among all $2N$ polarized channels (N channels for each level) we can pickup more extremal channels if we start from $(W_G^{A_1}, W_M^{A_0})$.

Numerical computations show another evidence which is:

$$Z\left(W_G^{A_0}\right) + Z\left(W_M^{A_1}\right) \geqslant Z\left(W_G^{A_1}\right) + Z\left(W_M^{A_0}\right).$$

See Figure 4.7. In this example we have used the standard ASK amplitude assignment, namely $\frac{A_1}{A_0} = 2$.

This evidence can be proved analytically as follows:

Proposition 4.3. *Consider the 4-ary AWGN channel whose input/output relationship is:*

$$Y_i = \Psi^{(0)} A_0 + \Psi^{(1)} A_1 + \eta$$

where $\eta \sim \mathcal{N}(0,1)$ *independent of* $\Psi^{(1)}, \Psi^{(0)}$, $A_1 \geqslant A_0$ *by assumption and* $\left(\Psi^{(0)}, \Psi^{(1)}\right)$ *is the channel input chosen uniformly from* $\{\pm 1\}^2$.

Let the two binary sub-channels seen in the two-stage decoding be:

$$\begin{aligned} W_M^{A_0} &: \Psi^{(0)} \mapsto A_0 \Psi^{(0)} + \left(\Psi^{(1)} A_1 + \eta\right), \\ W_G^{A_1} &: \Psi^{(1)} \mapsto A_1 \Psi^{(1)} + \eta \end{aligned}$$

respectively (if we decode from the low-amplitude bit to the high-amplitude bit) or:

$$\begin{aligned} W_M^{A_1} &: \Psi^{(1)} \mapsto A_1\Psi^{(1)} + \left(\Psi^{(0)}A_0 + \eta\right), \\ W_G^{A_0} &: \Psi^{(0)} \mapsto A_0\Psi^{(0)} + \eta \end{aligned}$$

respectively. Then

$$Z\left(W_G^{A_0}\right) + Z\left(W_M^{A_1}\right) \geqslant Z\left(W_G^{A_1}\right) + Z\left(W_M^{A_0}\right). \tag{4.49}$$

Proof. We know:

$$\begin{aligned} W_M^{A_0}\left(y|\Psi^{(0)}\right) &= \sum_{\Psi^{(1)}\in\{\pm1\}} \frac{1}{2}\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\left(y-A_1\Psi^{(1)}-A_0\Psi^{(0)}\right)^2} \\ W_G^{A_1}\left(y|\Psi^{(1)}\right) &= \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\left(y-A_1\Psi^{(1)}\right)^2} \end{aligned}$$

The Bhattacharyya parameter of the Gaussian sub-channel is known to be:

$$Z\left(W_G^{A_1}\right) = e^{-\frac{A_1^2}{2}}. \tag{4.50}$$

For the sub-channel with mixed noise we have:

$$\begin{aligned} W_M^{A_0}(y|1)\,W_M^{A_0}(y|-1) &= \sum_{\Psi^{(1)}\in\{\pm1\}} \frac{1}{2}\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\left(y-A_1\Psi^{(1)}-A_0\right)^2} \sum_{\Psi^{(1)}\in\{\pm1\}} \frac{1}{2}\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\left(y-A_1\Psi^{(1)}+A_0\right)^2} \\ &= \left(\frac{1}{2}\frac{1}{\sqrt{2\pi}}\right)^2 \Big[e^{-y^2+2A_1y-A_0^2-A_1^2} + e^{-y^2+2A_0A_1-A_0^2-A_1^2} + \\ &\qquad e^{-y^2-2A_0A_1-A_0^2-A_1^2} + e^{-y^2-2A_0y-A_0^2-A_1^2}\Big] \\ &= \left(\frac{1}{2}\frac{1}{\sqrt{2\pi}}\right)^2 e^{-\left(y^2+A_0^2+A_1^2\right)} \left[2\cosh(2A_1y) + 2\cosh(2A_0A_1)\right] \\ &\overset{(a)}{=} \left(\frac{1}{\sqrt{2\pi}}\right)^2 e^{-\left(y^2+A_0^2+A_1^2\right)} \left[\cosh^2(A_1y) + \cosh^2(A_0A_1) - 1\right] \end{aligned}$$

where in (a) we have used the identity $\cosh(2x) = 2\cosh^2(x) - 1$. The Bhattacharyya parameter of the sub-channel will therefore be:

$$\begin{aligned} Z\left(W_M^{A_0}\right) &= \int_y \sqrt{W_M^{A_0}(y|1)\,W_M^{A_0}(y|-1)}dy \\ &= \int_y \frac{1}{\sqrt{2\pi}} e^{-y^2/2} e^{-\left(A_0^2+A_1^2\right)/2} \sqrt{\cosh^2(A_1y) + \cosh^2(A_0A_1) - 1dy} \\ &= e^{-\left(A_0^2+A_1^2\right)/2} \mathbb{E}\left[\sqrt{\cosh^2(A_1\eta) + \cosh^2(A_0A_1) - 1}\right] \end{aligned} \tag{4.51}$$

On the other side, it is easy to verify $e^{-\alpha^2/2} = \mathbb{E}\left[\cosh(\alpha\eta)\right]$ which yields:

$$Z\left(W_G^{A_1}\right) = e^{-A_1^2/2} = e^{-\left(A_0^2+A_1^2\right)/2} e^{-A_0^2/2} = e^{-\left(A_0^2+A_1^2\right)/2} \mathbb{E}\left[\cosh(A_0\eta)\right]. \tag{4.52}$$

Combining (4.51) and (4.52) we can write:

$$Z\left(W_G^{A_1}\right) + Z\left(W_M^{A_0}\right) = e^{-\left(A_0^2+A_1^2\right)/2}\mathbb{E}\left[\cosh(A_0\eta) + \sqrt{\cosh^2(A_1\eta) + \cosh^2(A_0A_1) - 1}\right] \tag{4.53}$$

Similarly we can show:

$$Z\left(W_G^{A_0}\right) + Z\left(W_M^{A_1}\right) = e^{-\left(A_0^2+A_1^2\right)/2}\mathbb{E}\left[\cosh(A_1\eta) + \sqrt{\cosh^2(A_0\eta) + \cosh^2(A_0A_1) - 1}\right] \tag{4.54}$$

Therefore the difference between the left hand side and the right hand side of (4.49) is:

$$\begin{aligned}\left[Z\left(W_G^{A_0}\right) + Z\left(W_M^{A_1}\right)\right] - \left[Z\left(W_G^{A_1}\right) + Z\left(W_M^{A_0}\right)\right] = \\ e^{-\left(A_0^2+A_1^2\right)/2}\mathbb{E}\Bigg[\cosh(A_1\eta) + \sqrt{\cosh^2(A_0\eta) + \cosh^2(A_0A_1) - 1} \\ - \cosh(A_0\eta) - \sqrt{\cosh^2(A_1\eta) + \cosh^2(A_0A_1) - 1}\Bigg]\end{aligned} \tag{4.55}$$

Now we show as far as $A_1 \geqslant A_0$ the term inside the expectation is non-negative which proves the claim.

$$\begin{aligned}&\cosh(A_1\eta) + \sqrt{\cosh^2(A_0\eta) + \cosh^2(A_0A_1) - 1} \\ &\quad - \cosh(A_0\eta) - \sqrt{\cosh^2(A_1\eta) + \cosh^2(A_0A_1) - 1} \\ &= \cosh(A_1\eta) - \cosh(A_0\eta) \\ &\quad + \frac{\cosh^2(A_0\eta) - \cosh^2(A_1\eta)}{\sqrt{\cosh^2(A_0\eta) + \cosh^2(A_0A_1) - 1} + \sqrt{\cosh^2(A_1\eta) + \cosh^2(A_0A_1) - 1}} \\ &= (\cosh(A_1\eta) - \cosh(A_0\eta)) \\ &\quad \left[1 - \frac{\cosh(A_0\eta) + \cosh(A_1\eta)}{\sqrt{\cosh^2(A_0\eta) + \cosh^2(A_0A_1) - 1} + \sqrt{\cosh^2(A_1\eta) + \cosh^2(A_0A_1) - 1}}\right]\end{aligned} \tag{4.56}$$

First, since $A_1 \geqslant A_0$, $\cosh(A_1\eta) \geqslant \cosh(A_0\eta)$. Moreover, since $\cosh^2(A_0A_1) \geqslant 1$,

$$\sqrt{\cosh^2(A_0\eta) + \cosh^2(A_0A_1) - 1} \geqslant \cosh(A_0\eta).$$

Similarly

$$\sqrt{\cosh^2(A_1\eta) + \cosh^2(A_0A_1) - 1} \geqslant \cosh(A_1\eta),$$

hence:

$$\sqrt{\cosh^2(A_1\eta) + \cosh^2(A_0A_1) - 1} + \sqrt{\cosh^2(A_0\eta) + \cosh^2(A_0A_1) - 1} \geqslant \cosh(A_1\eta) + \cosh(A_0\eta).$$

which shows the term inside the brackets in (4.56) is non-negative. This proves the claim. □

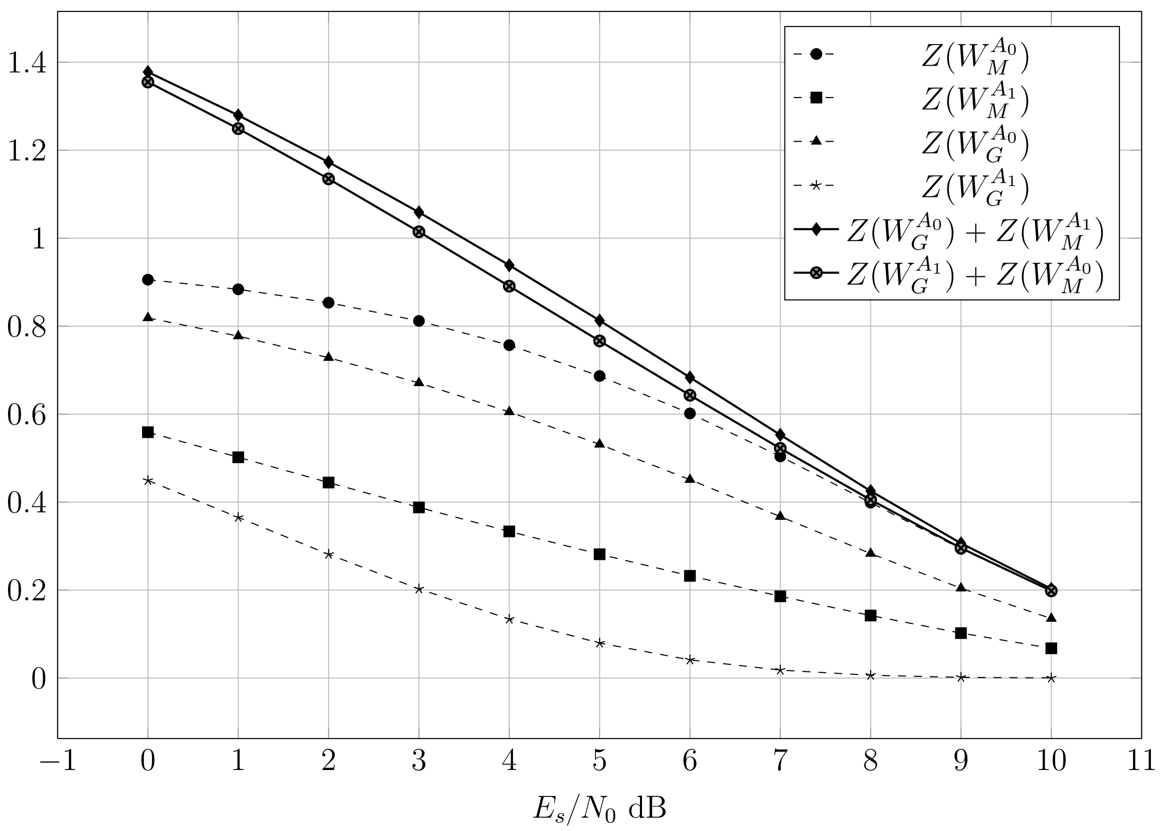


Figure 4.7: Comparison of Bhattacharyya parameters in different decoding orders

The third comparison showing the channel pair $(W_G^{A_1}, W_M^{A_0})$ is better than the pair $(W_G^{A_0}, W_M^{A_1})$ is the following: Assume we approximate the mixed-noise channel W_M with a Gaussian channel with the same SNR. Let $P_\ell \triangleq \frac{A_\ell^2}{2}, \quad \ell = 0, 1$. Hence $P_1 \geqslant P_0$.

By (4.48) we have:

$$\frac{1}{2}\log(1+P_0) + \frac{1}{2}\log\left(1+\frac{P_0}{1+P_1}\right) = \frac{1}{2}\log(1+P_1) + \frac{1}{2}\log\left(1+\frac{P_1}{1+P_0}\right) \tag{4.57}$$

Now let us compute the sum of cutoff rates of individual channels in each pair. Since the cutoff rate of a Gaussian channel is $\frac{1}{2}\log\left(1+\frac{\text{SNR}}{2}\right)$ we have:

$$R_0\left(W_G^{A_0}\right) + R_0\left(W_M^{A_1}\right) \approx \frac{1}{2}\left(\log\left(1+\frac{P_0}{2}\right) + \log\left(1+\frac{P_1}{2(1+P_0)}\right)\right) \tag{4.58}$$

$$R_0\left(W_G^{A_1}\right) + R_0\left(W_M^{A_0}\right) \approx \frac{1}{2}\left(\log\left(1+\frac{P_1}{2}\right) + \log\left(1+\frac{P_0}{2(1+P_1)}\right)\right) \tag{4.59}$$

Now we can show:

$$R_0\left(W_G^{A_0}\right) + R_0\left(W_M^{A_1}\right) \leqslant R_0\left(W_G^{A_1}\right) + R_0\left(W_M^{A_0}\right) \tag{4.60}$$

as follows:

$$
\begin{aligned}
& \log\left(\left(1+\frac{P_0}{2}\right)\left(1+\frac{P_1}{2(1+P_0)}\right)\right) \leqslant \log\left(\left(1+\frac{P_1}{2}\right)\left(1+\frac{P_0}{2(1+P_1)}\right)\right) \\
\Longleftrightarrow\quad & \left(1+\frac{P_0}{2}\right)\left(1+\frac{P_1}{2(1+P_0)}\right) \leqslant \left(1+\frac{P_1}{2}\right)\left(1+\frac{P_0}{2(1+P_1)}\right) \\
\Longleftrightarrow\quad & 2\left(P_0+\frac{P_1}{1+P_0}\right)+\frac{P_0P_1}{1+P_0} \leqslant 2\left(P_1+\frac{P_0}{1+P_1}\right)+\frac{P_1P_0}{1+P_1} \\
\Longleftrightarrow\quad & 2\left(P_0+\frac{P_1}{1+P_0}+\frac{P_0P_1}{1+P_0}\right)-\frac{P_0P_1}{1+P_0} \leqslant 2\left(P_1+\frac{P_0}{1+P_1}+\frac{P_0P_1}{1+P_1}\right)-\frac{P_0P_1}{1+P_1} \\
\overset{(a)}{\Longleftrightarrow}\quad & -\frac{P_0P_1}{1+P_0} \leqslant -\frac{P_0P_1}{1+P_1} \\
\Longleftrightarrow\quad & \frac{1}{1+P_0} \geqslant \frac{1}{1+P_1}
\end{aligned}
$$

where (a) is true since (4.57) implies $P_0+\frac{P_1}{1+P_0}+\frac{P_0P_1}{1+P_0}=P_1+\frac{P_0}{1+P_1}+\frac{P_0P_1}{1+P_1}$.

So if we accept the approximation by Gaussian channel, we conclude that even though both channel pairs $(W_G^{A_1}, W_M^{A_0})$ and $(W_G^{A_0}, W_M^{A_1})$ have equal overall capacity, the total cutoff rate of former is higher than that of latter. Hence, in some sense the first pair is better.

All these evidences, show that unlike our initial intuition, assigning the first codeword to weaker signal component should result in better performance when the codeword length is finite.

Conclusion 5

In this thesis, we mainly focused on the issues that arise while employing polar codes in practical communication systems.

In Chapter 1 we studied the polarization phenomena and showed that polar codes can achieve the capacity of all symmetric binary memoryless channels. Their low encoding and decoding complexity, in addition to their explicit construction , makes them good candidates to be employed in real systems.

In particular the successive cancellation decoder equations are more or less similar to the equations of the other decoders which are used currently in operational systems. Those equations are generally implemented using simple arithmetic operations of addition and comparison. However, the non-iterative nature of successive cancellation decoder is an important property that might lead to an analytical proof for robustness of the decoder to the approximations and quantization errors in computations.

The other difficulties with utilizing polar codes in real systems are code construction and sensitivity of code performance to changes in channel conditions. Using the notions of degraded and upgraded channels we described the approximation algorithms proposed by Vardy and Tal that quickly identify good and bad polarized channels.

Despite the exponentially decreasing error probability of polar codes in large block lengths, for low to moderate block lengths they do not perform very well. In Chapter 3 we analyzed the error events of the successive cancellation decoder and based on numerical computations concluded that at certain code rates, decoding errors stem from a single wrong decision which deviates the decoding path and causes the output of decoder to be far away from the sent codeword. This suggests that poor performance of polar codes with finite length at a range of rates not too close to channel's capacity can be significantly improved by employing a means for detecting the single decision error. Even though we couldn't provide any analytical proofs due to hardness of the problem, the simulation results confirm this suggestion.

The other implication of this observation is that by detecting the index of wrongly decoded bit, one can construct list decoders with maximum list size of 2 for polar codes whose performance is much better than that of ordinary successive cancellation decoder.

At this point, next steps can be taken in two different directions:

1. Investigating more properties of error events and proving the considerable gains in performance one can get by single error correction.

2. Discovering the structures and algorithms for detecting and correcting the single errors with negligible rate loss.

Many of the physical channels of interest have a capacity equivalent to more than one bit per channel use. AWGN channel is a well known example. Hence, non-binary symbols should be transmitted across these channels in order to exploit their capacity which emerges the need for non-binary channel coding schemes.

As we saw in Chapter 4 polar coding can be extended to non-binary channels. For this purpose, one alternative is the classical multilevel coding approach which divides the non-binary channel into several binary sub-channels. For each of that channels a separate binary polar code can be employed and combination of those codes achieve the capacity of non-binary channel. We showed that the explicit construction of polar code is a nice property that leads to an explicit rate assignment rule in multilevel polar coding. Low complexity and scalability are two important characteristics of multilevel polar codes. Since both encoder and decoder in this scheme reduce to multiple instances of binary encoder and decoder, implementation of multilevel polar codes will be robust and fast with basic arithmetic operations. Furthermore, we showed that efficient binary code construction algorithms are ready to be applied to binary sub-channels and construct the multilevel code.

The other alternative for non-binary channel coding is the Q-ary polar coding schemes proposed by Şaşoğlu. He has showed that with a slight change in channel combining relationships non-binary channels can be polarized in a fashion analogous to binary channels. Hence, non-binary polar codes can be designed similar to binary polar codes. The complexity of Q-ary decoding is higher than that of multilevel decoder. This difference becomes important especially when the alphabet size increases. The other difficulty with Q-ary decoder is the higher complexity of decoder equations which results in the need for high processing power to carry out multiplications and makes the computations more sensitive to quantization errors.

A substantial issue in Q-ary polar coding is code construction problem. Original code construction method of Arıkan can be extended to Q-ary codes, however, the procedure will be extremely long especially in large alphabet sizes. Consequently, we concluded that generalization of the binary approximation algorithms to non-binary setting is a key requirement for employing these codes.

We also provided some specific results on employing both of the mentioned schemes on Gaussian channels. Generally, a change in SNR of the channel will be equivalent to a change in the underlying channel for which the code is designed. This change, in turn, will require the code to be constructed again as the indices of good and bad channels might be changed. However, in case of the Gaussian channel, we saw that change in SNR is equivalent to upgrading or degrading the underlying memoryless channel. As a result, we can ensure that no significant changes will happen in performance of codes in case of this change so the coding schemes are not sensitive to SNR changes.

Our simulation results show that both of the non-binary coding schemes perform quite well on AWGN channel. Low complexity multilevel polar codes can achieve the capacity of channel with block length of $N = 4096$ which is practically feasible.

We expect the performance of Q-ary codes to be better than that of multilevel codes in return of their increased complexity. However, the simulation results are not consistent with this prediction. Nevertheless, it is notable that choice of polarizing map can affect the performance of code. So one might find polarizing maps other than the one suggested by Şaşoğlu to improve the performance of Q-ary polar codes.

In multilevel polar codes, simulation results show that order of decoding can affect the performance of code. Finding the optimal order of decoding analytically is an interesting problem to study in future.

Bibliography

[1] E. Arıkan, “Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels,” *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3051 –3073, July 2009.

[2] E. Arıkan and E. Telatar, “On the rate of channel polarization,” in *IEEE International Symposium on Information Theory (ISIT), 2009*, July 2009, pp. 1493 –1495.

[3] N. Hussami, R. Urbanke, and S. Korada, “Performance of polar codes for channel and source coding,” in *IEEE International Symposium on Information Theory (ISIT), 2009*, July 2009, pp. 1488 –1492.

[4] S. Hassani, K. Alishahi, and R. Urbanke, “On the scaling of polar codes: Ii. the behavior of un-polarized channels,” in *IEEE International Symposium on Information Theory (ISIT), 2010*, June 2010, pp. 879 –883.

[5] I. Tal and A. Vardy, “List decoding of polar codes,” in *IEEE International Symposium on Information Theory (ISIT), 2011*, August 2011, pp. 1 –5.

[6] ——, “How to construct polar codes,” 2011. [Online]. Available: http://arxiv.org/abs/1105.6164

[7] B. Xia and W. Ryan, “On importance sampling for linear block codes,” in *IEEE International Conference on Communications (ICC) 2003*, vol. 4, May 2003, pp. 2904 – 2908.

[8] S. Kakakhail, S. Reynal, D. Declercq, and V. Heinrich, “Efficient performance evaluation of forward error correcting codes,” in *11th IEEE International Conference on Communication Systems (ICCS), 2008, Singapore*, November 2008, pp. 263 –267.

[9] R. G. Gallager, “Low-density parity-check codes,” 1963.

[10] L. Duan, B. Rimoldi, and R. Urbanke, “Approaching the awgn channel capacity without active shaping,” in *IEEE International Symposium on Information Theory (ISIT), 1997*, June 1997, p. 374.

[11] U. Wachsmann, R. Fischer, and J. Huber, “Multilevel codes: Theoretical concepts and practical design rules,” *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1361 –1391, July 1999.

[12] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing).* Wiley-Interscience, 2006.

[13] E. Şaşoğlu, "Polar Coding Theorems for Discrete Systems," Ph.D. dissertation, Lausanne, 2011. [Online]. Available: http://library.epfl.ch/theses/?nr=5219

[14] E. Şaşoğlu, E. Telatar, and E. Arıkan, "Polarization for arbitrary discrete memoryless channels," in *IEEE Information Theory Workshop (ITW), 2009*, October 2009, pp. 144 –148.