Foundations and Trends® in Communications and Information Theory

Reed-Muller Codes

Suggested Citation: Emmanuel Abbe, Ori Sberlo, Amir Shpilka and Min Ye (2023), "Reed-Muller Codes", Foundations and Trends® in Communications and Information Theory: Vol. 20, No. 1–2, pp 1–156. DOI: 10.1561/0100000123.

Emmanuel Abbe
EPFL
emmanuel.abbe@epfl.ch

Ori Sberlo
Tel Aviv University
ori.sberlo@gmail.com

Amir Shpilka
Tel Aviv University
shpilka@tauex.tau.ac.il

Min Ye
Tsinghua-Berkeley Shenzhen Institute
yeemmi@gmail.com

now
the essence of knowledge
Boston — Delft

Contents

1 Introduction **3**

1.1 Outline of the Survey and Differences from a Previous Version 6

2 Basic Properties of RM Codes **8**

2.1 Definition and Parameters 8

2.2 Recursive Structure and Distance 10

2.3 Connection Between RM Codes and Polar Codes 11

2.4 Duality 13

2.5 Affine-Invariance 15

2.6 Punctured RM Codes are Cyclic Codes 16

2.7 Nonlinear Subcodes of RM Codes: Nordstrom-Robinson Code and Kerdock Codes 18

2.8 General Finite Fields and Locality 20

3 Performance Measures and Important Quantities in Channel Coding **22**

3.1 Information Measures of a Channel 22

3.2 Error Probabilities of a Code Over a Channel 24

3.3 Weight Enumerator of a Code 25

3.4 Sequential Entropy of a Code on a Channel 29

3.5 EXIT Function of a Code on a Channel 30

3.6 Connecting the Performance Measures 35

4 Bounds on the Weight Enumerator **40**
4.1 The Combinatorial Approach 47
4.2 Lower Bounds on the Weight Enumerator 54
4.3 The Analytical Approach 55

5 Bounding the Error Probability to Obtain Capacity-Achieving Results **69**
5.1 RM Codes Achieve Capacity at Low Rate [1], [134] 69
5.2 RM Codes Achieve Capacity on the BEC at High Rate [1], [25] 72
5.3 RM Codes Achieve Capacity on the BEC at Constant Rate [90] 74
5.4 RM Codes Achieve Capacity on BMS Channels at Constant Rate Under Bit-MAP Decoding [122] 78

6 Polarization **82**
6.1 Information Inequalities, Entropy Conservation and Polarization 82
6.2 Polar Codes Polarize and Achieve Capacity 87
6.3 RM Codes Polarize and Twin-RM Codes Achieve Capacity 93

7 Scaling Law **100**

8 Decoding Algorithms **107**
8.1 Reed's Algorithm [121]: Unique Decoding Up to Half the Code Distance 108
8.2 Decoding Algorithms with Good Practical Performance 110
8.3 Berlekamp-Welch Type Decoding Algorithm [130] 126

9 Applications of RM Codes Beyond Communication **130**
9.1 Low Degree Testing 131
9.2 Private Information Retrieval 134
9.3 Compressed Sensing 136

10 Open Problems **138**
10.1 Capacity-Achieving Results 138
10.2 Weight Enumerator 139
10.3 Algorithms 140

References **142**

Reed-Muller Codes

Emmanuel Abbe[1], Ori Sberlo[2], Amir Shpilka[2] and Min Ye[3*]

[1] *Mathematics Institute and the School of Computer and Communication Sciences, EPFL, Switzerland; emmanuel.abbe@epfl.ch*
[2] *Blavatnik School of Computer Science, Tel Aviv University, Israel; ori.sberlo@gmail.com, shpilka@tauex.tau.ac.il*
[3] *Tsinghua-Berkeley Shenzhen Institute, Tsinghua Shenzhen International Graduate School, China; yeemmi@gmail.com*

ABSTRACT

Reed-Muller (RM) codes are among the oldest, simplest and perhaps most ubiquitous family of codes. They are used in many areas of coding theory in both electrical engineering and computer science. Yet, many of their important properties are still under investigation. This work covers some of the developments regarding the weight enumerator and the capacity-achieving properties of RM codes, as well as some of the algorithmic developments. In particular, it discusses connections established between RM codes, thresholds of Boolean functions, polarization theory, hypercontractivity, and the techniques of approximating low weight codewords using lower degree polynomials (when codewords are viewed as evaluation vectors of degree r polynomials in m variables).

*Corresponding author: Min Ye, yeemmi@gmail.com. O. Sberlo received funding from the ERC (Grant 949499), and from the ISF (Grant Number 952/18). A. Shpilka received funding from the Israel Science Foundation (Grant Number 514/20) and from the Len Blavatnik and the Blavatnik Family foundation. M. Ye received funding from National Key R&D Program of China under Grant No. 2021YFA1001000 and Shenzhen Stable Support program under Grant No. WDZC20220811170401001.

Emmanuel Abbe, Ori Sberlo, Amir Shpilka and Min Ye (2023), "Reed-Muller Codes", Foundations and Trends® in Communications and Information Theory: Vol. 20, No. 1–2, pp 1–156. DOI: 10.1561/0100000123.

It then overviews some of the algorithms for decoding RM codes, giving both algorithms with provable performance guarantees for every block length, as well as algorithms with state-of-the-art performances in practical regimes, which do not perform as well for large block length. Finally, some applications of RM codes in theoretical computer science and signal processing are given.

1

Introduction

A large variety of codes have been developed over the past 70 years. These were driven by various objectives, in particular, achieving efficiently the Shannon capacity [137], constructing perfect or good codes in the Hamming worst-case model [67], matching the performance of random codes, improving the decoding complexity, the weight enumerator, the scaling law, the universality, the local properties of the code [28], [78], [79], [98], [99], [123], [154], and more objectives in theoretical computer science such as in cryptography (e.g., secrete sharing, private information retrieval), pseudorandomness, extractors, hardness amplification or probabilistic proof systems; see [1] for references. Among this large variety of code developments, one of the first, simplest and perhaps most ubiquitous code is the Reed-Muller (RM) code.

The RM code was introduced by Muller in 1954 [109], and Reed developed shortly after a decoding algorithm decoding up to half its minimum distance [121]. The code construction can be described with a greedy procedure. Consider building a linear code (with block length a power of two); it must contain the all-0 codeword. If one has to pick a second codeword, then the all-1 codeword is the best choice under any meaningful criteria. If now one has to keep these two codewords,

the next best choice to maximize the code distance is the half-0 half-1 codeword, and to continue building a basis sequentially, one can add a few more vectors that preserve a relative distance of half, completing the simplex code, which has an optimal rate for the relative distance half. Once saturation is reached at relative distance half, it is less clear how to pick the next codeword, but one can simply re-iterate the simplex construction on any of the support of the previously picked vectors, and iterate this after each saturation, reducing each time the distance by half. This gives the RM code, whose basis is equivalently defined by the evaluation vectors of bounded degree monomials.

As mentioned, the first order RM code is the augmented simplex code or equivalently the Hadamard code, and the simplex code is the dual of the Hamming code that is "perfect". This strong property is clearly lost once the RM code order gets higher, but RM codes preserve nonetheless a decent distance (at root block length for constant rate). Of course this does not give a "good" family of codes (i.e., a family of codes with asymptotically constant rate and constant relative distance), and it is far from achieving the distance that other combinatorial codes can reach, such as Golay codes, BCH codes or expander codes [99]. However, once put under the light of random errors, i.e., the Shannon setting, for which the minimum distance is no longer the right figure or merit, RM codes may perform well again. In [77], Levenshtein and co-authors showed that for the binary symmetric channel, there are codes that improve on the simplex code in terms of the error probability (with matching length and dimension). Nonetheless, in the lens of Shannon capacity, RM codes seem to perform very well. In fact, more than well; it is plausible that they achieve the Shannon capacity on any Binary-input Memoryless Symmetric (BMS) channel [1], [2], [43], [89], [90], [122] and perform comparably to random codes on criteria such as the scaling law [70] or the weight enumerator [82]–[84], [99], [127], [142].

The fact that RM codes have good performance in the Shannon setting, and that they seem to achieve capacity, has long been observed and conjectured. It is hard to track back the first appearance of this belief in the literature, but [89] reports that it was likely already present in the late 60s. The claim was mentioned explicitly in a 1993 talk by Shu Lin, entitled "RM Codes are Not So Bad" [95]. It appears that a 1994

paper by Dumer and Farrell contains the earliest printed discussion on this matter [50]. Since then, the topic has become increasingly prevalent[1] [1], [9], [11], [39], [43], [45], [104].

But the research activity has truly sparked with the emergence of polar codes [11]. Polar codes are close relatives of RM codes. They are derived from the same square matrix (the matrix whose rows correspond to evaluations of multilinear monomials) but with a different rule of row selection. The more sophisticated and channel dependent construction of polar codes gives them the advantage of being provably capacity-achieving on any BMS channel, due to the polarization phenomenon. Even more impressive is the fact that they possess an efficient decoding algorithm down to the capacity.

Shortly after the polar code breakthrough, and given the close relationship between polar and RM codes, the hope that RM codes could also be proved to achieve capacity on any BMS started to propagate, both in the electrical engineering and computer science communities. A first confirmation of this was obtained in extremal regimes of the BEC and BSC [1], exploiting new bounds on the weight enumerator [84], and a first complete proof for the BEC at constant rate was finally obtained in [90]. The paper [122] presented a major breakthrough proving that constant-rate RM codes indeed achieve capacity on all BMS channels under bit-MAP decoding. While [122] comes close to proving the conjecture, the question of whether RM codes achieve capacity under block-MAP decoding still remains open.

The papers mentioned in the previous paragraph however did not exploit the close connection between RM and polar codes. This connection was studied in [2] where it was shown that the RM transform is also polarizing and that a third variant of the RM code achieves capacity on any BMS channel. Furthermore [2] conjectured that this variant is indeed the RM code itself.

Polar codes and RM codes can be compared in different ways. In most performance metrics, and putting aside the decoding complexity, RM codes seem to be superior to polar codes [2], [104]. Namely, they seem

[1]The capacity conjecture for the BEC at constant rate was posed as one of the open problems at the Information Theory Semester at the Simons Institute, Berkeley, in 2015.

to achieve capacity universally and with an optimal scaling-law, while polar codes have a channel-dependent construction with a suboptimal scaling-law [66], [70], [71]. However, RM codes seem more complex both in terms of obtaining performance guarantees (as evidenced by the long standing conjectures) and in terms of their decoding complexity.

Efficient decoding of RM codes is the second main outstanding challenge. Many algorithms have been proposed since Reed's algorithm [121], such as [20], [48], [49], [51], [64], [125], [141], and newer ones have appeared in the post polar code period [129], [130], [153]. Some of these already show that at various block-lengths and rates that are relevant for communication applications, RM codes are indeed competing or even superior to polar codes [104], [153], even compared to the improved versions considered for 5G [57].

This survey is meant to overview these developments regarding both the performance guarantees (in particular on weight enumerator and capacity) and the decoding algorithms for RM codes. At the end of this survey, we discuss a few applications of RM codes in the areas beyond communication, e.g., applications in low degree testing, private information retrieval, and compressed sensing.

1.1 Outline of the Survey and Differences from a Previous Version

Part of this monograph was taken from a previous survey [4] written by the first author, the third author and the fourth author. At the same time, we have added quite a few new elements and optimized the presentation of the contents from [4]. Below we give the outline of this new survey and discuss the difference from [4].

We start in Section 2 with the main definitions and basic properties of RM codes. Most parts of this section already appeared in [4], e.g., the code parameters, recursive structure, duality, automorphism group, and local properties. We have, however, added two new subsections discussing the cyclic property of punctured RM codes and the nonlinear subcodes of RM codes. In Section 3, we introduce some performance measures and important quantities in channel coding. This is a new section that has not appeared in [4]. We then cover the bounds on the weight enumerator of RM codes in Section 4. In Section 5, we cover

the capacity-achieving results, using tools from the weight enumerator and sharp thresholds of monotone Boolean functions. In Section 6, we explore the connection between RM codes and polar codes. Although Sections 4–6 have appeared in [4], we have revised the organization of these 3 sections and added some proofs to better explain the results as well as covered results that appeared between the publication time of these two surveys. Section 7 is a new section that describes the finite-length scaling of random codes, RM codes and polar codes. We then cover various decoding algorithms in Section 8, providing pseudo-codes for them. This section is similar to the previous version [4]. Finally, in Section 9, we discuss some applications of RM codes beyond communication and channel coding, which were not covered in the previous version [4].

2

Basic Properties of RM Codes

2.1 Definition and Parameters

Codewords of binary Reed-Muller codes consist of the evaluation vectors of multivariate polynomials over the binary field $\mathbb{F}_2$. The encoding procedure of RM codes maps the information bits stored in the polynomial coefficients to the polynomial evaluation vector. Consider the polynomial ring $\mathbb{F}_2[x_1, x_2, \ldots, x_m]$ with m variables. For a polynomial $f \in \mathbb{F}_2[x_1, x_2, \ldots, x_m]$ and a binary vector $z = (z_1, z_2, \ldots, z_m) \in \mathbb{F}_2^m$, let $\text{Eval}_z(f) := f(z_1, z_2, \ldots, z_m)$ be the evaluation of f at the vector z, and let $\text{Eval}(f) := (\text{Eval}_z(f)\colon z \in \mathbb{F}_2^m)$ be the evaluation vector of f whose coordinates are the evaluations of f at all 2^m vectors in $\mathbb{F}_2^m$. Reed-Muller codes with parameters m and r consist of all the evaluation vectors of polynomials with m variables and degree no larger than r.

Definition 1. The r-th order (binary) Reed-Muller code $\text{RM}(m, r)$ code is defined as the following set of binary vectors

$$\text{RM}(m, r) := \{\text{Eval}(f)\colon f \in \mathbb{F}_2[x_1, x_2, \ldots, x_m],\ \deg(f) \leq r\}.$$

Note that in later sections, we might use $\text{Eval}(f)$ and f interchangeably to denote the codeword of RM codes. For a subset $A \subseteq [m] :=$

$\{1, 2, \ldots, m\}$, we use the shorthand notation $x_A := \prod_{i \in A} x_i$. Notice that we always have $x^n = x$ in $\mathbb{F}_2$ for any integer $n \geq 1$, so we only need to consider the polynomials in which the degree of each x_i is no larger than 1. All such polynomials with degree no larger than r are linear combinations of the following set of monomials

$$\{x_A \colon A \subseteq [m],\ |A| \leq r\}.$$

There are $\sum_{i=0}^{r} \binom{m}{i}$ such monomials, and the encoding procedure of $\mathrm{RM}(m, r)$ maps the coefficients of these monomials to their corresponding evaluation vectors. Therefore, $\mathrm{RM}(m, r)$ is a linear code with code length $n = 2^m$ and code dimension $\sum_{i=0}^{r} \binom{m}{i}$. Moreover, the evaluation vectors $\{\mathrm{Eval}(x_A) \colon A \subseteq [m], \quad |A| \leq r\}$ form a generator matrix of $\mathrm{RM}(m, r)$. Here we give a few examples of generator matrices for RM codes with code length 8:

$$\mathrm{RM}(3,0)\colon \begin{bmatrix}\mathrm{Eval}(1)\end{bmatrix} = \begin{bmatrix}1&1&1&1&1&1&1&1\end{bmatrix}$$

$$\mathrm{RM}(3,1)\colon \begin{bmatrix}\mathrm{Eval}(x_1)\\ \mathrm{Eval}(x_2)\\ \mathrm{Eval}(x_3)\\ \mathrm{Eval}(1)\end{bmatrix} = \begin{bmatrix}1&1&1&1&0&0&0&0\\ 1&1&0&0&1&1&0&0\\ 1&0&1&0&1&0&1&0\\ 1&1&1&1&1&1&1&1\end{bmatrix}$$

$$\mathrm{RM}(3,2)\colon \begin{bmatrix}\mathrm{Eval}(x_1x_2)\\ \mathrm{Eval}(x_1x_3)\\ \mathrm{Eval}(x_2x_3)\\ \mathrm{Eval}(x_1)\\ \mathrm{Eval}(x_2)\\ \mathrm{Eval}(x_3)\\ \mathrm{Eval}(1)\end{bmatrix} = \begin{bmatrix}1&1&0&0&0&0&0&0\\ 1&0&1&0&0&0&0&0\\ 1&0&0&0&1&0&0&0\\ 1&1&1&1&0&0&0&0\\ 1&1&0&0&1&1&0&0\\ 1&0&1&0&1&0&1&0\\ 1&1&1&1&1&1&1&1\end{bmatrix}$$

$$\mathrm{RM}(3,3)\colon \begin{bmatrix}\mathrm{Eval}(x_1x_2x_3)\\ \mathrm{Eval}(x_1x_2)\\ \mathrm{Eval}(x_1x_3)\\ \mathrm{Eval}(x_2x_3)\\ \mathrm{Eval}(x_1)\\ \mathrm{Eval}(x_2)\\ \mathrm{Eval}(x_3)\\ \mathrm{Eval}(1)\end{bmatrix} = \begin{bmatrix}1&0&0&0&0&0&0&0\\ 1&1&0&0&0&0&0&0\\ 1&0&1&0&0&0&0&0\\ 1&0&0&0&1&0&0&0\\ 1&1&1&1&0&0&0&0\\ 1&1&0&0&1&1&0&0\\ 1&0&1&0&1&0&1&0\\ 1&1&1&1&1&1&1&1\end{bmatrix}$$

From this example, we can see that $\mathrm{RM}(m,0)$ is the repetition code, and $\mathrm{RM}(m,m)$ consists of all the binary vectors of length $n=2^m$, i.e., the evaluation vectors $\{\mathrm{Eval}(x_A)\colon A\subseteq[m]\}$ form a basis of $\mathbb{F}_2^n$.

Another equivalent way of defining RM codes is the Plotkin $(u,u+v)$ construction, which we discuss in detail below in Section 2.2. We also note that RM codes can be defined as geometry codes and refer to [96], [99] for further details.

2.2 Recursive Structure and Distance

For any polynomial $f\in\mathbb{F}_2[x_1,x_2,\ldots,x_m]$, we can always decompose it into two parts, one part containing x_m and the other not containing x_m:

$$f(x_1,x_2,\ldots,x_m)=g(x_1,x_2,\ldots,x_{m-1})+x_m h(x_1,x_2,\ldots,x_{m-1}). \tag{2.1}$$

Here we use the fact that $x_m^n=x_m$ in $\mathbb{F}_2$ for any integer $n\geq 1$.

We can also decompose the evaluation vector $\mathrm{Eval}(f)$ into two subvectors, one subvector consisting of the evaluations of f at all $z=(z_1,\ldots,z_m)$'s with $z_m=0$ and the other subvector consisting of the evaluations of f at all $z=(z_1,\ldots,z_m)$'s with $z_m=1$. We denote the first subvector as $\mathrm{Eval}^{[z_m=0]}(f)$ and the second one as $\mathrm{Eval}^{[z_m=1]}(f)$. We also define their sum over $\mathbb{F}_2$ as $\mathrm{Eval}^{[/z_m]}(f):=\mathrm{Eval}^{[z_m=0]}(f)+\mathrm{Eval}^{[z_m=1]}(f)$. Note that all three vectors $\mathrm{Eval}^{[z_m=0]}(f),\mathrm{Eval}^{[z_m=1]}(f)$ and $\mathrm{Eval}^{[/z_m]}(f)$ have length 2^{m-1}, and their coordinates are indexed by $(z_1,z_2,\ldots,z_{m-1})\in\mathbb{F}_2^{m-1}$.

By (2.1), $\mathrm{Eval}^{[z_m=0]}(f)$ is the evaluation vector of $g(x_1,x_2,\ldots,x_{m-1})$, and $\mathrm{Eval}^{[/z_m]}(f)$ is the evaluation vector of $h(x_1,x_2,\ldots,x_{m-1})$. Now assume that $\mathrm{Eval}(f)\in\mathrm{RM}(m,r)$, or equivalently, assume that $\deg(f)\leq r$. Then we have $\deg(g)\leq r$ and $\deg(h)\leq r-1$. Therefore, $\mathrm{Eval}^{[z_m=0]}(f)\in\mathrm{RM}(m-1,r)$ and $\mathrm{Eval}^{[/z_m]}(f)\in\mathrm{RM}(m-1,r-1)$. This is called the Plotkin $(u,u+v)$ construction of RM codes, meaning that if we take a codeword $c\in\mathrm{RM}(m,r)$, then we can always divide its coordinates into two subvectors u and $u+v$ of length 2^{m-1}, where $u\in\mathrm{RM}(m-1,r)$, $v\in\mathrm{RM}(m-1,r-1)$ and $c=(u,u+v)$.

A consequence of this recursive structure is that the code distance of $\mathrm{RM}(m,r)$ is $d = 2^{m-r}$. We prove this by induction. It is easy to establish the induction basis. For the inductive step, suppose that the claim holds for $m-1$ and all $r \leq m-1$. Then we only need to show that for any $u \in \mathrm{RM}(m-1,r)$ and $v \in \mathrm{RM}(m-1,r-1)$, the Hamming weight of the vector $(u, u+v)$ is at least 2^{m-r} whenever $(u, u+v)$ is not the all-zero vector, i.e., we only need to show that $w(u)+w(u+v) \geq 2^{m-r}$, where $w(\cdot)$ is the Hamming weight of a vector. The proof is divided into two cases: **Case (i)** Suppose that v is the all-zero vector. Then u can not be all zero because otherwise $(u, u+v)$ is the all-zero vector. In this case, we have $w(u)+w(u+v) = 2w(u)$. By the induction hypothesis, $w(u) \geq 2^{m-1-r}$, so $w(u)+w(u+v) = 2w(u) \geq 2^{m-r}$. **Case (ii)** Suppose that v is not the all-zero vector. By the triangle inequality, we have $w(u)+w(u+v) \geq w(v)$. By the inductive hypothesis, $w(v) \geq 2^{m-r}$, so $w(u)+w(u+v) \geq 2^{m-r}$. This completes the proof of the code distance.

The Plotkin construction also implies a recursive relation between the generator matrices of RM codes. More precisely, let $G(m-1,r-1)$ be a generator matrix of $\mathrm{RM}(m-1,r-1)$ and let $G(m-1,r)$ be a generator matrix of $\mathrm{RM}(m-1,r)$. Then we can obtain a generator matrix $G(m,r)$ of $\mathrm{RM}(m,r)$ using the following relation:

$$G(m,r) = \begin{bmatrix} G(m-1,r) & G(m-1,r) \\ 0 & G(m-1,r-1) \end{bmatrix},$$

where 0 denotes the all-zero matrix with the same size as $G(m-1,r-1)$.

2.3 Connection Between RM Codes and Polar Codes

RM codes and polar codes with code length $n = 2^m$ share the same mother matrix. This mother matrix is an $n \times n$ square matrix whose row vectors are the evaluation vectors of all monomials in $\mathbb{F}_2[x_1, x_2, \ldots, x_m]$. We call it mother matrix because the generator matrices of both RM codes and polar codes are submatrices of this matrix. The difference between these two codes lies in how to choose row vectors from the mother matrix to form the generator matrix.

RM codes simply choose the row vectors corresponding to the lowest-degree monomials, or equivalently, the row vectors with the largest

Hamming weights. In contrast, the selection criterion of polar codes is more complicated and depends heavily on the communication channel. In order to explain the polar code construction, let us define

$$G_n := \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^{\otimes m}, \tag{2.2}$$

where $\otimes$ is the Kronecker product and $n = 2^m$. We first show that G_n is in fact the mother matrix, i.e., its row vectors are precisely the evaluation vectors of all monomials in $\mathbb{F}_2[x_1, x_2, \ldots, x_m]$. This claim clearly holds true for $n = 2$, and we will prove it for general values of n by induction. According to the definition, G_n satisfying the following relation

$$G_{2n} := \begin{bmatrix} G_n & \mathbf{0} \\ G_n & G_n \end{bmatrix},$$

where 0 represents the all-zero $n \times n$ matrix. The first n columns of G_{2n} correspond to the evaluation points whose indices satisfy $z_{m+1} = 1$, and the last n columns of G_{2n} correspond to the evaluation points whose indices satisfy $z_{m+1} = 0$. Recall the shorthand notation $x_A := \prod_{i \in A} x_i$ for $A \subseteq [m]$. By the induction hypothesis, the row vectors of G_n consists of the evaluation vectors of all monomials in the set $\{x_A \colon A \subseteq [m]\}$ with evaluation points ranging over the vector space $\mathbb{F}_2^m$. Therefore, the first n rows of G_{2n} are evaluation vectors of all monomials in the set $\{x_{A \cup \{m+1\}} \colon A \subseteq [m]\}$ with evaluation points ranging over the vector space $\mathbb{F}_2^{m+1}$, and the last n rows of G_{2n} are evaluation vectors of all monomials in the set $\{x_A \colon A \subseteq [m]\}$ with evaluation points ranging over the vector space $\mathbb{F}_2^{m+1}$. This establishes the inductive step and proves that G_n is indeed the mother matrix.

Next we describe how to choose rows from G_n to form the generator matrix of polar codes for a communication channel W. Let $U_1, U_2, \ldots, U_n$ be n i.i.d. Bernoulli-1/2 random variables. Define the random vector $(X_1, \ldots, X_n) = (U_1, \ldots, U_n)G_n$. For each $i \in [n]$, let Y_i be the random channel output after transmitting X_i through the channel W. Define the conditional entropy $H_i := H(U_i \mid U^{i-1}, Y^n)$. In the polar code construction, we include the ith row of G_n in the generator matrix if H_i is close to 0. In the seminal paper [11], Arıkan proved that when $n \to \infty$, almost all H_i's are very close to either 0

or 1, implying that polar codes achieve capacity for any binary-input memoryless channel with symmetric outputs. Although the construction of polar codes is not as straightforward as RM codes, efficient methods of calculating or approximating H_i's are well-known; see for example [11], [106]–[108], [113], [145].

A subset of row vectors in the generator matrix of polar codes also appears in the generator matrix of RM codes, and one can in fact calculate the asymptotic size of this subset when the code length n goes to infinity. More specifically, suppose that $\mathsf{C}_{\mathrm{polar}}$ is a polar code with code length n and code rate R constructed for a BMS channel W. As mentioned above, the generator matrix of $\mathsf{C}_{\mathrm{polar}}$ is formed by certain row vectors of G_n. We define a set $\mathcal{G}_{\mathrm{polar}} \subseteq \{1, 2, \ldots, n\}$ as follows: For each $1 \leq i \leq n$, $i \in \mathcal{G}_{\mathrm{polar}}$ if and only if the ith row of G_n is included in the generator matrix of $\mathsf{C}_{\mathrm{polar}}$. Let C_{RM} be a Reed-Muller code with code length n and code rate R'. We also define a set $\mathcal{G}_{\mathrm{RM}} \subseteq \{1, 2, \ldots, n\}$ in a similar way: For each $1 \leq i \leq n$, $i \in \mathcal{G}_{\mathrm{RM}}$ if and only if the ith row of G_n is included in the generator matrix of C_{RM}. By definition, we have $|\mathcal{G}_{\mathrm{polar}}| = nR$ and $|\mathcal{G}_{\mathrm{RM}}| = nR'$. Hassani *et al.* [73, Corollary 6] showed that

$$|\mathcal{G}_{\mathrm{polar}} \cap \mathcal{G}_{\mathrm{RM}}| = nI(W) \cdot \min\left(\frac{R}{I(W)}, R'\right) + o(n).$$

In particular, when $R = R'$, we have

$$|\mathcal{G}_{\mathrm{polar}} \cap \mathcal{G}_{\mathrm{RM}}| = nI(W)R + o(n),$$

indicating that the difference between $\mathcal{G}_{\mathrm{polar}}$ and $\mathcal{G}_{\mathrm{RM}}$ is not negligible whenever $I(W) < 1$.

We will give a more detailed discussion about polar codes and its relations to RM codes in Section 6.

2.4 Duality

The dual code of a binary linear code $\mathsf{C} \subseteq \mathbb{F}_2^n$ is defined as[1]

$$\mathsf{C}^{\perp} := \{x \in \mathbb{F}_2^n \colon \langle x, c\rangle = 0 \quad \forall c \in \mathsf{C}\}, \text{ where } \langle x, c\rangle = \sum_{i=1}^{n} x_i c_i.$$

[1] As we work over $\mathbb{F}_2$ all calculations are done in that field.

By definition, the dual code $\mathsf{C}^\perp$ is also a linear code, and we have

$$\dim(\mathsf{C}) + \dim(\mathsf{C}^\perp) = n. \tag{2.3}$$

Next we will show that the dual code of $\mathrm{RM}(m,r)$ is $\mathrm{RM}(m, m-r-1)$. First, observe that the Hamming weight of every codeword in $\mathrm{RM}(m, m-1)$ is even, i.e., for every $f \in \mathbb{F}_2[x_1, \ldots, x_m]$ with $\deg(f) \leq m-1$, we have $\sum_{z\in\mathbb{F}_2^m} \mathrm{Eval}_z(f) = 0$, where the summation is over $\mathbb{F}_2$. This is because the Hamming weight of $\mathrm{Eval}(x_A)$ is $2^{m-|A|}$, so $\sum_z \mathrm{Eval}_{z\in\mathbb{F}_2^m}(x_A) = 0$ for all subsets A with size $|A| \leq m-1$. For every $f \in \mathbb{F}_2[x_1, \ldots, x_m]$ with $\deg(f) \leq m-1$, we can write it as $f = \sum_{A\subset[m],|A|\leq m-1} u_A x_A$. Therefore,

$$\begin{aligned}\sum_{z\in\mathbb{F}_2^m} \mathrm{Eval}_z(f) &= \sum_{z\in\mathbb{F}_2^m} \mathrm{Eval}_z\left(\sum_{A\subset[m],|A|\leq m-1} u_A x_A\right)\\ &= \sum_{A\subset[m],|A|\leq m-1} \left(u_A \sum_{z\in\mathbb{F}_2^m} \mathrm{Eval}_z(x_A)\right) = 0.\end{aligned}$$

Suppose that $\mathrm{Eval}(f)$ is a codeword of $\mathrm{RM}(m,r)$ and $\mathrm{Eval}(g)$ is a codeword of $\mathrm{RM}(m, m-r-1)$. Then $\deg(f) \leq r$ and $\deg(g) \leq m-r-1$. Notice that $\langle \mathrm{Eval}(f), \mathrm{Eval}(g)\rangle = \sum_{z\in\mathbb{F}_2^m} \mathrm{Eval}_z(f)\,\mathrm{Eval}_z(g) = \sum_{z\in\mathbb{F}_2^m} \mathrm{Eval}_z(fg)$. Since $\deg(fg) \leq m-1$, we have $\langle \mathrm{Eval}(f), \mathrm{Eval}(g)\rangle = \sum_{z\in\mathbb{F}_2^m} \mathrm{Eval}_z(fg) = 0$. Therefore, every codeword of $\mathrm{RM}(m, m-r-1)$ belongs to the dual code of $\mathrm{RM}(m,r)$, i.e., $\mathrm{RM}(m, m-r-1) \subseteq \mathrm{RM}(m,r)^\perp$.

Since

$$\dim(\mathrm{RM}(m, m-r-1)) = \sum_{i=0}^{m-r-1} \binom{m}{i} = \sum_{i=0}^{m-r-1} \binom{m}{m-i} = \sum_{i=r+1}^{m} \binom{m}{i},$$

we have

$$\begin{aligned}\dim(\mathrm{RM}(m,r)) + \dim(\mathrm{RM}(m, m-r-1)) &= \sum_{i=0}^{r} \binom{m}{i} + \sum_{i=r+1}^{m} \binom{m}{i}\\ &= \sum_{i=0}^{m} \binom{m}{i} = 2^m = n.\end{aligned}$$

Combining this with (2.3), we know that $\dim(\mathrm{RM}(m, m-r-1)) = \dim(\mathrm{RM}(m,r)^\perp)$. Thus we conclude that

$$\mathrm{RM}(m, m-r-1) = \mathrm{RM}(m,r)^\perp.$$

Table 2.1: Important parameters of $\mathrm{RM}(m, r)$

Code	Code Length	Code Dimension	Code Distance	Dual Code
$\mathrm{RM}(m, r)$	$n = 2^m$	$k = \sum_{i=0}^{r} \binom{m}{i}$	$d = 2^{m-r}$	$\mathrm{RM}(m, m-r-1)$

This in particular tells us that the parity check matrix of $\mathrm{RM}(m, r)$ is the generator matrix of $\mathrm{RM}(m, m-r-1)$. The important parameters of $\mathrm{RM}(m, r)$ are summarized in Table 2.1.

2.5 Affine-Invariance

The automorphism group of a code C is the set of permutations under which C remains invariant. More precisely, the automorphism group of a code C with code length n is defined as $\mathcal{A}(\mathsf{C}) := \{\pi \in S_n \colon \pi(\mathsf{C}) = \mathsf{C}\}$, where $\pi(\mathsf{C}) := \{\pi(c) \colon c \in \mathsf{C}\}$, and $\pi(c)$ is vector obtained from permuting the coordinates of c according to π. It is easy to verify that $\mathcal{A}(\mathsf{C})$ is always a subgroup of the symmetric group S_n.

RM codes are affine-invariant in the sense that $\mathcal{A}(\mathrm{RM}(m, r))$ contains a subgroup isomorphic to the affine linear group. More specifically, since the codewords of RM codes are evaluation vectors and they are indexed by the vectors $z \in \mathbb{F}_2^m$, the affine linear transform $g_{A,b} \colon z \mapsto Az + b$ gives a permutation on the coordinates of the codeword when A is an $m \times m$ invertible matrix over $\mathbb{F}_2$ and $b \in \mathbb{F}_2^m$. Next we show that such a permutation indeed belongs to $\mathcal{A}(\mathrm{RM}(m, r))$. For any codeword $c \in \mathrm{RM}(m, r)$, there is a polynomial $f \in \mathbb{F}_2[x_1, \ldots, x_m]$ with $\deg(f) \leq r$ such that $c = \mathrm{Eval}(f)$. Since $g_{A,b}(c) = \mathrm{Eval}(f \circ g_{A,b})$ and $\deg(f \circ g_{A,b}) = \deg(f) \leq r$, we have $g_{A,b}(c) \in \mathrm{RM}(m, r)$. Therefore, $g_{A,b} \in \mathcal{A}(\mathrm{RM}(m, r))$, and RM codes are affine-invariant.

Recall that in Section 2.2 we showed that $\mathrm{Eval}^{[/z_m]}(f) \in \mathrm{RM}(m-1, r-1)$ if $\mathrm{Eval}(f) \in \mathrm{RM}(m, r)$. Using the affine-invariant property, we can replace z_m in this statement with any linear combination of $z_1, \ldots, z_m$. More specifically, for any $\ell = b_1 z_1 + \cdots + b_m z_m$ with nonzero coefficient vector $b = (b_1, \ldots, b_m) \neq 0$, we define $\mathrm{Eval}^{[\ell=0]}(f)$, $\mathrm{Eval}^{[\ell=1]}(f), \mathrm{Eval}^{[/\ell]}(f)$ in the same way as $\mathrm{Eval}^{[z_m=0]}(f), \mathrm{Eval}^{[z_m=1]}(f)$,

$\mathrm{Eval}^{[/z_m]}(f)$. For any such ℓ, one can always find an affine linear transform mapping z_m to ℓ. Since RM codes are invariant under such affine transforms, we have $\mathrm{Eval}^{[/\ell]}(f) \in \mathrm{RM}(m-1, r-1)$ whenever $\mathrm{Eval}(f) \in \mathrm{RM}(m, r)$. This observation will be used in several decoding algorithms in Section 8.

2.6 Punctured RM Codes are Cyclic Codes

A cyclic code C is a linear code satisfying the following property: If $(C_1, C_2, \ldots, C_n) \in \mathsf{C}$ is a codeword, then its cyclic shift $(C_n, C_1, C_2, \ldots, C_{n-1})$ is also a codeword in C.

Reed-Muller codes themselves are not cyclic codes. However, if we remove *any* coordinate of a Reed-Muller code with length $n = 2^m$ and rearrange the remaining $(n-1)$ coordinates in a specific order, then the resulting code is a cyclic code with length $n-1$. The procedure of removing one coordinate of a code is called puncturing, and the code obtained from puncturing a RM code is called a punctured RM code. As a concrete example, let us consider the code $\mathrm{RM}(m, r)$. Every codeword in $\mathrm{RM}(m, r)$ is an evaluation vector $\mathrm{Eval}(f) = (\mathrm{Eval}_z(f)\colon z \in \mathbb{F}_2^m)$ for some polynomial f with degree $\deg(f) \le r$. Suppose that we puncture the coordinate with the all-zero index $z = \mathbf{0}$. Then the punctured codeword is $(\mathrm{Eval}_z(f)\colon z \in \mathbb{F}_2^m \setminus \{\mathbf{0}\})$ and the punctured $\mathrm{RM}(m, r)$ code is

$$\{(\mathrm{Eval}_z(f)\colon z \in \mathbb{F}_2^m \setminus \{\mathbf{0}\})\colon f \in \mathbb{F}_2[x_1, x_2, \ldots, x_m],\ \deg(f) \le r\}.$$

Next we prove that if we arrange the coordinates of the punctured codewords $(\mathrm{Eval}_z(f)\colon z \in \mathbb{F}_2^m \setminus \{\mathbf{0}\})$ in a specific order, then the punctured RM codes are cyclic codes. We start with the simplest case of $r = 1$, i.e., the first-order punctured RM codes. A generator matrix of punctured $\mathrm{RM}(m, 1)$ is

$$G = \begin{bmatrix} (\mathrm{Eval}_z(x_1)\colon z \in \mathbb{F}_2^m \setminus \{\mathbf{0}\}) \\ \vdots \\ (\mathrm{Eval}_z(x_m)\colon z \in \mathbb{F}_2^m \setminus \{\mathbf{0}\}) \\ (\mathrm{Eval}_z(\mathbf{1})\colon z \in \mathbb{F}_2^m \setminus \{\mathbf{0}\}) \end{bmatrix}.$$

The first m rows of G are evaluation vectors of degree-1 monomials, and the last row is the all-one vector. Consider the matrix $\widetilde{G}$ consisting of the first m rows of G. The $(2^m - 1)$ columns of $\widetilde{G}$ are all distinct from each other, and these column vectors are precisely all the nonzero vectors in $\mathbb{F}_2^m$. On the other hand, it is well known that Hamming code with length $(2^m - 1)$ has a parity check matrix consisting of all the nonzero column vectors in $\mathbb{F}_2^m$. Therefore, $\widetilde{G}$ is in fact the generator matrix of the simplex code, the dual code of Hamming code [99, Chapter 1.9]. Both Hamming codes and simplex codes are cyclic codes [99, Theorem 2 and Theorem 4 of Chapter 7], so the row space of $\widetilde{G}$ is invariant under cyclic shifts. Since the generator matrix G is obtained by appending the all-one vector to $\widetilde{G}$, we conclude that the punctured $\mathrm{RM}(m, 1)$ code is indeed a cyclic code.

Now we move on to prove that the punctured $\mathrm{RM}(m, r)$ code is a cyclic code for general values of r. Consider a generator matrix $G(m, r)$ of $\mathrm{RM}(m, r)$, consisting of the evaluation vectors of all monomials with degree no larger than r. Note that the evaluation vector of a monomial with degree s is the coordinate-wise product of s distinct rows of $\widetilde{G}$. Therefore, the order of columns in $\widetilde{G}$ completely determines the order of columns in the generator matrix $G(m, r)$. As a consequence, the cyclic invariance of the row space of $\widetilde{G}$ implies that the row space of $G(m, r)$ is also invariant under cyclic shifts, i.e., the punctured $\mathrm{RM}(m, r)$ code is a cyclic code.

We have shown that by puncturing the coordinate with the all-zero index $z = \mathbf{0}$ from a RM code, we obtain a cyclic code. The affine-invariant property of RM codes then immediately implies that a cyclic code can be obtained by puncturing *any* coordinate of a RM code.

The above arguments are based on the classic paper [81]. As a final remark, cyclic codes are usually defined by their generator polynomials and/or check polynomials. A detailed discussion on generator polynomials and check polynomials of punctured RM codes (or general cyclic codes) is beyond the scope of this monograph. Interested readers may consult Chapter 13.5 of [99] for results on this subject.

2.7 Nonlinear Subcodes of RM Codes: Nordstrom-Robinson Code and Kerdock Codes

In this section we briefly review some nonlinear subcodes of second-order RM codes, including the Nordstrom-Robinson code [111] and the Kerdock codes [86]. These nonlinear codes are important because they contain more codewords than any known linear codes with the same code length and the same code distance.

A Kerdock code has a parameter $m \geq 4$, which is required to be an even number, and the code is denoted as $\mathcal{K}(m)$. The code length of $\mathcal{K}(m)$ is $n = 2^m$; the number of codewords in $\mathcal{K}(m)$ is $2^{2m} = n^2$; the code distance of $\mathcal{K}(m)$ is $d = 2^{m-1} - 2^{m/2-1} = \frac{1}{2}(n - \sqrt{n})$. The Nordstrom-Robinson code is simply $\mathcal{K}(4)$.

Kerdock codes contain all the codewords from the first-order RM codes, and all codewords of Kerdock codes are contained in the second-order RM codes. In other words, we have $\mathrm{RM}(m,1) \subseteq \mathcal{K}(m) \subseteq \mathrm{RM}(m,2)$. In addition to $\mathrm{RM}(m,1)$ itself, $\mathcal{K}(m)$ contains $2^{m-1} - 1$ cosets of $\mathrm{RM}(m,1)$ in $\mathrm{RM}(m,2)$. Since the number of codewords in $\mathrm{RM}(m,1)$ is 2^{m+1}, the number of codewords in $\mathcal{K}(m)$ is $2^{m-1} \cdot 2^{m+1} = 2^{2m}$. To better describe Kerdock codes, we associate each quadratic function in $\mathbb{F}_2[x_1, x_2, \ldots, x_m]$ with a coset of $\mathrm{RM}(m,1)$: For each $f \in \mathbb{F}_2[x_1, x_2, \ldots, x_m]$ with $\deg(f) = 2$, define its corresponding coset as

$$\mathrm{Coset}(f) = \{\mathrm{Eval}(f+g)\colon g \in \mathbb{F}_2[x_1, x_2, \ldots, x_m], \deg(g) \leq 1\}.$$

Let us consider a special choice of the quadratic function f. When m is even, let $f^* = x_1x_2 + x_3x_4 + x_5x_6 + \cdots + x_{2m-1}x_{2m}$. For this particular choice, one can show that the Hamming weights of all the codewords in $\mathrm{Coset}(f^*)$ can only take two values, either $\frac{1}{2}(n - \sqrt{n})$ or $\frac{1}{2}(n + \sqrt{n})$. Moreover, the number of codewords with Hamming weight $\frac{1}{2}(n - \sqrt{n})$ is the same as the number of codewords with Hamming weight $\frac{1}{2}(n + \sqrt{n})$, both equal to 2^m. The function f^* is not the only quadratic function whose corresponding coset has this particularly nice weight distribution. For example, we can obtain another quadratic function with this property by simply performing a permutation on $x_1, x_2, \ldots, x_m$ in the definition of f^*. Denote $f_0 = 0$ as the zero polynomial. The construction of Kerdock codes requires finding $2^{m-1} - 1$

distinct quadratic functions $f_1, f_2, \ldots, f_{2^{m-1}-1}$ such that the weight distribution of Coset$(f_i - f_j)$ is the same as the weight distribution of Coset(f^*) for all $0 \leq i < j \leq 2^{m-1} - 1$. We will not discuss how to find such quadratic functions since it is complicated and beyond the scope of this monograph. Here we only record one consequence of the weight distribution of these cosets. That is, we can completely characterize the weight distribution of $\mathcal{K}(m)$. Indeed, RM$(m, 1)$ has 1 codeword with weight 0, 1 codeword with weight n and $2^{m+1} - 2$ codewords with weight $n/2$. For every $1 \leq i \leq 2^{m-1} - 1$, the coset Coset$(f_i)$ has 2^m codewords with weight $\frac{1}{2}(n - \sqrt{n})$ and 2^m codewords with weight $\frac{1}{2}(n + \sqrt{n})$. Therefore, the weight distribution of $\mathcal{K}(m)$ is as follows: It has 1 codeword with weight 0, 1 codeword with weight n, $2^{m+1} - 2$ codewords with weight $n/2$, $2^m(2^{m-1} - 1)$ codewords with weight $\frac{1}{2}(n - \sqrt{n})$ and $2^m(2^{m-1} - 1)$ codewords with weight $\frac{1}{2}(n + \sqrt{n})$.

Delsarte and Goethals extended the construction of Kerdock codes to obtain a new family of codes called Delsarte-Goethals codes [44]. This family of codes is also nonlinear, and it sits between Kerdock codes and second-order RM codes, i.e., it is a subcode of second-order RM codes and it contains all codewords of Kerdock codes.

Finally, we note that although Nordstrom-Robinson code, Kerdock codes and Delsarte-Goethals codes are all nonlinear codes over the binary field, they can all be constructed as binary images under the Gray map of linear codes over $\mathbb{Z}_4$, the integer ring mod 4 [68]. We first use a simple example to explain how to map a $\mathbb{Z}_4$-linear code to a binary code using the Gray mapping. Consider the following 4×8 matrix over $\mathbb{Z}_4$:

$$\begin{bmatrix} 2 & 2 & 2 & 2 & 0 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 & 2 & 2 & 0 & 0 \\ 2 & 0 & 2 & 0 & 2 & 0 & 2 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}. \tag{2.4}$$

The row space of this matrix contains the following 32 linear combinations over $\mathbb{Z}_4$:

$$\{a_0(1,1,1,1,1,1,1,1) + a_1(1,1,1,1,0,0,0,0) + a_2(1,1,0,0,1,1,0,0) + a_3(1,0,1,0,1,0,1,0)\colon a_0 \in \{0,1,2,3\},\ a_1, a_2, a_3 \in \{0,2\}\}. \tag{2.5}$$

These 32 vectors form a $\mathbb{Z}_4$-linear code with code length 8. The Gray mapping ϕ maps a symbol in $\mathbb{Z}_4$ to two symbols in $\mathbb{F}_2$ as follows: $\phi(0) = (0,0), \phi(1) = (0,1), \phi(2) = (1,1), \phi(3) = (1,0)$. The mapping ϕ naturally induces a sequence of mappings ϕ_n that maps from a $\mathbb{Z}_4$ vector with length n to an $\mathbb{F}_2$ vector with length $2n$ as follows: $\phi_n((a_1, a_2, \ldots, a_n)) = (\phi(a_1), \phi(a_2), \ldots, \phi(a_n))$, where $a_1, \ldots, a_n \in \mathbb{Z}_4$. As a concrete example, $\phi_8((3,3,1,1,2,2,0,0)) = (1,0,1,0,0,1,0,1,1,1,1,1,0,0,0,0)$. If we apply the mapping ϕ_8 to each of the 32 vectors in (2.5), then we obtain a binary code with code length 16 that contains 32 codewords. It is easy to check that this binary code is the first-order RM code of length 16. In other words, RM$(4,1)$ is the binary image of the $\mathbb{Z}_4$-linear code with generator matrix (2.4) under the Gray map. The remarkable results of [68] show that all the nonlinear codes mentioned in this section can also be constructed as binary images of $\mathbb{Z}_4$-linear codes under the Gray map. For example, the Nordstrom-Robinson code is the binary image of the $\mathbb{Z}_4$-linear code with generator matrix

$$\begin{bmatrix} 1 & 1 & 3 & 1 & 0 & 0 & 0 & 2 \\ 1 & 3 & 2 & 2 & 1 & 1 & 0 & 2 \\ 1 & 0 & 3 & 2 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

If we multiply the first three rows of this matrix by 2 (mod 4), then we obtain the matrix (2.4), indicating that the Nordstrom-Robinson code contains all the codewords in RM$(4,1)$, as we already mentioned earlier. Readers who want to further explore these codes may consult [97] and Chapter 15 of [99], among many other references.

2.8 General Finite Fields and Locality

The definition of binary RM codes above can be naturally extended to more general finite fields $\mathbb{F}_q$. Let us consider the polynomial ring $\mathbb{F}_q[x_1, x_2, \ldots, x_m]$ of m variables. For a polynomial $f \in \mathbb{F}_q[x_1, x_2, \ldots, x_m]$, we again use $\mathrm{Eval}(f) := (\mathrm{Eval}_z(f)\colon z \in \mathbb{F}_q^m)$ to denote the evaluation vector of f. Since $x^q = x$ in $\mathbb{F}_q$, we only need to consider the polynomials in which the degree of each x_i is no larger than $q-1$, and the degree of such polynomials is no larger than $m(q-1)$.

Definition 2. Let $n := q^m$ and $r \leq m(q-1)$. The r-th order q-ary Reed-Muller code $\mathrm{RM}_q(m,r)$ code is defined as the following set of vectors in $\mathbb{F}_q^n$:

$$\mathrm{RM}_q(m,r) := \{\mathrm{Eval}(f)\colon f \in \mathbb{F}_q[x_1, x_2, \ldots, x_m], \deg(f) \leq r\}.$$

A locally decodable code (LDC) is an error-correcting code that allows a single bit of the original message to be decoded with high probability by only examining (or querying) a small number of bits of a possibly corrupted codeword. RM codes over large finite fields are the oldest and most basic family of LDC. When RM codes are used as LDC, the order r of RM codes is typically set to be smaller than the field size q. At a high level, local decoding of RM codes requires us to efficiently correct the evaluation of a multivariate polynomial at a given point z from the evaluation of the same polynomial at a small number of other points. The decoding algorithm chooses a set of points on an affine line that passes through z. It then queries the codeword for the evaluation of the polynomial on the points in this set and interpolates that polynomial to obtain the evaluation at z. We refer the readers to [154] for more details on this topic.

3

Performance Measures and Important Quantities in Channel Coding

3.1 Information Measures of a Channel

We consider a discrete memoryless channel (DMC) W: $\mathcal{X} \to \mathcal{Y}$ with input alphabet $\mathcal{X}$ and output alphabet $\mathcal{Y}$. The transition probability $W(y|x)$ is the conditional probability of obtaining the channel output $y \in \mathcal{Y}$ when the channel input is $x \in \mathcal{X}$.

Let $\mathbb{P}_X$ be a probability distribution on $\mathcal{X}$ and let X, Y be distributed according to $\mathbb{P}(X = x, Y = y) = \mathbb{P}_X(x)W(y|x)$, for all $x \in \mathcal{X}, y \in \mathcal{Y}$. The entropy of X, the conditional entropy of Y given $X = x$, and the conditional entropy of Y given X are defined as

$$\begin{aligned} H(X) &= -\sum_{x \in \mathcal{X}} \mathbb{P}_X(x) \log_2 \mathbb{P}_X(x) \\ H(Y|X = x) &= -\sum_{y \in \mathcal{Y}} W(y|x) \log_2 W(y|x) \\ H(Y|X) &= \sum_{x \in \mathcal{X}} H(Y|X = x)\mathbb{P}_X(x). \end{aligned}$$

Note that for a channel W and an input distribution $\mathbb{P}_X$ we have the induced "reverse" channel which we denote by $\widetilde{W}(y|x) = W(y|x)\mathbb{P}_X(x)/\mathbb{P}_Y(y)$ where $\mathbb{P}_Y(y) = \sum_{x \in \mathcal{X}} W(y|x)\mathbb{P}_X(x)$.

The mutual information of the channel W for the input distribution $\mathbb{P}_X$ is defined by

$$I(\mathbb{P}_X, W) = I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X).$$

The channel capacity of W is $I(W) := \sup_{\mathbb{P}_X} I(\mathbb{P}_X, W)$ and the symmetric capacity of W is defined as $I(\mathbb{U}_X, Y)$ where $\mathbb{U}_X$ is the uniform distribution on $\mathcal{X}$.

In this monograph, we focus on binary-input memoryless symmetric (BMS) channels, i.e., channels with input alphabet $\mathcal{X} = \{0,1\}$ that satisfy the following property: There is a permutation π on the output alphabet $\mathcal{Y}$ satisfying (i) $\pi^{-1} = \pi$ and (ii) $W(y|1) = W(\pi(y)|0)$ for all $y \in \mathcal{Y}$. The binary erasure channel (BEC), the binary symmetric channel (BSC), and the binary input additive white Gaussian noise channel (BIAWGN) are all BMS channels. An important feature of BMS channels is that the mutual information $I(X;Y)$ is maximized when the input random variable X takes the Bernoulli-1/2 distribution. Therefore, for a BMS channel W, we have $I(W) = 1 - H(X|Y)$, where $H(X|Y)$ is the conditional entropy.

We define the Bhattacharyya parameter of a BMS channel W as

$$\begin{aligned} Z(W) = Z(X|Y) &:= \sum_{y\in\mathcal{Y}} \sqrt{W(y|0)W(y|1)} \\ &= 2\sum_{y\in\mathcal{Y}} \sqrt{\mathbb{P}_{X,Y}(0,y)\mathbb{P}_{X,Y}(1,y)}, \end{aligned} \tag{3.1}$$

where $\mathbb{P}_{X,Y}(0,y) = \frac{1}{2}W(y|0)$ and $\mathbb{P}_{X,Y}(1,y) = \frac{1}{2}W(y|1)$, i.e., $\mathbb{P}_{X,Y}$ is joint distribution of X and Y when we assume X is a Bernoulli-1/2 random variable and Y is the corresponding channel output. The following relation holds for $H(X|Y)$ and $Z(X|Y)$:

$$(Z(X|Y))^2 \le H(X|Y) \le \log(1 + Z(X|Y)). \tag{3.2}$$

This inequality was proved in [10, Proposition 2]. Since X only takes two possible values 0 and 1, we have $0 \le H(X|Y) \le 1$ and $0 \le Z(X|Y) \le 1$. Both $H(X|Y)$ and $Z(X|Y)$ measure the uncertainty of X given Y. Inequality (3.2) implies that these two uncertainty measures are highly correlated in the following sense: (i) $Z(X|Y) \approx 0$ if and only if $H(X|Y) \approx 0$, (ii) $Z(X|Y) \approx 1$ if and only if $H(X|Y) \approx 1$.

Finally, we define the error probability when guessing a random variable X from the marginal distribution or from the conditional distribution given Y as

$$err(X) = 1 - \max_{x \in \mathcal{X}} \mathbb{P}_X(x) \leq H(X),$$
$$err(X|Y) = \sum_{y \in \mathcal{Y}} \left(1 - \max_{x \in \mathcal{X}} \mathbb{P}_{X|Y}(x|y)\right) \mathbb{P}_Y(y) \leq H(X|Y).$$

These two inequalities are proved in Section 6.1.

3.2 Error Probabilities of a Code Over a Channel

In the previous section, we considered information measures on a "one-shot" or "single-letter" channel. In particular, we considered the minimal error probability of guessing the input on a such channel, without any coding involved. We now consider codes, the induced channel for a given blocklength, and the bit and block error probabilities.

Consider a binary code $\mathsf{C} \subseteq \mathbb{F}_2^n$ of block length n and a BMS channel W. This channel has a memoryless extension to vectors of blocklength n which we denote as W^n, i.e., $W^n(y|c) = \prod_{i=1}^n W(y_i|c_i)$ for $c \in \mathbb{F}_2^n$ and $y \in \mathcal{Y}^n$. Then the block error probability of the code C over the channel W under the maximum a posteriori (MAP) decoding is defined as the probability that a randomly chosen codeword in C is not correctly decoded by the decoder that minimizes the block error probability, i.e., the decoder that declares the most likely codeword after observing the output Y of the channel. In this monograph, we use ML (maximum likelihood) decoding and MAP decoding interchangeably because we always assume that the codeword is uniformly chosen from the codebook. Formally speaking, we denote the block error probability of a code C over a channel W under the MAP decoding by

$$err(W, \mathsf{C}) = \mathbb{E}_{C \sim \mathbb{U}_\mathsf{C}} \, \mathbb{P}(C \neq \hat{C}^{\mathrm{MAP}})$$

where $\hat{C}^{\mathrm{MAP}} = \arg\max_{c \in \mathsf{C}} W^n(Y|c), Y \sim W^n(\cdot|C)$, and $\mathbb{U}_\mathsf{C}$ is the uniform distribution over the codebook C. In particular, for $\mathrm{RM}(m, r)$ codes over $\mathrm{BEC}(p)$ or $\mathrm{BSC}(p)$ channels, we denote this block error probability as $err(\mathrm{BEC}(p), \mathrm{RM}(m, r))$ or $err(\mathrm{BSC}(p), \mathrm{RM}(m, r))$.

Given a code C of code length n and a channel W, we now define the bit-MAP decoding

$$\hat{x_i}^{\mathrm{MAP}}(y) = \mathrm{argmax}_{x_i \in \{0,1\}} \mathbb{P}_{X_i|Y}(x_i|y), \tag{3.3}$$

where the random vector X is chosen uniformly from C, and the random vector Y is the corresponding channel output distributed according to $W^n(\cdot|X)$. The tie is broken arbitrarily if $\mathbb{P}_{X_i|Y}(0|y) = \mathbb{P}_{X_i|Y}(1|y)$. The bit-MAP decoding error probability is then given by

$$biterr(W, \mathsf{C}) = \frac{1}{n}\mathbb{E}_{X,Y}\left(\sum_{i=1}^{n} \mathbb{P}(\hat{x_i}^{\mathrm{MAP}}(Y) \neq X_i)\right). \tag{3.4}$$

3.3 Weight Enumerator of a Code

An important parameter of error correcting codes, closely related to the error probability of the ML decoder, is the weight enumerator of a code. Here we give the definition of the weight enumerator for RM codes. Below we will use f and $\mathrm{Eval}(f)$ interchangeably, i.e., we use f to denote both the polynomial and its evaluation vector.

Definition 3. We denote with $\mathcal{P}_{m,r}$ the set of all polynomials $f \in \mathbb{F}_2[x_1, \ldots, x_m]$ of degree at most r.

With this definition we have $\mathrm{RM}(m,r) = \{\mathrm{Eval}(f) \mid f \in \mathcal{P}_{m,r}\}$.

Definition 4 (Support, Weight, Bias)**.** We denote with $\mathrm{supp}(f)$ the set of nonzero coordinates of a codeword $\mathrm{Eval}(f) \in \mathrm{RM}(m,r)$. Namely, $\mathrm{supp}(f) = \{z \in \mathbb{F}_2^m \mid f(z) \neq 0\}$. We denote its hamming weight by $\mathrm{wt}(f) = |\{z \in \mathbb{F}_2^m \mid f(z) = 1\}| = |\mathrm{supp}(f)|$, and its relative-weight, with $\mathrm{wt_n}(f) = \mathrm{wt}(f)/2^m = \mathrm{wt}(f)/n$. When a codeword is of weight close to $1/2$ it will sometimes be more convenient to speak of its bias: $\mathrm{bias}(f) = \left|\mathbb{E}_Z[(-1)^{f(Z)}]\right| = |1 - 2\mathrm{wt_n}(f)|$, where Z is a random vector uniformly distributed in $\mathbb{F}_2^m$.

Definition 5 (Weight Enumerator)**.** We let $A_{m,r}(\cdot)$ denote the relative-weight enumerator function of $\mathrm{RM}(m,r)$. Specifically, for $\beta \in [0,1]$ we define $A_{m,r}(\beta) \triangleq |\{f \in \mathcal{P}_{r,m} \mid \mathrm{wt_n}(f) = \beta\}|$. We also abuse notation and denote $A_{m,r}(\leq \beta) \triangleq |\{f \in \mathcal{P}_{r,m} \mid \mathrm{wt_n}(f) \leq \beta\}|$.

In general, for a code $\mathsf{C} \subseteq \mathbb{F}_2^n$ which is not necessarily a Reed-Muller code, we define $A_{\mathsf{C}}(\beta) \triangleq |\{c \in \mathsf{C} \mid \mathrm{wt_n}(c) = \beta\}|$ and $A_{\mathsf{C}}(\leq \beta) \triangleq |\{c \in \mathsf{C} \mid \mathrm{wt_n}(c) \leq \beta\}|$ for $\beta \in [0,1]$, where $\mathrm{wt_n}(c) = \mathrm{wt}(c)/n$ is the relative-weight of the codeword c, and $\mathrm{wt}(c)$ is the Hamming weight of c.

Besides being a very natural parameter of a code, the weight enumerator plays an important role in understanding the amount of random or worst-case errors (or erasures) that the code can be decoded from. For example, if the minimum relative-weight of a nonzero codeword is δ, then for every $0 < \beta < \delta$ there are no codewords of weight β, so the code can be decoded from δ fraction of worst-case erasures and $\delta/2$ fraction of worst-case errors. In addition, the Bhattacharyya parameter (see Theorem 2) bounds the error probability of the ML decoder for decoding from random errors or erasures, using the weight enumerator.

In the remainder of this subsection, we show that if the weight enumerator of a linear code is similar to that of a random code, of the same rate, then the code achieves capacity on BMS channels, including, among others, the BEC and the BSC.

Definition 6 (Random Code Ensemble)**.** A binary equiprobable random code ensemble of length n and rate R consists of 2^{nR} random codewords $\{c^{(i)} = (c_1^{(i)}, c_2^{(i)}, \ldots, c_n^{(i)}) \colon 1 \leq i \leq 2^{nR}\}$, where the $n \cdot 2^{nR}$ binary random variables $\{c_j^{(i)} \colon 1 \leq i \leq 2^{nR}, 1 \leq j \leq n\}$ are i.i.d. Bernoulli-1/2 random variables.

By definition, we have

$$\mathbb{E}_{\mathsf{C} \sim \mathrm{RCE}(n,R)}[A_{\mathsf{C}}(\beta)] = 2^{nR} \cdot \binom{n}{n\beta} / 2^n, \tag{3.5}$$

where $\mathsf{C} \sim \mathrm{RCE}(n,R)$ means that the code C is randomly chosen from the random code ensemble.

Let us first prove that if $\mathsf{C} \sim \mathrm{RCE}(n,R)$, then C achieves the capacity of BEC. Indeed, suppose that the erasure probability of BEC is p, and its capacity is $1-p$. Let $\epsilon > 0$ be an arbitrarily small positive number, and let $R = 1 - p - \epsilon$. We use $\mathcal{E} \subseteq \{1, 2, \ldots, n\}$ to represent the set of erased coordinates. Then the number of erasures $|\mathcal{E}|$ satisfies $|\mathcal{E}| \leq n(p + \epsilon/2)$ with probability $1 - o(1)$. Without loss of generality,

we assume that the transmitted codeword is $c^{(1)}$. We define a set $\mathcal{D}(\mathcal{E})$ as

$$\mathcal{D}(\mathcal{E}) = \{(x_1, \ldots, x_n) \in \{0,1\}^n \colon x_j = c_j^{(1)} \text{ for all } j \notin \mathcal{E}\}.$$

It is clear that we can successfully decode if and only if there are no other codewords (apart from $c^{(1)}$) in the set $\mathcal{D}(\mathcal{E})$, i.e., $(\mathsf{C} \setminus \{c^{(1)}\}) \cap \mathcal{D}(\mathcal{E}) = \emptyset$. Since the random codeword $c^{(i)}$ is independent of $c^{(1)}$ and $\mathcal{E}$ for every $i > 1$, we have $\mathbb{P}(c^{(i)} \in \mathcal{D}(\mathcal{E})) = |\mathcal{D}(\mathcal{E})|/2^n = 2^{|\mathcal{E}|-n}$. By union bound,

$$\begin{aligned}\mathbb{P}(\exists i > 1 \text{ such that } c^{(i)} \in \mathcal{D}(\mathcal{E})) &\le (2^{nR} - 1)2^{|\mathcal{E}|-n} < 2^{|\mathcal{E}|+nR-n} \\ &= 2^{|\mathcal{E}|-np-n\epsilon}.\end{aligned}$$

Therefore, when $|\mathcal{E}| \le n(p + \epsilon/2)$, the decoding error probability is upper bounded by

$$\begin{aligned}\mathbb{P}((\mathsf{C} \setminus \{c^{(1)}\}) \cap \mathcal{D}(\mathcal{E}) \neq \emptyset) &= \mathbb{P}(\exists i > 1 \text{ such that } c^{(i)} \in \mathcal{D}(\mathcal{E})) \\ &< 2^{-n\epsilon/2} \to 0.\end{aligned}$$

Combining this with $\mathbb{P}(|\mathcal{E}| \le n(p + \epsilon/2)) = 1 - o(1)$, we conclude that C achieves the capacity of BEC.

Next we show that if $\mathsf{C} \sim \mathrm{RCE}(n, R)$, then C achieves the capacity of BSC. Suppose that the crossover probability of BSC is p, so its capacity is $1 - H(p)$, where $H(p) = -p \log_2 p - (1-p)\log_2(1-p)$ is the binary entropy function. Let $\epsilon > 0$ be an arbitrarily small positive number, and let $R = 1 - H(p) - \epsilon$. We again assume that $c^{(1)}$ is the transmitted codeword, and we use $y \in \{0,1\}^n$ to denote the channel output vector. Then $d_H(c^{(1)}, y) \le n(p + \delta)$ with probability $1 - o(1)$ for any positive constant δ, where $d_H(\cdot, \cdot)$ is the Hamming distance between two vectors. We pick a positive δ such that

$$H(p + \delta) < H(p) + \epsilon/2. \tag{3.6}$$

It is clear that if $d_H(c^{(i)}, y) > n(p + \delta)$ for all $i > 1$, then we can successfully decode. We define a Hamming ball $\mathcal{B}(x, r)$ with center $x \in \{0,1\}^n$ and radius r as

$$\mathcal{B}(x, r) = \{z \in \{0,1\}^n \colon d_H(x, z) \le r\}.$$

The volume of a Hamming ball only depends on its radius, and it has the following well-known approximation:

$$\mathrm{vol}(\mathcal{B}(x, n\beta)) = \sum_{i=0}^{n\beta} \binom{n}{i} = 2^{nH(\beta)+o(n)}.$$

Since the random codeword $c^{(i)}$ is independent of $c^{(1)}$ and y for every $i > 1$, we have

$$\begin{aligned}\mathbb{P}(d_H(c^{(i)}, y) \le n(p+\delta)) &= \mathbb{P}\left(c^{(i)} \in \mathcal{B}(y, n(p+\delta))\right) \\ &= \frac{\mathrm{vol}(\mathcal{B}(y, n(p+\delta)))}{2^n} = 2^{n(H(p+\delta)-1)+o(n)} < 2^{n(H(p)-1+\epsilon/2)+o(n)},\end{aligned}$$

where the last inequality follows from (3.6). Then by union bound,

$$\begin{aligned}\mathbb{P}(\exists i > 1 \text{ such that } d_H(c^{(i)}, y) \le n(p+\delta)) &< 2^{nR} 2^{n(H(p)-1+\epsilon/2)+o(n)} \\ &= 2^{-n\epsilon/2+o(n)} \to 0,\end{aligned}$$

where the last equality follows from the assumption $R = 1 - H(p) - \epsilon$. Combining this with $\mathbb{P}(d_H(c^{(1)}, y) \le n(p+\delta)) = 1 - o(1)$, we conclude that C achieves the capacity of BSC.

The capacity-achieving proofs above, especially the one for BSC, are based on the analyses in [16], [58]. It is worth mentioning that [16], [58] went beyond the capacity-achieving proof and further calculated the error exponents of random codes.

Next, we will analyze the performance of specific codes whose weight enumerators are close to that of random codes. To that end, we introduce some functions to measure the similarity of weight enumerators between a specific code and random codes.

Definition 7. Let C be a binary code with length n and rate R. For $1 \le i \le n$, we define

$$\alpha_i(\mathsf{C}) = \frac{A_{\mathsf{C}}(i/n)}{2^{nR-n}\binom{n}{i}},$$

where the quantity in the denominator is the expected weight enumerator of random codes; see (3.5). We further define

$$\alpha(\mathsf{C}) = \max_{1 \le i \le n} \alpha_i(\mathsf{C}).$$

If $\alpha(\mathsf{C})$ is small, then the weight enumerator of C is close to that of random codes.

Theorem 1. Let $\overline{\mathsf{C}}$ be a binary **linear** code with length n and rate R. If $\log \alpha(\overline{\mathsf{C}}) = o(n)$, then $\overline{\mathsf{C}}$ achieves the capacity of BEC and BSC.

A more rigorous way to state this theorem is to say that a family of linear codes $\{\overline{\mathsf{C}}^{(n)}\}$ with increasing code length n achieves capacity when $n \to \infty$. Here we only consider a code $\overline{\mathsf{C}}$ instead of a whole code family because we want to simplify the notation.

There are several ways to prove this theorem. The most straightforward way is to analyze the error probability of every pair of codewords, which only depends on their Hamming distance, and then apply the union bound using the pairwise error probability and the weight distribution of the code. One may consult [16, Section III] for the details of this proof method. Another proof method was given in [140]: We apply random permutations on the codewords and then show that the randomized code, which has the same error probability as the original code, achieves capacity. In fact, under the conditions in Theorem 1, one can further show that the code $\overline{\mathsf{C}}$ has a positive error exponent as long as R is strictly smaller than the channel capacity. The methods of calculating the error exponent were also given in [16], [140].

3.4 Sequential Entropy of a Code on a Channel

Let $X = (X_1, \ldots, X_n)$ be a codeword drawn uniformly at random in some code C and let $Y = (Y_1, \ldots, Y_n)$ be the output random vector when transmitting X through a BMS channel W. We can decompose the conditional entropy of X given Y using the chain rule of the Shannon entropy:

$$H(X|Y) = \sum_{i \in [n]} H(X_i|Y, X^{i-1}),$$

where X^{i-1} is the shorthand notation for the random vector $(X_1, X_2, \ldots, X_{i-1})$. Note that the above sum can be carried in any order of the indices, not necessarily from 1 to n, but any ordering covering all n indices.

The quantity $H(X_i|Y, X^{i-1})$ is the uncertainty on the i-th coordinate of the codeword given the channel output Y and the previous $i-1$

coordinates in the codeword. It is therefore a quantity relevant for the successive decoding of a codeword, which is a weaker decoder than the block-MAP decoder in terms of the decoding error probability. We will formally connect these "sequential entropies" to the successive decoder in Section 6.

3.5 EXIT Function of a Code on a Channel

In this section we briefly introduce the extrinsic information transfer (EXIT) function of binary linear codes and its properties. Let $\mathsf{C} \subseteq \mathbb{F}_2^n$ be a binary linear code of length n. A natural question is how much information is stored in a given subset of the coordinates. That is, given a random codeword $X \in \mathsf{C}$ and a subset of the coordinates $\mathcal{S} \subseteq [n]$, how much information does $X_\mathcal{S}$ reveal on X. Since the code is linear, restricting the code to a subset of the coordinates

$$\mathsf{C}_\mathcal{S} = \{x_\mathcal{S} \mid x \in \mathsf{C}\}$$

yields a linear subspace. If the rank of this subspace is equal to the code dimension of C, then this subset of coordinates completely determines the entire codeword. In general, suppose that we can only look at a subset of the coordinates $\mathcal{S} \subseteq [n]$ of a codeword $X \in \mathsf{C}$, then this tells us that X belongs to an affine subspace of dimension $\dim(C) - \dim(C_\mathcal{S})$. Moreover, if X is chosen uniformly at random then, conditioned on $X_\mathcal{S}$, X is distributed uniformly on that affine subspace. Thus we have

$$H(X|X_\mathcal{S}) = \dim(\mathsf{C}) - \dim(\mathsf{C}_\mathcal{S}).$$

In order to motivate the study of $\dim(\mathsf{C}_\mathcal{S})$ we explain its relation to decoding erasures. Let $X \in \mathsf{C}$ be a random codeword, and erase the coordinates outside of a set $\mathcal{S} \subseteq [n]$ to get a word Y (i.e., $Y_i =?$ for $i \notin \mathcal{S}$ and $Y_i = X_i$ for $i \in \mathcal{S}$). Then, we can decode X from Y if and only if $\dim(\mathsf{C}_\mathcal{S}) = \dim(\mathsf{C})$. In particular, the code C can be decoded over $\mathrm{BEC}(p)$ if and only if $\dim(\mathsf{C}_\mathcal{S}) = \dim(\mathsf{C})$ with high probability over the choice of a random subset $\mathcal{S}$, where the distribution on subsets is such that

$$\mathbb{P}\left(i \in \mathcal{S}\right) = 1 - p$$

independently for every $i \in [n]$.

It seems that trying to directly determine the probability in which $\dim(\mathsf{C}_{\mathcal{S}}) = \dim(\mathsf{C})$ is incredibly difficult. However, a similar quantity, which we shall now present, is evidently better behaved and easier to analyze.

Definition 3.1 (EXIT Function)**.** Let $\mathsf{C} \subseteq \mathbb{F}_2^n$ be a binary linear code of length n. The EXIT function of C is a function $h\colon [0,1] \to [0,1]$ defined by

$$h(p) = \frac{1}{n} \cdot \sum_{i=1}^{n} H(X_i | (Y_1, \ldots, Y_{i-1}, Y_{i+1}, \ldots, Y_n)),$$

where $X = (X_1, \ldots, X_n)$ is a uniformly random codeword in C, and $Y = (Y_1, \ldots, Y_n)$ is the channel output when transmitting X over $\mathrm{BEC}(p)$.

Recall the definitions of bit-MAP decoding and bit-MAP decoding error probability from (3.3)–(3.4). The EXIT function is closely related to the bit-MAP decoding error probability over BEC channels. Let us first consider the quantity $H(X_i | Y_1, \ldots, Y_n)$. For every realization $y_1, \ldots, y_n$ of the random vector $Y_1, \ldots, Y_n$, the conditional entropy $H(X_i | Y_1 = y_1, \ldots, Y_n = y_n)$ is either 0 or 1. More precisely, let $\mathcal{S} \subseteq [n]$ be the set of coordinates that are **not** erased in $y_1, \ldots, y_n$. Then

$$H(X_i | Y_1 = y_1, \ldots, Y_n = y_n) = \begin{cases} 0 & \dim(\mathsf{C}_{\mathcal{S}}) = \dim(\mathsf{C}_{\mathcal{S} \cup \{i\}}) \\ 1 & \dim(\mathsf{C}_{\mathcal{S}}) \neq \dim(\mathsf{C}_{\mathcal{S} \cup \{i\}}) \end{cases}$$

In the latter case, the bit-MAP decoding error probability for the ith coordinate is $1/2$ because X_i takes values 0 and 1 with equal probability. As a consequence, we have

$$\mathit{biterr}(\mathrm{BEC}(p), \mathsf{C}) = \frac{1}{2n} \sum_{i=1}^{n} H(X_i | Y_1, \ldots, Y_n).$$

Since

$$\begin{aligned} H(X_i | Y_1, \ldots, Y_n) &= (1-p) H(X_i | (Y_1, \ldots, Y_{i-1}, Y_{i+1}, \ldots, Y_n), Y_i = X_i) \\ &\quad + p H(X_i | (Y_1, \ldots, Y_{i-1}, Y_{i+1}, \ldots, Y_n), Y_i = ?) \\ &= p H(X_i | (Y_1, \ldots, Y_{i-1}, Y_{i+1}, \ldots, Y_n)), \end{aligned}$$

we obtain that

$$biterr(\mathrm{BEC}(p), \mathsf{C}) = \frac{1}{2} p h(p). \tag{3.7}$$

Apart from the connection with bit-MAP decoding error probability, the function $h(p)$ has many other useful properties, some of which we list below. Interested readers are referred to Chapter 3 in [123] for a detailed exposition of the EXIT function.

Lemma 3.2. Let $\mathsf{C} \subseteq \mathbb{F}_2^n$ be a binary code of length n. Then,

(1) Monotonicity: h is increasing. $h(1) = 1$. If $\dim(\mathsf{C}_{\mathcal{S}}) = \dim(\mathsf{C})$ for every subset $\mathcal{S} \subseteq [n]$ with size $|\mathcal{S}| = n - 1$, then $h(0) = 0$.

(2) Area Theorem: $\int_0^1 h(\lambda) \mathrm{d}\lambda = \mathrm{rate}\,(\mathsf{C})$.

(3) Duality: Denote by $h^{\perp}(p)$ the EXIT function of $C^{\perp}$. Then

$$h^{\perp}(p) = 1 - h(1-p).$$

(4) Let X be a uniformly random codeword in C, and let Y be the channel output vector after transmitting X over $\mathrm{BEC}(p)$. Then

$$H(X|Y) = n \cdot \int_0^p h(\lambda) \mathrm{d}\lambda$$

(5) For every $\lambda \in (0, 1)$

$$\begin{aligned} n \cdot \int_0^{\lambda} h(\lambda') \mathrm{d}\lambda' &= \lambda \cdot n - \mathbb{E}_{\mathcal{S} \sim \lambda}[\dim(\mathsf{C}_{\mathcal{S}}^{\perp})] \\ &= \dim(\mathsf{C}) - \mathbb{E}_{\mathcal{S} \sim \lambda}[\dim(\mathsf{C}_{\overline{\mathcal{S}}})], \end{aligned}$$

where $\mathcal{S} \sim \lambda$ means that each i belongs to $\mathcal{S}$ with probability λ independently for every $i \in [n]$, $\mathsf{C}^{\perp}$ is the dual code of C, and $\overline{\mathcal{S}} = [n] \setminus \mathcal{S}$ is the complement of $\mathcal{S}$.

(6) The EXIT function $h(p)$ equals the bit-MAP decoding error, i.e., $biterr(\mathrm{BEC}(p), \mathsf{C}) = \frac{1}{2} p h(p)$.

Proof.

(1) Monotonicity follows directly from the definition. For each $i \in [n]$, the ith summand in $h(p)$ is a conditional entropy $H(X_i|(Y_1, \ldots, Y_{i-1}, Y_{i+1}, \ldots, Y_n))$, which measures the uncertainty of X_i given the channel output of the BEC channel. The uncertainty clearly increases with the erasure probability p of the BEC channel, so $h(p)$ is an increasing function.

(2) Instead of standard BEC channels, let us now consider the following variation: Suppose that each X_i is erased independently with probability p_i for $i \in [n]$, and we denote the corresponding channel output as $Y_i(p_i)$. In this variation we allow the erasure probabilities $p_1, p_2, \ldots, p_n$ to take different values while in standard BEC channels they must be equal to each other. Then we have

$$\begin{aligned}
&H(X_1, \ldots, X_n|Y_1(p_1), \ldots, Y_n(p_n)) \\
&= H(X_i|Y_1(p_1), \ldots, Y_n(p_n)) \\
&\quad + H(X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n|X_i, Y_1(p_1), \ldots, Y_n(p_n)) \\
&= H(X_i|Y_1(p_1), \ldots, Y_n(p_n)) \\
&\quad + H(X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n|X_i, Y_1(p_1), \ldots, Y_{i-1}(p_{i-1}), \\
&\qquad Y_{i+1}(p_{i+1}), \ldots, Y_n(p_n)).
\end{aligned}$$

Since the term in the last line is independent of p_i, we obtain that

$$\frac{\partial H(X_1, \ldots, X_n|Y_1(p_1), \ldots, Y_n(p_n))}{\partial p_i} = \frac{\partial H(X_i|Y_1(p_1), \ldots, Y_n(p_n))}{\partial p_i}.$$

The partial derivative on the right-hand side can be calculated as follows.

$$\begin{aligned}
&H(X_i|Y_1(p_1), \ldots, Y_n(p_n)) \\
&= (1 - p_i)H(X_i|(Y_1(p_1), \ldots, Y_{i-1}(p_{i-1}), Y_{i+1}(p_{i+1}), \ldots, Y_n(p_n)), \\
&\quad Y_i(p_i) = X_i) + p_i H(X_i|(Y_1(p_1), \ldots, Y_{i-1}(p_{i-1}), \\
&\quad Y_{i+1}(p_{i+1}), \ldots, Y_n(p_n)), Y_i(p_i) = \text{erasure}) \\
&= p_i H(X_i|(Y_1(p_1), \ldots, Y_{i-1}(p_{i-1}), Y_{i+1}(p_{i+1}), \ldots, Y_n(p_n))).
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\frac{\partial H(X_1,\ldots,X_n|Y_1(p_1),\ldots,Y_n(p_n))}{\partial p_i}\\
&= \frac{\partial H(X_i|Y_1(p_1),\ldots,Y_n(p_n))}{\partial p_i}\\
&= H(X_i|(Y_1(p_1),\ldots,Y_{i-1}(p_{i-1}),Y_{i+1}(p_{i+1}),\ldots,Y_n(p_n))).
\end{aligned}$$

Thus we obtain that

$$\begin{aligned}
&\frac{\mathrm{d}H(X_1,\ldots,X_n|Y_1(p),\ldots,Y_n(p))}{\mathrm{d}p}\\
&= \sum_{i=1}^{n} \frac{\partial H(X_1,\ldots,X_n|Y_1(p_1),\ldots,Y_n(p_n))}{\partial p_i}\Big|_{p_1=\ldots=p_n=p}\\
&= \sum_{i=1}^{n} H(X_i|(Y_1(p),\ldots,Y_{i-1}(p),Y_{i+1}(p),\ldots,Y_n(p))) = nh(p).
\end{aligned} \tag{3.8}$$

Therefore,

$$\begin{aligned}
n\int_0^1 h(\lambda)\mathrm{d}\lambda &= H(X_1,\ldots,X_n|Y_1(1),\ldots,Y_n(1))\\
&\quad - H(X_1,\ldots,X_n|Y_1(0),\ldots,Y_n(0))\\
&= H(X_1,\ldots,X_n) - 0 = \dim(\mathsf{C}).
\end{aligned}$$

This proves the area theorem.

(3) Let $\mathsf{C}^\perp$ be the dual code of C. One can show that

$$\begin{aligned}
&H(X_i|(Y_1(p),\ldots,Y_{i-1}(p),Y_{i+1}(p),\ldots,Y_n(p)))\\
&= \sum_{\mathcal{S}\subseteq[n]\setminus\{i\}} p^{n-1-|\mathcal{S}|}(1-p)^{|\mathcal{S}|}(\dim(\mathsf{C}_{\mathcal{S}\cup\{i\}}) - \dim(\mathsf{C}_{\mathcal{S}}))\\
&= \sum_{\mathcal{S}\subseteq[n]\setminus\{i\}} (1-p)^{n-1-|\mathcal{S}|}p^{|\mathcal{S}|}(1 + \dim(\mathsf{C}_{\mathcal{S}}^\perp) - \dim(\mathsf{C}_{\mathcal{S}\cup\{i\}}^\perp)).
\end{aligned}$$

The relation $h^\perp(p) = 1 - h(1-p)$ follows from this equality.

(4) Follows directly from (3.8).

(5) Follows from 4.

(6) Already proved in (3.7).

□

Remark 1. EXIT charts were introduced by ten Brink [33] in the context of turbo decoding as a visual tool for understanding iterative decoding. The area theorem was originally proven by Ashikhmin *et al.* in [13] and further generalized by Méasson *et al.* in [102].

An important application of the area theorem lies in the capacity-achieving proofs. The first step of such proofs is to show that the EXIT function exhibits a sharp transition from 0 to 1. Then the area theorem is used to pin down the transition threshold. This proof method is discussed in detail in Section 4.3.2 for RM codes, and the sharp threshold phenomenon is illustrated in Figure 4.1.

The EXIT function defined in this subsection is associated with the BEC channel. Later in Section 5.4, we will introduce the generalized EXIT (GEXIT) function, which satisfies another version of the area theorem and can be used to analyze the performance of codes over general BMS channels; see (5.6)–(5.8).

3.6 Connecting the Performance Measures

The main objective in coding theory is to control the block error probability of data transmission over noisy channels. Recall that the block error probability of a code C over a channel W is denoted $err(W, \mathsf{C})$. We have the following bounds between the quantities defined in previous subsections and $err(W, \mathsf{C})$. First, we have a bound involving the Bhattacharyya parameter and the weight enumerator[1].

[1] Other variants can be used; see [1], [115]

Theorem 2 (Bhattacharyya Bound)**.** Let W be a BMS channel and let $\mathsf{C} \subseteq \mathbb{F}_2^n$ be a binary linear code of length n. Then, the block-MAP error probability is bounded via

$$err(W, \mathsf{C}) \leq \sum_{i=1}^{n} \mathrm{Z}(W)^i \cdot A_{\mathsf{C}}(i/n).$$

where $A_{\mathsf{C}}(\cdot)$ is the weight enumerator defined in Definition 5, and $\mathrm{Z}(W)$ is the Bhattacharyya parameter defined in (3.1). In particular, for the BEC and the BSC we have $\mathrm{Z}(\mathrm{BEC}(p)) = p$ and $\mathrm{Z}(\mathrm{BSC}(p)) = 2\sqrt{p(1-p)}$, so we obtain

$$\begin{aligned} err(\mathrm{BEC}(p), \mathsf{C}) &\leq \sum_{i=1}^{n} p^i \cdot A_{\mathsf{C}}(i/n), \\ err(\mathrm{BSC}(p), \mathsf{C}) &\leq \sum_{i=1}^{n} \left(2\sqrt{p(1-p)}\right)^i \cdot A_{\mathsf{C}}(i/n). \end{aligned}$$

Proof. Since C is a linear code, we can simply assume that the all-zero codeword is transmitted. Then the block-MAP decoder makes an error if there is a non-zero codeword $c = (c_1, \ldots, c_n) \in \mathsf{C}$ such that the channel output vector $(y_1, \ldots, y_n)$ satisfies

$$\prod_{i=1}^{n} W(y_i|c_i) \geq \prod_{i=1}^{n} W(y_i|0).$$

Define the set $\mathcal{N}(c) = \{i \in [n] \mid c_i = 1\}$. Then the above inequality is equivalent to

$$\prod_{i \in \mathcal{N}(c)} W(y_i|1) \geq \prod_{i \in \mathcal{N}(c)} W(y_i|0).$$

Let $\mathcal{Y}$ be the output alphabet of W. We define the following set of channel output vectors

$$\mathcal{B}(c) = \{(y_1, \ldots, y_n) \in \mathcal{Y}^n \mid \prod_{i \in \mathcal{N}(c)} W(y_i|1) \geq \prod_{i \in \mathcal{N}(c)} W(y_i|0)\}.$$

By definition, the block-MAP decoder makes an error when the output vector falls into the set $\mathcal{B}(c)$ for some non-zero codeword c. The

probability of obtaining an output vector in the set $\mathcal{B}(c)$ can be upper bounded as follows:

$$\begin{aligned}
&\mathbb{P}((Y_1,\ldots,Y_n)\in\mathcal{B}(c))\\
&=\sum_{(y_1,\ldots,y_n)\in\mathcal{B}(c)}\prod_{i=1}^{n}W(y_i|0)\\
&=\sum_{(y_1,\ldots,y_n)\in\mathcal{B}(c)}\Big(\Big(\prod_{i\in\mathcal{N}(c)}W(y_i|0)\Big)\Big(\prod_{i\notin\mathcal{N}(c)}W(y_i|0)\Big)\Big)\\
&\leq\sum_{(y_1,\ldots,y_n)\in\mathcal{B}(c)}\Big(\Big(\prod_{i\in\mathcal{N}(c)}W(y_i|0)\Big)\Big(\prod_{i\notin\mathcal{N}(c)}\sqrt{W(y_i|0)W(y_i|1)}\Big)\Big)\\
&\leq\sum_{(y_1,\ldots,y_n)\in\mathcal{Y}^n}\Big(\Big(\prod_{i\in\mathcal{N}(c)}W(y_i|0)\Big)\Big(\prod_{i\notin\mathcal{N}(c)}\sqrt{W(y_i|0)W(y_i|1)}\Big)\Big)\\
&=\prod_{i\in\mathcal{N}(c)}\Big(\sum_{y_i\in\mathcal{Y}}W(y_i|0)\Big)\prod_{i\notin\mathcal{N}(c)}\Big(\sum_{y_i\in\mathcal{Y}}\sqrt{W(y_i|0)W(y_i|1)}\Big)\\
&=\prod_{i\notin\mathcal{N}(c)}\Big(\sum_{y_i\in\mathcal{Y}}\sqrt{W(y_i|0)W(y_i|1)}\Big)=\prod_{i\notin\mathcal{N}(c)}Z(W)=(Z(W))^{\mathrm{wt}(c)}.
\end{aligned}$$

The upper bound on the block-MAP error probability is then obtained from applying the union bound on all non-zero codewords:

$$err(W,\mathsf{C})\leq\sum_{c\in\mathsf{C},c\neq 0}(Z(W))^{\mathrm{wt}(c)}=\sum_{i=1}^{n}\mathrm{Z}(W)^i\cdot A_{\mathsf{C}}(i/n).$$

This completes the proof of the theorem. □

In Section 4 we prove bounds on the weight enumerator of RM-codes. Specifically, in Section 4.1 we discuss the combinatorial approach for bounding the weight enumerator. This approach was initiated in the work of Kaufman *et al.* [84], and was later strengthened in [1], [134]. In Section 4.3 we discuss the analytical approach of [127] and the subsequent improvements in [76], [126]. These bounds are then used in Section 5 to prove upper bounds on the error probability of the ML decoder.

We then have a bound using the sequential entropies,

$$err(W,\mathsf{C})\leq H(X|Y)=\sum_{i\in[n]}H(X_i|Y,X^{i-1}),$$

where $X = (X_1, \ldots, X_n)$ is a random codeword uniformly chosen from C, and $Y = (Y_1, \ldots, Y_n)$ is the channel output vector after transmitting X over the channel W. We refer to Section 6.1 for a proof. Moreover, the above bound is often used by embedding the code in full dimension and taking advantage of the conservation of the entropy to show that the conditional entropies polarize and to obtain capacity results; this is discussed in Section 6.

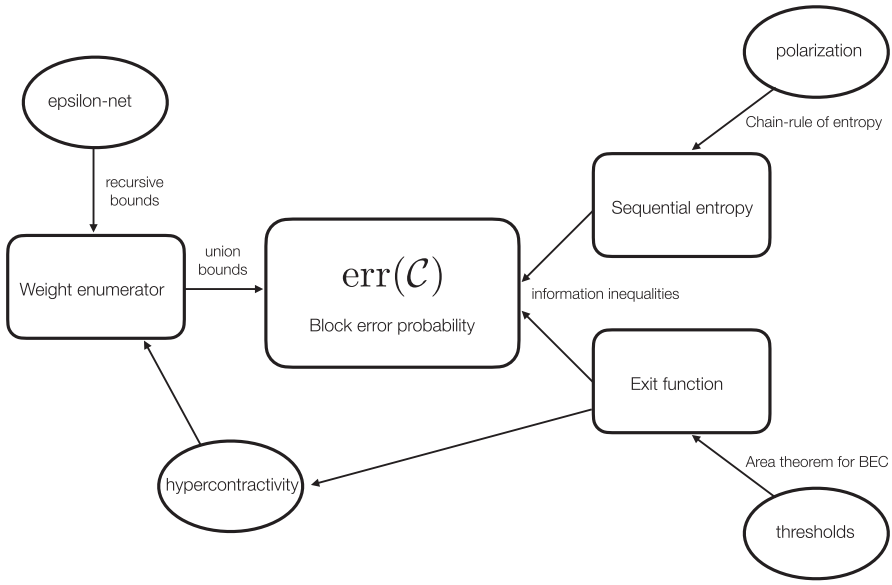


Figure 3.1: Relation between different performance measures and different methods to bound the error probability.

Finally, we also have the following bound for the bit error probability

$$err(\mathrm{BEC}(p), \mathsf{C}) \leq \frac{n}{d} biterr(\mathrm{BEC}(p), \mathsf{C}).$$

We refer to Proposition 11 in Section 5.3 for a proof. This bound is often used with the Area theorem (which uses the entropy interpolation), and exploits the fact that for the BEC, $biterr(\mathrm{BEC}(p), \mathsf{C})$ corresponds to the probability of a monotone Boolean property, and thus to show that threshold phenomena holds and conclude capacity results. For non-BEC channels, this approach does not apply directly. One may still use the Area Theorem and show that the bit error rate satisfies a threshold

phenomenon, but these no longer rely on general results from monotone Boolean properties. In particular, a recent paper [122] used this method to prove that RM codes achieve capacity of general BMS channels under the bit-MAP decoder, and this result is explained in Section 5.4.

Figure 3.1 shows the relation between different performance measures and different methods to bound the error probability. It also gives an outline of the next three sections.

4

Bounds on the Weight Enumerator

In this section we survey known results on the weight distribution of binary RM codes. We first discuss the results in chronological order and then give some proofs. Recall from Section 3.3 that the weight enumerator of a code is a function that for every weight $0 \leq \beta \leq 1$ returns the number of codewords of relative-weight β. Thus, $A_{m,r}(\leq \beta)$ counts the number of codewords of (relative) weight at most β. In particular, for $\beta < 2^{-r}$, $A_{m,r}(\leq \beta) = 1$.

As mentioned in Section 3.3, if the weight enumerator of a code is similar to that of a random code, of the same rate, then the code achieves capacity on the BEC and the BSC. Clearly, RM codes are quite different than random codes, e.g., we expect the relative-weight of every non-zero codeword of a random code to be roughly $(1/2 \pm \epsilon)$ (where ϵ is a constant depending on the rate), whereas RM codes contain many codewords of small weight. Nevertheless, as we shall see, if one can show that the weight enumerator drops quickly for $\beta < 1/2$ then this may be sufficient for proving that the code achieves capacity. Thus, proving strong upper bound on the weight enumerator for weights slightly smaller than $1/2$ is an interesting and in some cases also a fruitful approach to proving that RM codes achieve capacity.

Computing the weight enumerator of RM codes is a hard problem that is open in most ranges of parameters. The case of RM codes of degree 1 is very simple as codewords are linear functions and so

any nonconstant codeword is completely balanced. The case of degree two RM codes was completely solved by Sloane and Berlekamp [143]. However, for larger degrees it is quite difficult to exactly calculate the number of codewords of a given weight and so we try to give upper and lower estimates on this number. In 1970, Kasami and Tokura [82] characterized all codewords of weight less than twice the minimum distance. This was later improved in [83] to all codewords of weight less than 2.5 times the minimal distance. For degrees larger than 3 they obtained the following result.

Theorem 3 (Theorem 1 of [83]). If $r \geq 3$ and $f \in \mathcal{P}_{m,r}$ satisfies $\mathrm{wt}_{\mathrm{n}}(f) < 2.5 \cdot 2^{-r}$ then, up to an invertible linear transformation,

$$f = x_1 g(x_3, \ldots, x_m) + x_2 h(x_3, \ldots, x_m) + x_1 x_2 s(x_3, \ldots, x_m),$$

where $\deg(g) = \deg(h) = r - 1$ and $\deg(s) = r - 2$.

By counting the number of such representations one can get a good estimate (from above) on the number of codewords of such relative-weight. No significant progress was made for over thirty years until in [84] Kaufman, Lovett and Porat gave, for any constant degree $r = O(1)$, asymptotically tight bounds on the weight enumerator of RM codes of degree r. Unfortunately, as the degree gets higher, the estimate in Theorem 3.1 of [84] becomes less and less tight. Building on the techniques of [84], Abbe *et al.* [1] managed to get better bounds for degrees up to $m/4$, which they used to show that RM codes achieve capacity for the BEC and the BSC for degrees $r = o(m)$.

Sberlo and Shpilka [134] polished the techniques of [1] and managed to obtain good estimates for every degree.

Theorem 4 (Theorem 1.2 of [134]). Let $\gamma = r/m$. Then, for every integer ℓ,

$$\begin{aligned} A_{m,r}\left(\leq 2^{-\ell}\right) &\leq 2^{\left(\sum_{j=\ell}^{r} 17m(j-1)(j+2)+17(j+2)\binom{m-(j-1)}{\leq r-(j-1)}\right)} \\ &\leq 2^{\left(O(m^4)+17(c_\gamma \ell + d_\gamma)\binom{m-(\ell-1)}{\leq r-(\ell-1)}\right)} \\ &\leq 2^{\left(O(m^4)+17(c_\gamma \ell + d_\gamma)\gamma^{\ell-1}\binom{m}{\leq r}\right)}, \end{aligned}$$

where $c_\gamma = \frac{1}{1-\gamma}$ and $d_\gamma = \frac{2-\gamma}{(1-\gamma)^2}$.

To better understand the upper bound, note that the leading term in the exponent is $O\left(\ell\gamma^{\ell-1}\binom{m}{\leq r}\right)$ (when $0 < \gamma < 1$ is a constant). In contrast, if we were to calculate the same quantity in Theorem 3.3 of [1], its leading term would be $O\left(\ell^4\gamma^{\ell-1}\binom{m}{\leq r}\right)$. This is what prevented the authors of [1] from extending their results beyond degree $r = m/4$.

As mentioned in Section 3.3, the proofs in [1], [134] use the technique developed in [84], which we call the combinatorial approach for bounding the weight enumerator. In [127], Samorodnitsky presented a new technique, building on log-Sobolev type inequalities, and proved a remarkable general result regarding the weight enumerator of codes that either they or their dual code achieve capacity for the BEC. The results of [127] were later improved in [76], [126], [128], where the new proofs used information about the EXIT function of RM codes (see Section 3.5) to get tighter bounds on the relevant estimates in [127]. The results of [76], [126]–[128] hold for codes that are invariant under a doubly transitive permutation group, but when specialized for RM codes they yield slightly tighter results than for general codes.

Theorem 5 (Propositions 1.1 and 1.3 in [128])**.** Let $C = \mathrm{RM}(m, r)$ be the binary RM code of positive rate $0 < R < 1$. There exists R^* such that $|R - R^*| = \frac{o(1)}{\sqrt{n}}$ and such that for all $0 \leq b \leq n$ it holds that

$$A_{m,r}(b/n) \leq O\left((1 - R^*)^{-2b\cdot\ln 2}\right).$$

When b/n is close to $1/2$ we get a stronger result: For $0 \leq b \leq n$, let $b^* = \min\{b, n - b\}$. Let $\theta = R^{2\ln 2}$. If $\frac{1-\theta}{2} \cdot n \leq b^*$ then

$$A_{m,r}(b/n) \leq 2^{o(n)} \cdot \binom{n}{b^*} \cdot \frac{|C|}{2^n}.$$

We note that the second of these bounds implies that the weight distribution of $\mathrm{RM}(m, r)$ of rate R, is essentially upper-bounded by that of a random code of the same rate in the band of width $R^{2\ln 2}$ around $1/2$.

On the other hand, for small rates, e.g., when $r/m = (1 - \epsilon)/2$ for $0 < \epsilon < 1$, Theorem 4 gives better bounds than Theorem 5. Observe that, for small R (but not smaller than $1/\sqrt{n}$), the bound in

the first inequality of Theorem 5 is roughly $|C|^{2b/n}$, obtained as follows: Since R (and hence R^*) is small, we have $(1-R^*)^{-2b\cdot\ln 2} \approx e^{-R\cdot(-2b\cdot\ln 2)} = 2^{2R\cdot n\cdot b/n} = 2^{Rn\cdot 2b/n} = |C|^{2b/n}$. If $b/n = 2^{-\ell}$, then the upper bound is $|C|^{(1/2)^{\ell-1}}$, whereas Theorem 4 gives an upper bound of $|C|^{O\left(\ell\cdot(r/m)^{\ell-1}\right)} = |C|^{O\left(\ell\cdot(1-\epsilon)^{\ell-1}\cdot(1/2)^{\ell-1}\right)}$, which may be much smaller.

Note, however, that the bound in Theorem 4 is not so good when the relative-weight is close to $1/2$. As most codewords have weight close to $n/2$ it is interesting to understand the weight distribution at that regime. In particular, it is interesting to know how many codewords have weight smaller than $(1-\epsilon)n/2$ for small ϵ. To the best of our knowledge, the first such result was obtained by Ben-Eliezer *et al.* [23], who proved the following.

Theorem 6 (Lemma 2 in [23]). Let $m, r \in \mathbb{N}$ and $\delta > 0$ such that $r \leq (1-\delta)m$. Then there exist positive constants c_1, c_2 (which depend solely on δ) such that,

$$A_{m,r}\left(\leq \frac{1-2^{-c_1\frac{m}{r}}}{2}\right) \leq 2^{\left((1-c_2)\binom{m}{\leq r}\right)}.$$

This result was later extended to other prime fields in [18]. Theorem 6 does not give information about the number of codewords of weight $\frac{1-o(1)}{2}n$. Such a result was obtained in [134], and it played an important role in their results on the capacity of RM codes. [134] first proved a result for the case that $r < m/2$ and then for the general case (with a weaker bound).

Theorem 7 (Proposition 3.15 of [134]). Let $m, r, s, \ell \in \mathbb{N}$ such that $r \leq m$ and write $\gamma = r/m$. Then,

$$A_{r,m}\left(\frac{1-2^{-\ell}}{2}\right) \leq \exp_2\left(O(m^4) + \left(1-(1-\tilde{\gamma})^{2\ell+s+1} + 17(c_\gamma(s-1)+d_\gamma)\gamma^{s-2}\right)\cdot\binom{m}{\leq r}\right),$$

where $\tilde{\gamma} = \gamma\left(1+\frac{2\ell+s+1}{m-(2\ell+s+1)}\right)$, $c_\gamma = \frac{1}{1-\gamma}$, $d_\gamma = \frac{2-\gamma}{(1-\gamma)^2}$.

For a given r, m, ℓ one has to choose s to minimize the expression. In Theorem 1.4 of [134] the authors slightly weakened the bound in

order to get a simpler expression. Namely, in Theorem 1.4 of [134] they proved

$$A_{m,\gamma m}\left(\leq \frac{1-2^{-\ell}}{2}\right) \leq 2^{\left(O(m^4)+\left(1-2^{-c(\gamma,\ell)}\right)\binom{m}{\leq r}\right)}, \tag{4.1}$$

where $c(\gamma,\ell) = O\left(\frac{\gamma^2\ell+\gamma\log(1/1-2\gamma)}{1-2\gamma}+\gamma\right)$.

As even after the simplification the form of the bound is a bit complicated, the following remark was made in [134].

Remark 2. To make better sense of the parameters in the theorem we note the following.

- When $\gamma < 1/2$ is a constant, the exponent is essentially $2^{(1-\exp(-\ell))\binom{m}{\leq r}}$.
- The bound is meaningful up to degrees $r \leq \left(\frac{1}{2} - \Omega\left(\frac{\sqrt{\log m}}{\sqrt{m}}\right)\right) m$, but falls short of working for constant rate RM codes.
- For γ approaching $1/2$, i.e., $\gamma = 1/2 - o(1)$, there is a trade-off between how small the $o(1)$ is and the largest ℓ for which the bound is applicable to. Nevertheless, even if $\gamma = 1/2 - \Omega\left(\sqrt{\frac{\log m}{m}}\right)$ the statement still holds for $\ell = \Omega(\log m)$ (i.e., for a polynomially small bias).

A later work by Rao and Sprumont [119] proved a more general bound that improves earlier works for a certain range of parameters. Specifically, they obtained the best results for RM codes of non-constant rate and sub-constant bias, or when $\gamma = r/m$ is larger than some constant and the relative-weight is also larger than some constant. Calculating the exact constants is a bit cumbersome and requires a delicate optimization and comparison of the expressions in Theorems 4, 5 and 8.

Theorem 8 (Theorem 1 in [119], Restated)**.** For every $0 < \beta < 1$

$$A_{m,r}(\beta) \leq 2^{H(\beta)\cdot\binom{m}{\leq r}},$$

where $H(\beta) = -\beta \log_2 \beta - (1-\beta)\log_2(1-\beta)$ is the binary entropy function[1].

Although the exact constants are not mentioned in Theorem 6, one can verify that Theorem 8 subsumes the result in Theorem 6. As the proof of Theorem 8 is short and elegant we give it here.

Proof of Theorem 8. In our argument we can rearrange the coordinates of the codeword so let us assume, without loss of generality, that the first $Rn = \binom{m}{\leq r}$ coordinates are linearly independent.

Fix $0 < \beta < 1$ and let C_β be the set of all codewords of relative-weight β. Let $(Z_1, \ldots, Z_n)$ be a codeword sampled uniformly at random from C_β. Since the code is transitive, we have for every i, j that $H(Z_i) = H(Z_j)$. As the weight is fixed to be $\beta \cdot n$ symmetry (transitivity) implies that $\mathbb{P}_{Z_i}(Z_i = 1) = \beta$ and hence $H(Z_i) = H(\beta)$. We now get by the chain rule that

$$\begin{aligned}
\log |C_\beta| &= H(Z_1, \ldots, Z_n) \\
&= H(Z_1) + H(Z_2|Z_1) + \cdots + H(Z_{Rn}|Z_1, \ldots, z_{Rn-1}) \\
&\quad + H(Z_{Rn+1}, \ldots, Z_n|Z_1, \ldots, Z_{Rn}) \\
&\leq H(Z_1) + \cdots + H(Z_{Rn}) + 0 = RnH(\beta) = H(\beta) \cdot \binom{m}{\leq r},
\end{aligned}$$

where we have used the fact that, by our assumption, the first $Rn = \binom{m}{\leq r}$ coordinates are linearly independent and hence span the rest of the coordinates. □

As the proof shows, the result of Rao and Spurmont actually holds for codes that are invariant under the action of a transitive group.

We summarize the best upper bound results proved on the weight distribution of RM codes in Table 4.1.

To complete the picture we state two lower bounds on the weight enumerator. The first is a fairly straightforward observation.

[1] When we write $H(\beta)$ or $H(p)$, we refer to the binary entropy function. When we write $H(X)$ or $H(X|Y)$ for random variables/vectors X and Y, we refer to the entropy or conditional entropy of these random variables/vectors.

Table 4.1: Known upper bounds on the weight enumerator

Rate R/ degree r	**Weight**	$A_{m,r}(\cdot)$	**Comment**	**Reference**
$0 < R < 1$ constant	$\beta \in [\frac{1}{2} - R^{2\ln 2}, \frac{1}{2}]$	$2^{o(n)} \cdot \binom{n}{b} \cdot \frac{\lvert \mathrm{RM}(m,r)\rvert}{2^n}$	As in a random code	Thm. 5 [128]
	$\beta < \frac{1}{2} - R^{2\ln 2}$	$O\left((1-R^*)^{-2b\cdot\ln 2}\right)$	$\lvert R - R^*\rvert = \frac{o(1)}{\sqrt{n}}$	
$0 < \gamma = \frac{r}{m} < 1$	$\beta \le 2^{-\ell}$	$2^{\left(O_\gamma(\ell)\cdot\gamma^{\ell-1}\binom{m}{\le r}\right)}$	$A_{m,r}(\le \beta)$	Thm. 4 [134]
$0 < \gamma < \frac{1}{2}$ $-O\left(\frac{\sqrt{\log m}}{\sqrt{m}}\right)$	$\beta \le \frac{1-2^{-\ell}}{2}$	$2^{\left(\left(1-2^{-O_\gamma(\ell)}\right)\binom{m}{\le r}\right)}$	$A_{m,r}(\le \beta)$	Thm. 7 [134]
Any r	any β	$2^{H(\beta)\cdot\binom{m}{\le r}}$		Thm. 8 [119]

Observation 1.

$$\frac{1}{2}\cdot 2^{\binom{m-\ell+1}{\le r-\ell+1}} \le A_{m,r}\left(\le 2^{-\ell}\right).$$

Comparing to Theorems 4 and 5, we see that for $\ell = O(1)$ the lower bound in Observation 1 is roughly of the form $\frac{1}{2}2^{(r/m)^{\ell-1}\cdot\binom{m}{\le r}}$. Thus, for $r = m/2$ it is similar to what Theorem 5 gives, except that it has a smaller constant in the exponent: 1 versus $2\ln 2$. For $r = \gamma m$ Observation 1 gives a much smaller constant in the exponent compared to Theorem 4: 1 versus $17(c_\gamma \ell + d_\gamma)$. The main difference is for very small weights, say $\ell = (1-\epsilon)r$. In that case, the estimate in Observation 1 is much smaller than what Theorem 4 gives. This is mainly due to the fact that Theorem 4 heavily relies on estimates of binomial coefficients that become less and less good as $(r-\ell)$ gets smaller.

The second lower bounds was given in [134] and it concerns weights around $1/2$, i.e., codewords with small bias.

Theorem 9 (Theorem 1.8 of [134])**.** Let $20 \le r \le m \in \mathbb{N}$. Then for any integer $\ell < r/3$ and sufficiently large m it holds that

$$\frac{1}{2}\cdot 2^{\left(\sum_{j=1}^{\ell-1}\binom{m-j}{\le r-1}\right)} \le A_{m,r}\left(\le \frac{1-2^{-\ell}}{2}\right).$$

Comparing the upper bound in Theorem 7 to Theorem 9 we see that there is a gap between the two bounds. Roughly, the lower bound on

the number of polynomials that have bias at least ϵ matches the upper bound corresponding to bias at least $\sqrt{\epsilon}$. This may be a bit difficult to see when looking at Theorem 4 but we refer the reader to Remark 3.16 in [134] for a qualitative comparison.

The rest of the section is organized as follows. In Section 4.1 we discuss combinatorial approaches for bounding the weight enumerator of RM codes following the influential paper of Kaufman *et al.* [84]. In Section 4.3 we give another approach for bounding the weight enumerator using the approach of Samorodnitsky [127] and the techniques developed in Section 3.5.

Using the results that we survey in these sections we prove in Section 5 results about the ability of RM codes to decode from random errors and erasures. Namely, that constant rate RM codes can decode from a constant fraction of random errors and that RM codes achieve capacity for the BEC and BSC in certain parameter regimes.

4.1 The Combinatorial Approach

In this section we give the proofs of Theorems 4 and 7 as they share a similar structure. The proofs follow the strategy of [84] and its later refinements in [1], [134]. The approach as presented in [84] is as follows:

[84].(1) Prove that if a degree r polynomial has "small" weight then it can be expressed as a function $F(g_1, \ldots, g_t)$ of "few" polynomials of lower degree $\{g_i\} \subset \mathcal{P}_{m,r-1}$.

[84].(2) Use a counting argument to bound the number of such expressions $F(g_1, \ldots, g_t)$ and deduce an upper bound estimate on the number of polynomials of "small" weight.

In [1] this approach was strengethened in the following way.

[1].(1) Prove that if a degree r polynomial has "small" weight then it can be *approximated* by a function of the form $F(g_1, \ldots, g_s)$, where the $\{g_i\} \subset \mathcal{P}_{m,r-1}$.

The main difference between [84].(1) and [1].(1) is that instead of exactly computing the polynomial, as in [84].(1), in [1].(1) we compute a function

that is close in hamming distance to the low-weight polynomial. This allows the parameter s in [1].(1) (the number of g_is) to be much smaller compared to the parameter t in [84].(1). On the other hand, now that we only have approximations of low-weight codewords, it is not clear how to bound their number. The main idea of [1] is that all codewords that are ϵ close to $F(g_1, \ldots, g_s)$ are at distance at most 2ϵ from each other and therefore, we can bound their number by $A_{m,r}(\leq 2\epsilon)$, which we compute recursively.

More abstractly, this approach can be summarized as follows: in order to upper bound the number of polynomials, of certain weight, we should find a relatively small set, in the space of all functions $\mathbb{F}_2^m \to \mathbb{F}_2$, such that all low weight polynomials are contained in balls of radius at most ϵ (in relative Hamming distance) around the elements of the set (thisis the set of all functions $F(g_1, \ldots, g_s)$ described above). We call such a set an ϵ-net for $\mathrm{RM}(m, r)$. Assuming we have found such a net, we can upper bound the number of low weight codewords by the number of codewords in each ball times the size of the net. The crux of the argument is to note that the number of codewords in each ball is upper bounded by $A_{m,r}(\leq 2\epsilon)$. This gives rise to a recursive approach whose base case is when the radius of the ball is smaller than half the minimum distance (and then there is at most one codeword in the ball). Formally, this idea is captured by the next simple observation.

Observation 2. Let $S \subseteq \mathcal{P}_{m,r}$ be a subset of polynomials with an ϵ-net $\mathcal{N}$. Then,

$$|S| \leq |\mathcal{N}| \cdot A_{m,r}(\leq 2\epsilon).$$

Thus, to get strong upper bounds on the number of low weight/bias polynomials we would like the ϵ-net to be as effective as possible. This means that on the one hand we would like the ϵ-net to be small and on the other hand that no ball around an element of the net should contain too many codewords.

We now explain the main idea of Kaufman *et al.* [84] for constructing the ϵ-net. For this we will need the notion of *discrete derivative*. The discrete derivative of a function $f\colon \mathbb{F}_2^m \to \mathbb{F}_2$ at direction $y \in \mathbb{F}_2^m$ is the function

$$\Delta_y f\colon x \mapsto \Delta_y f(x) \triangleq f(x+y) + f(x). \tag{4.2}$$

It is not hard to see that if $f(x)$ is a degree r polynomial then, for every y, $\Delta_y f(x)$ has degree at most $r-1$ as a polynomial in x. Another basic observation, that follows immediately from Equation (4.2), is that if a function $f\colon \mathbb{F}_2^m \to \mathbb{F}_2$ has relative-weight β, then, for each x,

$$\mathbb{P}_Y\left(\Delta_Y f(x) = f(x)\right) = 1-\beta,$$

where by Y we mean a random variable that is distributed uniformly over $\mathbb{F}_2^m$. Thus, if $\beta < 1/2$ then if we fix x and pick a random Y then with good probability $\Delta_Y f(x)$ gives a (somewhat) good estimate for $f(x)$. We are next going to use the Chernoff bound to move from a reasonable estimate to a very good approximation.

Theorem 4.1 (Chernoff's Inequality)**.** Let $X_1,\ldots,X_n \in \{0,1\}$ independent random variables such that $\mathbb{P}\left(X_i = 1\right) = p$. Then, for $0 < \epsilon < 1$

$$\mathbb{P}\left(\sum_i X_i \leq (1-\epsilon)pn\right) \leq \exp\left(-pn\epsilon^2/2\right).$$

For proofs see e.g., [103]. Hence, if for any t directions $y_1,\ldots,y_t \in \mathbb{F}_2^m$ we define

$$F_{y_1,\ldots,y_t}(x) \triangleq \text{Majority}\left(\Delta_{y_1} f(x),\ldots,\Delta_{y_t} f(x)\right),$$

then we get from the Chernoff bound that

$$\mathbb{E}_{X,Y_1,\ldots,Y_t}[\mathbf{1}[F_{Y_1,\ldots,Y_t}(X) \neq f(X)]] \approx \exp(-t).$$

Picking $t = O(\log 1/\epsilon)$ we see that for each codeword $\text{Eval}(f) \in \text{RM}(m,r)$ there is some $F_{y_1,\ldots,y_t}$ at hamming distance at most ϵ. Thus, the set of all such $F_{y_1,\ldots,y_t}$ forms an ϵ-net for polynomials of weight at most β in $\text{RM}(m,r)$. All that is left to do is to count the number of such functions $F_{y_1,\ldots,y_t}$ to obtain a bound on the size of the net. In fact, one can carry the same approach further and rather than approximating f by its first order derivatives, use instead higher order derivatives. For a set of k directions $\mathcal{Y} = \{y_1,\ldots,y_k\}$, where each $y_i \in \mathbb{F}_2^m$, we define[2]

$$\Delta_{\mathcal{Y}} f(x) = \Delta_{y_k}\Delta_{y_{k-1}}\cdots\Delta_{y_1} f(x).$$

The following version of Lemma 2.2 of [84] appeared in [134].

[2]It is not hard to see that the order of derivatives does not matter.

Lemma 1. Let $f\colon \mathbb{F}_2^m \to \mathbb{F}_2$ be a function such that $\mathrm{wt}_n(f) \le 2^{-k-1}$ for $k \ge 2$ and let $\epsilon > 0$. Then, there exist sets of k directions $\mathcal{Y}_1, \ldots, \mathcal{Y}_t$ such that[3]

$$\mathbb{P}\left(f(X) \neq \text{Majority}\left(\Delta_{\mathcal{Y}_1} f(X), \ldots, \Delta_{\mathcal{Y}_t} f(X)\right)\right) \le \epsilon,$$

where $t = \lceil 17 \log(1/\epsilon) \rceil$.

Proof Sketch. The proof is based on the following proposition that can be proved by induction

Proposition 1. Let $f\colon \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a function with $\mathrm{bias}(f) \le 2^{-k}$. Then,

$$(-1)^{f(x)} = \mathbb{E}_{Y \in (\mathbb{F}_2^m)^{k-1}} \left[\alpha_Y \cdot (-1)^{\Delta_Y f(x)}\right], \tag{4.3}$$

where,

$$\alpha_Y = \frac{1}{\mathrm{bias}(f) \cdot \mathrm{bias}\,\Delta_{y_1} f \cdots \mathrm{bias}\,\Delta_{y_{k-2}} \cdots \Delta_{y_1} f}.$$

This implies the following approximation scheme for f: sample from the distribution $\alpha_Y \cdot (-1)^{\Delta_Y f(x)}$ independently and take the sign of the average. Noting that $\alpha_Y < 3.5$, the upper bound then follows from Hoeffding's inequality that is given below. □

Theorem 4.2 (Hoeffding's Inequality)**.** Let $X_1, \ldots, X_t$ independent random variables where each X_i is supported on the interval $[a_i, b_i]$. Then,

$$\mathbb{P}\left(\frac{1}{t}\sum_{i=1}^{t} X_i - \mu \ge \epsilon\right) \le \exp\left(\frac{2\epsilon^2 t^2}{\sum_{i=1}^{t}(b_i - a_i)^2}\right),$$

with $\mu = \mathbb{E}\left[\frac{1}{t}\sum_{i=1}^{t} X_i\right]$.

Denote

$$F_{\mathcal{Y}_1,\ldots,\mathcal{Y}_t} \triangleq \text{Majority}\left(\Delta_{\mathcal{Y}_1} f, \ldots, \Delta_{\mathcal{Y}_t} f\right)$$

and

$$\mathcal{N}_{k,t} \triangleq \left\{F_{\mathcal{Y}_1,\ldots,\mathcal{Y}_t} \mid \mathcal{Y}_1, \ldots, \mathcal{Y}_t \in (\mathbb{F}_2^m)^k,\ f \in \mathcal{P}_{m,r} \text{ and } \mathrm{wt}_n(f) \le 2^{-k-1}\right\}.$$

Combining Observation 2 with recursive applications of Lemma 1, we obtain the following bound on the weight enumerator.

[3] Actually, we have to take a weighted majority, but for sake of clarity we ignore this detail in our presentation.

Corollary 1. Let $r, m, \ell \in \mathbb{N}$ such that $r \leq m$. Then,

$$A_{m,r}\left(\leq 2^{-\ell}\right) \leq |\mathcal{N}_{\ell-1,t}| \cdot A_{m,r}\left(2^{-\ell-1}\right),$$

where $t = 17(\ell+2)$. Consequently,

$$A_{m,r}\left(\leq 2^{-\ell}\right) \leq \prod_{j=\ell}^{r} \left|\mathcal{N}_{j-1,17(j+2)}\right|.$$

The way that Kaufman *et al.* [84] bounded the size of $\mathcal{N}_{k,t}$ was simply to say that each $F_{\mathcal{Y}_1,\ldots,\mathcal{Y}_t}$ is an explicit function of t polynomials of degree $r-k$ and hence the size of the net is at most $|\mathrm{RM}(m, r-k)|^t$. One idea in the improvement of [1] over [84] is that derivatives of polynomials can be represented as polynomials in fewer variables. Specifically, one can think of $\Delta_{\mathcal{Y}} f$ as a polynomial defined on the vector space $\mathrm{span}(\mathcal{Y})^{\perp}$. This allows for some saving in the counting argument, namely,

$$|\mathcal{N}_{k,t}| \leq 2^{mkt} \cdot |\mathrm{RM}(m-k, r-k)|^t,$$

where the term 2^{mkt} comes from the fact that now we need to explicitly specify the sets $\mathcal{Y}_1, \ldots, \mathcal{Y}_t$. The proof of Theorem 4 follows by using the bound from Lemma 2 in Corollary 1.

Lemma 2 (Implicit in the Proof of Theorem 3.3 of [1])**.** For any $k, t \in \mathbb{N}$ we have,

$$|\mathcal{N}_{k,t}| \leq \exp_2\left(mtk + t\binom{m-k}{\leq r-k}\right).$$

Proof Sketch. Each derivative $\Delta_{\mathcal{Y}} f$ can be expressed as a degree $r-k$ polynomial in $m-k$ variables. This is easy to see when $\mathcal{Y}$ is the first k coordinate vectors, but it is also true in general. The bound follows from simple counting. □

When $k = 1$, [134] further improved the upper bound by noting that different derivatives contain information about each other. I.e., they share monomials. This allowed them to get a better control of the amount of information encoded in the list of derivatives and as a result they obtained a better bound on the size of the net. This proved significant for bounding the number of codewords having small bias. See Lemma 4 below.

The discussion above relied on Lemma 1 that works for weights at most 1/4. For weights closer to 1/2 Kaufman *et al.* [84] changed their approach and instead of considering independent directions, they picked highly dependent directions, forming a subspace. Specifically, they pick a random subspace of dimension t and consider all $2^t - 1$ first order derivatives according to all nonzero directions in the subspace. As before each directional derivative gives a somewhat good approximation of the codeword but we can no longer use Chernoff's bound as the derivatives are not independent. Instead they observed that the directions are 2-wise independent and therefore they could use the Chebyshev bound to bound t as a function of the minimum distance of the RM code. Lemma 2.4 of [84] as stated in [134][4] gives:

Lemma 3 (Lemma 3.11 in [134], Lemma 2.4 in [84])**.** Let $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$ be a function such that $\mathrm{bias}(f) \geq \delta > 0$ and let $\epsilon > 0$. Then, for $t = \lceil \log(1/\epsilon) + 2\log(1/\delta) + 1 \rceil$, there exist directions $y_1, \ldots, y_t \in \mathbb{F}_2^m$ such that,

$$\mathbb{P}_X \left(f(X) = \mathrm{Majority}\left(\Delta_{\sum_{i \in I} y_i} f(X) \colon \emptyset \neq I \subseteq [t] \right) \right) \geq 1 - \epsilon.$$

Proof Sketch. The proof goes along similar lines to the proof of Lemma 1 only now we use Chebyshev's inequality rather than Chernoff's. □

Corollary 2. For any $t \in \mathbb{N}$ define,

$$\mathcal{B}_t = \left\{ \mathrm{Majority}\left(\Delta_{\sum_{i \in I} y_i} f(x) \mid \emptyset \neq I \subseteq [t] \right) : f \in \mathcal{P}_{m,r},\ y_1, \ldots, y_t \in \mathbb{F}_2^m \right\}.$$

Then, for $t = \lceil \log(1/\epsilon) + 2\log(1/\delta) + 1 \rceil$, $\mathcal{B}_t$ is an ϵ-net for $\{\mathrm{Eval}(f) \in \mathrm{RM}(m,r) \mid \mathrm{bias}(f) \geq \delta\}$.

To upper bound $|\mathcal{B}_t|$ we note that first order derivatives in directions $y_1, \ldots, y_t$ determine the derivatives in every direction within $\mathrm{span}\{y_1, \ldots, y_t\}$. Hence, $|\mathcal{B}_t| \leq |\mathcal{N}_{1,t}|$. Earlier we computed $|\mathcal{N}_{k,t}|$ for $k \geq 2$ so we need the following estimate.

[4] [84] only gave a rough upper bound on t and [134] did a more careful analysis to obtain this result.

Lemma 4. Let $m, r, t \in \mathbb{N}$ such that $t, r \leq m$ and write $\gamma = r/m$. Then,

$$|\mathcal{N}_{1,t}| \leq \exp_2\left(mt + \sum_{j=1}^{t} \binom{m-j}{\leq r-1}\right)$$
$$\leq \exp_2\left(mt + \left(1-(1-\tilde{\gamma})^t\right)\binom{m}{\leq r}\right),$$

where $\tilde{\gamma} = \gamma\left(1 + \frac{t}{m-t}\right)$.

Proof Sketch. For simplicity, after applying a linear transformation, we may assume without loss of generality that $y_i = e_i$ and so $\Delta_{y_i} f$ is just the formal derivative with respect to x_i. Therefore the sequence,

$$(\Delta_{y_1} f, \Delta_{y_2} f, \ldots, \Delta_{y_t} f)$$

is determined only by the monomials of f containing x_i for some $i = 1, \ldots, t$. Thus, if we count the number of monomials containing x_1, then those that contain x_2 but not x_1 etc. we get that there are exactly,

$$\sum_{j=1}^{t} \binom{m-j}{\leq r-1} = \binom{m}{\leq r} - \binom{m-t}{\leq r}$$

such monomials. Hence, there are at most $\exp_2\left(\binom{m}{\leq r} - \binom{m-t}{\leq r}\right)$ such distinct sequences. This estimate holds for fixed directions $y_1, \ldots, y_t \in \mathbb{F}_2^m$. In order to get an upper bound on $|\mathcal{N}_{1,t}|$, we need to take the union over all directions which gives another factor of 2^{mt}. The bound now follows from a careful estimate of binomial coefficients. □

Proof Sketch of Theorem 7. Observation 1 and Corollary 2 (with parameters $\epsilon = 2^{-\ell}$ and $\delta = 2^{-s}$) imply that

$$A_{r,m}\left(\frac{1-2^{-\ell}}{2}\right) \leq |\mathcal{B}_t| \cdot A_{r,m}\left(2^{-s+1}\right).$$

As $|\mathcal{B}_t| \leq |\mathcal{N}_{1,t}|$ the theorem follows from combining the estimate above with Lemma 4 and Theorem 4 (and a lot of calculations). □

To conclude, the ϵ-net approach works as follows. We first show that each polynomial of relative-weight at most β can be approximated

by an explicit function of some of its derivatives. We then count the number of such possible representations and then continue recursively to bound the number of codewords that are close to each such function.

4.2 Lower Bounds on the Weight Enumerator

The proofs of both Observation 1 and Theorem 9 are based on exhibiting a large set of polynomials having the claimed weight.

Observation 1 follows from the simple fact that, with probability $1/2$, for a polynomial $g(x_\ell, \ldots, x_m)$ sampled uniformly from $\mathcal{P}_{m-\ell,r-\ell+1}$, the degree r polynomial $f(x_1, \ldots, x_m) = x_1 \cdot x_2 \ldots x_{\ell-1} \cdot g$ will have relative-weight at most $2^{-\ell}$ (as half of the polynomials g have weight at most $1/2$).

Proof Sketch of Theorem 9. To prove Theorem 9 we consider all polynomials of the form

$$g(x_1, \ldots, x_m) = \sum_{i=1}^{\ell} x_i f_i(x_{i+1}, \ldots, x_{m-i}),$$

where $f_i \in \mathcal{P}_{m-i,r-1}$. It is not hard to see that different choices of $(f_1, \ldots, f_\ell)$ yield different polynomials $g(x) \in \mathcal{P}_{m,r}$. The main idea is proving that with probability at least $1/2$, over the choice of $\{f_i\}$, it holds that $\operatorname{bias} g \geq 2^{-\ell+1}$. As there are $\exp_2\left(\sum_{j=1}^{\ell} \binom{m-j}{\leq r-1}\right)$ such different polynomials g, the lower bound follows.

For $(a_1, \ldots, a_\ell) \in \mathbb{F}_2^\ell$ define

$$g\big|_{(a_1,\ldots,a_\ell)}(x_{\ell+1}, \ldots, x_m) = \sum_{i=1}^{\ell} a_i f_i(a_{i+1}, \ldots, a_\ell, x_{\ell+1}, \ldots, x_m).$$

Note that

$$\operatorname{bias} g = 2^{-\ell} + \mathbb{E}_{(Y_1,\ldots,Y_\ell)\neq(0,\ldots,0)}[\operatorname{bias} g\big|_{(Y_1,\ldots,Y_\ell)}], \tag{4.4}$$

where the $2^{-\ell}$ term comes from the case $(a_1, \ldots, a_\ell) = (0, \ldots, 0)$. Observe that for any $(a_1, \ldots, a_\ell) \neq (0, \ldots, 0)$, $g\big|_{(a_1,\ldots,a_\ell)}$ is a uniformly random polynomial, over the variables $x_{\ell+1}, \ldots, x_m$, of degree at most

$r-1$. From Theorem 8 and the union bound we get that with probability at least

$$1 - 2^{\ell} \cdot 2^{m} \exp\left(\left(-1 + H\left(\frac{1-2^{-\ell-1}}{2}\right)\right)\binom{m-\ell}{\leq r-1}\right) > 1/2,$$

it holds that $\text{bias}\, g|_{(a_1,\ldots,a_\ell)} > -2^{-\ell-1}$ for every $(a_1,\ldots,a_t) \neq (0,\ldots,0)$. Hence,

$$\mathbb{P}\left(\text{bias}\, g \geq 2^{-\ell-1}\right) \geq \mathbb{P}\Big(\text{bias}\, g|_{(a_1,\ldots,a_\ell)} \leq -2^{-\ell-1} \quad \forall (a_1,\ldots,a_\ell) \neq (0,\ldots,0)\Big) > 1/2. \qquad \square$$

4.3 The Analytical Approach

4.3.1 Boolean Analysis

Boolean analysis is the study of functions from the boolean hypercube to the complex numbers $\psi\colon \{0,1\}^n \to \mathbb{C}$. One of the most useful tools in studying boolean functions is the discrete Fourier transform.

Definition 8 (Character Functions)**.** For $\mathcal{S} \subseteq [n]$ denote the character function $\chi_{\mathcal{S}}(x) = (-1)^{\sum_{i\in S} x_i}$.

Definition 9 (Fourier Decomposition)**.** Let $\psi\colon \{0,1\}^n \to \mathbb{C}$ be a boolean function. Denote its Fourier transform via $\widehat{\psi}(\mathcal{S}) = \frac{1}{2^n} \cdot \sum_{x\in\{0,1\}^n}[f(x) \cdot \chi_{\mathcal{S}}(x)]$.

We identify that collections of subsets $\mathcal{P}([n]) = \{\mathcal{S} \mid \mathcal{S} \subseteq [n]\}$ with the boolean hypercube in the natural way and so the Fourier transform is also a boolean function.

As the characters form an orthonormal basis to the space of functions from the boolean cube to $\mathbb{C}$ with respect to the inner product $\langle \psi_1, \psi_2 \rangle = \mathbb{E}_{x\in\{0,1\}^n} \psi_1(x)\psi_2(x)$ we have that:

Proposition 2. Let $\psi_1, \psi_2\colon \{0,1\}^n \to \mathbb{C}$. Then,

$$\psi_1(x) = \sum_{\mathcal{S}\subseteq[n]} \widehat{\psi_1}(\psi)\chi_{\mathcal{S}}(x),$$

$$\langle \psi_1, \psi_2 \rangle = \sum_{\mathcal{S}\subseteq\{0,1\}^n} \widehat{\psi_1}(\mathcal{S})\widehat{\psi_2}(\mathcal{S}).$$

A very special and important class of boolean functions for which the theory of boolean analysis is particularly useful is that of indicators of monotone sets.

Definition 10 (Monotone Sets)**.** Let $x, y \in \{0,1\}^n$. We say that $x \leq y$ if $x_i \leq y_i$ for all $i \in [n]$. A subset $\Omega \subseteq \{0,1\}^n$ is monotone if $x \in \Omega$ and $x \leq y$ imply that $y \in \Omega$.

Definition 11 (Biased Measure)**.** We defined the p-biased measure of a vector

$$\mu_p(x) = p^{\sum_{i=1}^n x_i}(1-p)^{n-\sum_{i=1}^n x_i}.$$

For a subset $\Omega \subseteq \{0,1\}^n$ we define $\mu_p(\Omega) = \sum_{x\in\Omega} \mu_p(x)$.

One of the key concepts in the analysis of boolean functions is that of *pivotality* or *influence*. Intuitively speaking, the influence of a variable quantifies its influence on the value of the function. For simplicity, we shall define these concepts only for monotone sets. In that case, the influence can be described in terms of the *boundary* of a set.

Definition 12 (Boundary)**.** Let $\Omega \subseteq \{0,1\}^n$ be a monotone set. Define the boundary sets

$$\partial_j\Omega = \{x \mid (x_1,\ldots,x_{j-1},0,x_{j+1},\ldots,x_n) \notin \Omega \text{ and } (x_1,\ldots,x_{j-1},1,x_{j+1},\ldots,x_n) \in \Omega\}$$

Given the characteristic function $\psi\colon \{0,1\}^n \to \mathbb{C}$ of a monotone set $\Omega \subseteq \{0,1\}^n$, the influence of the variable x_j, with respect to the p-biased distribution, is defined $I^{(p)}(\psi) = \mu_p(\partial_j\Omega)$. In probabilistic terms, $\mu_p(\partial_j\Omega)$ equals the probability that flipping the variable x_j of a random input changes the value of the function (when x is chosen according to the p-biased measure). Since we only consider monotone sets, we shall use the more explicit notation $\mu_p(\partial_j\Omega)$ rather than $I^{(p)}(\psi)$.

Finally, we state and prove a lemma by Russo and Margulis.

Proposition 3 (Russo-Margulis Lemma)**.** For any monotone set $\Omega \subseteq \{0,1\}^n$

$$\frac{\mathrm{d}\mu_p(\Omega)}{\mathrm{d}p} = \sum_{i=1}^n \mu_p(\partial_i\Omega).$$

Proof. Since Ω is a monotone set, we have $x_i = 1$ for all $x \in \Omega \cap \partial_i \Omega$. Now let us consider a different measure $\mu_{p_1,\ldots,p_n}$ defined by

$$\mu_{p_1,\ldots,p_n}(x) = \prod_{i=1}^{n} (p_i^{x_i}(1-p_i)^{1-x_i})$$

for $x = (x_1, \ldots, x_n) \in \{0,1\}^n$. If we set $p_1 = \cdots = p_n = p$, then $\mu_{p_1,\ldots,p_n}$ is the same as μ_p. The measure $\mu_{p_1,\ldots,p_n}(\Omega)$ can be decomposed as follows:

$$\begin{aligned}\mu_{p_1,\ldots,p_n}(\Omega) &= \sum_{x\in\Omega\cap\partial_i\Omega} \mu_{p_1,\ldots,p_n}(x) + \sum_{x\in\Omega\setminus\partial_i\Omega} \mu_{p_1,\ldots,p_n}(x) \\ &= p_i \sum_{x\in\Omega\cap\partial_i\Omega} \prod_{j\neq i} (p_j^{x_j}(1-p_j)^{1-x_j}) + \frac{1}{2} \sum_{x\in\Omega\setminus\partial_i\Omega} \prod_{j\neq i} (p_j^{x_j}(1-p_j)^{1-x_j}).\end{aligned}$$

The last equality holds because (i) $x_i = 1$ for all $x \in \Omega \cap \partial_i \Omega$; (ii) the set $\Omega \setminus \partial_i \Omega$ can be partitioned into disjoint pairs such that the two elements in each pair only differ in the ith coordinate. Since the second term in the last line does not contain p_i, we have

$$\begin{aligned}\frac{\partial \mu_{p_1,\ldots,p_n}(\Omega)}{\partial p_i} &= \sum_{x\in\Omega\cap\partial_i\Omega} \prod_{j\neq i} \left(p_j^{x_j}(1-p_j)^{1-x_j}\right) \\ &= p_i \sum_{x\in\Omega\cap\partial_i\Omega} \prod_{j\neq i} \left(p_j^{x_j}(1-p_j)^{1-x_j}\right) \\ &\quad + (1-p_i) \sum_{x\in\Omega\cap\partial_i\Omega} \prod_{j\neq i} \left(p_j^{x_j}(1-p_j)^{1-x_j}\right) = \mu_{p_1,\ldots,p_n}(\partial_i\Omega).\end{aligned}$$

The last equality follows from the definition of the boundary $\partial_i \Omega$. Therefore,

$$\frac{\mathrm{d}\mu_p(\Omega)}{\mathrm{d}p} = \sum_{i=1}^{n} \frac{\partial \mu_{p_1,\ldots,p_n}(\Omega)}{\partial p_i}\bigg|_{p_1=\cdots=p_n=p} = \sum_{i=1}^{n} \mu_p(\partial_i\Omega).$$

This completes the proof of the lemma. □

A thorough and comprehensive introduction to the analysis of boolean functions can be found at [24], [112].

4.3.2 On the EXIT Function of RM Codes

Kudekar *et al.* [90], proved the following estimate on the EXIT function of RM codes with constant rate.

Theorem 10. Let $h(p)$ denote the EXIT function of the RM code $\mathrm{RM}(m, r)$ code with constant rate R. Then,

$$h(1 - R - o(1) - \epsilon) \leq 2^{-\Omega(\epsilon m \log(m))}.$$

Theorem 10 asserts that for values of p slightly smaller than $1 - R$ the EXIT function $h(p)$ is asymptotically 0. Also, for values of p larger than $1-R$ the EXIT function $h(p)$ is asymptotically 1. The latter follows from the channel capacity limit of the BEC and the fact that $h(p)$ is associated with the bit-error probability over the BEC (See Lemma 3.2). Thus, Theorem 10 suggests that the EXIT function of RM codes behaves like a step function jumping from 0 to 1 at $p = 1 - R$. This phenomenon is known as *sharp threshold* and is illustrated in Figure 4.1 for short block-lengths of RM codes.

We now proceed with a proof of Theorem 10. The following proposition gives a condition under which a monotone function has a sharp threshold.

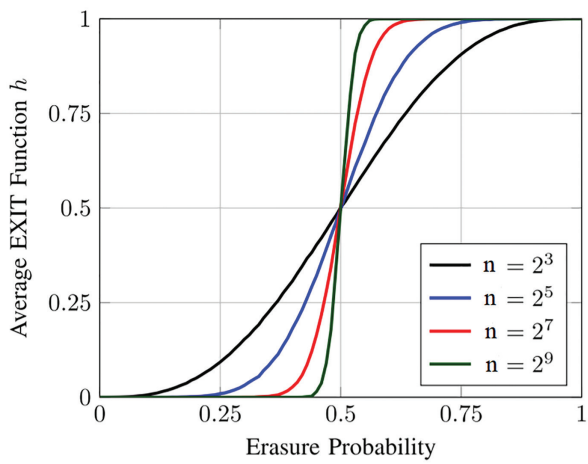


Figure 4.1: The average EXIT function of the rate-1/2 Reed-Muller code with blocklength n. This is originally Figure 1 in [90].

Proposition 4 (Lemma 34 in [90]). Assume a monotone function h: $[0,1] \to [0,1]$ satisfies

$$\forall p \in [a,b] \quad \frac{\mathrm{d}h}{\mathrm{d}p} \geq c \cdot h(p)(1-h(p)). \tag{4.5}$$

Then for any $0 < \epsilon_1 < \epsilon_2 \leq 1$

$$h^{-1}(\epsilon_2) - h^{-1}(\epsilon_1) \leq \frac{1}{c} \ln \frac{(1-\epsilon_1)\epsilon_2}{\epsilon_1(1-\epsilon_2)} + \delta(\epsilon_1, \epsilon_2)$$

where $\delta(\epsilon_1, \epsilon_2) = \max\{a - h^{-1}(\epsilon_1), 0\} + \max\{h^{-1}(\epsilon_2) - b, 0\}$. In particular if $[a,b] = [\delta, 1-\delta]$ then

$$h(h^{-1}(1/2) - \epsilon) \leq \exp(-c(\epsilon - 2\delta)), \text{ and } h(h^{-1}(1/2) + \epsilon) \leq \exp(-c(\epsilon - 2\delta)).$$

Proof Sketch. The idea is to consider the function $g(p) = \log \frac{h(p)}{1-h(p)}$. One can easily verify that

$$g'(p) = h(p)(1-h(p))$$

so $g'(p) \geq c \cdot h(p)(1-h(p))$ for all $p \in [a,b]$. The first assertion is obtained by integrating $g'(p)$ along ϵ_1 to ϵ_2. For the second assertion, use $\epsilon_1 = h(h^{-1}(1/2) - \epsilon)$ and $\epsilon_2 = 1/2$. □

To apply this to RM codes, Kudekar *et al.* [90] proved that the EXIT function of RM codes satisfies the requirement in the proposition. For this they also needed the following claim concerning monotone sets.

Theorem 11. Let $\Omega \subseteq \{0,1\}^n$ be a monotone set with equal influences $\mu_p(\partial_{j_1}\Omega) = \mu_p(\partial_{j_2}\Omega)$. Then,

$$\frac{\mathrm{d}\mu_p(\Omega)}{\mathrm{d}p} \geq (c(p) - o_n(1)) \cdot \ln(n) \cdot \mu_p(\Omega)(1-\mu_p(\Omega)),$$

where[5] $c(p) = \frac{1-2p}{p(1-p)\ln\left(\frac{1-p}{p}\right)}$.

Remark 3. In [59] Theorem 11 was proved with a different constant. The tighter bound using $c(p)$ was obtained in [124].

[5] At $p = 1/2$ the function $c(p)$ has a removable discontinuity and so we define $c(1/2) = \lim_{p \to 1/2} c(p) = 2$.

We can now prove that the EXIT function of RM codes satisfies the condition of Proposition 4. From this point on we shall restrict ourselves to the EXIT function of RM(m, r), and abusing notation, denote it by h: $[0, 1] \to [0, 1]$ suppressing the dependence on m and r.

Proposition 5. Let $h(p)$ denote the EXIT function of the RM code RM(m, r). Then,

$$\frac{\mathrm{d}h(p)}{\mathrm{d}p} \geq (c(p) - o(1)) \cdot \ln(n) \cdot h(p)(1 - h(p)),$$

where $c(p)$ is as in Theorem 11.

Proof. For each $z \in \mathbb{F}_2^m$ define the set

$$\Omega_z = \{y \in \{0,1\}^{n-1} \mid \exists f \in \mathcal{P}_{m,r} \text{ such that } (\mathrm{Eval}(f) \leq y_{z\to 1}) \text{ and } f(z) = 1\}, \tag{4.6}$$

where $y_{z\to 1}$ is a vector that equals y except that its z'th coordinate is set to 1 (it may be the case that $y = y_{z\to 1}$), and In terms of erasures, the set Ω_z consists of all erasure patterns over $\mathbb{F}_2 \setminus \{z\}$ for which the z'th coordinate cannot be decoded (given that the z'th coordinate is erased). Indeed, this follows from the following simple observation.

Observation 3. Let $\mathsf{C} \subset \mathbb{F}_2^n$ be a linear code. Then, an erasure pattern $\mathcal{S} \subseteq [n]$ can be corrected if and only if no codeword is supported on the pattern. I.e., there is no codeword $x \in \mathsf{C}$ such that $\mathrm{supp}(x) \subseteq \mathcal{S}$.

The last item in Lemma 3.2 implies that

$$h(p) = \mathbb{E}_z \mu_p(\Omega_z)$$

and by linearity of differentiation we see that

$$\frac{\mathrm{d}h}{\mathrm{d}p} = \mathbb{E}_z \left[\frac{\mathrm{d}\mu_p(\Omega_z)}{\mathrm{d}p}\right].$$

The proof is concluded using the following two claims.

Lemma 5. For any $z_1 \neq z_2$ we have $\mu_p(\Omega_{z_1}) = \mu_p(\Omega_{z_2})$.

Lemma 6. For any $z_1 \neq z_2$ and z we have $\mu_p(\partial_{z_1}\Omega_z) = \mu_p(\partial_{z_2}\Omega_z)$.

Proof of Lemma 5. Let $T\colon \mathbb{F}_2^m \to \mathbb{F}_2^m$ be a linear transformation satisfying $T(z_1) = z_2$. As composing a degree r polynomial with a linear transformation gives another degree r polynomial, it follows that the mapping

$$(x_z)_{z\in\mathbb{F}_2^m} \mapsto (x_{T(z)})_{z\in\mathbb{F}_2^m}$$

induces a bijection between $\partial_{z_1}\Omega_z$ and $\partial_{z_2}\Omega_z$. This bijection also preserves the measure of the set as $\mu_p((x_z)_{z\in\mathbb{F}_2^m}) = \mu_p((x_{T(z)})_{z\in\mathbb{F}_2^m})$. □

Proof of Lemma 6. Similar to the proof of Lemma 5, this time using an affine transformation T satisfying $T(z) = z$, $T(z_1) = z_2$. □

Lemma 5 implies that $\frac{\mathrm{d}h}{\mathrm{d}p} = \mu_p(\Omega_z)$ for any choice of $z \in \mathbb{F}_2^m$. Using an arbitrary z, Lemma 6 implies that Ω_z satisfies the assumption of Theorem 11, which concludes the proof. □

From Propositions 4 and 5 it follows that the EXIT function of RM codes has a sharp threshold. The area theorem now implies that the threshold point must be around $1 - R$. This is made quantitative in the following corollary (see e.g., Theorem 19 in [90]).

Corollary 3 (Implicit in [90])**.** For any $\epsilon \le 1/2$

$$h^{-1}(1/2) \ge 1 - R - (2\epsilon R + g(\epsilon)),$$

where $g(\epsilon) = h^{-1}(1-\epsilon) - h^{-1}(\epsilon)$.

Proof. Using the Area theorem (Lemma 3.2) we get $h^{-1}(1-\epsilon) \ge 1 - R + \frac{\epsilon R}{1-\epsilon}$ as

$$R = \int_0^1 h(p)\mathrm{d}p \ge \int_{h^{-1}(1-\epsilon)}^1 h(p)\mathrm{d}p \ge (1 - h^{-1}(1-\epsilon)) \cdot (1-\epsilon).$$

Also, by the definition of $g(\epsilon)$ we have

$$h^{-1}(\epsilon) = h^{-1}(1-\epsilon) - g(\epsilon).$$

Putting the two together we get

$$h^{-1}(\epsilon) \ge 1 - R - (2\epsilon R + g(\epsilon))$$

Using the monotonicity of the EXIT function (Lemma 3.2) the claim follows. □

Putting it altogether we obtain the following.

Corollary 4. Let $h(p)$ denote the EXIT function of the RM code $\mathrm{RM}(m,r)$ with constant rate R and $c(p)$ as in Theorem 11. Then,

$$h(1-R-o(1)-\epsilon) \leq n^{-k(1-R)\cdot\epsilon},$$

where $k(p) = \begin{cases} c(p)-o(1) & p<1/2 \\ c(1/2)-o(1) & p \geq 1/2 \end{cases}$.

In particular, $h(1-R-o(1)-\epsilon) = n^{-\Omega(1)}$.

Proof. Combining Propositions 4 and 5 we get

$$g(\epsilon) = h^{-1}(1-\epsilon) - h^{-1}(\epsilon) \leq \frac{2}{\ln(n)} \ln\left(\frac{1}{\epsilon}\right)$$

Using Corollary 3 with $\epsilon = \frac{\ln\ln(n)}{\ln(n)}$ we get

$$h^{-1}(1/2) \geq 1 - R - O\left(\frac{\ln\ln n}{\ln n}\right).$$

The result follows from Proposition 4, and the "in particular" part follows since $k(p) = \Omega(1)$. □

Note that Corollary 4 is weaker than Theorem 10. We proceed by giving the more sharper bound of Theorem 10. Let us start by highlighting the key property of RM codes underlying the preceding arguments - their *symmetry group*.

Definition 13. The symmetry group of a set $\Omega \subseteq \{0,1\}^n$ is

$$\mathrm{Sym}(\Omega) = \left\{\pi \in S_n \mid \forall x\ \pi(x_1,\ldots,x_n) \in \Omega \iff (x_{\pi(1)},\ldots,x_{\pi(n)}) \in \Omega\right\}.$$

The proof of Proposition 5 used that the symmetry group of Ω_z satisfies the following property: For any given indices $i_1, i_2, j_1, j_2 \in \mathbb{F}_2^m$ with $i_1 \neq i_2$ there exists a permutation $\pi \in \mathrm{Sym}(\Omega_z)$ satisfying $\pi(i_1) = j_1$ and $\pi(i_2) = j_2$. This property is called *double-transitivity*. The improvement of Theorem 10 over Corollary 4 follows by utilizing a more refined symmetry property of $\mathrm{Sym}(\Omega_z)$: It contains a subgroup isomorphic to $\mathrm{GL}_m(\mathbb{F}_2)$.

Claim 1. Let Ω_z be as in Equation (4.6) then $\mathrm{Sym}(\Omega_z)$ has a subgroup isomorphic to $\mathrm{GL}_m(\mathbb{F}_2)$ the group of all linear transformations.

Proof. Without the loss of generality $z = \overline{0}$ the zero vector. Any transformation $T \in \mathrm{GL}_m(\mathbb{F}_2)$ fixes the zero vector and $T \in \mathrm{Sym}(\mathrm{RM}(m,r))$. Thus, Ω_z is invariant under permuting the coordinates by T. □

In [32] Bourgain and Kalai proved a sharper estimate than the one in Theorem 11 that enables us to leverage this symmetry property.

Theorem 12 (Theorem 1 & Corollary 4.1 in [32])**.** Let $\Omega \subseteq \{0,1\}^n$ be a monotone set that its symmetry group contains a subgroup isomorphic to $\mathrm{GL}_m(\mathbb{F}_2)$ where $n = 2^m$. Then,

$$\forall p \in [\delta(m), 1-\delta(m)] \quad \frac{\mathrm{d}\mu_p(\Omega)}{\mathrm{d}p} \geq c \cdot m \ln(m) \cdot \mu_p(\Omega)(1-\mu_p(\Omega)),$$

for some universal constant $c > 0$ and $\delta(m) = m^{-\Omega(1)}$.

Applying Theorem 12 to the case of RM codes, [90] obtained an asymptotic improvement over Corollary 4.

Corollary 5. Let $h(p)$ denote the EXIT function of the RM code $\mathrm{RM}(m,r)$ of constant rate R. Then,

$$h(1-R-o(1)-\epsilon) \leq 2^{-\Omega(m \log(m) \cdot \epsilon)}.$$

Vanishing Rates. The statements of Theorem 10 and Corollary 4 are only meaningful for constant rate RM codes since otherwise the $o(1)$ might be larger than the rate R. A careful look at the analysis reveals that this $o(1)$ term can be replaced with $O(m^{-1} \ln m)$ in the case of Corollary 4, and $m^{-\Omega(1)}$ in the case of Theorem 10. Therefore, it is possible to derive similar statements for RM codes with the appropriate rates.

4.3.3 The Approach of Samorodnitsky

In [127] Samorodnitsky used boolean analysis to argue about the weight enumerator of certain error correcting codes, and particularly RM codes.

For brevity we describe the result for arbitrary binary linear codes and then continue with RM codes.

Samorodnitsky's argument is based upon an inequality on boolean functions. In order to state this inequality, we shall need a few definitions.

Definition 4.1 (Norm)**.** We define the ℓ_q norm of a boolean function $\psi\colon \{0,1\}^n \to \mathbb{C}$ by

$$||\psi||_q = \left(\mathbb{E}_x \left|\psi(x)\right|^q\right)^{1/q}.$$

Definition 14 (Noise Operator)**.** For $x \in \{0,1\}^n$ and $-1 \leq \rho \leq 1$, let $y \sim N_\rho(x)$ be a random element of $\{0,1\}^n$ with each coordinate y_i being i.i.d equal to x_i with probability $(1+\rho)/2$ and flipped with probability $(1-\rho)/2$. Let $\psi\colon \{0,1\}^n \to \mathbb{R}$ and $\rho \in [-1,1]$. Define the function $T_\rho\psi\colon \{0,1\}^n \to \mathbb{R}$ by

$$T_\rho\psi(x) = \mathbb{E}_{y\sim N_\rho(x)}\psi(y).$$

Definition 15 (Conditional Expectation Operator)**.** Let $\psi\colon \{0,1\}^n \to \mathbb{C}$ and $\mathcal{S} \subseteq [n]$ be a subset of coordinates. We define another function $\mathbb{E}[\psi|\mathcal{S}](x) = \mathbb{E}_{y_\mathcal{S}=x_\mathcal{S}}\psi(y)$. That is, we take the expectation of the value of ψ on inputs that are equal to x on the coordinate set $\mathcal{S}$.

Theorem 13. Let $\psi\colon \{0,1\}^n \to \mathbb{R}^{\geq 0}$ be a non-negative function, and $\rho \in (0,1)$. Then,

$$\log \|T_\rho\psi\|_2 \leq \mathbb{E}_{\mathcal{S}\sim\lambda(\rho)} \log \|\mathbb{E}[\psi|\mathcal{S}]\|_2\,,$$

where $\lambda(\rho) = \log(1+\rho^2)$ and $\mathcal{S} \sim \lambda$ is a random subset $\mathcal{S}$ of $[n]$ in which each element is included independently with probability λ.

Applying the inequality for the special case in which $\psi = \mathbf{1}_C$ is the indicator function of a binary linear code $\mathsf{C} \subseteq \mathbb{F}_2^n$ Samoronitsky obtained the following.[6]

[6]This statement first appeared in [76], which is obtained via the original inequality in [126], [127] and standard algebraic manipulations.

Theorem 14 ([76], [126], [127]). Let $\mathsf{C} \subseteq \mathbb{F}_2^n$ be a linear code, $0 \leq \lambda \leq 1$ and $A_i = A_{m,r}(i/n)$ denote the number of codewords of weight exactly i. Then,

$$\log\left(\sum_{i=0}^{n} A_i \cdot (2^{\lambda}-1)^i\right) \leq H(X|Y).$$

Here X and Y are random variables such that X is a random uniform codeword in C, and Y is the result of transmitting X over the channel $\mathrm{BEC}(\lambda)$.

We now briefly explain how Theorem 14 is derived from Theorem 13. Let us start by stating three elementary claims on boolean functions and linear subspaces.

Proposition 6 (See Chapter 2 in [112]). For any boolean function $\psi\colon \{0,1\}^n \to \mathbb{R}$ we have $\widehat{T_\rho \psi}(S) = \rho^{|S|} \cdot \hat{\psi}(S)$.

Proposition 7 (See Chapter 3 in [112]). For any linear subspace $\mathsf{C} \subseteq \mathbb{F}_2^n$ we have that $\widehat{\mathbf{1}_{\mathsf{C}}}(S) = \frac{\mathbf{1}_{\mathsf{C}^\perp}(S)}{|\mathsf{C}^\perp|}$.

Proposition 8 (See Proof of Proposition 1.3 in [127]). Let $\mathsf{C} \subseteq \mathbb{F}_2^n$ be a binary linear code and $\mathcal{S} \subseteq [n]$

$$\mathbb{E}[\mathbf{1}_{\mathsf{C}}|\mathcal{S}](x) = \begin{cases} 2^{|\mathcal{S}|-\dim(\mathsf{C}^\perp_{\mathcal{S}})} & \exists y \in \mathsf{C}^\perp \text{ such that } x_{\mathcal{S}} = y_{\mathcal{S}} \\ 0 & \text{otherwise} \end{cases}$$

Proof of Theorem 14. Let $\psi = |\mathsf{C}| \cdot \mathbf{1}_{\mathsf{C}^\perp}$ and $(A_0, \ldots, A_n)$ the weight enumerator of C. Using Propositions 6 and 7 we get (for $\lambda = \lambda(\rho) = \log(1+\rho^2)$)

$$\|T_\rho \psi\|_2^2 = \sum_{i=0}^{n} A_i \cdot (2^{\lambda}-1)^i.$$

Using Proposition 8 and Lemma 3.2 we get

$$\mathbb{E}_{\mathcal{S}\sim\lambda(\rho)} \log \|\mathbb{E}[\psi|\mathcal{S}]\|_2 = H(X|Y).$$

The claim now follows from Theorem 13. □

We now focus on RM codes $\mathsf{C} = \mathrm{RM}(m,r)$ with constant rate. It immediately follows from Theorem 14 that sufficiently strong bounds

on $H(X|Y)$ imply bounds on the weight distribution. Recall that by Lemma 3.2

$$H(X|Y) = n \cdot \int_0^{\lambda} h(p)\mathrm{d}p$$

where $h(p)$ is the EXIT function, which means that we can use the upper bounds from Section 4.3.2. As before we denote $A_i = A_{m,r}\,(i/n)$.

Corollary 6. Let $\mathrm{RM}(m,r)$ with constant rate $R \in (0,1)$. Then for sufficiently large m

$$\sum_{i=1}^{2^m} A_i (2^{1-R-o(1)} - 1)^i \leq 1.$$

In particular,

$$A_i \leq (2^{1-R-o(1)} - 1)^{-i}.$$

Proof. Apply Corollary 5 with $\epsilon = \frac{\log\log m}{\log m}$ to obtain

$$\sum_{i=0}^{2^m} A_i (2^{1-R-o(1)} - 1)^i \leq 2^m \cdot h(1-R-o(1)) \leq 2^{-\omega(m)}.$$

Removing the first term $A_0 = 1$ we get

$$\sum_{i=0}^{2^m} A_i (2^{1-R-o(1)} - 1)^i = o(1).$$

Thus, for sufficiently large m the RHS is bounded by 1. □

By combining Theorem 14 with *MacWilliams identity* – an identity that relates the weight enumerator of a code with that of its dual – and using the fact that the dual code of an RM code is another RM code (See Section 2.4), we get another estimate for the weight enumerator.

Proposition 9 (MacWilliams Identity [100])**.** Let $(a_0, \ldots, a_n)$ be the weight enumerator of a binary code C and $(b_0, \ldots, b_n)$ the weight enumerator of its dual $\mathsf{C}^{\perp}$ (i.e., there are exactly a_i codewords in C of weight i etc.). Then, for every $\theta \in \mathbb{R}$

$$\sum_{i=0}^{n} a_i \theta^i = \frac{1}{|\mathsf{C}^{\perp}|} \cdot \sum_{i=0}^{n} b_i (1-\theta)^i (1+\theta)^{n-i}.$$

Proof. Use Parseval's identity (Proposition 2) with the functions $\mathbf{1}_{\mathsf{C}}$ and $\theta^{\sum_{i=1}^{n} x_i}$. □

Proposition 10. Let $\mathrm{RM}(m,r)$ with constant rate $R \in (0,1)$. Then,

$$A_{m,r}\left(i/n\right) \leq 2^{(R+H(i/n)-1)n}.$$

for $i \in [(1-2^{R-1-o(1)})n, 2^{R-1-o(1)}n]$.

Proof. We start with the dual code $\mathrm{RM}(m,r)^{\perp} = \mathrm{RM}(m-r-1,m)$ which has rate $1-R$. It follows from Corollary 6 that for any $\lambda \leq R-o(1)$ the weight distribution of the dual code satisfies

$$\sum_{i=1}^{2^m} A_{m,m-r-1}\left(i/n\right) \cdot (2^{\lambda}-1)^i \leq 1.$$

Applying MacWilliams identity (Proposition 9), and using the fact that the dual code of an RM code is another RM code, we obtain that for all $\theta \leq 2^{\lambda}-1$

$$\sum_{i=1}^{2^m} A_{m,r}\left(i/n\right)(1-\theta)^i(1+\theta)^{n-i} \leq |\mathrm{RM}(m,r)|,$$

and so in particular

$$A_{m,r}\left(i/n\right) \leq \frac{|\mathrm{RM}(m,r)|}{\max_{0\leq\theta\leq 2^{R-o(1)}-1}(1-\theta)^i(1+\theta)^{n-i}},$$

Using elementary calculus we find that the optimal value of θ is

$$\max_{0\leq\theta\leq 2^{R-o(1)}-1}(1-\theta)^i(1+\theta)^{n-i} = \begin{cases} 2^{n(1-H(i/n))} & \theta \geq 2^{R-o(1)}-1 \\ (2-2^{R-o(1)})^i(2^{R-o(1)})^{n-i} & \theta < 2^{R-o(1)}-1 \end{cases}.$$

This completes the proof. Note that the bound obtained for $\theta < 2^{R-o(1)}-1$ matches the bound in Corollary 6 and so omitted. □

We remark an astonishing qualitative consequence of Proposition 10 – the weight enumerator of a constant rate RM code for weights sufficiently close to $n/2$ is as that of a random code. A random binary code C have

roughly $\frac{|\mathsf{C}|}{2^n} \cdot \binom{n}{i}$ codewords of weight i. Using the well-known estimate (e.g., by Stirling's formula)

$$2^{n \cdot H(i/n)} \leq O(\sqrt{n}) \cdot \binom{n}{i}$$

we see that

$$A_{r,m}(i) \leq 2^{O(m)} \cdot \binom{2^m}{i}$$

for $i \in [(1 - 2^{-R-o(1)}) \cdot 2^m, 2^{-R-o(1)} \cdot 2^m]$.

The estimates in Corollary 6 and Proposition 10 are illustrated in Figure 4.2. For readability, we plot the points (α, β) for which we have the bound $A_{r,m}(\alpha) \leq 2^{\beta n}$ where n is the block-length. The solid blue curve is the upper bound of Proposition 10, and matches the weight distribution of random codes (the dashed curve is a continuation of the solid blue curve, capturing the weight distribution of random graphs in the range where Proposition 10 does not apply). This bound holds roughly until weight $\alpha = 2^{R-1}$, from which we have the bound of Corollary 6 illustrated by the solid red curve.

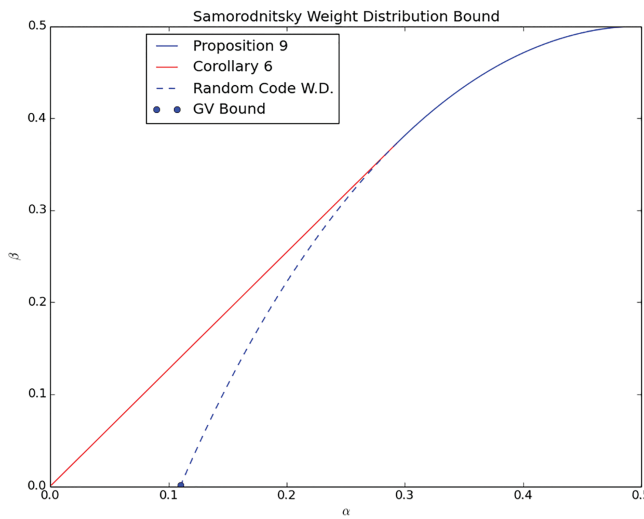


Figure 4.2: Samorodnitsky's Weight Distribution Bounds for RM code of rate $\frac{1}{2}$.

5

Bounding the Error Probability to Obtain Capacity-Achieving Results

In this section we survey results concerning the performance of RM codes on BMS channels, with a focus on the BEC and the BSC, at certain ranges of parameters. In particular we cover the result of [90] that prove that constant-rate RM codes achieve capacity for the BEC; the results of [1], [134] showing that RM codes of low- and high-rates achieve capacity for the BEC and BSC; and the results of [76], [128] that show that constant-rate RM codes can correct a constant fraction of random errors. Finally, we discuss a recent paper [122] which showed that constant-rate RM codes achieve capacity on all BMS channels under the bit-MAP decoding.

5.1 RM Codes Achieve Capacity at Low Rate [1], [134]

The results on the weight enumerator that were described in Section 4 and the bounds of Theorem 2 can be combined to prove that law- and high-rate RM codes achieve capacity for the BEC and the BSC. Moreover, it can show that, in many regimes of parameters, RM code can decode a number of errors that is close to (but does not match) the number of errors that capacity achieving codes of the same rate can decode from.

Note that in the case of rates close to 0 achieving capacity means the following: for the BEC achieving capacity means that we can correct a fraction $p \geq 1 - R(1+\epsilon)$ of random erasures (with high probability), and for the BSC achieving capacity means that we can correct a fraction p of random errors (with high probability) for p satisfying $H_2(p) \geq 1 - R(1+\epsilon)$. See [1] for a discussion of achieving capacity in extremal regimes.

5.1.1 BEC

In this section we relate the weight distribution of RM codes to the probability of correctly decoding from random erasures. In [1], [134] this was used to conclude that RM codes achieve capacity for the BEC at certain parameter range.

Theorem 15 (Theorem 1.9 from [134])**.** For any $r \leq m/50$, $\mathrm{RM}(m,r)$ achieves capacity on the BEC.

Proof Sketch. We need to prove that for any $\delta > 0$ it holds that $err(\mathrm{BEC}(p), \mathrm{RM}(m,r)) = o(1)$, where for $R = \binom{m}{\leq r}/2^m$, $p = (1-\delta)R$. According to Lemma 2,

$$err(\mathrm{BEC}(p), \mathrm{RM}(m,r)) \leq \sum_{\beta \neq 0} p^{\beta 2^m} \cdot A_{m,r}(\beta). \tag{5.1}$$

Thus, it is enough to show that $A_{m,r}(\beta)$ decays faster than $p^{\beta 2^m}$. In order to estimate the sum in Equation (5.1), [134] considered three different regimes:

- The typical case: Polynomials with an extremely small bias (including negative bias), i.e., all polynomials f satisfying $\mathrm{bias}(f) \leq \delta/8$.
- Relatively small bias: Polynomials with a not too large bias: $\delta/8 \leq \mathrm{bias}(f) \leq 3/4$.
- Low weight: Polynomials of weight $\mathrm{wt}_{\mathrm{n}}(f) \leq 1/8$.

For "typical" polynomials it is not hard to see (since the bias is so small) that $A_{m,r}(\beta)$ decays quickly (for $\beta \in [\frac{1-\delta/8}{2}, \frac{1+\delta/8}{2}]$) and hence their contribution to the sum in Equation (5.1) is negligible.

In the third case they partitioned the summation over β to the dyadic intervals $[2^{-k-1}, 2^{-k}]$ (for $k \geq 3$) and using the estimate from Theorem 4 proved that each such interval contributes to the overall sum a small quantity.

To bound the contribution of the polynomials satisfying the "relatively small bias" condition they again partitioned the interval to sub-intervals of the form $2^{-k} \leq \text{bias}(f) \leq 2^{-k+1}$, and used Theorem 7 to bound the contribution to the overall sum. In this case the calculations are more delicate but still give the required result. □

In addition, [134] shows that RM codes of higher degrees, specifically, RM codes of rate $\leq 1/\operatorname{poly}(\log n)$, can correct a fraction $1 - o(1)$ of random erasures. The proof is essentially the same only now we don't strive to achieve capacity and so we can take p to be a bit smaller, which makes the calculation work.

Theorem 16. For any $r < m/2 - \Omega\left(\sqrt{m \log m}\right)$, $\text{RM}(m, r)$ can efficiently correct a fraction of $1 - o(1)$ random erasures (as m increases).

While this result is not enough to deduce that the code achieves capacity, it shows that it is very close to doing that.

5.1.2 BSC

Similarly to what we did in Section 5.1.1, we shall bound the sum

$$err(\text{BSC}(p), \text{RM}(m, r)) \leq \sum_{\beta \neq 0} (2\sqrt{p(1-p)})^{\beta 2^m} \cdot A_{m,r}(\beta). \tag{5.2}$$

As in the case of the BEC, Sberlo and Shpilka partition the sum to small interval around weight $1/2$ and then to dadic interval of the form $[2^{-k-1}, 2^{-k}]$ and bound the contribution of each individual interval to the overall sum using Theorems 7 and 4, respectively. This leads the the following result.

Theorem 17 (Theorem 1.10 of [134])**.** For any $r \leq m/70$, $\text{RM}(m, r)$ achieves capacity on the BSC.

Similarly to the BEC case, they show that up to degrees close to $m/2$, RM codes can correct a fraction $1/2 - o(1)$ of random errors.

Theorem 18. For any $r < m/2 - \Omega\left(\sqrt{m \log m}\right)$ the maximum likelihood decoder for $\mathrm{RM}(m, r)$ can correct a fraction of $1/2 - o(1)$ random errors.

As in Theorem 16, this is not enough to show that RM codes achieve capacity at this range of parameters (as the $o(1)$ term is not the correct one), but it gives a good indication that it does.

5.2 RM Codes Achieve Capacity on the BEC at High Rate [1], [25]

In this section we explain the high level idea of the proof of [1], [25] that RM codes of very high degree achieve capacity for the BEC. Similarly to the case of $R \to 0$, we say that a family of codes of rate $R \to 1$ achieves capacity for the BEC if it can correct (with high probability) a fraction $p \geq (1 - \epsilon)(1 - R)$ of random erasures. Thus, for such a code to achieve capacity for the BEC it must hold that, with high probability, $(1 - (1 - \epsilon)(1 - R))\, n$ random rows of the generating matrix span the row space. This is equivalent to saying that a random subset of $(1 - \epsilon)(1 - R)n$ columns of the parity check matrix of the code has full rank (i.e., the columns are linearly independent) with high probability.

When dealing with very high rates (i.e., very high degrees) it is more convenient to use the notation $\mathrm{RM}(m, m - r - 1)$ (i.e., RM code of degree $m - r - 1$). As the parity check matrix of $\mathrm{RM}(m, m - r - 1)$ is the generating matrix of $\mathrm{RM}(m, r)$, the discussion above gives rise to the question that we consider next.

For an input $z \in \mathbb{F}_2^m$ and degree r denote with z^r the column of R_n corresponding to the evaluation point z (recall Equation (6.11)). In other words, z^r contains the evaluation of all multilinear monomials of degree at most r at z. Thus, the code $\mathrm{RM}(m, m - r - 1)$ achieves capacity for the BEC if it holds with high probability that a random subset $\mathcal{Z} \subset \mathbb{F}_2^m$, generated by adding each vector to the set uniformly and independently at random with probability $p = (1 - \epsilon)(1 - R)$, satisfies that the set $\{z^r \mid z \in \mathcal{Z}\}$ is linearly independent. This claim can be equally stated in terms of the dimension of the dual space. Indeed, a set $\{z_1^r, \ldots, z_K^r\}$ is linearly independent, if and only if the dual space has dimension $\binom{m}{\leq r} - K$. As one can think of vectors in the

dual space as coefficient vectors of degree r multilinear polynomials, we introduce the following notation. For a set $\mathcal{Z} = \{z_1, \ldots, z_K\} \subseteq \mathbb{F}_2^m$ let $\mathbb{I}_r(\mathcal{Z})$ denote the set of degree r multilinear polynomials that vanish on all points in $\mathcal{Z}$. Thus, the set $\{z^r \mid z \in \mathcal{Z}\}$ is linearly independent if and only if $\dim(\mathbb{I}_r(\mathcal{Z})) = \binom{m}{\leq r} - K$.

Abbe *et al.* [1] showed that for $r = o(\sqrt{m/\log m})$ and $K = (1-\epsilon) \cdot \binom{m}{\leq r}$ it holds that $\dim(\mathbb{I}_r(\mathcal{Z})) = \binom{m}{\leq r} - K$. Bhandari *et al.* [25] improved the results of [1] by proving the following theorem.

Theorem 19 (Theorem 1.1 of [25])**.** There exists a constant $\gamma_0 > 0$ such that for all $\epsilon > 0$ the following is true. Let $K = (1-\epsilon) \cdot \binom{m}{\leq r}$ where $r < \gamma_0 m$. Then

$$\mathbb{P}_{\mathcal{Z}}\left[\dim(\mathbb{I}_r(\mathcal{Z})) = \binom{m}{\leq r} - K\right] = 1 - o(1),$$

where $\mathcal{Z}$ is a a uniformly random set of K distinct points in $\mathbb{F}_2^m$.

The corollary below follows immediately from Theorem 19 and the discussion above.

Theorem 20 ([25])**.** Let γ_0 be the constant in Theorem 19. For $r < \gamma_0 m$, $\mathrm{RM}(m, m-r-1)$ achieves capacity on the BEC.

The idea behind the proof of Theorem 19 is to argue that with high probability, $|\mathbb{I}_r(\mathcal{Z})| = 2^{\binom{m}{\leq r}-K}(1+o(1))$. It is not hard to see that the probability that a uniformly random polynomial $f \in \mathcal{P}_{m,r}$ belongs to $\mathbb{I}_r(\mathcal{Z})$ is exactly $(1-\mathrm{wt}(f))^K$. We thus have

$$\mathbb{E}\left[|\mathbb{I}_r(\mathcal{Z})|\right] = \sum_{f \in \mathcal{P}_{m,r}} (1-\mathrm{wt}(f))^K. \tag{5.3}$$

As an overwhelming majority of the polynomials $f \in \mathcal{P}_{m,r}$ have weight close to $1/2$, we expect that the sum is close to $2^{\binom{m}{\leq r}-K}$ as required. To make this intuition work, [25] use the upper bounds proved in Theorems 4 and 7. As in the proof of Theorem 15, they partition the set of polynomials to polynomials with small bias and polynomials with large bias, and bound the contribution of each set separately to the sum in Equation (5.3), to conclude that the expectation is

$$\mathbb{E}\left[|\mathbb{I}_r(\mathcal{Z})|\right] = 2^{\binom{m}{\leq r}-K}(1+o(1)).$$

Finally, we note that it is enough to bound the expectation. Indeed, as $\mathbb{I}_r(\mathcal{Z})$ is a vector space defined by K linear equations, its dimension is always at least $\binom{m}{\leq r} - K$. Thus, if we denote $q = \mathbb{P}_{\mathcal{Z}}\left[\dim(\mathbb{I}_r(\mathcal{Z})) = \binom{m}{\leq r} - K\right]$ then

$$\mathbb{E}\left[|\mathbb{I}_r(\mathcal{Z})|\right] \geq q \cdot 2^{\binom{m}{\leq r}-K} + (1-q) \cdot 2^{1+\binom{m}{\leq r}-K}.$$

Hence,

$$q \cdot 2^{\binom{m}{\leq r}-K} + (1-q) \cdot 2^{1+\binom{m}{\leq r}-K} \leq \mathbb{E}\left[|\mathbb{I}_r(\mathcal{Z})|\right] = 2^{\binom{m}{\leq r}-K}(1+o(1))$$

and it follows that

$$\mathbb{P}_{\mathcal{Z}}\left[\dim \mathbb{I}_r(\mathcal{Z}) = \binom{m}{\leq r} - K\right] = q = 1 - o(1)$$

as claimed.

5.3 RM Codes Achieve Capacity on the BEC at Constant Rate [90]

In [90] the authors proved that constant rate RM codes achieve capacity for the BEC. The main technical effort in their work is to show that the EXIT function of constant rate RM codes admits a sharp threshold at the point of capacity. These bounds were already covered previously in Section 4.3.2, and we now conclude that constant rate RM codes achieve capacity for the BEC using these estimates.

At first, it seems that both bounds provided in Corollary 5 and Corollary 4 suffice to conclude that constant rate RM codes achieve capacity. However, there is a slight technicality - the EXIT function does not capture the error probability over the BEC in the classical sense of ML decoding, but rather bit-MAP decoding (See Section 3.5).

Proposition 11. We have the following inequality:

$$\begin{aligned} biterr(\mathrm{BEC}(p), \mathrm{RM}(m,r)) &\leq err(\mathrm{BEC}(p), \mathrm{RM}(m,r)) \\ &\leq \frac{2^m}{d} biterr(\mathrm{BEC}(p), \mathrm{RM}(m,r)) \end{aligned}$$

where $d = 2^r$ is the distance of $\mathrm{RM}(m,r)$.

Proof Sketch. The first inequality is immediate as if the ML decoder fails than at least one bit was not decoded. We proceed with the second inequality. If all but at most $d-1$ coordinates were decoded then the codeword is uniquely determined. By Markov's inequality this does not occur with probability at most

$$\frac{\mathbb{E}[\#\text{bits not decoded}]}{d}.$$

Recall that the expected number of bits that cannot be decoded is exactly

$$2^m \cdot biterr(\mathrm{BEC}(p), \mathrm{RM}(m,r))$$

by definition. This completes the proof. □

Theorem 21. Constant rate RM codes achieve capacity for the BEC.

Proof. Recall that $biterr(\mathrm{BEC}(p), \mathrm{RM}(m,r)) = ph(p)$ where h is the EXIT function of the RM code $\mathrm{RM}(m,r)$ (See Lemma 3.2). Thus, we can bound $h(p)$ using Corollary 5 and then the result follows from the inequality in Proposition 11. □

The above proof crucially depends on the sharp estimate of Bourgain-Kalai (Theorem 12) rather than the standard sharp threshold theorem (Theorem 11) as otherwise the bounds on the EXIT function are insufficient. While Theorem 11 is a "textbook result" in the field of boolean functions, Theorem 12 is considered to be notoriously difficult. Thus, it is desirable to have a proof that relies on Theorem 11 without resorting to the heavy machinery of the Bourgain-Kalai result. Such alternate proof was given in [91]. The proof is especially interesting because it utilizes both the combinatorial and analytical approach - the weight enumerator bounds from Section 4.1, and the EXIT function bounds from Section 4.3.2.

The following can be thought of as a hybrid between Proposition 11 and the Bhattacharyya bound for the BEC.

Proposition 12. Let $p \in (0,1)$, k a positive integer which is at most 2^m and $h(p)$ the EXIT function of $\mathrm{RM}(m,r)$. Then,

$$err(\mathrm{BEC}(p), \mathrm{RM}(m,r)) \leq \sum_{1 \leq i < k} A_{r,m}(i)\, p^i + \frac{2^m}{k} \cdot biterr(\mathrm{BEC}(p), \mathrm{RM}(m,r))$$

Proof Sketch. Without the loss of generality we may focus on the zero codeword. We shall analyze the probability in which the MAP decoder decodes to a nonzero $f \in \mathcal{P}_{m,r}$ in the case $\mathrm{awt}(f) \leq k$ and $\mathrm{awt}(f) > k$. If $\mathrm{awt}(f) \leq k$ then it is not hard to see that the probability in which the MAP decoder decodes to f is at most $p^{\mathrm{awt}(f)}$. Let us analyze the event in which the MAP decoder decodes to some $g \in \{f \in \mathcal{P}_{m,r} \mid \mathrm{awt}(f) \geq k\}$. In that case, it has to be that least k coordinates were not decoded successfully. By Markov's inequality this occurs with probability at most

$$\frac{\mathbb{E}[\#\text{bits not decoded}]}{k}.$$

Recall that the expected number of bits that cannot be decoded is exactly $2^m \cdot biterr(\mathrm{BEC}(p), \mathrm{RM}(m,r))$ then this event is bounded by

$$\frac{2^m}{k} \cdot biterr(\mathrm{BEC}(p), \mathrm{RM}(m,r)).$$

The error probability is then bounded by the sum of these events which concludes the proof. □

Lemma 7 (Lemmas 3, 4 in [91])**.** Let $\mathrm{RM}(m,r)$ RM code with constant $R \in (0,1)$ rate. Then, for any constants $z, \beta \in (0,1)$

$$\sum_{2^{-r} \leq \alpha \leq 2^{-\beta m}} z^{2^m \alpha} A_{r,m}(\alpha) = o(1) \tag{5.4}$$

Proof Sketch. Write $\alpha = 2^{-\ell}$ and $z^{2^m \alpha} = 2^{-\log(1/z) 2^{m-\ell}}$, and recall the estimate Theorem 4 on the weight distribution

$$A_{r,m}\left(2^{-\ell}\right) \leq 2^{O(m^4) + O(\ell) \cdot \binom{m-\ell}{\leq r-\ell}}.$$

Since β is constant, and the assumption on the rate we have $\frac{r-\ell}{m-\ell} \leq 1/2 - \Omega(\beta)$ so we can use the bound $\binom{m-\ell}{\leq r-\ell} \leq 2^{(m-\ell) H_2\left(\frac{r-\ell}{m-\ell}\right)}$. Thus, a

single summand in Equation (5.4) is upper bounded by

$$2^{-\log(1/z)2^{m-\ell}+O(m^4)+2^{(m-\ell)H_2\left(\frac{r-\ell}{m-\ell}\right)+\log(\ell)+O(1)}}.$$

Applying the Taylor series of the binary entropy around 1/2 we have

$$H_2\left(\frac{r-\ell}{m-\ell}\right) \leq 1 - \frac{1}{2\ln 2}\left(1-\frac{2(r-\ell)}{m-\ell}\right)^2$$

Neglecting the $O(m^4)$ and $\log(\ell) + O(1)$ terms, and noting that $H_2\left(\frac{r-\ell}{m-\ell}\right) < 1 - \Omega(\beta^2)$ we get that the term $-\log(1/z)2^{m-\ell}$ dominates the exponent. Taking into account that there are at most 2^m terms in the summand, then a trivial estimate for Equation (5.4) is

$$2^m \cdot z^{2^{m-r}}.$$

Using the constant rate assumption $m - r \geq m/2 + o(1)$, and as z is constant the above is clearly $o(1)$. □

We now put it altogether to obtain an alternate proof for Theorem 21 that does not rely on the Bourgain-Kalai sharp threshold estimate [32] but rather on the more conservative estimate in Theorem 11.

Alternate Proof for Theorem 21. First, recall that $ph(p)$ equals the bit-MAP error probability where $h(p)$ is the EXIT function of $\mathrm{RM}(m,r)$ (See Lemma 3.2). Applying Proposition 12 with $k = 2^{(1-\beta)m}$ for sufficiently small constant β that will be determined later

$$err(\mathrm{BEC}(p), \mathrm{RM}(m,r)) \leq \sum_{1\leq i<k} A_{r,m}(i)\, p^i + \frac{2^m}{k} h(p). \tag{5.5}$$

The first term can be estimated using Equation (5.4). For the second term, Corollary 4 implies that assuming $\mathrm{RM}(m,r)$ has constant rate R

$$h(1-R-o(1)) = n^{-\Omega(1)}$$

where $n = 2^m$ is the block-length. Therefore, the second term is bounded by $2^{(\beta-\Omega(1))m}$ and so for sufficiently small β this term is $o(1)$ as well. Together this shows that Equation (5.5) is $o(1)$ as required. □

5.4 RM Codes Achieve Capacity on BMS Channels at Constant Rate Under Bit-MAP Decoding [122]

Recently, Reeves and Pfister [122] proved that constant-rate RM codes achieve capacity on all BMS channels under bit-MAP decoding. Before explaining their proof, we would like to point out two differences between the result in [122] and the results discussed in previous subsections. The first difference is that the result in [122] holds for all BMS channels while the results in previous subsections only hold for some specific channels—either BEC or BSC. However, the generality of the result in [122] was obtained at the expense of a weaker bound on the decoding error probability. In all the previous subsections, when we say "achieving capacity" we mean the classical definition of "achieving capacity under block-MAP decoding", i.e., the code rate approaches channel capacity and the **block error probability** approaches 0. In contrast, [122] proved that RM codes "achieve capacity under bit-MAP decoding", which means that the code rate approaches channel capacity and the **bit error probability** approaches 0. Achieving capacity under bit-MAP decoding is weaker than achieving capacity under block-MAP decoding, as one usually needs $biterr(W, \mathsf{C}) = o(1/n)$ to show that $err(W, \mathsf{C}) = o(1)$, where n is the code length of C.

We first introduce some notation for general BMS channels. Rather than focusing on a specific BMS channel W, we consider a family of channels $\{W(t)\}$ indexed by a parameter $t \in [0, 1]$, where each $W(t)$ is a channel from a common input alphabet $\mathcal{X}$ to a common output alphabet $\mathcal{Y}$. We assume throughout that $t = 0$ is the perfect channel (i.e., $I(W(0)) = 1$) and $t = 1$ is the completely noisy channel (i.e., $I(W(1)) = 0$). We still use $X = (X_1, \ldots, X_n)$ to denote a codeword chosen uniformly at random from some code C. The channel output random vector $Y = (Y_1, \ldots, Y_n)$ is obtained as follows: Suppose that each X_i is transmitted through the channel $W(t_i)$ for some $t_i \in [0, 1]$, and Y_i is the corresponding channel output. Below we use the notation $Y(t_1, \ldots, t_n)$ to make the dependence on the channel parameters explicit. If $t_1 = t_2 = \cdots = t_n = t$, then we simply write $Y(t_1, \ldots, t_n)$ as $Y(t)$.

Since $H(X|Y(0)) = 0$ (the perfect channel) and $H(X|Y(1)) = H(X) = \dim(\mathsf{C})$ (the completely noisy channel), we have

$$\begin{aligned}\dim(\mathsf{C}) &= \int_0^1 \left\{ \frac{\mathrm{d}}{\mathrm{d}s} H(X \mid Y(s)) \right\}_{s=t} \mathrm{d}t \\ &= \sum_{i=1}^{n} \int_0^1 \underbrace{\left\{ \frac{\partial}{\partial s_i} H\left(X \mid Y\left(s_1, \ldots, s_n\right)\right) \right\}_{s_1=\cdots=s_n=t}}_{\text{GEXIT function of entry } i} \mathrm{d}t, \end{aligned} \tag{5.6}$$

where the generalized EXIT (GEXIT) function of the ith entry is defined as

$$\mathrm{GEXIT}_i(t) = \frac{\partial}{\partial s_i} H\left(X \mid Y\left(s_1, \ldots, s_n\right)\right) \Big|_{s_1=\cdots=s_n=t} \quad \text{for } t \in [0,1]. \tag{5.7}$$

For the rest of this subsection we suppose that the code C is a RM code. Then the symmetry and transitivity of RM codes imply that $\mathrm{GEXIT}_1(t) = \mathrm{GEXIT}_2(t) = \cdots = \mathrm{GEXIT}_n(t)$. In this case, Equation (5.6) becomes

$$\int_0^1 \mathrm{GEXIT}_i(t)\, \mathrm{d}t = \frac{\dim(\mathsf{C})}{n} = \mathrm{rate}\,(\mathsf{C}) \tag{5.8}$$

for all $i \in [n]$. This equality is known as the GEXIT area theorem.

The analysis of the GEXIT function is difficult due to its complicated definition. Instead, [122] chose to analyze the extrinsic MMSE function and proved that this function transitions quickly from 0 to 1 as $n \to \infty$. Since the extrinsic MMSE function is closely related to the GEXIT function, the GEXIT area theorem allows us to pin down the transition point of the the extrinsic MMSE function, which is precisely the capacity of the underlying BMS channel.

In order to define the extrinsic MMSE function, we need a slight modification on the code C. Previously, we have always assumed that $\mathsf{C} \subseteq \{0,1\}^n$, i.e., each coordinate in a codeword takes value in $\{0,1\}$. In this subsection we will assume that $\mathsf{C} \subseteq \{-1,1\}^n$. More precisely, we start with a standard binary code $\mathsf{C} \subseteq \{0,1\}^n$. Then for every codeword in C, we apply the following mapping to each of its coordinates:

$$0 \mapsto 1 \quad \text{and} \quad 1 \mapsto -1.$$

In this way, we obtain a code $\mathsf{C} \subseteq \{-1, 1\}^n$. Readers familiar with communication systems would recognize that the above mapping is simply the BPSK modulation.

We still use X to denote a random codeword, but this time each X_i takes value in $\{-1, 1\}$. The extrinsic MMSE function of the ith coordinate is defined as

$$M_i(t) := \mathbb{E}\left[\left(X_i - \mathbb{E}\left[X_i \mid Y_{\sim i}(t)\right]\right)^2\right], \quad t \in [0, 1],$$

where $Y_{\sim i}(t)$ is the vector obtained from discarding the ith coordinate in $Y(t)$. Again by the symmetry and transitivity of RM codes, we have $M_1(t) = \cdots = M_n(t)$. Sometimes we will omit the subscript i and simply denote the extrinsic MMSE by $M(t)$. The main technical contribution of [122] is the proof of the sharp threshold property of the extrinsic MMSE function. More precisely, for the code $\mathrm{RM}(m, r)$, [122] proved that

$$\int_0^1 M(t)(1 - M(t))\mathrm{d}t = O\left(\frac{\ln m}{\sqrt{m}}\right) \tag{5.9}$$

which implies that $M(t)$ cannot be too different from a step function that jumps from 0 to 1.

The proof of (5.9) starts with the following equality

$$M_i(t)\,(1 - M_i(t)) = \frac{1}{2}\mathbb{E}\left[\left(\mathbb{E}\left[X_i \mid Y_{\sim i}(t)\right] - \mathbb{E}\left[X_i \mid Y'_{\sim i}(t)\right]\right)^2\right], \tag{5.10}$$

where $Y'(t)$ is an independent second use of the channel with the same input X. For a subset $S \subseteq [n]$, define

$$\Delta_i^S(t) = \frac{1}{2}\mathbb{E}\left[\left(\mathbb{E}\left[X_i \mid Y_{\sim i}(t)\right] - \mathbb{E}\left[X_i \mid, Y^S_{\sim i}(t)\right]\right)^2\right]$$

where $Y^S(t)$ is a modified version of $Y(t)$ in which the entries indexed by S have been resampled according to the same input X. For a set with a single element, we simply write $\Delta_i^j(t)$ instead of $\Delta_i^{\{j\}}(t)$. An application of the Efron-Stein inequality to the right-hand side of (5.10) gives us

$$M_i(t)\,(1 - M_i(t)) \le \Delta_i^B(t) + \sum_{j \in A} \Delta_i^j(t), \quad \text{where } A = [n] \setminus (B \cup \{i\}).$$

The term $\Delta_i^B(t)$ is bounded as follows: One can show that if the random codeword X in the definitions of $M_i(t)$ and $\Delta_i^B(t)$ is chosen uniformly at random from the code $\mathrm{RM}(m,r)$, then for all integers $k \leq m$, there exists for each $i \in [n]$ a set $B \subset [n]\backslash i$ of size $2^m - 2^{m-k} - 1$ such that

$$\int_0^1 \Delta_i^B(t)\mathrm{d}t \leq 4\ln(2)(R(m,r) - R(m+k,r)), \tag{5.11}$$

where $R(m,r)$ is the rate of the $\mathrm{RM}(m,r)$ code. This inequality is proved via a clever use of the nesting property of RM codes, i.e., there are more than one copy of $\mathrm{RM}(m,r)$ codes embedded inside the longer code $\mathrm{RM}(m+k,r)$. The difference of code rates can be upper bounded by

$$R(m,r) - R(m+k,r) \leq \frac{3k+4}{5\sqrt{m}}.$$

Therefore, the right-hand side of (5.11) approaches 0 if we set $k = o(\sqrt{m})$.

The term $\Delta_i^j(t)$ is bounded as follows: One can show that if the input distribution has a doubly transitive symmetry group, then the following bound holds for all $i \neq j$,

$$\int_0^1 \Delta_i^j(t)\mathrm{d}t \leq \frac{4\ln 2}{n-1}$$

Finally, the bound (5.9) is obtained by setting $k = \lceil \frac{1}{2}\log_2(m) \rceil$ in (5.11), where k is the parameter that determines the size of the set B.

6

Polarization

Previous sections were concerned with estimating the weight enumerator of RM codes, using the analytical or combinatorial approach. In both cases, this is then used for the problem of evaluating the capacity of RM codes on a given channel by relying on upper-bounds on the error probability in terms of the weight enumerator. Therefore, the weight enumerator is used as a proxy to upper-bound the error probability of the code. We now switch to a different type of proxy to bound the error probability, namely, information measures.

6.1 Information Inequalities, Entropy Conservation and Polarization

We start by recalling a few basic properties. Consider a random variable X taking value in a finite set $\mathcal{X}$. If one had to guess the outcome of X, and if the benchmark is to minimize the probability of getting the wrong guess, which we denote $err(X)$, then the optimal rule is obviously to declare the most likely outcome of X, i.e.,

$$\hat{x}^{\mathrm{MAP}} = \mathrm{argmax}_{x \in \mathcal{X}} \, \mathbb{P}_X(x).$$

The tie is broken arbitrarily if there are multiple maximizers. This leads to the error probability

$$err(X) = 1 - \mathbb{P}_X(\hat{x}^{\mathrm{MAP}}) = 1 - \max_{x \in \mathcal{X}} \mathbb{P}_X(x).$$

Lemma 8. Let X be a discrete random variable taking values in a finite set $\mathcal{X}$. Then $err(X) \leq H(X)$.

Proof. The proof is divided into two cases. **Case (i)**: Suppose that $err(X) \leq 1/2$. Then

$$\begin{aligned} H(X) &= \sum_{x \in \mathcal{X}} \mathbb{P}_X(x) \log_2 \frac{1}{\mathbb{P}_X(x)} \\ &\geq \sum_{x \in \mathcal{X}, x \neq \hat{x}^{\mathrm{MAP}}} \mathbb{P}_X(x) \log_2 \frac{1}{\mathbb{P}_X(x)} \\ &\geq \sum_{x \in \mathcal{X}, x \neq \hat{x}^{\mathrm{MAP}}} \mathbb{P}_X(x) \log_2 \frac{1}{1 - \mathbb{P}_X(\hat{x}^{\mathrm{MAP}})} \\ &= (1 - \mathbb{P}_X(\hat{x}^{\mathrm{MAP}})) \log_2 \frac{1}{1 - \mathbb{P}_X(\hat{x}^{\mathrm{MAP}})} \\ &= err(X) \log_2 \frac{1}{err(X)} \geq err(X), \end{aligned}$$

where the last inequality follows from the assumption $err(X) \leq 1/2$. **Case (ii)**: Suppose that $err(X) > 1/2$. Then

$$\begin{aligned} H(X) = \sum_{x \in \mathcal{X}} \mathbb{P}_X(x) \log_2 \frac{1}{\mathbb{P}_X(x)} \geq \sum_{x \in \mathcal{X}} \mathbb{P}_X(x) \log_2 \frac{1}{\mathbb{P}_X(\hat{x}^{\mathrm{MAP}})} \\ = \log_2 \frac{1}{\mathbb{P}_X(\hat{x}^{\mathrm{MAP}})} = \log_2 \frac{1}{1 - err(X)} > 1 \geq err(X). \end{aligned}$$

This completes the proof of the lemma. □

Consider now two random variables X, Y taking values in the set $\mathcal{X} \times \mathcal{Y}$ according to the joint distribution $\mathbb{P}_{X,Y}$. Suppose that Y is observed and we want to guess the value of X given the observation of Y. When we observe $Y = y$, the optimal guess is

$$\hat{x}^{\mathrm{MAP}}(y) = \mathrm{argmax}_{x \in \mathcal{X}} \, \mathbb{P}_{X|Y}(x|y).$$

The tie is broken arbitrarily if there are multiple maximizers. The corresponding error probability is

$$err(X|Y=y) = 1 - \mathbb{P}_{X|Y}(\hat{x}^{\mathrm{MAP}}(y)|y) = 1 - \max_{x\in\mathcal{X}} \mathbb{P}_{X|Y}(x|y).$$

We further define

$$err(X|Y) = \sum_{y\in\mathcal{Y}} err(X|Y=y)\mathbb{P}_Y(y).$$

Lemma 8 immediately implies that $err(X|Y=y) \le H(X|Y=y)$ for all $y \in \mathcal{Y}$. Therefore,

$$err(X|Y) \le H(X|Y). \tag{6.1}$$

This inequality also holds when X and Y are random vectors.

Now we can apply this to the setting where $X^n = (X_1, \ldots, X_n)$ is a codeword[1] drawn uniformly at random in some code C, such as $\mathrm{RM}(m, r)$, and $Y^n = (Y_1, \ldots, Y_n)$ is the output random vector of X^n through a given BMS channel. Inequality (6.1) implies that if $H(X^n|Y^n) \approx 0$, then the block-MAP decoding error probability $err(X^n|Y^n)$ is also close to 0. A "good" code should have decoding error probability approaching 0 while containing as many codewords as possible. Since $H(X^n|Y^n)$ is an upper bound on the decoding error probability, and $H(X^n) = \log_2(|\mathsf{C}|)$ measures the number of codewords in C, the above condition for "good" codes translates into

$$H(X^n|Y^n) \text{ vanishing while } H(X^n) \text{ as large as possible.} \tag{6.2}$$

For the case of linear codes, the (random) codeword X is generated from a $k \times n$ generator matrix $G_{k\times n}$. More precisely, $(X_1, \ldots, X_n) = (U_1, \ldots, U_k)G_{k\times n}$, where $(U_1, \ldots, U_k)$ is the message vector consisting of i.i.d. Bernoulli-1/2 components. In this setting, the requirement in (6.2) becomes $H(U_1, \ldots, U_k|Y)$ vanishing while k as large as possible.

One way to construct a good generator matrix $G_{k\times n}$ is first constructing an $n \times n$ square matrix G_n that is full rank. Then pick k rows from G_n to form the matrix $G_{k\times n}$. This time we use a message vector of length

[1] In previous sections, we use X to denote the vector $(X_1, \ldots, X_n)$. Here we use X^n to follow the notation in the polar coding literature.

n, denoted by $U^n = (U_1, \ldots, U_n)$, still consisting of i.i.d. Bernoulli-1/2 components. The (random) codeword $X^n = (X_1, \ldots, X_n)$ is given by $X^n = U^n G_n$. Since G_n is invertible, $X_1, \ldots, X_n$ are also i.i.d. Bernoulli-1/2 random variables. As a consequence, $H(X^n|Y^n) = nH(X_1|Y_1)$. Therefore,

$$\sum_{i\in[n]} H(U_i|Y^n, U^{i-1}) = H(U^n|Y^n) = H(X^n|Y^n) = nH(X_1|Y_1), \quad (6.3)$$

where U^{i-1} is the shorthand notation for $(U_1, U_2, \ldots, U_{i-1})$. The conditional entropy $H(U_i|Y^n, U^{i-1})$ has a natural connection to the successive decoder, which decodes the message bits one by one from U_1 to U_n. When decoding U_i, the successive decoder already knows the values of all previous message bits in U^{i-1} and all the channel outputs in Y. By (6.1), $H(U_i|Y^n, U^{i-1})$ is an upper bound on the decoding error probability of the successive decoder when it decodes U_i. In light of (6.3), we can not hope that $H(U_i|Y^n, U^{i-1})$ vanishes for all $i \in [n]$. In fact, since $H(U_i|Y^n, U^{i-1}) \leq 1$ for all i, the conditional entropy $H(U_i|Y^n, U^{i-1})$ is not vanishing for at least $nH(X_1|Y_1)$ values of i. Therefore, $H(U_i|Y^n, U^{i-1}) \approx 0$ for at most $n - nH(X_1|Y_1)$ values of i, and this happens if and only if the conditional entropies *polarize*, i.e., for all but a vanishing fraction of $i \in [n]$,

$$H(U_i|Y^n, U^{i-1}) \text{ is very close to either 0 or 1.}$$

More formally, polarization means that for some $\epsilon = o(1/n)$,

$$|\{i \in [n]\colon H(U_i|Y^n, U^{i-1}) \in (\epsilon, 1-\epsilon)\}| = o(n).$$

Now denoting the set of "deterministic" components by $D_{\epsilon,n} := \{i \in [n]\colon H(U_i|Y^n, U^{i-1}) < \epsilon\}$, the above polarization condition together with (6.3) implies that

$$|D_{\epsilon,n}| = n(1 - H(X_1|Y_1)) - o(n).$$

Moreover, we have

$$\begin{aligned} &err((U_i, i \in D_{\epsilon,n}) \mid Y^n, (U_i, i \notin D_{\epsilon,n})) \\ &\quad \leq H((U_i, i \in D_{\epsilon,n}) \mid Y^n, (U_i, i \notin D_{\epsilon,n})) \\ &\quad \leq \sum_{i\in D_{\epsilon,n}} H(U_i|Y^n, U^{i-1}) \leq \epsilon |D_{\epsilon,n}| = o(1), \end{aligned}$$

where the first inequality follows from (6.1), the second inequality follows from the fact that conditioning on more variables reduces entropy, and the last equality follows from the assumption $\epsilon = o(1/n)$. Thus, we can pick the rows of G_n with row indices in $D_{\epsilon,n}$ to form the generator matrix $G_{k\times n}$. The resulting code has a vanishing error probability and code rate $1 - H(X_1|Y_1) - o(1)$. Since $1 - H(X_1|Y_1) = I(W)$ is the channel capacity of the underlying BMS channel W, this gives us a capacity-achieving code.

The benefit of working in the "full rank" setting described in the previous paragraphs is that the total entropy is preserved via the linear transform G_n; see (6.3). Under this invariant, if some of the individual entropies $H(U_i|Y^n, U^{i-1})$ go up, others must go down. In particular, this gives hope to use recursive arguments where individual entropies get pushed iteratively towards the extreme. In the case of the BEC channel, such conditional entropies have an explicit algebraic interpretation in terms of "conditional ranks" [5], but for general channels, these quantities do not have an explicit relation to the algebraic properties of the code.

We next discuss such polarization processes, underlying how the polar codes and RM codes ordering are different but related. In a nutshell, both polar codes and RM codes are derived from the same full rank matrix

$$G_n := \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^{\otimes m},$$

where $n = 2^m$. This matrix can also be viewed, up to a permutation of the rows, as the matrix obtained by evaluating all multivariate monomials on the m-dimensional Boolean hypercube. See Section 2.3 for a discussion on this. Thus, one can have different orderings from the same full rank matrix, and each of them can polarize. We also know that not all orderings polarize; for instance, proceeding from the nth row down to the first row with G_n leaves all conditional entropies equal to the original $H(X_1|Y_1)$. Polar codes give one specific ordering that polarizes by simply taking the increasing ordering of the row indices in G_n. Another natural ordering is the one corresponding to the increasing weights of the rows in G_n, which seems relevant for RM codes since

RM codes select rows of large weights. In fact, as we will explain in Section 6.3, the code obtained from this ordering for a given channel is equivalent to RM codes if and only if RM codes achieve capacity on this channel.

6.2 Polar Codes Polarize and Achieve Capacity

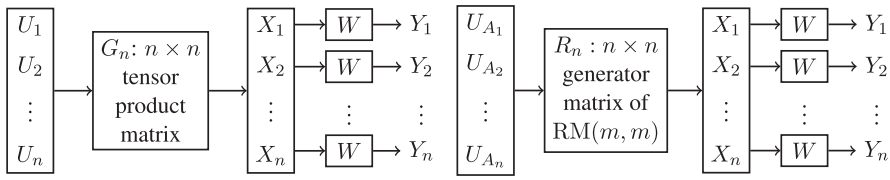
We now consider an arbitrary BMS channel W. We use the communication model in Figure 6.1 to transmit information over W. As in the last subsection, the input vector U^n consists of n i.i.d. Bernoulli-1/2 random variables. We encode U^n by multiplying it with an $n \times n$ invertible matrix G_n and denote the resulting vector as $(X_1, \ldots, X_n) = (U_1, \ldots, U_n)G_n$. Then we transmit each X_i through an independent copy of W. Given the channel output vector Y^n, our task is to recover the input vector U^n, and we use a successive decoder to do so. The successive decoder decodes the input vector bit by bit from U_1 to U_n. When decoding U_i, it makes use of all the channel outputs Y^n and all the previously decoded[2] inputs U^{i-1}. The conditional entropy

$$H_i := H(U_i|U^{i-1}, Y^n)$$

indicates whether U_i is noisy or noiseless under the successive decoder: If $H_i \approx 0$, then U_i is (almost) noiseless and can be correctly decoded with high probability. If H_i is bounded away from 0, then so is the decoding error probability of decoding U_i.

As mentioned in the last subsection, we say that the matrix G_n polarizes if almost all $H_i, 1 \leq i \leq n$ are close to either 0 or 1, or equivalently, if almost all $U_i, i \leq i \leq n$ become either noiseless or completely noisy. As discussed in the previous section, an important consequence of polarization is that every polarizing matrix automatically gives us a capacity-achieving code under the successive decoder. Indeed, if G_n polarizes, then we can construct the capacity achieving code by putting all the information in the U_i's whose corresponding H_i is close to 0 and freezing all the other U_i's to be 0, i.e., we only put information in the (almost) noiseless U_i's. Let $\mathcal{G}$ be the set of indices of the noiseless

[2] Assuming no decoding errors up to U^{i-1}.



(a) Polarization framework

(b) Polarization framework for RM codes

Figure 6.1: Polarization framework: The input vector U^n are i.i.d. Bernoulli(1/2) random variables. Given the channel output vector Y^n, we use successive decoder to decode the input vector U^n one by one from top to bottom.

U_i's. Then the generator matrix of this code is the submatrix of G_n obtaining by retaining only the rows whose indices belong to $\mathcal{G}$. To show that this code achieves capacity, we only need to argue that $|\mathcal{G}|$ is asymptotic to $nI(W)$, and this directly follows from

$$\sum_{i=1}^{n} H_i = H(U^n|Y^n) = H(X^n|Y^n) = nH(X_i|Y_i) = n(1 - I(W)), \quad (6.4)$$

where the last equality relies on the assumption that W is symmetric. Since almost all H_i's are close to either 0 or 1, by the equation above we know that the number of H_i's that are close to 1 is asymptotic to $n(1 - I(W))$, so the number of H_i's that are close to 0 is asymptotic $nI(W)$, i.e., $|\mathcal{G}|$ is asymptotic to $nI(W)$.

In his influential paper [11], Arıkan gave an explicit construction of a polarizing matrix

$$G_n := \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^{\otimes m},$$

where $\otimes$ is the Kronecker product and $n = 2^m$. For example,

$$G_4 := \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^{\otimes 2} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

Polar codes are simply the capacity-achieving codes constructed from G_n. More precisely:

Theorem 22 (Polarization for G_n)**.** For any BMS channel W and any $0 < \epsilon < 1/2$,

$$|\{i \subseteq [2^m] : H_i \in (\epsilon, 1-\epsilon)\}| = o(2^m). \tag{6.5}$$

Here we give a brief explanation about why this theorem holds without delving into technical details. When $m = 1$ and $n = 2$, we have $G_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$. Define $H(W) := 1 - I(W)$. The chain rule of conditional entropy gives

$$2H(W) = H(U_1|Y_1, Y_2) + H(U_2|Y_1, Y_2, U_1).$$

Since $H(U_2|Y_1, Y_2, U_1) \leq H(U_2|Y_2) = H(W)$, we can create two synthetic channels $W^-\colon U_1 \to (Y_1, Y_2)$ and $W^+\colon U_2 \to (Y_1, Y_2, U_1)$ such that $H(W^-) \geq H(W) \geq H(W^+)$. More precisely, let $\Delta := H(W) - H(W^+)$. Then we have

$$H(W^-) = H(W) + \Delta \geq H(W) \geq H(W^+) = H(W) - \Delta, \tag{6.6}$$

and the above inequalities are strict unless $\Delta = H(W) - H(W^+) = 0$. It has been established that $\Delta = 0$ if and only if $H(W) = 0$ or 1, i.e., W is either noiseless or completely noisy. In fact, a quantitative version of this claim is known:

Lemma 9 (Lemma 2.2 of [132])**.** Let W be a BMS channel. For any $\epsilon \in (0, 1/2)$, there exists a constant $\delta(\epsilon) > 0$, which only depends on ϵ and does not depend on W, such that

$$H(W) \in (\epsilon, 1-\epsilon) \implies \Delta = H(W) - H(W^+) = H(W^-) - H(W) > \delta(\epsilon). \tag{6.7}$$

Therefore, unless the initial channel W is noiseless or completely noisy, the synthesized channel W^+ is strictly better than W, and the synthesized channel W^- is strictly worse than W.

When we move to $m = 2$ and $n = 4$, we can create four synthetic channels $W^{--}, W^{-+}, W^{+-}, W^{++}$ corresponding to the channels mapping U_i to (Y^4, U^{i-1}) where the binary expansion of i (mapping 0 to $-$ and 1 to $+$) gives the channel index. Note that the behavior of these channels at $m = 2$ can be related to that at $m = 1$, as suggested by the

notation $W^{s_1 s_2}$, $s_1, s_2 \in \{-, +\}$. Namely, the two channels (W^{*-}, W^{*+}) are the synthesized channels obtained by composing two independent copies of W^*, $* \in \{-, +\}$ with the transformation G_2, as done for $m = 1$.

This recursive structure continues to hold as we increase the value of m. Namely, each time by increasing the value of m by 1, we produce twice more synthetic channels which are the $+$ and $-$ versions of the original synthetic channels. For each synthetic channel we have

$$H(W^{s-}) = H(W^s) + \Delta_s \geq H(W^s) \geq H(W^{s+}) = H(W^s) - \Delta_s, \tag{6.8}$$

with $\Delta_s := H(W^s) - H(W^{s+})$, $s \in \{-, +\}^m$. This follows by the inductive property of G_{2n}:

$$G_{2n} = \begin{bmatrix} G_n & 0 \\ G_n & G_n \end{bmatrix}.$$

Note that if W is a BMS channel, then each synthetic channel W^s is still a BMS channel. Therefore, Lemma 9 implies that for every $\epsilon \in (0, 1/2)$ and every $s \in \{-, +\}^m$,

$$\begin{aligned} H(W^s) \in (\epsilon, 1-\epsilon) \implies \Delta_s &= H(W^s) - H(W^{s+}) \\ &= H(W^{s-}) - H(W^s) > \delta(\epsilon), \end{aligned} \tag{6.9}$$

where $\delta(\epsilon)$ is a universal constant that is independent of s. The polarization result is then a consequence of this recursive process: if one tracks the entropies of the 2^m channels at level m, at the next iteration, i.e., at level $m + 1$, one breaks symmetrically each of the previous entropies into one strictly lower and one strictly larger value, as long as the entropies at level m are not extremal, i.e., not 0 or 1. Therefore, extremal configurations are the only stable points of this process, and most of the values end up at these extremes, as stated by Theorem 22. One way to prove this is based on the martingale convergence theorem, together with the fact that the only two fixed points of the $-, +$ transforms are at the extremes, or conversely, that for values other than 0 and 1, a strict movement takes place as in (6.9).

Here we give an elementary derivation of Theorem 22 without using the martingale convergence theorem. To prove Theorem 22, it suffices to show that

$$\lim_{m\to\infty} \frac{1}{2^m} \sum_{s\in\{-,+\}^m} H(W^s)(1-H(W^s)) = 0. \tag{6.10}$$

Define $\Gamma_m := \frac{1}{2^m}\sum_{s\in\{-,+\}^m} H(W^s)(1-H(W^s))$. We first prove that $\Gamma_{m+1} \le \Gamma_m$ for all m, i.e., Γ_m is decreasing with m. By definition,

$$\begin{aligned}
\Gamma_{m+1} &= \frac{1}{2^{m+1}} \sum_{s\in\{-,+\}^m} [H(W^{s+})(1-H(W^{s+})) \\
&\quad + H(W^{s-})(1-H(W^{s-}))] \\
&= \frac{1}{2^{m+1}} \sum_{s\in\{-,+\}^m} [H(W^{s+}) + H(W^{s-}) \\
&\quad - (H(W^{s+}))^2 - (H(W^{s-}))^2] \\
&= \frac{1}{2^{m+1}} \sum_{s\in\{-,+\}^m} [2H(W^s) - \frac{1}{2}(H(W^{s+}) + H(W^{s-}))^2 \\
&\quad - \frac{1}{2}(H(W^{s+}) - H(W^{s-}))^2] \\
&\overset{(a)}{=} \frac{1}{2^{m+1}} \sum_{s\in\{-,+\}^m} [2H(W^s) - 2(H(W^s))^2 - 2\Delta_s^2] \\
&= \Gamma_m - \frac{1}{2^m} \sum_{s\in\{-,+\}^m} \Delta_s^2,
\end{aligned}$$

where equality (a) follows from $H(W^{s-}) - H(W^{s+}) = 2\Delta_s$. Since $\{\Gamma_m\}_{m=1}^\infty$ is a decreasing sequence, it does have a limit. Lemma 10 below further tells us that if Γ_m is bounded away from 0, then $\Gamma_m - \Gamma_{m+1} = \frac{1}{2^m}\sum_{s\in\{-,+\}^m}\Delta_s^2$ is also bounded away from 0. Therefore, the sequence $\{\Gamma_m\}_{m=1}^\infty$ can only converge to 0. This proves (6.10) and establishes Theorem 22.

Lemma 10. If $\Gamma_m \ge \epsilon > 0$, then $\Gamma_m - \Gamma_{m+1} > 2\epsilon\left(\delta(\frac{1-\sqrt{1-2\epsilon}}{2})\right)^2$, where the function $\delta(\cdot)$ is given in Lemma 9 and inequality (6.9).

Proof. Since $x(1-x) \le 1/4$ for $0 \le x \le 1$, we have $H(W^s)(1-H(W^s)) \le 1/4$ for all $s \in \{-,+\}^m$, and so $\epsilon \le \Gamma_m \le 1/4$. Define a set

$\mathcal{D} := \{s \in \{-,+\}^m \colon H(W^s)(1 - H(W^s)) > \epsilon/2\}$. Then

$$\begin{aligned}
\epsilon \le \Gamma_m &= \frac{1}{2^m} \sum_{s \in \{-,+\}^m} H(W^s)(1 - H(W^s)) \\
&= \frac{1}{2^m} \sum_{s \in \mathcal{D}} H(W^s)(1 - H(W^s)) + \frac{1}{2^m} \sum_{s \notin \mathcal{D}} H(W^s)(1 - H(W^s)) \\
&\le \frac{1}{2^m} \frac{1}{4} |\mathcal{D}| + \frac{1}{2^m} \frac{\epsilon}{2} (2^m - |\mathcal{D}|) \\
&\le \frac{1}{2^m} \frac{1}{4} |\mathcal{D}| + \frac{\epsilon}{2}.
\end{aligned}$$

Thus we obtain that $\frac{|\mathcal{D}|}{2^m} \ge 2\epsilon$. On the other hand, if $H(W^s)(1 - H(W^s)) > \epsilon/2$, then $\frac{1-\sqrt{1-2\epsilon}}{2} < H(W^s) < \frac{1+\sqrt{1-2\epsilon}}{2}$. Therefore, $\Delta_s > \delta(\frac{1-\sqrt{1-2\epsilon}}{2})$ for all $s \in \mathcal{D}$, where the function $\delta(\cdot)$ is given in Lemma 9 and inequality (6.9). Then we have

$$\begin{aligned}
\Gamma_m - \Gamma_{m+1} &= \frac{1}{2^m} \sum_{s \in \{-,+\}^m} \Delta_s^2 \ge \frac{1}{2^m} \sum_{s \in \mathcal{D}} \Delta_s^2 \\
&> \frac{1}{2^m} |\mathcal{D}| \left(\delta\left(\frac{1 - \sqrt{1 - 2\epsilon}}{2} \right) \right)^2 \ge 2\epsilon \left(\delta(\frac{1 - \sqrt{1 - 2\epsilon}}{2}) \right)^2 .
\end{aligned}$$

This completes the proof of the lemma. □

Stronger versions of Theorem 22 hold for certain choices of $\epsilon = \epsilon_n$ that is not a constant and varies with n. In particular, Theorem 22 still holds for $\epsilon = \epsilon_n = n^{-2}$ (in fact, ϵ_n can even decay exponentially with roughly the square-root of n). Therefore, the polar code retaining only rows i of G_n such that $H_i \le \epsilon_n$ has a block error probability that is upper-bounded by $n\epsilon_n$. Since $\epsilon_n = o(1/n)$, the block error probability is upper-bounded by $n\epsilon_n = o(1)$, and the code achieves capacity.

The encoding procedure of polar codes amounts to finding $\mathcal{G}$, the set of noiseless (or "good") bits, and efficient algorithms for finding these were proposed and analyzed in [11], [106]–[108], [113], [145]. In [11], Arıkan also showed that the successive decoder for polar codes allows for an $O(n \log n)$ implementation. Later in [146], a list decoding version of the successive decoder was proposed, and its performance is nearly the same as the Maximum Likelihood (ML) decoder of polar codes for a wide range of parameters.

Remark 4 (Universality of Polar Codes)**.** It was proved in [72] that polar codes are not universal under the successive cancellation decoder. Specifically, let $\mathcal{G}_{\mathrm{BEC},n}$ be the set of noiseless bits for a BEC channel with capacity $1/2$, where n is the code length. Similarly, let $\mathcal{G}_{\mathrm{BSC},n}$ be the set of noiseless bits for a BSC channel with capacity $1/2$. The results of [72] showed that

$$\lim_{n\to\infty} \frac{1}{n} |\mathcal{G}_{\mathrm{BEC},n} \cap \mathcal{G}_{\mathrm{BSC},n}| < 1/2,$$

indicating that the difference between these two sets is not negligible.

However, it is interesting to note that polar codes are universal under the MAP decoder. More precisely, it was proved in [131, pp. 87–89] that when using the MAP decoder, the polar code constructed for the BSC channel has vanishing decoding error probability under all BMS channels with the same capacity as the BSC channel.

Finally, we note that it is possible to modify the construction of polar codes to make them universal under the successive cancellation decoder [74], [133].

6.3 RM Codes Polarize and Twin-RM Codes Achieve Capacity

In [2], the authors develop a similar polarization framework to analyze RM codes; see Figure 6.1(b) for an illustration. More precisely, the monomials $x_{A_1}, \ldots, x_{A_n}$ defined by the $n := 2^m$ subsets $A_1, \ldots, A_n$ of $[m]$, are used in replacement to the increasing integer index i in $[n]$. We arrange these subsets in the following order: Larger sets always appear before smaller sets; for sets with equal size, we use the lexicographic order. Specifically, we always have $|A_i| \geq |A_j|$ for $i < j$. Define the matrix

$$R_n := \begin{bmatrix} \mathrm{Eval}(x_{A_1}) \\ \mathrm{Eval}(x_{A_2}) \\ \vdots \\ \mathrm{Eval}(x_{A_n}) \end{bmatrix} \tag{6.11}$$

whose row vectors are arranged according to the order of the subsets. By definition, R_n is a generator matrix of $\mathrm{RM}(m, m)$. Here we give a

concrete example of the order of sets and R_n for $m = 3$ and $n = 2^m = 8$:

$$\begin{array}{c} A_1 = \{1,2,3\} \\ A_2 = \{1,2\} \\ A_3 = \{1,3\} \\ A_4 = \{2,3\} \\ A_5 = \{1\} \\ A_6 = \{2\} \\ A_7 = \{3\} \\ A_8 = \emptyset \end{array} \qquad R_8 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Note that R_n is a row permutation of G_n. Let $U_{A_1}, \ldots, U_{A_n}$ be the (random) coefficients of the monomials $x_{A_1}, \ldots, x_{A_n}$. Multiplying the coefficient vector $(U_{A_1}, \ldots, U_{A_n})$ with R_n gives us a mapping from the coefficient vector to the evaluation vector $X^n := (U_{A_1}, \ldots, U_{A_n})R_n$. Then we transmit each X_i through an independent copy of W and get the channel output vector Y^n. We still use the successive decoder to decode the coefficient vector bit by bit from U_{A_1} to U_{A_n}. Similarly to H_i, we define the conditional entropy

$$H_{A_i} := H(U_{A_i} | U_{A_1}, \ldots, U_{A_{i-1}}, Y^n),$$

and by the chain rule we also have the balance equation

$$\sum_{i=1}^{n} H_{A_i} = n(1 - I(W)), \tag{6.12}$$

In [2], a polarization result for H_{A_i} is obtained, showing that with this ordering too, almost all H_{A_i} are close to either 0 or 1. More precisely:

Theorem 23 (Polarization of RM Codes)**.** For any BMS channel W and any $0 < \epsilon < 1/2$,

$$|\{i \in [2^m] \colon H_{A_i} \in (\epsilon, 1 - \epsilon)\}| = o(2^m). \tag{6.13}$$

In particular, the above still holds for some choices of $\epsilon = \epsilon_n$ that decay faster than $1/n$. Therefore the code obtained by retaining only the monomials x_{A_i} in R_n corresponding to A_i's such that $H_{A_i} \leq \epsilon_n$ has a vanishing block error probability and achieves capacity; this follows

from the same reasoning as in polar codes. We call this code the Twin-RM code as it is not necessarily the RM code. In fact, if the following implication were true,

$$|A| > |B| \overset{?}{\implies} H_A \geq H_B, \tag{6.14}$$

then the Twin-RM would be exactly the RM code, and the latter would also achieve capacity on any BMS. The same conclusion would hold if (6.14) held true for most sets; it is nonetheless conjectured in [2] that (6.14) holds in the strict sense. To further support this claim, [2] provides two partial results:

(i) Partial order:

$$A \supseteq B \implies H_A \geq H_B, \tag{6.15}$$

more generally, the implication is shown to hold if there exists $\tilde{B}$ s.t. $A \supseteq \tilde{B}$, $|\tilde{B}| = |B|$ and $\tilde{B}$ is less than B and each component of $\tilde{B}$ is smaller than or equal to the corresponding component of B (as integers).

(ii) For the BSC, (6.14) is proved up to $2^m = 16$, and numerically verified for some larger block lengths.

It is also shown in [2] that it suffices to check (6.14) for specific subsets. It is useful at this point to introduce the division of the input bits of RM(m, r) into $m + 1$ layers, where the jth layer corresponds to the subsets of $[m]$ with size j, and the range of j is from 0 to m. Therefore, the 0th layer only has one bit U_1, and the first layer has m bits $U_2, U_3, \ldots, U_{m+1}$. In general, the ith layer has $\binom{m}{i}$ bits. It is shown in [2] that it suffices to check (6.14) for subsets A, B that are respectively the last and first subsets in consecutive layers, as these are shown to achieve respectively the largest and least entropy within layers. Further, it was shown in [3] that the above ordering, together with a generalized ordering, allow to show that a δ-almost RM code, i.e., a linear code spanned by the evaluations of all but a δ fraction of the monomials of degree at most d, achieves positive rate on the BSC. Namely, that for any $\delta > 0$ and any $\epsilon > 0$, there exists a family

of δ-almost Reed–Muller codes of constant rate that correct $1/2 - \epsilon$ fraction of random errors with high probability.

We now present the proof technique for Theorem 23.

Proof technique for Theorem 23. For RM codes, the inductive argument for polar codes is broken. In particular, we no longer have a symmetric break of the conditional entropies as in (6.8), hence no obvious martingale argument. Nonetheless, we next argue that the loss of the inductive/symmetric structure takes place in a favorable way, i.e., the spread in (6.8) tends to be greater for RM codes than for polar codes. In turn, we claim that the conditional entropies polarize faster in the RM code ordering (see [2]). We next explain this and show how one can take a short-cut to show that RM codes polarize using increasing chains of subsets, exploiting the Plotkin recursive structure of RM codes and known inequalities from polar codes. We first need to define increasing chains.

Definition 6.1 (Increasing Chains)**.** We say that $\emptyset = B_0 \subseteq B_1 \subseteq B_2 \subseteq \ldots \subseteq B_m = [m]$ is an increasing chain if $|B_i| = i$ for all $i = 0, 1, 2, \ldots, m$.

As for polar codes, we will make use of the recursive structure of RM codes, i.e., the fact that $\mathrm{RM}(m+1, m+1)$ can be decomposed into two independent copies of $\mathrm{RM}(m, m)$. In order to distinguish H_A's for RM codes with different parameters, we add a superscript to the notation, writing $H_A^{(m)}$ instead of H_A. A main step in our argument consist in proving the following theorem:

Theorem 24 (RM Polarization on Chains)**.** For every BMS channel W, every positive m and every increasing chain $\{B_i\}_{i=0}^m$, we have

$$H_{B_0}^{(m)} \leq H_{B_1}^{(m)} \leq H_{B_2}^{(m)} \leq \ldots \leq H_{B_m}^{(m)}. \tag{6.16}$$

Further, for any $\epsilon \in (0, 1/2)$, there is a constant $D(\epsilon)$ such that for every positive m and every increasing chain $\{B_i\}_{i=0}^m$,

$$\left|\left\{i \in \{0, 1, \ldots, m\} \colon \epsilon < H_{B_i}^{(m)} < 1 - \epsilon\right\}\right| \leq D(\epsilon). \tag{6.17}$$

Note that $D(\epsilon)$ does not depend on m here. This theorem relies strongly on the following interlacing property over chains.

Lemma 11 (Interlacing Property)**.** For every BMS channel W, every positive m and every increasing chain $\{B_i\}_{i=0}^{m}$, we have

$$H_{B_i}^{(m+1)} \leq H_{B_i}^{(m)} \leq H_{B_{i+1}}^{(m+1)} \quad \forall i \in \{0,1,\ldots,m\}. \tag{6.18}$$

The proof of Theorem 24 relies mainly on the previous lemma and the following polar-like inequality: for any $\epsilon \in (0,1/2)$, there is $\delta(\epsilon) > 0$ such that for any increasing chain and any $i \in \{0,1,\ldots,m\}$,

$$H_{B_i}^{(m)} \in (\epsilon, 1-\epsilon) \implies H_{B_i}^{(m)} - H_{B_i}^{(m+1)} > \delta(\epsilon) \text{ and } H_{B_{i+1}}^{(m+1)} - H_{B_i}^{(m)} > \delta(\epsilon). \tag{6.19}$$

Note that (6.19) is analogous to (6.9) except that (i) it does not involve the $+,-$ transform of polar codes but the augmentation or not of a monomial with a new element, (ii) the resulting spread is not necessarily symmetrical as in (6.8). This is where it can be seen that RM codes have a bigger spread of polarization than polar codes, as further discussed below.

We first note that (6.16) follows directly from the interlacing property (6.18); see Figure 6.2 for an illustration of this. To prove (6.17), we combine (6.16) with the polar-like inequality (6.19). Indeed, by (6.19) we know that as long as $H_{B_i}^{(m)} > \epsilon$ and $H_{B_{i+1}}^{(m)} < 1-\epsilon$, we have $H_{B_{i+1}}^{(m)} - H_{B_i}^{(m)} > 2\delta$; see Figure 6.2 for an illustration. Let j be the smallest index such that $H_{B_j}^{(m)} > \epsilon$, and let j' be the largest index such that $H_{B_{j'}}^{(m)} < 1-\epsilon$. Then

$$\left|\left\{i \in \{0,1,\ldots,m\} \colon \epsilon < H_{B_i}^{(m)} < 1-\epsilon\right\}\right| = j' - j + 1.$$

Since $H_{B_i}^{(m)}$ increases with i, we have $H_{B_{j'}}^{(m)} - H_{B_j}^{(m)} = \sum_{i=j}^{j'-1}(H_{B_{i+1}}^{(m)} - H_{B_i}^{(m)}) > 2(j'-j)\delta$, and since $H_{B_{j'}}^{(m)} - H_{B_j}^{(m)}$ is upper bounded by 1, we have $j' - j < \frac{1}{2\delta}$. Therefore,

$$\left|\left\{i \in \{0,1,\ldots,m\} \colon \epsilon < H_{B_i}^{(m)} < 1-\epsilon\right\}\right| < \frac{1}{2\delta} + 1.$$

Thus we have proved (6.17) with the choice of $D(\epsilon) = 1/(2\delta(\epsilon)) + 1$.

Now we are left to explain how to prove (6.18)–(6.19). In Figure 6.1(b), we define the bit-channel $W_{A_i}^{(m)}$ as the binary-input

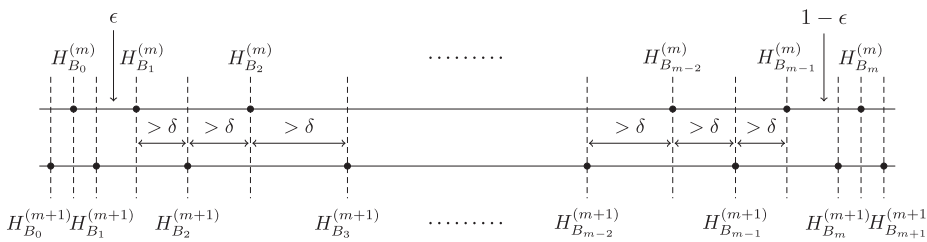


Figure 6.2: Illustration of the interlacing property in (6.18) used in the proofs of Theorem 24.

channel that takes $U_{A_i}^{(m)}$ as input and $Y^n, (U_{A_j}^{(m)}: j < i)$ as outputs, i.e., $W_{A_i}^{(m)}$ is the channel seen by the successive RM decoder when decoding $U_{A_i}^{(m)}$. By definition, we have $H_{A_i}^{(m)} = 1 - I(W_{A_i}^{(m)})$. Making use of the fact that $\mathrm{RM}(m+1, m+1)$ can be decomposed into two independent copies of $\mathrm{RM}(m, m)$, one can show the larger spread of RM code split. More precisely, for every $A_i \subseteq [m]$, the bit-channel $W_{A_i}^{(m+1)}$ is always better than the "+" polar transform of $W_{A_i}^{(m)}$, and the bit-channel $W_{A_i \cup \{m+1\}}^{(m+1)}$ is always worse than the "−" polar transform of $W_{A_i}^{(m)}$, i.e.,

$$H_{A_i}^{(m+1)} \leq H((W_{A_i}^{(m)})^+) \leq H_{A_i}^{(m)} \leq H((W_{A_i}^{(m)})^-) \leq H_{A_i \cup \{m+1\}}^{(m+1)}.$$

Therefore, the gap between $H_{A_i \cup \{m+1\}}^{(m+1)}$ and $H_{A_i}^{(m)}$ is even larger than the gap between $H((W_{A_i}^{(m)})^-)$ and $H_{A_i}^{(m)}$. Similarly, the gap between $H_{A_i}^{(m)}$ and $H_{A_i}^{(m+1)}$ is even larger than the gap between $H_{A_i}^{(m)}$ and $H((W_{A_i}^{(m)})^+)$; see Figure 6.3 for an illustration. Combining this with the polar inequality (6.9), we have shown that (6.18)–(6.19) hold for any $B_{i+1} = B_i \cup \{m+1\}$. Then by the symmetry of RM codes, one can show that $H_{B_i \cup \{j\}}^{(m+1)} \geq H_{B_i \cup \{m+1\}}^{(m+1)}$ for all $j \in [m] \setminus B_i$. This proves (6.18)–(6.19) and Theorem 24.

Theorem 24 proves a polarization on each increasing chain (See Equation (6.17)). In order to obtain the global polarization of RM codes (Theorem 23), we observe that there are in total $m!$ increasing chains, and a careful averaging argument over these $m!$ chains gives the result in Theorem 23.

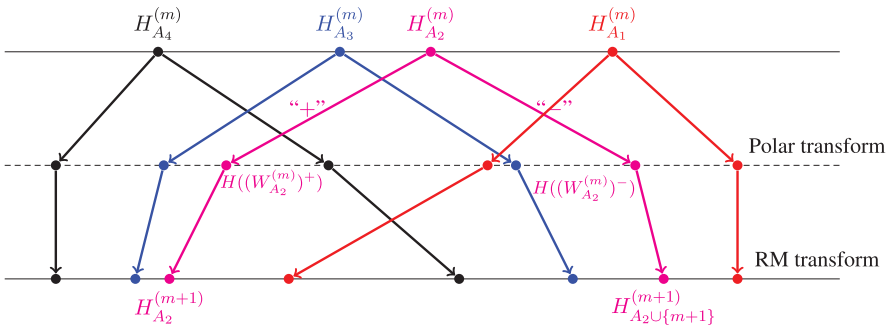


Figure 6.3: The fast polar transform with block size 4. The dots on the second line are the results of the standard polar transform, and the dots on the third line are the results of fast polar transform. In the fast polar transform, the bit-channel obtained by adding a monomial gets even worse (i.e., has even more entropy) than what would the classical polar $-$ transform produce on that bit-channel, and similarly, the better bit-channel obtained by not adding the monomial is even better (less entropic) that what the polar $+$ transform would produce. Therefore, the gap between $H_{A_i\cup\{m+1\}}^{(m+1)}$ and $H_{A_i}^{(m+1)}$ is always larger than the gap between $H((W_{A_i}^{(m)})^-)$ and $H((W_{A_i}^{(m)})^+)$. Intuitively, this explains why RM codes polarize and do so even faster than polar codes (although the formal proof uses the increasing chain argument).

7

Scaling Law

In this section, we review and explain some basic concepts in communication and information therory, including finite-length scaling and error exponents. We will also give an overview of results on the finite-length scaling of RM codes and polar codes.

Shannon's celebrated channel coding theorem asserts that reliable communication can be established if and only if the code rate R is below the channel capacity. More precisely, given a channel W and two (arbitrarily small) positive constants $\epsilon, \delta > 0$, we can always find a code with code rate $R > I(W) - \delta$ and large enough code length such that the decoding error probability $err < \epsilon$.

We say that a code is optimal for a channel W if it achieves the smallest decoding error probability for the transmission over W among all codes with the same length and same code rate. Then the channel coding theorem can be interpreted in two different ways: (i) Suppose that we fix the code rate $R < I(W)$, or equivalently, we fix the gap to capacity $I(W) - R$ to be some positive (and arbitrarily small) constant. For a family of optimal codes with this fixed code rate R and increasing code length, the decoding error probability $err \to 0$ as the code length $n \to \infty$. (ii) Suppose that we fix the decoding error probability to be

some (arbitrarily small) positive constant err. For a family of optimal codes with this fixed decoding error probability and increasing code length, the gap to capacity $I(W) - R \to 0$ as the code length $n \to \infty$.

These two interpretations naturally induce the following two questions for optimal codes: (i) When we fix the code rate R, we know that $err \to 0$ as $n \to \infty$, but how fast does err decay as a function of n? (ii) When we fix the decoding error probability err, we know that the gap to capacity $I(W) - R \to 0$ as $n \to \infty$, but how fast does $I(W) - R$ decay as a function of n? These two questions have been studied for decades, and numerous results have been established. For the first question, it is well known that when we fix R to be some constant that is strictly smaller than $I(W)$, err decays exponentially fast in n. More precisely, err scales as $\exp(-E(R)n)$, where $E(R)$ is a constant that depends only on R, and this constant is referred to as the *error exponent* [16], [29], [54], [58], [60], [61], [138]. For the second question, when we fix the decoding error probability err to be some (arbitrarily small) constant, the gap to capacity $I(W) - R$ scales as $\Theta(1/\sqrt{n})$ for optimal codes as well as the random code ensemble [46], [75], [116], [144], [151], [152].

After answering the two fundamental questions above, let us consider a more general setting, where both err and $I(W) - R$ scales as some functions of n. Now suppose that we want err to scale as $\exp(-\Theta(n^\pi))$ for some constant $0 \leq \pi \leq 1$, and we ask how does $I(W) - R$ scale as a function of n in this regime. This question has also been extensively studied [7], [8], [117]. For optimal codes as well as the random code ensemble, it has been established that $I(W) - R = \Theta(n^{-\frac{1}{2}(1-\pi)})$ when err scales as $\exp(-\Theta(n^\pi))$. Note that this more general setting includes the two fundamental settings above as special cases: When $\pi = 0$, we get back to the constant err setting; when $\pi = 1$, we get back to the constant code rate setting.

The Analysis of Optimal/Random Codes Over the BEC Channel. Let us use a BEC channel with erasure probability ϵ as a toy example to interpret and explain the results discussed above. For simplicity, let us consider linear codes. Suppose that the code length is n and code dimension is $k = Rn$. First we recognize that if more than $n - k$ bits are erased, then there is no way to recover the original codeword. By

the Central Limit Theorem, we know that the number of erasures is $\epsilon n \pm \Theta(\sqrt{n})$ with high probability. Therefore, if $n - k - \epsilon n = o(\sqrt{n})$, then with probability $\geq 1/2$ more than $n - k$ bits are erased, and so the decoding error probability is bounded away from 0. By noticing that $n - k - \epsilon n = o(\sqrt{n})$ is equivalent to $I(W) - R \ll 1/\sqrt{n}$, we conclude that $I(W) - R \geq \Theta(1/\sqrt{n})$ is necessary in order to achieve a small (but constant) err. The other direction, i.e., that $I(W) - R = \Theta(1/\sqrt{n})$ is also sufficient to achieve an arbitrarily small (but constant) err, requires some random coding arguments which we will not cover in this monograph.

Now consider the regime where $I(W) - R = \Theta(n^{-\frac{1}{2}(1-\pi)})$ for some constant $0 \leq \pi \leq 1$. This is equivalent to $n - k = \epsilon n + \Theta(n^{\frac{1}{2}(1+\pi)})$. The number of erasures is a binomial random variable with parameters n and ϵ. It is well known[1] that $\mathbb{P}(\text{Binom}(n, \epsilon) \geq \epsilon n + \Theta(n^{\frac{1}{2}(1+\pi)})) = \exp(-\Theta(n^{\pi}))$. Since the number of erasures $< n - k$ is a necessary condition to recover the original codeword, we conclude that the decoding error probability err is at least $\exp(-\Theta(n^{\pi}))$. One can further show that $err = \exp(-\Theta(n^{\pi}))$ is also achievable with random codes, which we will not cover in this monograph.

The Analysis of Optimal/Random Codes Over the BSC Channel. We now consider a BSC channel with crossover probability p, which we denote as BSC(p). The channel capacity of BSC(p) is $1 - H_2(p)$, where $H_2(p) := -p\log_2(p) - (1-p)\log_2(1-p)$ is the binary entropy function. A simple sphere-packing argument explains why the code rate cannot exceed $1 - H_2(p)$ for reliable communication: In the n-dimensional Hamming space $\{0,1\}^n$, the number of points contained in a Hamming ball with radius np is roughly $2^{nH_2(p)}$. We associate each codeword with a radius-np Hamming ball centered at this codeword. Reliable communication over BSC(p) requires that almost all vectors in each radius-np Hamming ball are decoded as the codeword lying at its center. In other words, we require that all these radius-np Hamming balls with codewords as their centers have no intersections with each other. The

[1] One can prove this using the Chernoff bound together with the fact that Chernoff bound is tight up to constant factors in the exponent.

n-dimensional Hamming space has 2^n vectors in total, so the number of codewords cannot exceed $2^{n-nH_2(p)}$, i.e., the code rate is upper bounded by $1 - H_2(p)$.

A more refined version of the above argument further tells us that $I(W) - R \geq \Theta(1/\sqrt{n})$ is necessary in order to achieve a small (but constant) err. We prove this by contradiction. Suppose on the contrary that $I(W) - R \ll 1/\sqrt{n}$, i.e., $n - k - nH_2(p) = o(\sqrt{n})$. Then the number of codewords is $2^{n-nH_2(p)-o(\sqrt{n})}$, so the average number of channel output vectors that are decoded into the same codeword is $2^{nH_2(p)+o(\sqrt{n})}$, which can only cover a Hamming ball with radius $\leq np + o(\sqrt{n})$. By the Central Limit Theorem, when the crossover probability is p, the number of errors is $np \pm \Theta(\sqrt{n})$ with high probability. Therefore, with probability $1/2$, the number of errors is $np + \Theta(\sqrt{n})$, which cannot be corrected by a code with $2^{n-nH_2(p)-o(\sqrt{n})}$ codewords, and so the decoding error probability of such a code is bounded away from 0. This explains why the gap to capacity $I(W) - R$ needs to be $\Theta(1/\sqrt{n})$ in order to achieve arbitrarily small (but constant) err over BSC.

The Definition of Scaling Exponent. In the above discussion, we analyzed how err and $I(W) - R$ jointly scale as functions of code length n for optimal codes and random codes. The same questions can be asked for any capacity-achieving[2] code family. We say that a capacity-achieving code family has a *polynomial gap to capacity* if there exists a constant $\mu > 0$ such that for any fixed decoding error probability $err > 0$, the gap to capacity satisfies $I(W) - R < n^{-1/\mu}$ when n is large enough. The smallest μ that satisfies the inequality $I(W) - R < n^{-1/\mu}$ for large enough n is called the *scaling exponent* of the code family. According to this definition, smaller scaling exponent means that the code has a smaller gap to capacity. Since the scaling exponent of optimal codes and random codes is 2, the scaling exponent of any other code family is at least 2.

[2]We require the code family to be capacity achieving because otherwise the gap to capacity $I(W) - R$ will not decrease to 0 as $n \to \infty$.

The Scaling of Polar Codes. It has been established that polar codes have a polynomial gap to capacity [66], [71]. In particular, [71] proved that when using the successive cancellation (SC) decoder, the scaling exponent of polar codes is at least 3.579 for any BMS channel. The authors of [105] further proved an upper bound $\mu \leq 4.714$ valid for any BMS channel and improved this upper bound to $\mu \leq 3.639$ for the case of the BEC. Numerical simulations in [88] showed that the scaling exponent of polar codes is about 3.63 for BEC and 4.01 for binary-input AWGN channels. These numbers imply that polar codes with the SC decoder have a much larger gap to capacity than optimal codes and random codes. At the same time, improvement on the gap to capacity (or equivalently, the scaling exponent) is available for polar codes at the expense of higher encoding and decoding complexity. It has been shown in [56], [65] that if we replace the matrix G_n in (2.2) with $G^{\otimes m}$ for some large enough square matrix G, then the resulting polar codes have scaling exponent 2 under the SC decoder for any BMS channel. However, since we use a very large G in the code construction procedure, both the encoding and the SC decoding of this new polar code have much higher complexity than the standard polar codes. The results in [65] were further extended in [150] to show that by using a large enough matrix G in the polar code construction, polar codes in fact achieve the jointly optimal scaling of $I(W) - R = \Theta(n^{-\frac{1}{2}(1-\pi)})$ and $err = \exp(-\Theta(n^{\pi}))$ under the SC decoder. Note that the joint scaling between the gap to capacity and error probability for standard polar codes was analyzed in [105].

The Scaling of RM Codes. The definition of scaling exponent can be extended to more general code families which are not necessarily capacity-achieving. More precisely, for a given code C with code length n, we use $err_{\mathrm{ML}}(p)$ to denote its block error probability under the ML decoder when transmitting over the BSC(p) channel. Given a positive number $0 < \delta < 1/2$, define $width(\delta) := err_{\mathrm{ML}}^{-1}(1-\delta) - err_{\mathrm{ML}}^{-1}(\delta)$ as the amount of change in the channel crossover probability when the ML decoding error decreases from $1-\delta$ to δ, where err_{ML}^{-1} is the inverse function of err_{ML}. Note that $width(\delta)$ depends on the code C although we omit this dependence in the notation. The scaling exponent of a

code family is defined as the smallest μ that satisfies the inequality $width(\delta) < n^{-1/\mu}$ for any $0 < \delta < 1/2$ and large enough n. If we set δ to be very close to 0, the inequality $width(\delta) < n^{-1/\mu}$ implies that the ML decoding error probability transitions from (almost) 0 to (almost) 1 in a narrow window of channel crossover probability, and the width of the window approaches 0 as $n \to \infty$. For the class of capacity-achieving codes, this narrow window locates at $H_2^{-1}(1-R)$, where R is the rate of the code and H_2^{-1} is the inverse of the binary entropy function. For a general code family, it is notoriously difficult to locate the window of the sharp transition. However, a classic result from Tillich and Zemor [148] asserts that for binary linear codes, the width of the transition window is $O(1/\sqrt{d})$, where d is the distance of the code. This result allows us to establish the sharp transition of the ML decoding error probability for a linear code family even if we don't know whether this code family achieves capacity or not.

In order to apply the result of [148] to RM codes, we need the following result about the distance of RM codes: If we fix the code rate of RM codes to be some positive constant $R \in (0,1)$ and let the code length n goes to infinity, then the code distance $d = \Omega(n^{1/2-\alpha})$ for arbitrarily small constant $\alpha > 0$. To see this, we recall that the distance of the code $\mathrm{RM}(m,r)$ is $d = 2^{m-r}$ and the code rate is

$$R = \frac{\binom{m}{0} + \binom{m}{1} + \binom{m}{2} + \cdots + \binom{m}{r}}{2^m}.$$

Let $Z_1, Z_2, \ldots, Z_m$ be m i.i.d. Bernoulli-1/2 random variables. It is easy to see that

$$R = \mathbb{P}\left[Z_1 + Z_2 + \cdots + Z_m \leq r\right].$$

Since we consider the regime of $m \to \infty$, Central Limit Theorem tells us that R is bounded away from 0 and 1 if and only if $r = m/2 \pm \Theta(\sqrt{m})$. Therefore, for RM codes, the code rate $R \in (0,1)$ implies that $d = 2^{m/2 \pm \Theta(\sqrt{m})} = \Omega(n^{1/2-\alpha})$ for arbitrarily small constant $\alpha > 0$. Taking $d = \Omega(n^{1/2-\alpha})$ into the result of [148], we obtain that the width of the transition window for constant-rate RM codes is $O(1/n^{1/4-\alpha})$ for arbitrarily small constant $\alpha > 0$. Recently, [70] improved the bound on the transition width of RM codes to $O(1/n^{1/2-\alpha})$ for arbitrarily small constant $\alpha > 0$. The result of [70] implies that *if RM codes*

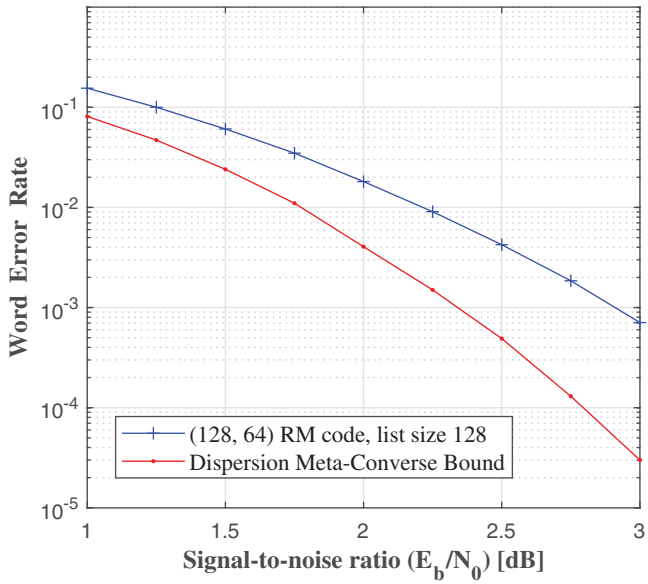


Figure 7.1: Comparison between the performance of RM$(7, 3)$ and the meta-converse bound in [116].

achieve capacity, then RM codes' gap to capacity is almost optimal since $\Theta(1/\sqrt{n})$ is the smallest possible gap to capacity for any code.

In Figure 7.1, we compare the performance of the RM$(7, 3)$ code with the dispersion meta-converse bound in [116]. More specifically, we use Dumer's recursive list decoder [48], [49], [51] to decode the RM$(7, 3)$ code. We choose the list size to be 128, which allows the decoding error probability of the recursive list decoder to be very close to the ML decoder for the RM$(7, 3)$ code. The code length and code dimension of RM$(7, 3)$ are $n = 128, k = 64$. The dispersion meta-converse bound in [116] gives a lower bound on the decoding error probability for any code with the same code length and code dimension. As we can see from Figure 7.1, there is still a gap between the ML performance of RM codes and the meta-converse bound.

8

Decoding Algorithms

We will survey various decoding algorithms for RM codes in this section. We divide these algorithms into three categories. The first category (Section 8.1) only consists of Reed's algorithm [121]. This is the first decoding algorithm for RM codes, designed for the worst-case error correction, and it can efficiently correct any error pattern with Hamming weight up to half the code distance. The second category (Section 8.2) includes efficient algorithms designed for correcting random errors or additive Gaussian noise. These algorithms afford good practical performance, especially in the short to medium code length regime or for low-rate RM codes. Yet due to complexity constraints, most of them are not efficient for decoding RM codes with long code length. More specifically, we will cover the Fast Hadamard Transform decoder [20], [64] for first-order RM codes, Sidel'nikov-Pershakov algorithm and its variants [125], [141], Dumer's list decoding algorithm [48], [49], [51] and Recursive Projection-Aggregation algorithm [153] as well as an algorithm based on minimum-weight parity checks [129]. Finally, the last category (Section 8.3) again only consists of a single decoding algorithm—a Berlekamp-Welch type decoding algorithm proposed in [130]. This algorithm is designed for correcting random errors. Its

performance guarantee (i.e., its polynomial run-time estimate) was established for decoding RM codes of degrees up to $r = o(\sqrt{m})$ while all the previous decoding algorithms discussed in this section only have performance guarantee for constant value of r (i.e., we do not have polynomial upper bounds on their run time at other regimes of parameters). In fact, this algorithm also gives interesting results for degrees $r = m - o(\sqrt{m/\log m})$.

8.1 Reed's Algorithm [121]: Unique Decoding Up to Half the Code Distance

In this section, we recap Reed's decoding algorithm [121] for $\mathrm{RM}(m, r)$. It can correct any error pattern with Hamming weight less than 2^{m-r-1}, half the code distance.

For a subset $A \subseteq [m]$, we write $\overline{A} = [m] \setminus A$ and we use $V_A := \{z \in \mathbb{F}_2^m \colon z_i = 0 \ \forall i \in \overline{A}\}$ to denote the $|A|$-dimensional subspace of $\mathbb{F}_2^m$, i.e., V_A is the subspace obtained by fixing all z_i's to be 0 for i outside of A. For a subspace V_A in $\mathbb{F}_2^m$, there are $2^{m-|A|}$ cosets of the form $V_A + b := \{z + b \colon z \in V_A\}$, where $b \in \mathbb{F}_2^m$. For any $A \subseteq [m]$ and any $b \in \mathbb{F}_2^m$, we always have

$$\sum_{z \in (V_A + b)} \mathrm{Eval}_z(x_A) = 1, \tag{8.1}$$

and we also have that for any $A \not\subset B$,

$$\sum_{z \in (V_A + b)} \mathrm{Eval}_z(x_B) = 0. \tag{8.2}$$

The sums in (8.1)–(8.2) are both over $\mathbb{F}_2$. To see (8.1), notice that $\mathrm{Eval}_z(x_A) = 1$ if and only if $z_i = 1$ for all $i \in A$, and there is only one such $z \in (V_A + b)$. To see (8.2): Since $A \not\subseteq B$, there is $i \in (A \setminus B)$. The value of z_i does not affect the evaluation $\mathrm{Eval}_z(x_B)$. Therefore, $\sum_{z \in (V_A+b), z_i = 0} \mathrm{Eval}_z(x_B) = \sum_{z \in (V_A+b), z_i = 1} \mathrm{Eval}_z(x_B)$, and (8.2) follows immediately.

Suppose that the binary vector $y = (y_z \colon z \in \mathbb{F}_2^m)$ is a noisy version of a codeword $\mathrm{Eval}(f) \in \mathrm{RM}(m, r)$ such that y and $\mathrm{Eval}(f)$ differ in less than 2^{m-r-1} coordinates. Reed's algorithm recovers the original

codeword from y by decoding the coefficients of the polynomial f. Since $\deg(f) \le r$, we can always write $f = \sum_{A\subseteq[m],|A|\le r} u_A x_A$, where u_A's are the coefficients of the corresponding monomials. Reed's algorithm first decodes the coefficients of all the degree-r monomials, and then it decodes the coefficients of all the degree-$(r-1)$ monomials, so on and so forth, until it decodes all the coefficients.

To decode the coefficients u_A for $|A| = r$, Reed's algorithm first calculates the sums $\sum_{z\in(V_A+b)} y_z$ over each of the 2^{m-r} cosets of the subspace V_A, and then it performs a majority vote among these 2^{m-r} sums: If there are more 1's than 0's, then we decode u_A as 1. Otherwise we decode it as 0. Notice that if there is no error, i.e., if $y = \mathrm{Eval}(f)$, then we have

$$\begin{aligned}\sum_{z\in(V_A+b)} y_z &= \sum_{z\in(V_A+b)} \mathrm{Eval}_z\left(\sum_{B\subseteq[m],|B|\le r} u_B x_B\right)\\ &= \sum_{B\subseteq[m],|B|\le r} u_B \sum_{z\in(V_A+b)} \mathrm{Eval}_z(x_B).\end{aligned}$$

According to (8.1)–(8.2), for the subsets $B \subseteq [m]$ with $|B| \le r = |A|$, $\sum_{z\in(V_A+b)} \mathrm{Eval}_z(x_B) = 1$ if and only if $B = A$. Therefore, $\sum_{z\in(V_A+b)} y_z = u_A$ for all the 2^{m-r} cosets of the form $V_A + b$ if $y = \mathrm{Eval}(f)$. Since we assume that y and $\mathrm{Eval}(f)$ differ in less than 2^{m-r-1} coordinates, there are less than 2^{m-r-1} cosets for which $\sum_{z\in(V_A+b)} y_z \ne u_A$. After the majority voting among these 2^{m-r} sums, we will obtain the correct value of u_A.

After decoding all the coefficients of the degree-r monomials, we can calculate

$$y' = y - \mathrm{Eval}\Big(\sum_{B\subseteq[m],|B|=r} u_B x_B\Big).$$

This is a noisy version of the codeword $\mathrm{Eval}(f - \sum_{B\subseteq[m],|B|=r} u_B x_B) \in \mathrm{RM}(m, r-1)$, and the number of errors in y' is less than 2^{m-r-1} by assumption. Now we can use the same method to decode the coefficients of all the degree-$(r-1)$ monomials from y'. We then repeat this procedure until we decode all the coefficients of f.

Theorem 25. For a fixed r and growing m, Reed's algorithm corrects any error pattern with Hamming weight less than 2^{m-r-1} in $O(n \log^r n)$ time when decoding $\mathrm{RM}(m, r)$.

Reed's algorithm is summarized below:

Algorithm 1 Reed's algorithm for decoding RM(m, r)

Input: Parameters m and r of the RM code, and a binary vector $y = (y_z \colon z \in \mathbb{F}_2^m)$ of length $n = 2^m$

Output: A codeword $c \in \mathrm{RM}(m, r)$

$t \leftarrow r$
while $t \geq 0$ **do**
  for each subset $A \subseteq [m]$ with $|A| = t$ **do**
    Calculate $\sum_{z \in (V_A + b)} y_z$ for all the 2^{m-t} cosets of V_A
    $\mathtt{num1} \leftarrow$ number of cosets $(V_A + b)$ such that $\sum_{z \in (V_A + b)} y_z = 1$
    $u_A \leftarrow \mathbf{1}[\mathtt{num1} \geq 2^{m-t-1}]$
  end for
  $y \leftarrow y - \mathrm{Eval}(\sum_{A \subseteq [m], |A| = t} u_A x_A)$
  $t \leftarrow t - 1$
end while
$c \leftarrow \mathrm{Eval}(\sum_{A \subseteq [m], |A| \leq r} u_A x_A)$
Output c

8.2 Decoding Algorithms with Good Practical Performance

8.2.1 Fast Hadamard Transform (FHT) for First Order RM Codes [20], [64]

The dimension of the first order RM code RM$(m, 1)$ is $m + 1$, so there are in total $2^{m+1} = 2n$ codewords. A naive implementation of the Maximum Likelihood (ML) decoder requires $O(n^2)$ operations. In this section we recap an efficient implementation of the ML decoder based on FHT which requires only $O(n \log n)$ operations. We will focus on the soft-decision version of this algorithm, and the hard-decision version can be viewed as a special case.

Consider a binary-input memoryless channel $W \colon \{0, 1\} \to \mathcal{W}$. The log-likelihood ratio (LLR) of an output symbol $x \in \mathcal{W}$ is defined as

$$\mathrm{LLR}(x) := \ln \left(\frac{W(x|0)}{W(x|1)} \right).$$

We still use $y = (y_z \colon z \in \mathbb{F}_2^m)$ to denote the noisy version of a codeword in $\mathrm{RM}(m,1)$. Given the channel output vector y, the ML decoder for first order RM codes aims to find $c \in \mathrm{RM}(m,1)$ to maximize $\prod_{z\in\mathbb{F}_2^m} W(y_z|c_z)$. This is equivalent to finding c which maximizes the following quantity:

$$\prod_{z\in\mathbb{F}_2^m} \frac{W(y_z|c_z)}{\sqrt{W(y_z|0)W(y_z|1)}},$$

which is further equivalent to maximizing

$$\sum_{z\in\mathbb{F}_2^m} \ln\left(\frac{W(y_z|c_z)}{\sqrt{W(y_z|0)W(y_z|1)}}\right). \tag{8.3}$$

As the codeword c is a binary vector,

$$\ln\left(\frac{W(y_z|c_z)}{\sqrt{W(y_z|0)W(y_z|1)}}\right) = \begin{cases} \frac{1}{2}\,\mathrm{LLR}(y_z) & \text{if } c_z = 0 \\ -\frac{1}{2}\,\mathrm{LLR}(y_z) & \text{if } c_z = 1 \end{cases}.$$

From now on we will use the shorthand notation

$$L_z := \mathrm{LLR}(y_z),$$

and the formula in (8.3) can be written as

$$\frac{1}{2}\sum_{z\in\mathbb{F}_2^m} \left((-1)^{c_z} L_z\right), \tag{8.4}$$

so we want to find $c \in \mathrm{RM}(m,1)$ to maximize this quantity.

By definition, every $c \in \mathrm{RM}(m,1)$ corresponds to a polynomial in $\mathbb{F}_2[x_1, x_2, \ldots, x_m]$ of degree one, so we can write every codeword c as a polynomial $u_0 + \sum_{i=1}^m u_i x_i$. In this way, we have $c_z = u_0 + \sum_{i=1}^m u_i z_i$, where $z_1, z_2, \ldots, z_m$ are the coordinates of the vector z. Now our task is to find $u_0, u_1, u_2, \ldots, u_m \in \mathbb{F}_2$ to maximize

$$\sum_{z\in\mathbb{F}_2^m} \left((-1)^{u_0+\sum_{i=1}^m u_i z_i} L_z\right) = (-1)^{u_0} \sum_{z\in\mathbb{F}_2^m} \left((-1)^{\sum_{i=1}^m u_i z_i} L_z\right). \tag{8.5}$$

For a binary vector $u = (u_1, u_2, \ldots, u_m) \in \mathbb{F}_2^m$, we define

$$\hat{L}(u) := \sum_{z\in\mathbb{F}_2^m} \left((-1)^{\sum_{i=1}^m u_i z_i} L_z\right).$$

To find the maximizer of (8.5), we only need to compute $\hat{L}(u)$ for all $u \in \mathbb{F}_2^m$, but the vector $(\hat{L}(u)\colon u \in \mathbb{F}_2^m)$ is exactly the Hadamard Transform of the vector $(L_z\colon z \in \mathbb{F}_2^m)$, so it can be computed using the Fast Hadamard Transform with complexity $O(n \log n)$. Once we know the values of $(\hat{L}(u), u \in \mathbb{F}_2^m)$, we can find $u^* = (u_1^*, u_2^*, \ldots, u_m^*) \in \mathbb{F}_2^m$ that maximizes $|\hat{L}(u)|$. If $\hat{L}(u^*) > 0$, then the decoder outputs the codeword corresponding to $u_0^* = 0, u_1^*, u_2^*, \ldots, u_m^*$. Otherwise, the decoder outputs the codeword corresponding to $u_0^* = 1, u_1^*, u_2^*, \ldots, u_m^*$. This completes the description of the soft-decision FHT decoder for first order RM codes.

The hard-decision FHT decoder is usually used for random errors, or equivalently, used for error corrections over BSC. For BSC, the channel output $y = (y_z\colon z \in \mathbb{F}_2^m)$ is a binary vector. Suppose that the crossover probability of BSC is $p < 1/2$, then $L_z = \ln(\frac{1-p}{p})$ if $y_z = 0$, and $L_z = -\ln(\frac{1-p}{p})$ if $y_z = 1$. Since rescaling the LLR vector by a positive factor does not change the maximizer of (8.5), we can divide the LLR vector by $\ln(\frac{1-p}{p})$ when decoding RM codes over BSC(p). This is equivalent to setting $L_z = 1$ for $y_z = 0$ and $L_z = -1$ for $y_z = 1$. Then the rest of the hard-decision FHT decoding is the same as the soft-decision version.

Theorem 26. The FHT decoder finds the ML decoding result in $O(n \log n)$ time when decoding first order RM codes.

FHT can also be used for list decoding of first order RM codes. For list decoding with list size s, we find s vectors $u^{(1)}, \ldots, u^{(s)}$ that give the largest values of $|\hat{L}(u)|$ among all vectors in $\mathbb{F}_2^m$.

As a final remark, we mention that first order RM codes can also be decoded efficiently as geometry codes [147].

8.2.2 Sidel'nikov-Pershakov Algorithm [141] and Its Variant [125]

In [141], Sidel'nikov and Pershakov proposed a decoding algorithm that works well for second order RM codes with short or medium code length (e.g., ≤ 1024). A version of their decoding algorithm also works for higher-order RM codes, but the performance is not as good as the one for second order RM codes.

Algorithm 2 FHT decoder for first order RM codes

Input: Code length $n = 2^m$, and the LLR vector $(L_z\colon z \in \mathbb{F}_2^m)$ of the received (noisy) codeword

Output: A codeword $c \in \mathrm{RM}(m, 1)$

1: $(\hat{L}(u)\colon u \in \mathbb{F}_2^m) \leftarrow \mathrm{FHT}(L_z\colon z \in \mathbb{F}_2^m)$
2: $u^* = (u_1^*, u_2^*, \ldots, u_m^*) \leftarrow \mathrm{argmax}_{u \in \mathbb{F}_2^m} |\hat{L}(u)|$
3: **if** $\hat{L}(u^*) > 0$ **then**
4: $\quad c \leftarrow \mathrm{Eval}(\sum_{i=1}^m u_i^* x_i)$
5: **else**
6: $\quad c \leftarrow \mathrm{Eval}(1 + \sum_{i=1}^m u_i^* x_i)$
7: **end if**
8: Output c

In this section, we recap Sidel'nikov-Pershakov algorithm for second order RM codes. Consider a polynomial $f \in \mathbb{F}_2[x_1, \ldots, x_m]$ with $\deg(f) \leq 2$:

$$f(z_1, \ldots, z_m) = \sum_{1 \leq i < j \leq m} u_{i,j} z_i z_j + \sum_{i=1}^m u_i z_i + u_0.$$

For a vector $b = (b_1, \ldots, b_m) \in \mathbb{F}_2^m$, we have

$$\begin{aligned}
\mathrm{Eval}_{z+b}(f) + \mathrm{Eval}_z(f) &= \sum_{1 \leq i < j \leq m} u_{i,j}(z_i + b_i)(z_j + b_j) \\
&\quad + \sum_{1 \leq i < j \leq m} u_{i,j} z_i z_j + \sum_{i=1}^m u_i b_i \\
&= \sum_{1 \leq i < j \leq m} u_{i,j} z_i b_j + \sum_{1 \leq i < j \leq m} u_{i,j} b_i z_j \\
&\quad + \sum_{1 \leq i < j \leq m} u_{i,j} b_i b_j + \sum_{i=1}^m u_i b_i \\
&= bUz^T + \sum_{1 \leq i < j \leq m} u_{i,j} b_i b_j + \sum_{i=1}^m u_i b_i, \qquad (8.6)
\end{aligned}$$

where the matrix U is defined as

$$U := \begin{bmatrix} 0 & u_{1,2} & u_{1,3} & \dots & u_{1,m} \\ u_{1,2} & 0 & u_{2,3} & \dots & u_{2,m} \\ u_{1,3} & u_{2,3} & 0 & \dots & u_{3,m} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ u_{1,m} & u_{2,m} & u_{3,m} & \dots & 0 \end{bmatrix}.$$

Note that $\mathrm{Eval}_{z+b}(f) + \mathrm{Eval}_z(f)$ is the coordinate of the discrete derivative of f at direction b, as defined in (4.2).

We first describe the decoder for BSC and then generalize it to other binary-input channels. Suppose that we transmit the codeword $c = \mathrm{Eval}(f) \in \mathrm{RM}(m, 2)$ through some BSC, and we denote the channel output vector as $y \in \mathbb{F}_2^n$. For a fixed b, the vector $(y_{z+b} + y_z \colon z \in \mathbb{F}_2^m)$ is the noisy version of the codeword in $\mathrm{RM}(m, 1)$ corresponding to the polynomial in (8.6). Note that the vector bU consist of the coefficients of all the degree-1 monomials in this polynomial. Therefore, we can decode bU from the noisy codeword $(y_{z+b} + y_z \colon z \in \mathbb{F}_2^m)$. A naive way to do so is to decode each bU separately using the FHT decoder for different vectors $b \in \mathbb{F}_2^m$. Sidel'nikov and Pershakov instead proposed to decode bU for all $b \in \mathbb{F}_2^m$ collectively: The first step is to calculate s candidates for bU that have the largest posterior probability by decoding $(y_{z+b} + y_z \colon z \in \mathbb{F}_2^m)$ with the FHT list decoder described at the end of Section 8.2.1. We denote these s candidates as $D_b^{(1)}, \dots, D_b^{(s)}$ and associate each of them with a reliability value initialized as its posterior probability. Since $bU = b'U + (b + b')U$ for all $b' \in \mathbb{F}_2^m$, the correct candidates $(D_b^* \colon b \in \mathbb{F}_2^m)$ satisfy $D_b^* = D_{b'}^* + D_{b+b'}^*$. In order to find D_b^*, for each $i = 1, \dots, s$, we check for all $b' \in \mathbb{F}_2^m$ whether there are certain i_1 and i_2 such that $D_b^{(i)} = D_{b'}^{(i_1)} + D_{b+b'}^{(i_2)}$. Each time when we find such b' and i_1, i_2, we increase the reliability value of $D_b^{(i)}$ by some function[1] of the reliability values of $D_{b'}^{(i_1)}$ and $D_{b+b'}^{(i_2)}$. Finally, we set D_b^* to be $D_b^{(i)}$ with the largest reliability value among all $i = 1, \dots, s$.

At this point, we have obtained D_b^* for all $b \in \mathbb{F}_2^m$. Notice that D_b^* is the noisy version of bU, and in particular, the first coordinate of

[1] The choice of this function is somewhat ad hoc, and we omit the precise definition here.

D_b^* is the noisy version of $u_{1,2}b_2 + u_{1,3}b_3 + \cdots + u_{1,m}b_m$. Therefore, if we pick the first coordinate of D_b^* for all $b \in \mathbb{F}_2^m$, we will obtain the noisy version of a codeword from $\mathrm{RM}(m, 1)$, and this codeword is the evaluation vector of the polynomial $u_{1,2}x_2 + u_{1,3}x_3 + \cdots + u_{1,m}x_m$. After decoding this noisy codeword using the FHT decoder, we will obtain the coefficients $u_{1,2}, u_{1,3}, \ldots, u_{1,m}$, which form the first column of the matrix U. Similarly, we can also pick the ith coordinate of D_b^* for all $b \in \mathbb{F}_2^m$ and decode it with the FHT decoder. This will allow us to calculate the ith column of U. Once we decode all the entries in U, we have the coefficients of all the degree-2 monomials in f. Then we use the FHT decoder again to decode all the other coefficients in f, which gives us the final decoding result.

For more general binary-input channels other than BSC, we are not able to calculate $y_{z+b} + y_z$ since the two summands are not binary any more. We instead work with the LLRs $L_z := \mathrm{LLR}(y_z)$. Given L_{z+b} and L_z, we want to estimate how likely $\mathrm{Eval}_{z+b}(f) + \mathrm{Eval}_z(f)$ is 0 or 1. The LLR of the sum $\mathrm{Eval}_{z+b}(f) + \mathrm{Eval}_z(f)$ can be calculated as

$$\ln\left(\exp\left(L_{z+b} + L_z\right) + 1\right) - \ln\left(\exp(L_z) + \exp(L_{z+b})\right). \tag{8.7}$$

Once we replace $y_{z+b} + y_z$ with this LLR, we can follow the decoding procedure described above for BSC to decode the output vector of more general binary-input channels.

In [141], Sidel'nikov and Pershakov showed that for second order RM codes $\mathrm{RM}(m, 2)$, their algorithm can correct almost all error patterns with Hamming weight no more than $(n - Cm^{1/4}n^{3/4})/2$ for any constant $C > \ln 4$ when the code length $n \to \infty$.

In [125], Sakkour proposed a simplified and improved version of the Sidel'nikov-Pershakov algorithm for decoding second order RM codes. The main change in Sakkour's algorithm is to use a simple majority voting to obtain D_b^* from D_b, replacing the more complicated procedure in Sidel'nikov-Pershakov algorithm. Such a simplification also leads to smaller decoding error probability. We summarize Sakkour's algorithm below in Algorithm 3 since it is simpler and has better performance:

Algorithm 3 Sakkour's algorithm for decoding second order RM codes over BSC

Input: The code length $n = 2^m$; the received (noisy) codeword $y = (y_z\colon z \in \mathbb{F}_2^m)$;

Output: A codeword $c \in \mathrm{RM}(m, 2)$

1: **for** every $b \in \mathbb{F}_2^m$ **do**
2: $D_b \leftarrow$ `FHT_Decoder`$(y_{z+b} + y_z\colon z \in \mathbb{F}_2^m)$ ▷ `FHT_Decoder` for $\mathrm{RM}(m, 1)$
3: **end for**
4: **for** every $b \in \mathbb{F}_2^m$ **do**
5: $D_b^* \leftarrow \mathrm{Majority}(D_{b+b'} + D_{b'} : b' \in \mathbb{F}_2^m)$
6: ▷ Majority function picks the vector occurring the largest number of times
7: **end for**
8: **for** $i = 1, 2, \ldots, m$ **do**
9: $E_i \leftarrow$ (the i-th coordiante of $D_b^*\colon b \in \mathbb{F}_2^m$) ▷ E_i is a vector of `length` n
10: $\hat{E}_i \leftarrow$ `FHT_Decoder`(E_i) ▷ `FHT_Decoder` for $\mathrm{RM}(m, 1)$
11: $(u_{\{i,j\}}\colon j \in [m] \setminus \{i\}) \leftarrow$ coefficients of the polynomial corresponding to $\hat{E}_i$
12: **end for**
13: $y \leftarrow y - \mathrm{Eval}(\sum_{A \subseteq [m], |A|=2} u_A x_A)$
14: $\hat{D} \leftarrow$ `FHT_Decoder`(y) ▷ `FHT_Decoder` for $\mathrm{RM}(m, 1)$
15: $(u_A\colon A \subseteq [m], |A| \leq 1) \leftarrow$ coefficients of the polynomial corresponding to $\hat{D}$
16: $c \leftarrow \mathrm{Eval}(\sum_{A \subseteq [m], |A| \leq 2} u_A x_A)$
17: Output c

8.2.3 Dumer's Recursive List Decoding [48], [49], [51]

Dumer's recursive list decoding makes use of the Plotkin $(u, u + v)$ construction of RM codes (see Section 2.2 for discussions), and it works well for short or medium length RM codes. More precisely, for RM codes with length ≤ 256, Dumer's recursive list decoding algorithm can efficiently approach the decoding error probability of the ML decoder.

For code length 512 or 1024, Dumer's algorithm works well for RM codes with low code rates. In [51], Dumer and Shabunov also proposed to construct subcodes of RM codes that have better performance than RM codes themselves under the recursive list decoding algorithm.

We start with the basic version of Dumer's recursive decoding algorithm (without list decoding). Suppose that we transmit a codeword $c = \mathrm{Eval}(f) \in \mathrm{RM}(m, r)$ through some binary-input channel W, and we denote the output vector as y and the corresponding LLR vector as L. The original codeword $\mathrm{Eval}(f)$ has two components $\mathrm{Eval}^{[z_m=0]}(f)$ and $\mathrm{Eval}^{[z_m=1]}(f)$. We also devide the LLR vector L into two subvectors $L^{[z_m=0]}$ and $L^{[z_m=1]}$ in the same way so that $L^{[z_m=0]}$ and $L^{[z_m=1]}$ are the LLR vectors of $\mathrm{Eval}^{[z_m=0]}(f)$ and $\mathrm{Eval}^{[z_m=1]}(f)$, respectively. We then construct the LLR vector of $\mathrm{Eval}^{[/z_m]}(f)$ from $L^{[z_m=0]}$ and $L^{[z_m=1]}$ and we denote it as $L^{[/z_m]}$: Each coordinate of $L^{[/z_m]}$ is obtained by combining the corresponding coordinates in $L^{[z_m=0]}$ and $L^{[z_m=1]}$ using formula (8.7).[2]

We first decode $\mathrm{Eval}^{[/z_m]}(f) \in \mathrm{RM}(m-1, r-1)$ from $L^{[/z_m]}$ and denote the decoding result as $\hat{c}^{[/z_m]}$. Then we use $\hat{c}^{[/z_m]}$ together with $L^{[z_m=0]}$ and $L^{[z_m=1]}$ to form an updated LLR vector of $\mathrm{Eval}^{[z_m=0]}(f)$, which we denote as $\tilde{L}^{[z_m=0]}$. The updating rule is as follows: For each $z = (z_1, \ldots, z_{m-1}) \in \mathbb{F}_2^{m-1}$, if $\hat{c}_z^{[/z_m]} = 0$, then we set $\tilde{L}_z^{[z_m=0]} = L_z^{[z_m=0]} + L_z^{[z_m=1]}$, and if $\hat{c}_z^{[/z_m]} = 1$, then we set $\tilde{L}_z^{[z_m=0]} = L_z^{[z_m=0]} - L_z^{[z_m=1]}$. As the next step, we decode $\mathrm{Eval}^{[z_m=0]}(f) \in \mathrm{RM}(m-1, r)$ from $\tilde{L}^{[z_m=0]}$ and denote the decoding result as $\hat{c}^{[z_m=0]}$. Finally, we combine $\hat{c}^{[/z_m]}$ and $\hat{c}^{[z_m=0]}$ to form the final decoding result $\hat{c} = (\hat{c}^{[z_m=0]}, \hat{c}^{[z_m=0]} + \hat{c}^{[/z_m]})$.

In this recursive decoding algorithm, we decompose the decoding of $\mathrm{RM}(m, r)$ into two tasks: First, we decode a codeword from $\mathrm{RM}(m-1, r-1)$. After that, we decode another codeword from $\mathrm{RM}(m-1, r)$. Then the decoding of $\mathrm{RM}(m-1, r-1)$ and $\mathrm{RM}(m-1, r)$ are further decomposed into decoding another four codewords from RM codes with shorter code length and smaller order. This decomposition procedure continues until we reach codewords from first order RM codes $\mathrm{RM}(i, 1)$ for some i or full RM codes $\mathrm{RM}(j, j)$ for some j. For first order RM

[2] In [51], Dumer and Shabunov proposed to work with the quantity $W(x|1) - W(x|0)$ instead of LLR, but one can show that formula (8.7) for LLR is equivalent to the combining method in [51] expressed in terms of $W(x|1) - W(x|0)$.

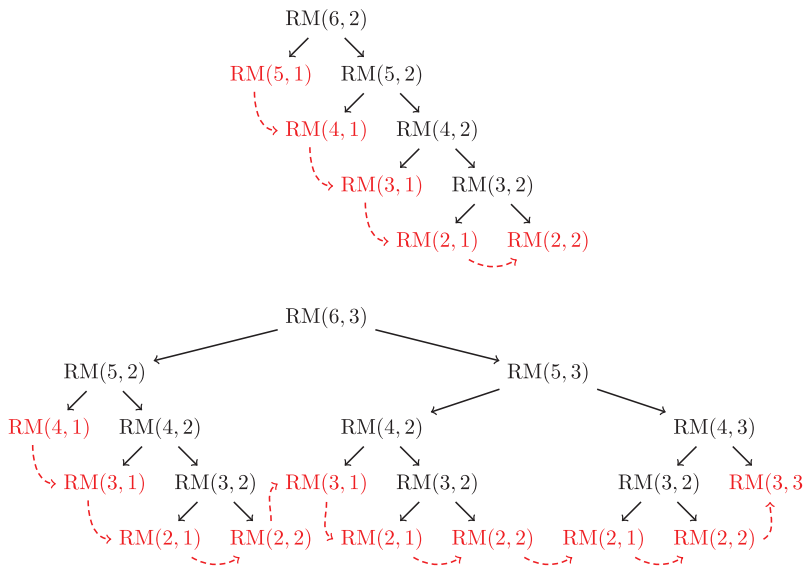


Figure 8.1: The recursive decoding algorithm Φ_r^m for $\mathrm{RM}(m, r)$ and an illustration of how it works for $\mathrm{RM}(6, 2)$ and $\mathrm{RM}(6, 3)$: We decompose $\mathrm{RM}(6, 2)$ and $\mathrm{RM}(6, 3)$ until we reach the leaf nodes, which are the first order or full RM codes marked in red. Eventually we only need to decode these RM codes on the leaf nodes using FHT or ML decoders. The order of decoding these leaf nodes is indicated by the red dashed arrows.

codes we use the FHT decoder, and for full RM codes we simply use the ML decoder. The summary of the algorithm and an illustration of how it works for $\mathrm{RM}(6, 2)$ and $\mathrm{RM}(6, 3)$ are given in Figure 8.1.

Next we briefly discuss the recursive list decoding algorithm. In the list decoding version, we usually stop at the zero order RM codes instead of the first order ones [3]. Note that the zero order RM codes are simply repetition codes with dimension 1. We still go through the same procedure as illustrated in Figure 8.1, i.e., we keep decomposing the RM codes and eventually we only decode the RM codes on "leaf nodes". In the list decoding algorithm, the codes on "leaf nodes" are

[3] In fact, stopping at first order RM codes in list decoding allows one to achieve smaller decoding error probability than stopping at zero order RM codes. In this work, we only describe the version of list decoding stopping at zero order RM codes for two reasons: First, it is easier to describe; second, this is the version presented in Dumer and Shabunov's original paper [51, Section III].

either repetition codes or full RM codes. Each time when we decode a new leaf node, we examine several possible decoding results of this new node for every candidate in the list: If this new leaf node is a repetition code, then there are only two possible decoding results–all zero or all one; if this new leaf node is a full RM code, then we take the 4 most likely decoding results of it for every candidate in the list. In this way, we will increase the list size by a factor of 2 or 4 at each step, depending on whether the new leaf node is a repetition code or a full RM code. We then calculate a reliability value for each candidate in the new list. When the list size is larger than some pre-specified value μ, we prune the list down to size μ by only keeping the candidates with the largest reliability values. Clearly, large μ leads to longer running time of the algorithm but smaller decoding error probability.

In [51], a family of subcodes of RM codes were also proposed. The subcodes have smaller decoding probability under the recursive list decoding algorithm. The idea is quite natural: Each repetition code on the "leaf nodes" only contain one information bit. Some of these information bits are relatively noisy, and the others are relatively noiseless. Dumer and Shabunov proposed to set all the noisy bits to be 0. In this way, one can get smaller decoding error at the cost of decreasing the code rate.

8.2.4 Recursive Projection Aggregation Decoding [153]

The Recursive Projection Aggregation (RPA) decoding algorithm was proposed recently by Ye and Abbe [153]. It works well for second and third order RM codes with short or medium code length (e.g., ≤ 1024). In particular, the RPA algorithm can efficiently achieve the same decoding error probability as the ML decoder for second order RM codes with length ≤ 1024. Moreover, RPA decoder naturally allows parallel implementation.

We will focus mainly on the RPA decoder for BSC channels and briefly mention how to adapt it to general communication channels at the end of this section. In Section 2.5, we have shown that $\mathrm{Eval}^{[/p]}(f) \in \mathrm{RM}(m-1, r-1)$ whenever $\mathrm{Eval}(f) \in \mathrm{RM}(m, r)$ for any $p = b_1 z_1 + \cdots + b_m z_m$ with nonzero coefficients $b = (b_1, \ldots, b_m) \neq 0$. Let $\mathbb{B}$ be

Algorithm 4 Dumer's Algorithm Φ_r^m for decoding $\mathrm{RM}(m, r)$

Input: LLR vector $L = (L^{[z_m=0]}, L^{[z_m=1]})$
Output: $\hat{c}$

if $1 < r < m$ **then**
    Calculate $L^{[/z_m]}$ from $L^{[z_m=0]}$ and $L^{[z_m=1]}$
    $\hat{c}^{[/z_m]} \leftarrow \Phi_{r-1}^{m-1}(L^{[/z_m]})$
    Calculate $\tilde{L}^{[z_m=0]}$ from $L^{[z_m=0]}, L^{[z_m=1]}$ and $\hat{c}^{[/z_m]}$
    $\hat{c}^{[z_m=0]} \leftarrow \Phi_r^{m-1}(\tilde{L}^{[z_m=0]})$
    $\hat{c} \leftarrow (\hat{c}^{[z_m=0]}, \hat{c}^{[z_m=0]} + \hat{c}^{[/z_m]})$
else if $r = 1$ **then**
    use FHT decoder
else if $r = m$ **then**
    use ML decoder
end if

the one-dimensional subspace of $\mathbb{F}_2^m$ consisting of the 0 vector and b. Then $\mathrm{Eval}^{[/p]}(f)$ is obtained by taking the sums in each of the 2^{m-1} cosets of $\mathbb{B}$, i.e., each coordinate in $\mathrm{Eval}^{[/p]}(f)$ is the sum $\sum_{z \in T} \mathrm{Eval}_z(f)$ for some coset T of $\mathbb{B}$. For this reason, we will use the two notations $\mathrm{Eval}^{[/\mathbb{B}]}(f)$ and $\mathrm{Eval}^{[/p]}(f)$ interchangeably from now on. For a noisy codeword y, we also use $y^{[/\mathbb{B}]}$ and $y^{[/p]}$ interchangeably, and we call $y^{[/\mathbb{B}]}$ the projection of y onto the cosets of $\mathbb{B}$. There are in total $n-1$ one-dimensional subspaces in $\mathbb{F}_2^m$. We denote them as $\mathbb{B}_1, \ldots, \mathbb{B}_{n-1}$.

Suppose that we transmit a codeword $c = \mathrm{Eval}(f) \in \mathrm{RM}(m, r)$ through BSC, and that the channel output is y. The RPA decoder for $\mathrm{RM}(m, r)$ consists of three steps: First, the projection step, then the recursive decoding step, and third, the aggregation step. More precisely, the first step is to project the noisy codeword y onto all $n-1$ one-dimensional subspaces $\mathbb{B}_1, \ldots, \mathbb{B}_{n-1}$. Note that this projection step also appears in Sidel'nikov-Pershakov algorithm [141] and Sakkour's algorithm [125]; see Section 8.2.2. The second step is to decode each $y^{[/\mathbb{B}_i]}$ using the RPA decoder for $\mathrm{RM}(m-1, r-1)$. If $r = 2$, then we simply use the FHT decoder for $y^{[/\mathbb{B}_i]}$. We denote the decoding result of the second step as $\hat{y}^{[/\mathbb{B}_i]}$. Note that $\hat{y}^{[/\mathbb{B}_1]}, \ldots, \hat{y}^{[/\mathbb{B}_{n-1}]}$ consist of the (noisy) estimates of the sum $\mathrm{Eval}_z(f) + \mathrm{Eval}_{z'}(f)$ for all $z \neq z'$. We

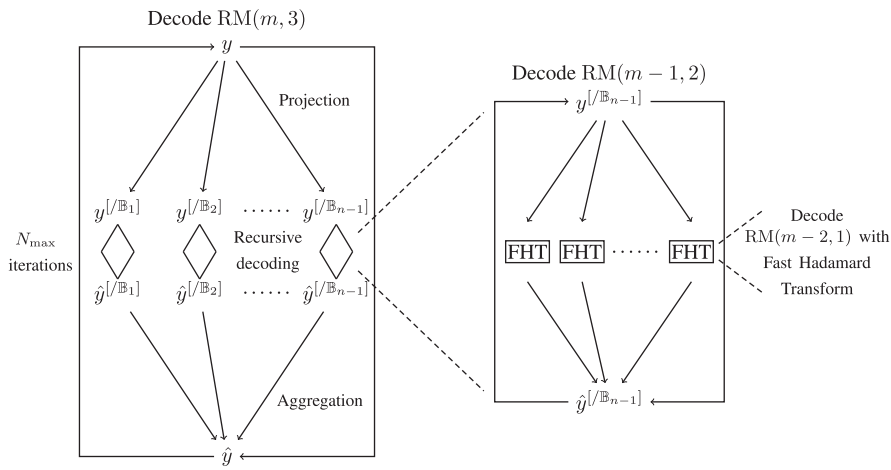


Figure 8.2: Recursive Projection-Aggregation decoding algorithm for third order RM codes.

denote the estimate of $\mathrm{Eval}_z(f) + \mathrm{Eval}_{z'}(f)$ as $\hat{y}_{(z,z')}$. Finally, in the aggregation step, observe that $\hat{y}_{(z,z')} + y_{z'}$ is an estimate of $\mathrm{Eval}_z(f)$ for all $z' \neq z$. For a fixed z, we have in total $n-1$ such estimates of $\mathrm{Eval}_z(f)$, and we perform a majority vote among these $n-1$ estimates to obtain $\hat{y}_z$, i.e., we count the number of 0's and 1's in the set $\{\hat{y}_{(z,z')} + y_{z'} \colon z' \in \mathbb{F}_2^m, z' \neq z\}$: If there are more 1's than 0's, then we set $\hat{y}_z$ to be 1. Otherwise, we set it to be 0. Next we replace the original channel output vector y with $\hat{y} = (\hat{y}_z \colon z \in \mathbb{F}_2^m)$, and run the Projection-Recursive decoding-Aggregation cycle again for a few more rounds[4]. The vector $\hat{y} = (\hat{y}_z \colon z \in \mathbb{F}_2^m)$ in the last round is the final decoding result of the RPA decoder. See Figure 8.2 for a high-level illustration of the RPA decoder.

For general communication channels we need to work with LLR, and we only need to make two changes in the RPA decoder for BSC. The first change is in the projection step: In order to calculate $y^{[/\mathbb{B}]}$, we need to calculate the sums $y_z + y_{z'}$ for the BSC case. We cannot do this for general communication channels because the channel output vector is not binary anymore. Instead, we calculate the projected LLR

[4]In practice, usually three rounds are enough for the algorithm to converge.

vectors $L^{[/\mathbb{B}]}$ using (8.7), and in the recursive decoding step, we decode from $L^{[/\mathbb{B}_1]}, \ldots, L^{[/\mathbb{B}_{n-1}]}$. The second change is in the aggregation step: We replace the majority vote with a weighted sum of the LLRs. More precisely, for a fixed z, we calculate the sum $\hat{L}_z = \frac{1}{n-1} \sum_{z' \neq z} \tilde{y}_{(z,z')} L_{z'}$, where we set $\tilde{y}_{(z,z')} = 1$ if $\hat{y}_{(z,z')} = 0$ and $\tilde{y}_{(z,z')} = -1$ if $\hat{y}_{(z,z')} = 1$. After each round, we replace the LLR vector of the original channel output with $\hat{L} = (\hat{L}_z \colon z \in \mathbb{F}_2^m)$ and run the Projection-Recursive decoding-Aggregation cycle again. After the last round, we decode $\hat{y}_z$ as 0 if $\hat{L}_z > 0$ and otherwise we decode $\hat{y}_z$ as 1. The vector $\hat{y} = (\hat{y}_z \colon z \in \mathbb{F}_2^m)$ is the final decoding result of the RPA decoder.

In practical implementation, we combine the RPA decoder with the following list decoding procedure proposed by Chase [40] to boost the performance. We first sort $|L_z|, z \in \mathbb{F}_2^m$ from small to large. Assume for example that $|L_{z_1}|, |L_{z_2}|, |L_{z_3}|$ are the three smallest components in the LLR vector, meaning that $y_{z_1}, y_{z_2}, y_{z_3}$ are the three most noisy symbols in the channel outputs. Next we enumerate all 8 the possible cases of these three bits: We set $L_{z_i} = \pm L_{\max}$ for $i = 1, 2, 3$, where $L_{\max}$ is some large real number. In practice, we can choose $L_{\max} := 2 \max(|L_z| \colon z \in \mathbb{F}_2^m)$. For each of these 8 cases, we use the RPA decoder to obtain a decoded codeword (candidate). Finally, we calculate the posterior probability for each of these 8 candidates, and choose the largest one as the final decoding result, namely, we perform the ML decoding among the 8 candidates in the list. This list decoding version of the RPA decoder allows us to efficiently achieve the same decoding error probability as the ML decoder for second order RM codes with length ≤ 1024.

8.2.5 Additional Methods

Santi *et al.* [129] applied iterative decoding to a highly-redundant parity-check (PC) matrix that contains only the minimum-weight dual codewords as rows. In particular, [129] proposed to use the peeling decoder for the binary erasure channel, linear-programming and belief propagation (BP) decoding for the binary-input additive white Gaussian noise channel, and bit-flipping and BP decoding for the binary symmetric channel. For short block lengths, it was shown that near-ML performance

Algorithm 5 RPA decoder for RM codes over BSC

Input: The parameters of the Reed-Muller code m and r; the received (noisy) codeword $y = (y_z \colon z \in \mathbb{F}_2^m)$; the maximal number of iterations $N_{\max}$

Output: $\hat{y} = (\hat{y}_z \colon z \in \mathbb{F}_2^m) \in \mathbb{F}_2^n$

for $j = 1, 2, \ldots, N_{\max}$ **do**
  $\hat{y}^{/\mathbb{B}_i} \leftarrow \texttt{RPA}(m-1, r-1, y^{/\mathbb{B}_i}, N_{\max})$ for $i = 1, 2, \ldots, n-1$
  $\{\hat{y}_{(z,z')} : z, z' \in \mathbb{F}_2^m, z \neq z'\} \leftarrow$ coordinates of $\hat{y}^{/\mathbb{B}_1}, \ldots, \hat{y}^{/\mathbb{B}_{n-1}}$
  for every $z \in \mathbb{F}_2^m$ **do**
    $\texttt{num1} \leftarrow$ number of $z' \in \mathbb{F}_2^m \setminus \{z\}$ such that $\hat{y}_{(z,z')} + y_{z'} = 1$
    $\hat{y}_z \leftarrow \mathbf{1}[\texttt{num1} > \frac{n-1}{2}]$
  end for
  if $y = \hat{y}$ **then**
    break
  end if
  $y \leftarrow \hat{y}$
end for
Output $\hat{y}$

can indeed be achieved in many cases. [129] also proposed a method to tailor the PC matrix to the received observation by selecting only a small fraction of useful minimum-weight PCs before decoding begins. This allows one to both improve performance and significantly reduce complexity compared to using the full set of minimum-weight PCs. Some other RM decoders recently proposed in [55], [63], [94] share some similarities with the RPA decoder in [153] because they all make use of the symmetry and/or the automorphism group of RM codes.

Geiselhart *et al.* [63] proposed a general decoding framework for RM codes and polar codes, which makes use of the automorphism groups of these two code families. The block diagram of their automorphism-based decoding framework is given in Figure 8.3. Suppose that we want to decode an RM code over a BSC channel, and $\boldsymbol{y}$ in Figure 8.3 is the channel output vector. The first step of the automorphism-based decoding is to find M permutations in the automorphism group of the RM code. Here M is a parameter up to our own choice, and we denote

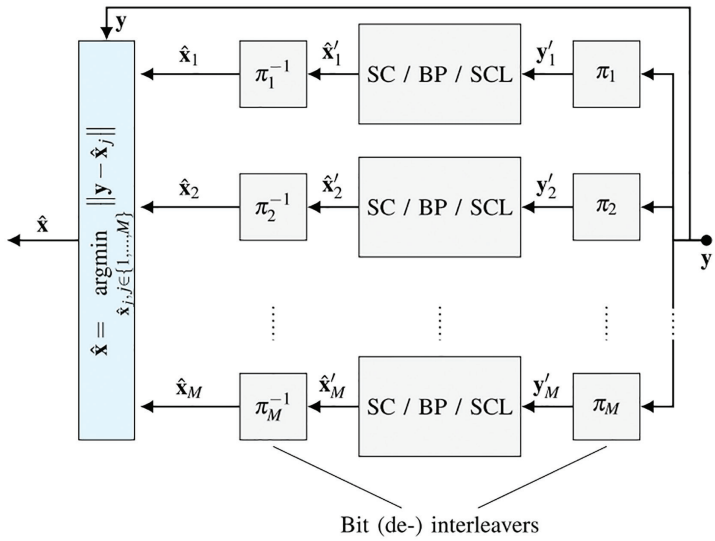


Figure 8.3: Block diagram of the automorphism-based decoding in [63]. This is originally Figure 1 in [63].

the permutations as $\pi_1, \pi_2, \ldots, \pi_M$. Larger M leads to smaller decoding error probability at the cost of a higher decoding complexity. In the next step, we apply these M permutations to the channel output vector $\boldsymbol{y}$ and denote the permuted versions of $\boldsymbol{y}$ as $\boldsymbol{y}'_1, \boldsymbol{y}'_2, \ldots, \boldsymbol{y}'_M$. Then we pick a classic decoder to decode each $\boldsymbol{y}'_i$ for $1 \leq i \leq M$. This classic decoder could be the SC decoder, BP decoder, SCL decoder, or recursive list decoder, among other choices. We denote the decoded version of $\boldsymbol{y}'_i$ as $\hat{\boldsymbol{x}}'_i$ and apply the inverse permutation π_i^{-1} to it. At this point, we obtain M decoding candidates $\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2, \ldots, \hat{\boldsymbol{x}}_M$. Finally, we perform the ML decoding among these M candidates to obtain the final decoding result $\hat{\boldsymbol{x}}$.

In recent years, machine learning techniques have been used to build new decoders for RM codes and other linear codes [41], [101], [110]. Nachmani *et al.* [110] introduced neural decoders as a generalization of the classic Belief Propagation (BP) decoding algorithms. In particular, they viewed the Trellis graph in the BP algorithm as a neural network and optimized the weights in the Trellis graph by training the neural network. The method in [110] is quite general, and it works for all linear

codes. Later in [41], a new class of neural decoders was specifically designed for cyclic codes, which exploits the cyclic invariant structure of the codes by imposing a shift-invariant structure on the weights of the neural decoder. Since punctured RM codes are cyclic codes, the neural decoders in [41] can be applied to them. Makkuva *et al.* [101] used machine learning methods to construct a new code called the Kronecker Operation (KO) codes, which share a similar structure as RM codes and polar codes.

Mondelli *et al.* [104] explored the relationship between polar and RM codes, and they proposed a coding scheme which improves upon the performance of the standard polar codes at practical block lengths. The starting point is the experimental observation that RM codes have a smaller error probability than polar codes under MAP decoding. This motivates one to introduce a family of codes that "interpolates" between RM and polar codes, call this family $\mathsf{C}_{\text{inter}} = \{\mathsf{C}_\alpha \colon \alpha \in [0,1]\}$, where $\mathsf{C}_\alpha|_{\alpha=1}$ is the original polar code, and $\mathsf{C}_\alpha|_{\alpha=0}$ is an RM code. Based on numerical observations, one can see that the error probability under MAP decoding is an increasing function of α. MAP decoding has in general exponential complexity, but empirically the performance of polar codes at finite block lengths is boosted by moving along the family $\mathsf{C}_{\text{inter}}$ even under low-complexity decoding schemes such as belief propagation or successive cancellation list decoder. The performance gain was also demonstrated in [104] via numerical simulations for transmission over the erasure channel as well as the Gaussian channel.

A recent paper [93] also made use of the connection between RM codes and polar codes to invent a new family of codes called the Adjacent-Bits-Swapped (ABS) polar codes. ABS polar codes were inspired by [2], which conjectured that RM codes polarize even faster than polar codes. The authors of [2] further conjectured that the reason for faster polarization is that RM codes reordered the rows so that the conditional entropy of each message bit given previous message bits and all the channel outputs becomes completely ordered. Inspired by this conjecture, the authors of [93] added a permutation layer after each polar transform layer in the ABS polar code construction. In each permutation layer, the "unordered" adjacent bits are swapped to speed up polarization. ABS polar codes can be viewed as an intermediate point between polar codes

and RM codes: On the one hand, ABS polar codes have permutation layers to speed up the polarization—this is somewhat similar to RM codes because RM codes also reorder/permute the rows of the square matrix $G_2^{\otimes m}$. On the other hand, ABS polar codes only swap a small number of adjacent bits so that the overall structure is still close to polar codes, which allows us to use a modified SCL decoder to efficiently decode ABS polar codes. After the invention of ABS polar codes, Li *et al.* proposed ABS+ polar codes in [92], which is a further generalization and improvement of ABS polar codes. ABS+ polar codes consistently improve upon standard polar codes by 0.15 dB–0.35 dB for a wide range of code parameters while keeping the same decoding time.

8.3 Berlekamp-Welch Type Decoding Algorithm [130]

In this section we explain the algorithm of Saptharishi *et al.* [130] for decoding RM codes of degrees up to $r = o(\sqrt{m})$. In fact, their algorithm also gives interesting results for degrees $r = m - o(\sqrt{m/\log m})$. The algorithm is similar in spirit to the work of Pellikaan [114] and Duursma and Kötter [52], which abstracts the Berlekamp-Welch algorithm.

Before stating their main theorem we will need the following notation. For $u, v \in \mathbb{F}_q^n$, we denote by $u * v \in \mathbb{F}_q^n$ the vector $(u_1 v_1, \ldots, u_n v_n)$. For $\mathcal{A}, \mathcal{B} \subseteq \mathbb{F}_q^n$ we similarly define $\mathcal{A} * \mathcal{B} = \{u * v \mid u \in \mathcal{A}, v \in \mathcal{B}\}$. For a code $\mathsf{N} \subseteq \mathbb{F}_q^n$ and a subset $\mathcal{U} \subset [n]$ we say that N can correct the erasure pattern $\mathbb{1}_{\mathcal{U}}$ if we can correct every codeword whose $\mathcal{U}$-coordinates were erased (i.e., replaced with "?").

The algorithm considers three codes C, E and N, all subsets of $\mathbb{F}_q^n$, such that $\mathsf{E} * \mathsf{C} \subseteq \mathsf{N}$. It is able to correct in C those *error* patterns that are correctable as *erasures* in N, through the use of an *error-locating code* E.

Theorem 27. Let $\mathbb{F}_q$ be a finite field and $\mathsf{E}, \mathsf{C}, \mathsf{N} \subseteq \mathbb{F}_q^n$ be codes with the following properties.

1. $\mathsf{E} * \mathsf{C} \subseteq \mathsf{N}$.
2. For any pattern $\mathbb{1}_{\mathcal{U}}$ that is correctable from erasures in N, and for any coordinate $i \notin \mathcal{U}$, there exists a codeword $\mathbf{a} \in \mathsf{E}$ such that $\mathbf{a}_j = 0$ for all $j \in \mathcal{U}$ and $\mathbf{a}_i = 1$.

Algorithm 6 Decoding Algorithm of [130]

Require: received word $\mathbf{y} \in \mathbb{F}_q^n$ such that $\mathbf{y} = \mathbf{c} + \mathbf{e}$, with $\mathbf{c} \in \mathsf{C}$ and $\mathbf{e}$ is supported on a set $\mathcal{U} \subseteq [n]$.

1: Solve for $\mathbf{a} \in \mathsf{E}, \mathbf{b} \in \mathsf{N}$, the linear system $\mathbf{a} * \mathbf{y} = \mathbf{b}$.
2: Let $\{\mathbf{a}_1, \ldots, \mathbf{a}_k\}$ be a basis for the solution space of $\mathbf{a}$, and let $\mathcal{E}$ denote the common zeros of $\{\mathbf{a}_i \mid i \in [k]\}$.
3: For every $j \in \mathcal{E}$, replace $\mathbf{y}_j$ with "?", to get a new word $\mathbf{y}'$.
4: Correct $\mathbf{y}'$ from erasures in C.

Then Algorithm 6 corrects in C any error pattern $\mathbb{1}_{\mathcal{U}}$, which is correctable from erasures in N.

It is worth pointing out the differences between Algorithm 6 and the abstract Berlekamp-Welch decoder of [114] and [52]. Similarly, [52], [114] set up codes E, C and N such that $\mathsf{E} * \mathsf{C} \subseteq \mathsf{N}$. However, instead of Property 2, they require that for any $\mathbf{e} \in \mathsf{E}$ and $\mathbf{c} \in \mathsf{C}$, if $\mathbf{e} * \mathbf{c} = 0$ then $\mathbf{e} = 0$ or $\mathbf{c} = 0$ (alternatively they impose some requirements on the distances of E and C that guarantee this property). This property as well as the distance properties, do not hold in the case of Reed-Muller codes, which is the main application of Theorem 27.

Proof Sketch. It is relatively easy to show that Property 2 in the statement of the theorem guarantees that every erasure pattern that is correctable in N is also correctable in C. The main point of the algorithm is that, under the hypothesis of the theorem, the common zeros of the possible solutions for $\mathbf{a}$ are exactly the corrupted coordinates (errors).

We first note that the system of equations $\mathbf{a} * \mathbf{y} = \mathbf{b}$ is indeed a linear system in the coordinates of $\mathbf{a}, \mathbf{b}$ and therefore can be solved efficiently.

Denote $\mathbf{y} = \mathbf{c} + \mathbf{e}$, where $\mathbf{c} \in \mathsf{C}$ is the transmitted codeword and $\mathbf{e}$ is supported on the set of error locations $\mathcal{U}$. The following two claims guarantee that the algorithm correctly finds the set of error locations. The first claim shows that error locations are common zeros and the second claim shows that no other coordinate is a common zero.

Claim 2. For every $\mathbf{a} \in \mathsf{E}, \mathbf{b} \in \mathsf{N}$ such that $\mathbf{a} * \mathbf{y} = \mathbf{b}$, it holds that $\mathbf{a} * \mathbf{e} = 0$.

Proof. Observe that $\mathbf{a} * \mathbf{e} = \mathbf{a} * \mathbf{y} - \mathbf{a} * \mathbf{c}$ is also a codeword in N. As $\mathbf{a} * \mathbf{e}$ is supported on $\mathcal{U}$, and since $\mathcal{U}$ is an erasure-correctable pattern in N, it must be the zero codeword. □

Claim 3. For every $i \notin \mathcal{U}$ there exists $\mathbf{a} \in \mathsf{E}, \mathbf{b} \in \mathsf{N}$ such that $\mathbf{a}$ is 0 on $\mathcal{U}$, $\mathbf{a}_i = 1$ and $\mathbf{a} * \mathbf{y} = \mathbf{b}$.

Proof. Property 2 implies that since $\mathcal{U}$ is correctable from erasures in N, for every $i \notin \mathcal{U}$ there is $\mathbf{a} \in \mathsf{E}$ such that $\mathbf{a}$ is 0 on $\mathcal{U}$ and $\mathbf{a}_i = 1$. Set $\mathbf{b} = \mathbf{a} * \mathbf{y}$. As $\mathbf{b} = \mathbf{a} * \mathbf{c} + \mathbf{a} * \mathbf{e} = \mathbf{a} * \mathbf{c}$, it follows that $\mathbf{b} \in \mathsf{N}$. □

Together, Claims 2 and 3 imply the correctness of the algorithm. □

To apply Theorem 27 to RM codes we note that for $m \in \mathbb{N}$ and $r \leq m/2 - 1$ the codes $\mathsf{C} = \mathrm{RM}(m, m - 2r - 2)$, $\mathsf{N} = \mathrm{RM}(m, m - r - 1)$ and $\mathsf{E} = \mathrm{RM}(m, r + 1)$ satisfy the conditions of Theorem 27.

Theorem 21 shows that $\mathrm{RM}(m, m - r - 1)$ achieves capacity for $r = m/2 \pm O(\sqrt{m})$. Letting $r = m/2 - o(\sqrt{m})$ and looking at the code $\mathrm{RM}(m, m - 2r - 2) = \mathrm{RM}(m, o(\sqrt{m}))$ so that $\binom{m}{\leq r} = (1/2 - o(1))2^m$, Saptharishi *et al.* obtained the following corollary to Theorem 27.

Corollary 7. There exists an efficient (deterministic) algorithm that is able to correct a fraction of $(1/2 - o(1))$ random errors in $\mathrm{RM}(m, o(\sqrt{m}))$, with probability $1 - o(1)$.

Similar arguments allow [130] to obtain results for high rate RM codes. In particular, combining 19 with the argument of [130], [25] gives the following result.

Theorem 28 (Corollary 1.3 of [25])**.** Let γ_0 be the constant in 19. Then, for $r < \gamma_0 m$, there is an efficient algorithm that can correct $\mathrm{RM}(m, m - (2r + 2))$ from a random set of $(1 - o(1))\binom{m}{\leq r}$ errors. Moreover, the running time of the algorithm is $2^m \cdot \mathrm{poly}(\binom{m}{\leq r})$.

In [87], Kopparty and Potukuchi improved the running time of the decoding algorithm of Theorem 28 to be a polynomial in the length of the syndrome, that is the algorithm can decode in time $\text{poly}(\binom{m}{\leq r})$.

It is an intriguing open problem to find an efficient algorithm that can decode from more random errors.

9

Applications of RM Codes Beyond Communication

Reed-Muller codes (over both large and small finite fields) have been extremely influential even beyond communication and channel coding, playing a central role in some important developments in several areas. In cryptography, they have been used e.g., in secret sharing schemes [135], instance hiding constructions [19] and private information retrieval (see the surveys [62], [154]). In the theory of randomness, they have been used in the constructions of many pseudo-random generators and randomness extractors, e.g., [31], [139]. These in turn were used for hardness amplification, program testing and eventually in various interactive and probabilistic proof systems, e.g., the celebrated results NEXP=MIP [14], IP=PSPACE [136] and NP=PCP [12]. In circuit lower bounds for some low complexity classes one argues that every circuit in the class is close to a codeword, so any function far from the RM code cannot be computed by such circuits (e.g., [120]). In distributed computing they were used to design fault-tolerant information dispersal algorithms for networks [118]. The hardness of approximation of many optimization problems is greatly improved by the "short code" [15], which uses the optimal testing result of [26]. In compressed sensing [36]–[38], [47], RM codes and Delsarte-Goethals codes (discussed in

Section 2.7) are used to construct good sensing matrices for the purpose of recovering sparse signals from very few measurements [17], [34], [35]. And the list goes on. Needless to say, the properties used in these works are properties of low-degree polynomials (such as interpolation, linearity, partial derivatives, self-reducibility, affine-invariance, heredity under various restrictions to variables, etc.), and in some of these cases, specific coding-theoretic perspective such as distance, unique-decoding, list-decoding, local testing and decoding etc. play important roles. Finally, polynomials are basic objects to understand computationally from many perspectives (e.g., testing identities, factoring, learning, etc.), and this study interacts well with the study of coding theoretic questions regarding RM codes.

We shall next give a taste of three of these applications without diving into too many details. As the focus of this manuscript is on binary RM codes, we shall not discuss RM codes over larger fields. The readers interested in more applications of error correcting codes in theoretical computer science are encouraged to read the survey [149].

9.1 Low Degree Testing

As mentioned above, RM codes played a very important role in the early days of interactive proofs, including the original proof of the PCP theorem. The main property that was used is that RM codes are locally testable. Local testability is the task of deciding, with high probability, by querying few positions in a received word, whether it is a valid RM codeword or is far (in Hamming distance) from all RM codewords. Arguably these early local testing results for RM codes (starting with the testing algorithm for Hadamard codes, which are RM codes of degree one, due to Blum *et al.* [30]) were the main driving force that established the now thriving field of property testing.

Most of the applications described above rely on RM codes over large fields. Specifically, the RM codes used in the proof of the PCP theorem and in the work on hardness-amplification require the field size to be larger than the degree of the multivariate polynomials being evaluated. However, testing of binary RM codes also received a lot of attention in the CS literature [6], [27], [69], [80], [85].

To explain the problem of low degree testing we start with some basic notation. All the notions below can be naturally extended to larger fields. As before, for f, g: $\mathbb{F}_2^m \to \mathbb{F}_2$ we let $\delta(f,g) = \mathbb{P}_x\left(f(x) \neq g(x)\right)$ denote the relative-distance between f and g, where the probability is over x chosen uniformly at random from $\mathbb{F}_2^m$. Let $\delta_r(f) = \min_{g \in \mathcal{P}_{m,r}}\{\delta(f,g)\}$ denote the distance of f from the space of degree r polynomials. We say f is δ-far from g if $\delta(f,g) \geq \delta$ and δ-close otherwise. We say f is δ-far from the set of degree r polynomials if $\delta_r(f) \geq \delta$. The goal of low-degree testing is to design a test to distinguish the case where $\delta_r(f)$ is zero from the case where it is relatively large.

Definition 9.1 (Local Tester)**.** A (q, ϵ, δ)-local tester (for $\mathcal{P}_{m,r}$) is a probabilistic algorithm $T = T(m,r)$ that when given oracle access to f: $\mathbb{F}_2^m \mapsto \mathbb{F}_2$,[1] makes at most q queries to f and accepts $f \in \mathcal{P}_{m,r}$ with probability 1, while rejecting any $f \notin \mathcal{P}_{m,r}$, such that $\delta_r(f) \geq \delta$, with probability at least ϵ. We call ϵ the δ-soundness parameter of the algorithm.

We say T is *absolutely sound* if there exists $\epsilon > 0$ such that for every δ, r and m, $T = T(m,r)$ is a $(q, \epsilon \cdot \delta, \delta)$-local tester.

Roughly, absolute soundness means that the guarantee on the tester degrades "continuously" as the distance to $\mathcal{P}_{m,r}$ decreases.

For an affine subspace A in $\mathbb{F}_2^m$, let $\dim(A)$ denote its dimension. For function f: $\mathbb{F}_2^m \to \mathbb{F}_2$ and affine subspace A, let $f|_A$: $A \to \mathbb{F}_2$ denote the restriction of f to A. For a function f, we let $\deg(f)$ denote its degree as a polynomial. We use the fact that $f|_A$ can be viewed as a $\dim(A)$-variate polynomial with $\deg(f|_A) \leq \deg(f)$. A special subclass of tests for $\mathcal{P}_{m,r}$ would simply pick an affine subspace A of $\mathbb{F}_2^m$ and verify that $\deg(f|_A) \leq r$. The concept of testing dimension (defined below) captures the minimal dimension for which such a test has positive soundness, regardless of the number of variables.

Definition 9.2 (Testing Dimension)**.** For a non-negative r, the *testing dimension* of polynomials of degree r over $\mathbb{F}_2$ is the smallest integer t

[1] Oracle access means that the algorithm can ask for the value of f at any input of its choice. The number of queries is counted as part of the running time of the algorithm.

satisfying the following: For every positive integer m and every function $f\colon \mathbb{F}_2^m \to \mathbb{F}_2$ with $\deg(f) > r$, there exists an affine subspace A of dimension at most t such that $\deg(f|_A) > r$. We use t_r to denote the testing dimension for polynomials of degree r.

This notion was studied in [85] who proved the following fact.

Proposition 13. The testing dimension $t_r = r + 1$.

In other words, if a function is not a degree r polynomial then there is an affine subspace A of dimension at most $r + 1$ such that $\deg(f|_A) > r$. Observe that for every such affine subspace there is a codeword in $\mathrm{RM}(m, r)^\perp$ that is supported on A. Thus, another way of interpreting Proposition 13 is that dual codewords supported on affine subspaces of dimension $r + 1$ span $\mathrm{RM}(m, r)^\perp$. A natural question to ask is does any function of degree larger than r violates many of these dimension-$(r + 1)$ constraints? The test proposed by [85] relies on a positive answer to this question:

Description 1 (t-Dimensional (Degree r) Tester of [85])**.** Given oracle access to $f\colon \mathbb{F}_2^m \to \mathbb{F}_2$, pick a random affine subspace A with $\dim(A) = t$ and accept if $\deg(f|_A) \leq r$.

Kaufman and Ron [85] shows that the t_r-dimensional test, which has query complexity 2^{t_r} and that accepts any $f \in \mathcal{P}_{m,r}$ with probability 1, has δ-soundness roughly $\Omega(\delta 2^{-t_r})$. The works [27] and [69] proved that the test is absolutely sound. Specifically, if we let $\rho_r(f, t)$ denote the probability that the t-dimensional test rejects a function f, then they proved that

Theorem 29 ([27], [69])**.** There exist constants $\epsilon_1, \epsilon_2 > 0$ such that for every r and m and every function $f\colon \mathbb{F}_2^m \to \mathbb{F}_2$, it is the case that $\rho_r(f, t_r) \geq \min\{\epsilon_1 2^{t_r}\delta(f), \epsilon_2\}$. In other words the t_r-dimensional test rejects f with probability $\min\{\epsilon_1 2^{t_r}\delta(f), \epsilon_2\}$.

Theorem 29 shows that every function that is "far" from $\mathrm{RM}(m, r)$ violates a large fraction of the $(r + 1)$-dimensional constraints. The proof of the theorem is a bit technical so we will not get into it. Instead we will just give a high level overview of the proof.

The idea is to prove the claim by induction on m. Consider a function f that is δ-far from every degree r polynomial. For a "hyperplane", i.e., an $(m-1)$-dimensional affine subspace A of $\mathbb{F}_2^m$, let $f|_A$ denote the restriction of f to A. We first note that the test can be interpreted as first picking a random hyperplane A in $\mathbb{F}_2^m$ and then picking a random $(r+1)$-dimensional affine subspace A' inside A and testing if $f|_{A'}$ is a degree r polynomial. Now, if on every hyperplane A, $f|_A$ is still δ-far from degree r polynomials then we would be done by the inductive hypothesis. The insight of the analysis is understanding what happens when $f|_A$ is close to some degree r polynomial g_A for several (but just $O(2^r)$) hyperplanes. In this case they prove that one can "glue" the different polynomials g_A (each defined on some $(m-1)$-dimensional subspace within $\mathbb{F}_2^m$) into a degree-r polynomial g that agrees with all the g_A's. They then show that this polynomial is close to f, completing the argument.

9.2 Private Information Retrieval

A κ-server Private Information Retrieval (PIR) scheme allows a user to retrieve a data item of its choice from a database, replicated among κ servers, while preventing the servers from gaining information about the identity of this item (the servers are not allowed to communicate with each other). The goal is to design such PIR schemes that minimize the communication cost (defined as the worst-case number of bits transferred between the user and the servers in the protocol). This problem was introduced by Chor *et al.* [42] and attracted a lot of research, see [53] and references within.

We shall denote the database by an n-bit string $x \in \{0,1\}^n$ where the user, holding some retrieval index i, wishes to learn the i-th data bit x_i. In the case of a single server ($\kappa = 1$), a trivial solution is to send the entire database x to the user. The communication cost of this solution is n - the server sends all the n bits of the database to the user. This was shown to be optimal in [42]. As we wish to minimize the communication cost, Chor *et al.* suggested that the user accesses $\kappa > 1$ replicated copies of the database stored at different servers, requiring that each individual server gets absolutely no information about i.

We shall present a construction due to Beimel, Ishay and Kushilevitz that is based on encoding the database as a codeword of an appropriate RM code [21].

Let $x \in \{0,1\}^n$ be the database from which the user wishes to retrieve information. Let $r = 2\kappa + 1$ and choose m such that $\binom{m}{r} \geq n$. Clearly, $m = O(\kappa n^{\frac{1}{2\kappa+1}})$ suffices. We identify each index $i \in [n]$ with a unique r-tuple in $[m]$ (by choice of m there are at least n such tuples). Denote by $E(i) \in \{0,1\}^m$ the tuple corresponding to i. Let $f(y_1, \ldots, y_m)$ be a polynomial of degree r over $\mathbb{F}_2$ such that $f(E(i)) = x_i$. It is not hard to see that such f exists. Note that the polynomial $f \in \mathcal{P}_{m,r}$ encodes the database. Thus, we can imagine that our task is the following. The κ servers hold f and our goal is to obtain the value $f(E(i))$ by asking the the different servers for the value of f at different points, that have no correlation to f in a way that will allow us to compute $f(E(i))$.

For $j \in [\kappa]$ let $z_j = (z_{j,1}, \ldots, z_{j,\kappa}) \in (\mathbb{F}_2^m)^\kappa$. Consider the polynomial $\tilde{f}(z_1, \ldots, z_\kappa) = f(z_1 + \cdots + z_\kappa)$. Each monomial $M(z_1, \ldots, z_\kappa)$ of $\tilde{f}$ contains at most r variables $z_{j,i}$. Thus, there is some $j \in [\kappa]$ such that the number of variables from z_j appearing in M is at most $\lceil \frac{r}{\kappa} \rceil = 1$. We associate the monomial M with the jth server (if there is more than one such j then we associate M to the first server having this property).

Given an index $i \in [n]$ the user picks at random $\kappa - 1$ vectors $v_i \in \{0,1\}^m$ and sets $v_\kappa = E(i) - \sum_{j=1}^{\kappa-1} v_j$. Thus, $E(i) = \sum_{j=1}^{\kappa} v_j$. Observe that any $\kappa - 1$ vectors among $v_1, \ldots, v_\kappa$ are completely random and independent.

The user, who wishes to compute the value $f(E(i)) = f(v_1 + \cdots + v_\kappa)$, sends all vectors except the jth one to the jth server. Each of the servers now evaluates the monomials associated to it on its input vectors. Observe that by the way we associated monomials to servers, each server now holds a linear function in the variables unknown to it. That is, the jth server now holds a linear function $\ell_j(z_{j,1}, \ldots, z_{j,m})$. Next, each server sends its linear function to the user (by sending the coefficients). Finally the user computes $\sum_{j=1}^{\kappa} \ell_j(v_j) = f(v_1 + \cdots + v_\kappa) = f(E(i))$. Overall, the user sent $O(\kappa^2 m)$ bits and the servers communicated $O(\kappa m)$ bits. Thus the total communication is $O(\kappa^2 m) = O(\kappa^3 n^{\frac{1}{2\kappa-1}})$. As each server

received $\kappa - 1$ completely random vectors, the servers learn nothing about the retrieved index i.

We note that the best known construction of PIR schemes also relies on RM codes, but in a more complicated way [22]. It achieves communication cost of $n^{O\left(\frac{\log\log\kappa}{\kappa\log\kappa}\right)}$.

9.3 Compressed Sensing

The goal of compressed sensing is to recover sparse signals from very few measurements. The two fundamental questions in compressed sensing are: (i) how to construct suitable sensing matrices; (ii) how to efficiently recover the sparse signal from the measurements. The mathematical foundation of compressed sensing was laid in the seminal works [36]–[38], [47]. In particular, the Restricted Isometry Property (RIP) was formulated in [38]: A sensing matrix satisfies the k-Restricted Isometry Property if it acts as a near isometry on all k-sparse vectors, where k-sparse means that the vector has at most k non-zero entries. To ensure unique and stable reconstruction of k-sparse vectors, it is sufficient that the sensing matrix satisfies $2k$-RIP.

We use Φ to denote a sensing matrix of size $m \times n$. For a subset $\mathcal{S} \subseteq [n]$ of size k, we write $\Phi_{\mathcal{S}}$ to refer to the $m \times k$ submatrix of Φ formed of the columns with indices in $\mathcal{S}$. We say Φ is (k, δ)-RIP if

$$\left\|\Phi_{\mathcal{S}}^T\Phi_{\mathcal{S}} - I\right\|_2 \leq \delta \text{ for every subset } \mathcal{S} \subseteq [n] \text{ of size } |\mathcal{S}| = k,$$

where I is the identity matrix, and $\|\cdot\|_2$ is the spectral norm (the largest singular value). In the compressed sensing problem, k is the sparsity of the signal, m is the number of measurements, and n is the length of the signal. Both m and k are much smaller than n. The objective is to design a sensing matrix Φ such that every k-sparse signal $x \in \mathbb{R}^n$ can be recovered from the measurements Φx with small errors in terms of the ℓ_1 or ℓ_2 norm. The results of [38] showed that this requirement is satisfied as long as Φ is $(2k, \delta)$-RIP for some small δ.

As a natural variation of the original compressed sensing problem, one may not seek to recover all k-sparse vectors, but only aim to recover most of them. This variation leads to the Statistical Restricted Isometry Property (StRIP) of the sensing matrix, defined as follows.

Suppose that a subset $\mathcal{S} \subset N$ of size $|\mathcal{S}| = k$ is chosen uniformly at random from $[n]$. Then Φ is said to have the (k, δ, ϵ)-StRIP if

$$\mathbb{P}\left(\left\|\Phi_{\mathcal{S}}^{T}\Phi_{\mathcal{S}} - I\right\|_2 \geq \delta\right) < \epsilon.$$

It was shown in [17], [34], [35] that certain RM codes and Delsarte-Goethals codes (discussed in Section 2.7) form good sensing matrices and satisfy the Statistical Restricted Isometry Property. In particular, [17] showed that the number of measurements is $m = O(k)$ if we construct StRIP sensing matrices from RM codes and Delsarte-Goethals codes. Meanwhile, if we require the RIP property for the sensing matrix, then the number of measurements must be at least $m = \Omega(k \log(n/k))$. Therefore, the sensing matrices constructed from RM codes and Delsarte-Goethals codes are more efficient in the sense that they require much fewer measurements, although they provide weaker recovery guarantees than sensing matrices with RIP properties, e.g., matrices with random Gaussian or Bernoulli entries.

10

Open Problems

10.1 Capacity-Achieving Results

1. *Capacity-achieving for general BMS channels under the block-MAP decoder*

   In the constant-rate regime, the following two capacity-achieving results are known: (i) RM codes achieve the capacity of BEC under the block-MAP decoder [90]; (ii) RM codes achieve the capacity of all BMS channels under the bit-MAP decoder [122]. These two results are discussed in Sections 5.3 and 5.4, respectively. Since achieving capacity under the block-MAP decoder is stronger than achieving capacity under the bit-MAP decoder, and represents the classical definition of Shannon's capacity, the natural open problem is to show that RM codes achieve the capacity of all BMS channels under the block-MAP decoder.

2. *Relation between the twin-RM codes and RM codes*

   As mentioned in Section 6.3, the twin-RM code, obtained by retaining the low-entropy components of the squared RM code (proceeding in the RM ordering) is proved to achieve capacity

on any BMS. There are now three possible outcomes: (i) the twin-RM code is exactly the RM code, (ii) the twin-RM code is equivalent to the RM code, in that only a vanishing fraction of rows selected by the two codes are different, (iii) the twin-RM code is not equivalent to the RM code. If (i) or (ii) is true, then RM codes also achieve the capacity of all BMS channels. If (iii) is true instead, then the RM code is not capacity-achieving (for the considered BMS channel).

Conjecture 1. The twin-RM code is the RM code, i.e., with the notation of Section 6.3 where $U^n = R_n X^n$ and R_n is the squared RM code matrix, for $i, j \in [n]$,

$$|A_i| > |A_j| \implies H(U_i|U^{i-1}, Y^n) \geq H(U_j|U^{j-1}, Y^n), \quad (10.1)$$

in words, the conditional entropy is non-decreasing as we go to lower degree layers.

10.2 Weight Enumerator

1. *Unified bounds for two different regimes*

   As mentioned in Section 4, we now have two different approaches to bound the number of codewords in two different regimes. More precisely, the method proposed in [127] gives strong and in some cases nearly optimal bounds in the constant relative weight regime for constant rate codes. Yet this method does not produce meaningful bound when the code rate approaches 0 or in the small weight regime. On the other hand, the method in [134] gives good bound for small rate codes or in small weight regime, but it does not work well in the linear weight regime for constant rate codes. A natural open problem is thus to obtain a unified bound that is effective in both regimes.

2. *Use the capacity-achieving results for BEC to prove the conjecture for BSC*

   As discussed in Section 4.3.3, Samorodnitsky gave a nearly optimal upper bound in a certain linear weight regime for any code that

achieves capacity for the BEC. This in particular means that the weight distribution of RM codes in this regime is (nearly) the same as that of random codes. Since random codes achieve capacity of BSC, can we thus extend this result to prove that RM codes achieve capacity on the BSC?

3. *Close the gap between the upper and lower bounds for small weight codewords*

   There is a gap between the existing upper and lower bounds on small weight codewords; see Theorem 7 and Theorem 9. A natural open problem is to close this gap.

10.3 Algorithms

1. *Efficient decoding algorithms with near-ML performance for general code parameters*

   Till now, all the available decoding algorithms for RM codes only work well in one of the following three regimes: (i) low code rate regime—code rate is close to 0; (ii) high code rate regime—code rate is close to 1; (iii) short code length regime. When the code parameters are outside of these three regimes, currently available RM decoders have either an unbearably long running time or a much larger decoding error probability than that of an ML decoder. The major open problem in this direction is to design new decoding algorithms with low complexity and near-ML performance for RM codes with general parameters. From a practical perspective, if one could design such a decoder for code length up to 2048 and code rates around 0.5, it would certainly be considered as a breakthrough in this area.

2. *Efficient decoding algorithms with theoretical guarantees*

   The results of [134], as described in Section 5.1, show that $\mathrm{RM}(m, m/2 - O(\sqrt{m \log m}))$ can correct a fraction of $1/2 - o(1)$ random errors. Currently, the best algorithm that we have can

only handle RM codes with degrees up to $o(\sqrt{m})$ [130] (see Section 8.3). It is a natural open problem to extend these up to any constant fraction of errors.

References

[1] E. Abbe, A. Shpilka, and A. Wigderson, "Reed–Muller codes for random erasures and errors," *IEEE Transactions on Information Theory*, vol. 61, no. 10, pp. 5229–5252, 2015.

[2] E. Abbe and M. Ye, "Reed-Muller codes polarize," in *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, IEEE, pp. 273–286, 2019.

[3] E. Abbe, J. Hazla, and I. Nachum, "Almost–Reed–Muller codes achieve constant rates for random errors," *IEEE Transactions on Information Theory*, vol. 67, no. 12, pp. 8034–8050, 2021.

[4] E. Abbe, A. Shpilka, and M. Ye, "Reed–Muller codes: Theory and algorithms," *IEEE Transactions on Information Theory*, vol. 67, no. 6, pp. 3251–3277, 2021.

[5] E. Abbe and Y. Wigderson, "High-girth matrices and polarization," in *2015 IEEE International Symposium on Information Theory (ISIT)*, pp. 2461–2465, 2015.

[6] N. Alon, T. Kaufman, M. Krivelevich, S. Litsyn, and D. Ron, "Testing Reed–Muller codes," *IEEE Trans. Inf. Theory*, vol. 51, no. 11, pp. 4032–4039, 2005.

[7] Y. Altuğ and A. B. Wagner, "Moderate deviation analysis of channel coding: Discrete memoryless case," in *2010 IEEE International Symposium on Information Theory*, IEEE, pp. 265–269, 2010.

[8] Y. Altuğ and A. B. Wagner, "Moderate deviations in channel coding," *IEEE Transactions on Information Theory*, vol. 60, no. 8, pp. 4417–4426, 2014.

[9] E. Arikan, "A survey of Reed–Muller codes from polar coding perspective," in *2010 IEEE Information Theory Workshop on Information Theory (ITW 2010, Cairo)*, IEEE, pp. 1–5, 2010.

[10] E. Arikan, "Source polarization," in *2010 IEEE International Symposium on Information Theory*, pp. 899–903, 2010.

[11] 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, 2009.

[12] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, "Proof verification and the hardness of approximation problems," *Journal of the ACM (JACM)*, vol. 45, no. 3, pp. 501–555, 1998.

[13] A. Ashikhmin, G. Kramer, and S. ten Brink, "Extrinsic information transfer functions: Model and erasure channel properties," *IEEE Transactions on Information Theory*, vol. 50, no. 11, pp. 2657–2673, 2004.

[14] L. Babai, L. Fortnow, and C. Lund, "Nondeterministic exponential time has two-prover interactive protocols," in *Foundations of Computer Science, 1990. Proceedings., 31st Annual Symposium on*, IEEE, pp. 16–25, 1990.

[15] B. Barak, P. Gopalan, J. Håstad, R. Meka, P. Raghavendra, and D. Steurer, "Making the long code shorter," in *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pp. 370–379, 2012.

[16] A. Barg and G. D. Forney, "Random codes: Minimum distances and error exponents," *IEEE Transactions on Information Theory*, vol. 48, no. 9, pp. 2568–2573, 2002.

[17] A. Barg, A. Mazumdar, and R. Wang, "Restricted isometry property of random subdictionaries," *IEEE Transactions on Information Theory*, vol. 61, no. 8, pp. 4440–4450, 2015.

[18] P. Beame, S. O. Gharan, and X. Yang, "On the bias of Reed–Muller codes over odd prime fields," *arXiv preprint arXiv:1806.06973*, 2018.

[19] D. Beaver and J. Feigenbaum, "Hiding instances in multioracle queries," in *STACS 90*, Springer, 1990, pp. 37–48.

[20] Y. Be'ery and J. Snyders, "Optimal soft decision block decoders based on fast Hadamard transform," *IEEE Transactions on Information Theory*, vol. 32, no. 3, pp. 355–364, 1986.

[21] A. Beimel, Y. Ishai, and E. Kushilevitz, "General constructions for information-theoretic private information retrieval," *Journal of Computer and System Sciences*, vol. 71, no. 2, pp. 213–247, 2005.

[22] A. Beimel, Y. Ishai, E. Kushilevitz, and J. Raymond, "Breaking the $O(n^{1/(2k-1)})$ barrier for information-theoretic private information retrieval," in *43rd Symposium on Foundations of Computer Science (FOCS 2002), 16-19 November 2002, Vancouver, BC, Canada, Proceedings*, pp. 261–270, IEEE Computer Society, 2002.

[23] I. Ben-Eliezer, R. Hod, and S. Lovett, "Random low-degree polynomials are hard to approximate," *Computational Complexity*, vol. 21, no. 1, pp. 63–81, 2012.

[24] I. Benjamini, G. Kalai, and O. Schramm, "Noise sensitivity of boolean functions and applications to percolation," *Publications Mathématiques de l'Institut des Hautes Études Scientifiques*, vol. 90, no. 1, pp. 5–43, 1999.

[25] S. Bhandari, P. Harsha, R. Saptharishi, and S. Srinivasan, "Vanishing spaces of random sets and applications to Reed–Muller codes," in *37th Computational Complexity Conference, CCC 2022, July 20–23, 2022, Philadelphia, PA, USA*, S. Lovett, Ed., ser. LIPIcs, vol. 234, 31:1–31:14, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[26] A. Bhattacharyya, S. Kopparty, G. Schoenebeck, M. Sudan, and D. Zuckerman, "Optimal testing of reed–muller codes," in *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, ser. FOCS '10, pp. 488–497, 2010.

[27] A. Bhattacharyya, S. Kopparty, G. Schoenebeck, M. Sudan, and D. Zuckerman, "Optimal testing of Reed–Muller codes," in *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23–26, 2010, Las Vegas, Nevada, USA*, pp. 488–497, IEEE Computer Society, 2010.

[28] R. E. Blahut, *Algebraic codes for data transmission*. Cambridge university press, 2003.

[29] R. Blahut, "Hypothesis testing and information theory," *IEEE Transactions on Information Theory*, vol. 20, no. 4, pp. 405–417, 1974.

[30] M. Blum, M. Luby, and R. Rubinfeld, "Self-testing/correcting with applications to numerical problems," *J. Comput. Syst. Sci.*, vol. 47, no. 3, pp. 549–595, 1993.

[31] A. Bogdanov and E. Viola, "Pseudorandom bits for polynomials," *SIAM J. Comput.*, vol. 39, no. 6, pp. 2464–2486, 2010.

[32] J. Bourgain and G. Kalai, "Influences of variables and threshold intervals under group symmetries," *Geometric and Functional Analysis*, vol. 7, no. 3, pp. 438–461, 1997.

[33] S. ten Brink, "Convergence of iterative decoding," *Electronics Letters*, vol. 35, no. 10, pp. 806–808, 1999.

[34] R. Calderbank, S. Howard, and S. Jafarpour, "Construction of a large class of deterministic sensing matrices that satisfy a statistical isometry property," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 358–374, 2010.

[35] R. Calderbank and S. Jafarpour, "Reed Muller sensing matrices and the LASSO," in *International Conference on Sequences and Their Applications*, Springer, pp. 442–463, 2010.

[36] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[37] E. J. Candes, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, vol. 59, no. 8, pp. 1207–1223, 2006.

[38] E. J. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?" *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5406–5425, 2006.

[39] C. Carlet and P. Gaborit, "On the construction of balanced Boolean functions with a good algebraic immunity," in *Proceedings. International Symposium on Information Theory, 2005. ISIT 2005.*, IEEE, pp. 1101–1105, 2005.

[40] D. Chase, "Class of algorithms for decoding block codes with channel measurement information," *IEEE Transactions on Information Theory*, vol. 18, no. 1, pp. 170–182, 1972.

[41] X. Chen and M. Ye, "Cyclically equivariant neural decoders for cyclic codes," in *International Conference on Machine Learning*, PMLR, pp. 1771–1780, 2021.

[42] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *J. ACM*, vol. 45, no. 6, pp. 965–981, 1998.

[43] D. J. Costello and G. D. Forney, "Channel coding: The road to channel capacity," *Proceedings of the IEEE*, vol. 95, no. 6, pp. 1150–1177, 2007.

[44] P. Delsarte and J.-M. Goethals, "Alternating bilinear forms over GF(q)," *Journal of Combinatorial Theory, Series A*, vol. 19, no. 1, pp. 26–50, 1975.

[45] F. Didier, "A new upper bound on the block error probability after decoding over the erasure channel," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4496–4503, 2006.

[46] R. L. Dobrushin, "Mathematical problems in the Shannon theory of optimal coding of information," in *Proc. 4th Berkeley Symp. Mathematics, Statistics, and Probability*, vol. 1, pp. 211–252, 1961.

[47] D. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[48] I. Dumer, "Recursive decoding and its performance for low-rate Reed-Muller codes," *IEEE Transactions on Information Theory*, vol. 50, no. 5, pp. 811–823, 2004.

[49] I. Dumer, "Soft-decision decoding of Reed-Muller codes: A simplified algorithm," *IEEE Transactions on Information Theory*, vol. 52, no. 3, pp. 954–963, 2006.

[50] I. Dumer and P. Farrell, "Erasure correction performance of linear block codes," in *Workshop on Algebraic Coding*, Springer, pp. 316–326, 1993.

[51] I. Dumer and K. Shabunov, "Soft-decision decoding of Reed-Muller codes: Recursive lists," *IEEE Transactions on information theory*, vol. 52, no. 3, pp. 1260–1266, 2006.

[52] I. M. Duursma and R. Kötter, "Error-locating pairs for cyclic codes," *IEEE Transactions on Information Theory*, vol. 40, no. 4, pp. 1108–1121, 1994.

[53] Z. Dvir and S. Gopi, "2-server PIR with subpolynomial communication," *J. ACM*, vol. 63, no. 4, pp. 39:1–39:15, 2016.

[54] A. G. i Fabregas, I. Land, and A. Martinez, "Extremes of random coding error exponents," in *2011 IEEE International Symposium on Information Theory Proceedings*, IEEE, pp. 2896–2898, 2011.

[55] D. Fathollahi, N. Farsad, S. A. Hashemi, and M. Mondelli, "Sparse multi-decoder recursive projection aggregation for Reed–Muller codes," in *2021 IEEE International Symposium on Information Theory (ISIT)*, IEEE, pp. 1082–1087, 2021.

[56] A. Fazeli, H. Hassani, M. Mondelli, and A. Vardy, "Binary linear codes with optimal scaling: Polar codes with large kernels," *IEEE Transactions on Information Theory*, vol. 67, no. 9, pp. 5693–5710, 2021.

[57] *Final report of 3GPP TSG RAN WG1 #87 v1.0.0.* URL: http://www.3gpp.org/ftp/tsg_ran/WG1_RL1/TSGR1_87/Report/.

[58] D. Forney, "On exponential error bounds for random codes on the BSC," *unpublished manuscript*, 2001.

[59] E. Friedgut and G. Kalai, "Every monotone graph property has a sharp threshold," *Proceedings of the American Mathematical Society*, vol. 124, no. 10, pp. 2993–3002, 1996.

[60] R. Gallager, "A simple derivation of the coding theorem and some applications," *IEEE Transactions on Information Theory*, vol. 11, no. 1, pp. 3–18, 1965.

[61] R. Gallager, "The random coding bound is tight for the average code (corresp.)," *IEEE Transactions on Information Theory*, vol. 19, no. 2, pp. 244–246, 1973.

[62] W. Gasarch, "A survey on private information retrieval," in *Bulletin of the EATCS*, Citeseer, 2004.

[63] M. Geiselhart, A. Elkelesh, M. Ebada, S. Cammerer, and S. ten Brink, "Automorphism ensemble decoding of Reed–Muller codes," *IEEE Transactions on Communications*, vol. 69, no. 10, pp. 6424–6438, 2021.

[64] R. R. Green, "A serial orthogonal decoder," *JPL Space Programs Summary*, vol. 37, pp. 247–253, 1966.

[65] V. Guruswami, A. Riazanov, and M. Ye, "Arikan meets Shannon: Polar codes with near-optimal convergence to channel capacity," in *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 552–564, 2020.

[66] V. Guruswami and P. Xia, "Polar codes: Speed of polarization and polynomial gap to capacity," *IEEE Transactions on Information Theory*, vol. 61, no. 1, pp. 3–16, 2015.

[67] R. W. Hamming, "Error detecting and error correcting codes," *The Bell system technical journal*, vol. 29, no. 2, pp. 147–160, 1950.

[68] A. Hammons, P. Kumar, A. Calderbank, N. Sloane, and P. Sole, "The $\mathbb{Z}_4$-linearity of Kerdock, Preparata, Goethals, and related codes," *IEEE Transactions on Information Theory*, vol. 40, no. 2, pp. 301–319, 1994.

[69] E. Haramaty, A. Shpilka, and M. Sudan, "Optimal testing of multivariate polynomials over small prime fields," *SIAM J. Comput.*, vol. 42, no. 2, pp. 536–562, 2013.

[70] H. Hassani, S. Kudekar, O. Ordentlich, Y. Polyanskiy, and R. Urbanke, "Almost optimal scaling of Reed-Muller codes on BEC and BSC channels," in *2018 IEEE International Symposium on Information Theory (ISIT)*, IEEE, pp. 311–315, 2018.

[71] S. H. Hassani, K. Alishahi, and R. L. Urbanke, "Finite-length scaling for polar codes," *IEEE Transactions on Information Theory*, vol. 60, no. 10, pp. 5875–5898, 2014.

[72] S. H. Hassani, S. B. Korada, and R. Urbanke, "The compound capacity of polar codes," in *2009 47th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 16–21, 2009.

[73] S. H. Hassani, R. Mori, T. Tanaka, and R. L. Urbanke, "Rate-dependent analysis of the asymptotic behavior of channel polarization," *IEEE Transactions on Information Theory*, vol. 59, no. 4, pp. 2267–2276, 2013.

[74] S. H. Hassani and R. Urbanke, "Universal polar codes," in *2014 IEEE International Symposium on Information Theory*, pp. 1451–1455, 2014.

[75] M. Hayashi, "Information spectrum approach to second-order coding rate in channel coding," *IEEE Transactions on Information Theory*, vol. 55, no. 11, pp. 4947–4966, 2009.

[76] J. Hazla, A. Samorodnitsky, and O. Sberlo, "On codes decoding a constant fraction of errors on the BSC," in *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21–25, 2021*, S. Khuller and V. V. Williams, Eds., pp. 1479–1488, ACM, 2021.

[77] T. Helleseth, T. Kløve, and V. I. Levenshtein, "The simplex codes and other even-weight binary linear codes for error correction," *IEEE Transactions on Information Theory*, vol. 50, no. 11, pp. 2818–2823, 2004.

[78] T. Helleseth, T. Kløve, and V. I. Levenshtein, "Error-correction capability of binary linear codes," *IEEE Transactions on Information Theory*, vol. 51, no. 4, pp. 1408–1423, 2005.

[79] W. C. Huffman and V. Pless, *Fundamentals of error-correcting codes*. Cambridge university press, 2010.

[80] C. S. Jutla, A. C. Patthak, A. Rudra, and D. Zuckerman, "Testing low-degree polynomials over prime fields," *Random Struct. Algorithms*, vol. 35, no. 2, pp. 163–193, 2009.

[81] T. Kasami, S. Lin, and W. Peterson, "New generalizations of the Reed–Muller codes–I: Primitive codes," *IEEE Transactions on Information Theory*, vol. 14, no. 2, pp. 189–199, 1968.

[82] T. Kasami and N. Tokura, "On the weight structure of reed–muller codes," *IEEE Transactions on Information Theory*, vol. 16, no. 6, pp. 752–759, 1970.

[83] T. Kasami, N. Tokura, and S. Azumi, "On the weight enumeration of weights less than 2.5 d of reed–muller codes," *Information and Control*, vol. 30, no. 4, pp. 380–395, 1976.

[84] T. Kaufman, S. Lovett, and E. Porat, "Weight distribution and list-decoding size of Reed–Muller codes," *IEEE Transactions on Information Theory*, vol. 58, no. 5, pp. 2689–2696, 2012.

[85] T. Kaufman and D. Ron, "Testing polynomials over general fields," *SIAM J. Comput.*, vol. 36, no. 3, pp. 779–802, 2006.

[86] A. M. Kerdock, "A class of low-rate nonlinear binary codes," *Information and Control*, vol. 20, no. 2, pp. 182–187, 1972.

[87] S. Kopparty and A. Potukuchi, "Syndrome decoding of Reed–Muller codes and tensor decomposition over finite fields," in *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pp. 680–691, 2018.

[88] S. B. Korada, A. Montanari, E. Telatar, and R. Urbanke, "An empirical scaling law for polar codes," in *2010 IEEE International Symposium on Information Theory*, pp. 884–888, 2010.

[89] S. Kudekar, S. Kumar, M. Mondelli, H. D. Pfister, E. Şaşoğlu, and R. Urbanke, "Reed-Muller codes achieve capacity on erasure channels," in *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing (STOC)*, ACM, pp. 658–669, 2016.

[90] S. Kudekar, S. Kumar, M. Mondelli, H. D. Pfister, E. Şaşoğlu, and R. Urbanke, "Reed–Muller codes achieve capacity on erasure channels," *IEEE Transactions on Information Theory*, vol. 63, no. 7, pp. 4298–4316, 2017.

[91] S. Kudekar, S. Kumar, M. Mondelli, H. D. Pfister, and R. Urbanke, "Comparing the bit-map and block-map decoding thresholds of Reed–Muller codes on bms channels," in *2016 IEEE International Symposium on Information Theory (ISIT)*, pp. 1755–1759, 2016.

[92] G. Li, M. Ye, and S. Hu, abs+ polar codes: Exploiting more linear transforms on adjacent bits, arXiv:2209.02461, 2022.

[93] G. Li, M. Ye, and S. Hu, adjacent-bits-swapped polar codes: A new code construction to speed up polarization, arXiv:2202.04454, 2022.

[94] M. Lian, C. Häger, and H. D. Pfister, "Decoding Reed–Muller codes using redundant code constraints," in *2020 IEEE International Symposium on Information Theory (ISIT)*, IEEE, pp. 42–47, 2020.

[95] S. Lin, "RM codes are not so bad," in *IEEE Inform. Theory Workshop*, Invited talk, 1993.

[96] S. Lin and D. J. Costello, *Error control coding*. Prentice hall, 2001.

[97] J. H. van Lint, "Kerdock codes and Preparata codes," *Congressus Numerantium*, vol. 39, pp. 25–41, 1983.

[98] J. H. van Lint, *Introduction to coding theory*. Springer, 1999.

[99] F. J. MacWilliams and N. J. A. Sloane, *The theory of error-correcting codes*. Elsevier, 1977.

[100] J. MacWilliams, "A theorem on the distribution of weights in a systematic code," *Bell System Technical Journal*, vol. 42, no. 1, pp. 79–94, 1963.

[101] A. V. Makkuva, X. Liu, M. V. Jamali, H. Mahdavifar, S. Oh, and P. Viswanath, "KO codes: Inventing nonlinear encoding and decoding for reliable wireless communication via deep-learning," in *International Conference on Machine Learning*, PMLR, pp. 7368–7378, 2021.

[102] C. Méasson, A. Montanari, and R. Urbanke, "Maxwell construction: The hidden bridge between iterative and maximum a posteriori decoding," *IEEE Transactions on Information Theory*, vol. 54, no. 12, pp. 5277–5307, 2008.

[103] M. Mitzenmacher and E. Upfal, *Probability and Computing – Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[104] M. Mondelli, S. H. Hassani, and R. L. Urbanke, "From polar to Reed-Muller codes: A technique to improve the finite-length performance," *IEEE Transactions on Communications*, vol. 62, no. 9, pp. 3084–3091, 2014.

[105] M. Mondelli, S. H. Hassani, and R. L. Urbanke, "Unified scaling of polar codes: Error exponent, scaling exponent, moderate deviations, and error floors," *IEEE Transactions on Information Theory*, vol. 62, no. 12, pp. 6698–6712, 2016.

[106] M. Mondelli, S. H. Hassani, and R. L. Urbanke, "Construction of polar codes with sublinear complexity," *IEEE Transactions on Information Theory*, vol. 65, no. 5, pp. 2782–2791, 2019.

[107] R. Mori and T. Tanaka, "Performance and construction of polar codes on symmetric binary-input memoryless channels," in *2009 IEEE International Symposium on Information Theory*, pp. 1496–1500, 2009.

[108] R. Mori and T. Tanaka, "Performance of polar codes with the construction using density evolution," *IEEE Communications Letters*, vol. 13, no. 7, pp. 519–521, 2009.

[109] D. E. Muller, "Application of Boolean algebra to switching circuit design and to error detection," *Transactions of the IRE Professional Group on Electronic Computers*, no. 3, pp. 6–12, 1954.

[110] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be'ery, "Deep learning methods for improved decoding of linear codes," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 119–131, 2018.

[111] A. W. Nordstrom and J. P. Robinson, "An optimum nonlinear code," *Information and Control*, vol. 11, no. 5-6, pp. 613–616, 1967.

[112] R. O'Donnell, *Analysis of boolean functions*. Cambridge University Press, 2014.

[113] R. Pedarsani, S. H. Hassani, I. Tal, and E. Telatar, "On the construction of polar codes," in *2011 IEEE International Symposium on Information Theory Proceedings*, pp. 11–15, 2011.

[114] R. Pellikaan, "On decoding by error location and dependent sets of error positions," *Discrete Mathematics*, vol. 106-107, pp. 369–381, 1992.

[115] G. Poltyrev, "Bounds on the decoding error probability of binary linear codes via their spectra," *IEEE Transactions on Information Theory*, vol. 40, no. 4, pp. 1284–1292, 1994.

[116] Y. Polyanskiy, H. V. Poor, and S. Verdú, "Channel coding rate in the finite blocklength regime," *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2307–2359, 2010.

[117] Y. Polyanskiy and S. Verdú, "Channel dispersion and moderate deviations limits for memoryless channels," in *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, IEEE, pp. 1334–1339, 2010.

[118] M. O. Rabin, "Efficient dispersal of information for security, load balancing, and fault tolerance," *Journal of the ACM (JACM)*, vol. 36, no. 2, pp. 335–348, 1989.

[119] A. Rao and O. Sprumont, on list decoding transitive codes from random errors, 2022.

[120] A. A. Razborov, "Lower bounds on the size of bounded depth circuits over a complete basis with logical addition," *Math. Notes*, vol. 41, no. 4, pp. 333–338, 1987.

[121] I. Reed, "A class of multiple-error-correcting codes and the decoding scheme," *Transactions of the IRE Professional Group on Information Theory*, vol. 4, no. 4, pp. 38–49, 1954.

[122] G. Reeves and H. D. Pfister, "Reed–Muller codes achieve capacity on BMS channels," arXiv:2110.14631, 2021.

[123] T. Richardson and R. Urbanke, *Modern coding theory*. Cambridge university press, 2008.

[124] R. Rossignol, "Threshold for monotone symmetric properties through a logarithmic sobolev inequality," *The Annals of Probability*, vol. 34, no. 5, pp. 1707–1725, 2006.

[125] B. Sakkour, "Decoding of second order Reed-Muller codes with a large number of errors," in *IEEE Information Theory Workshop*, IEEE, pp. 176–178, 2005.

[126] A. Samorodnitsky, "An improved bound on ℓ_q norms of noisy functions," *arXiv preprint arXiv:2010.02721*, 2020.

[127] A. Samorodnitsky, "An upper bound on ℓ_q norms of noisy functions," *IEEE Transactions on Information Theory*, vol. 66, no. 2, pp. 742–748, 2020.

[128] A. Samorodnitsky and O. Sberlo, "On codes decoding a constant fraction of errors on the BSC," arXiv:2008.07236, 2020.

[129] E. Santi, C. Hager, and H. D. Pfister, "Decoding Reed-Muller codes using minimum-weight parity checks," in *2018 IEEE International Symposium on Information Theory (ISIT)*, IEEE, pp. 1296–1300, 2018.

[130] R. Saptharishi, A. Shpilka, and B. L. Volk, "Efficiently decoding Reed–Muller codes from random errors," *IEEE Transactions on Information Theory*, vol. 63, no. 4, pp. 1954–1960, 2017.

[131] E. Şaşoğlu, "Polar coding theorems for discrete systems," Ph.D. dissertation, EPFL, 2011.

[132] E. Şaşoğlu, "Polarization and polar codes," *Foundations and Trends® in Communications and Information Theory*, vol. 8, no. 4, pp. 259–381, 2012.

[133] E. Şaşoğlu and L. Wang, "Universal polarization," *IEEE Transactions on Information Theory*, vol. 62, no. 6, pp. 2937–2946, 2016.

[134] O. Sberlo and A. Shpilka, "On the performance of Reed–Muller codes with respect to random errors and erasures," in *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SIAM, pp. 1357–1376, 2020.

[135] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[136] A. Shamir, "Ip= pspace," *Journal of the ACM (JACM)*, vol. 39, no. 4, pp. 869–877, 1992.

[137] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[138] C. E. Shannon, R. G. Gallager, and E. R. Berlekamp, "Lower bounds to error probability for coding on discrete memoryless channels. I," *Information and Control*, vol. 10, no. 1, pp. 65–103, 1967.

[139] A. Ta-Shma, D. Zuckerman, and S. Safra, "Extractors from reed–muller codes," *J. Comput. Syst. Sci.*, vol. 72, no. 5, pp. 786–812, 2006.

[140] N. Shulman and M. Feder, "Random coding techniques for nonrandom codes," *IEEE Transactions on Information Theory*, vol. 45, no. 6, pp. 2101–2104, 1999.

[141] V. M. Sidel'nikov and A. S. Pershakov, "Decoding of Reed-Muller codes with a large number of errors," *Problemy Peredachi Informatsii*, vol. 28, no. 3, pp. 80–94, 1992.

[142] N. Sloane and E. Berlekamp, "Weight enumerator for second-order Reed–Muller codes," *IEEE Transactions on Information Theory*, vol. 16, no. 6, pp. 745–751, 1970.

[143] N. Sloane and E. Berlekamp, "Weight enumerator for second-order Reed–Muller codes," *IEEE Transactions on Information Theory*, vol. 16, no. 6, pp. 745–751, 1970.

[144] V. Strassen, "Asymptotische abschatzugen in shannon's informationstheorie," in *Transactions of the Third Prague Conference on Information Theory etc, 1962. Czechoslovak Academy of Sciences, Prague*, pp. 689–723, 1962.

[145] I. Tal and A. Vardy, "How to construct polar codes," *IEEE Transactions on Information Theory*, vol. 59, no. 10, pp. 6562–6582, 2013.

[146] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2213–2226, 2015.

[147] L. G. Tallini and B. Bose, "Reed–Muller codes, elementary symmetric functions and asymmetric error correction," in *2011 IEEE International Symposium on Information Theory Proceedings*, IEEE, pp. 1051–1055, 2011.

[148] J. P. Tillich and G. Zémor, "Discrete isoperimetric inequalities and the probability of a decoding error," *Combinatorics, Probability and Computing*, vol. 9, no. 5, pp. 465–479, 2000.

[149] L. Trevisan, "Some applications of coding theory in computational complexity," *CoRR*, vol. cs.CC/0409044, 2004.

[150] H. P. Wang and I. M. Duursma, "Polar codes' simplicity, random codes' durability," *IEEE Transactions on Information Theory*, vol. 67, no. 3, pp. 1478–1508, 2021.

[151] L. Weiss, "On the strong converse of the coding theorem for symmetric channels without memory," *Quarterly of Applied Mathematics*, vol. 18, no. 3, pp. 209–214, 1960.

[152] J. Wolfowitz, "The coding of messages subject to chance errors," *Illinois Journal of Mathematics*, vol. 1, no. 4, pp. 591–606, 1957.

[153] M. Ye and E. Abbe, "Recursive projection-aggregation decoding of Reed–Muller codes," *IEEE Transactions on Information Theory*, vol. 66, no. 8, pp. 4948–4965, 2020.

[154] S. Yekhanin, "Locally decodable codes," *Foundations and Trends® in Theoretical Computer Science*, vol. 6, no. 3, pp. 139–255, 2012.