Local List-Decoding of Reed-Muller Codes over $\mathbb{F}_2$

Original paper by Gopalan, Klivans, and Zuckerman [6, 7]

Sahil Singla
Computer Science Department
Carnegie Mellon University
ssingla@cmu.edu

Manzil Zaheer
Electrical and Computer Engineering
Carnegie Mellon University
manzil@cmu.edu

December 7, 2014

Abstract

In this report we study the paper titled, "List-Decoding Reed-Muller Codes over Small Fields" by Parikshit Gopalan, Adam R. Klivans and David Zuckerman [6, 7]. The Reed-Muller codes are multivariate generalization of Reed-Solomon codes, which use polynomials in multiple variables to encode messages through functional encoding. Next, any list decoding problem for a code has two facets: Firstly, what is the maximal radius up to which any ball of that radius contains only a constant number of codewords. Secondly, how to find these codewords in an efficient manner.

The paper by GKZ presents the first local list-decoding algorithm for the r-th order Reed-Muller code RM(r, m) over $\mathbb{F}_2$ for $r \geq 2$. Given an oracle for a received word $R : \mathbb{F}_2^m \to \mathbb{F}_2$, their randomized local list-decoding algorithm produces a list containing all degree r polynomials within relative distance $(2^{-r} - \epsilon)$ from R for any $\epsilon > 0$ in time $poly(m^r, \epsilon^r)$. Since RM(r, m) has relative distance 2^r, their algorithm beats the Johnson bound for $r \geq 2$. Moreover, as the list size could be exponential in m at radius 2^{-r}, their bound is optimal in the local setting.

Contents

1 Introduction

We begin by providing basic definitions of code and comparing the two approaches – unique and list – for decoding in Section 1.1. Then we go over the Reed-Muller codes (RM Codes) and its unique decoding approaches in Section 1.2. Next, marketing of the GKZ approach for locally list decoding RM codes is carried out in Section 1.3. Then we proceed to describe the known list decoding method for Hadamard codes and port the method to RM codes in Section 2. In doing so we face two major obstacles, which we tackle over the next sections 3-4. Finally we conclude in Section 5.

1.1 Basics of Code

Definition 1 *A code over a finite field $\mathbb{F}$ is a triplet $(\mathcal{C}, \mathsf{Enc}, \mathsf{Dec})$ consisting of:*

- ***Codebook:*** *The codebook $\mathcal{C}$ is a subset of $\mathbb{F}^n$. The elements of the codebook, i.e. $\mathbf{c} \in \mathcal{C}$ are called codewords and the parameter n is called the blocklength.*
- ***Encoding Map:*** *Associated with a codebook is an encoding map $\mathsf{Enc} : \mathcal{M} \to \mathbb{F}^n$ which maps the message set $\mathcal{M}$, identified in some canonical way with $\{1, 2, ..., |\mathcal{C}|\}$ say, to codewords belonging to $\mathbb{F}^n$. The codebook can then be also defined the image of the encoding map.*
- ***Decoding Map:*** *Associated with a codebook is an decoding map $\mathsf{Dec} : \mathbb{F}^n \to \mathcal{M}$ which maps each vector in $\mathbb{F}^n$ to some message in the message set $\mathcal{M}$, identified in some canonical way with $\{1, 2, ..., |\mathcal{C}|\}$ say.*

Ideally, for any $\mathbf{x} \in \mathcal{M}$ we would like to have $\mathsf{Dec}(\mathsf{Enc}(\mathbf{x})+\text{noise}) = \mathbf{x}$, for every "reasonable" noise pattern that the channel might induce. In Section 1.1.2 and 1.1.3 we will see two meanings of "reasonable" noise pattern and how well we can recover the original message.

Definition 2 *The rate of a codebook $\mathcal{C} \in \mathbb{F}^n$, denoted $R(\mathcal{C})$, is defined by:*

$$R(\mathcal{C}) = \frac{\log |\mathcal{C}|}{n \log |\mathbb{F}|} \tag{1}$$

Definition 3 *The minimum distance, or simply distance, of a codebook $\mathcal{C}$, denoted by $\Delta(\mathcal{C})$, is defined to be the minimum Hamming distance between two distinct codewords of $\mathcal{C}$, i.e.*

$$\Delta(\mathcal{C}) = \min_{\mathbf{c}_1 \neq \mathbf{c}_2 \in \mathcal{C}} \Delta(\mathbf{c}_1, \mathbf{c}_2) \tag{2}$$

The relative distance of $\mathcal{C}$, denoted $\delta(\mathcal{C})$, is the normalized quantity $\Delta(\mathcal{C})/n$, where n is the block length of $\mathcal{C}$.

The general goal in coding theory is to come up with a code having high rate and minimum distance along with computationally efficient encoder and decoder. A general code might have no structure and not admit any representation other than listing the entire codebook. We now focus on an important subclass of codes with additional structure called linear codes.

Definition 4 *Let $\mathbb{F}$ be a field and $\mathcal{C} \subseteq \mathbb{F}^n$ is a subspace of $\mathbb{F}^n$ then $\mathcal{C}$ is said to be a linear code.*

If $\mathcal{C}$ has dimension k, then a matrix $\mathbf{G}$ consisting of columns as basis of the subspace is called the generator matrix for $\mathcal{C}$. The generator matrix $\mathbf{G}$ provides the encoding map, $\mathsf{Enc} : \mathbb{F}^k \to \mathbb{F}^n$ for the linear code as linear transformation, in which a a message $\mathbf{x} \in \mathbb{F}^n$ (thought of as a column vector) is encoded as the codeword $\mathbf{Gx} \in \mathcal{C} \subseteq \mathbb{F}^n$. Although, even this additional structure does not immediately give rise to efficient decoders. Mostly efficient decoders are designed specially for each class of linear code, though there are some general strategies like *Syndrome decoding* or *Coset* based nearest neighbour.

Examples: Parity check codes, Cyclic codes, Hamming codes, Golay codes, Reed-Solomon codes, BCH codes, Reed-Muller codes, Low density parity check codes.

1.1.1 Johnson Bound

Among the rich set of bounds available in coding theory, we will make use of Johnson bound for the problem of list decoding RM codes, which we state below:

Theorem 1 *For any code $\mathcal{C}$ with distance δn and any $\mathbf{r} \in \{0,1\}^n$*

- *Number of $\mathbf{c} \in \mathcal{C}$ such that $\delta(\mathbf{r}, \mathbf{c}) < J(\delta) - \gamma$ is at most $O(\gamma^{-2})$*
- *Number of $\mathbf{c} \in \mathcal{C}$ such that $\delta(\mathbf{r}, \mathbf{c}) < J(\delta)$ is at most $2n$*

where $J(\delta) = \frac{1}{2}(1 - \sqrt{1 - 2\delta})$.

1.1.2 Unique-Decoding

As discussed earlier, for decoding problem we would ideally like to have $\mathsf{Dec}(\mathsf{Enc}(\mathbf{x}) + \text{noise}) = \mathbf{x}$ for any $\mathbf{x} \in \mathcal{M}$. Now suppose we have a code $\mathcal{C}$ with minimum distance d. Then distance between any two codewords is atleast d. So for any $\mathbf{r} \in \mathbb{F}^n$, there can be only one codeword within a distance of $(d-1)/2$ from $\mathbf{r}$ (follows from the triangle inequality). Consequently, if the received word $\mathbf{r}$ has at most $\eta < (d-1)/2n$ error fraction, then the transmitted codeword can be uniquely identified. For example, in Figure 1 if the error rate is limited to

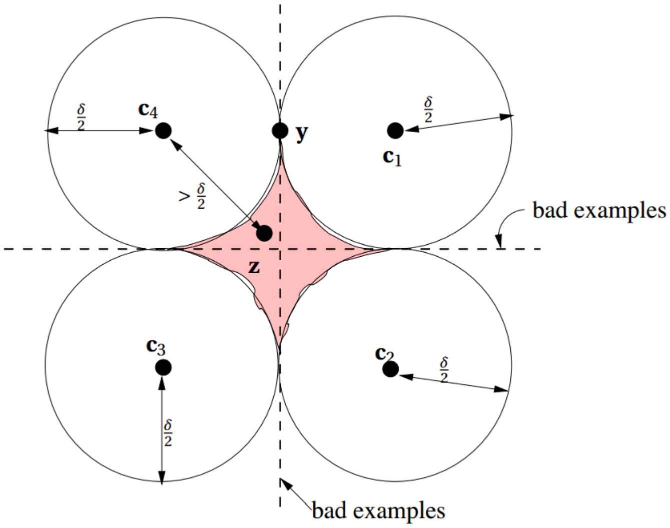


Figure 1: Unique and List decoding. Figure copied from [10]

$\eta = (d-1)/2n$, then the received word $\mathbf{r}$ lies within the ball of radius $(d-1)/2$ from the transmitted codeword, and hence can be uniquely decoded. Under such decoding scheme, when using a code of minimum distance d, a noise pattern of $d/2$ or more errors cannot always be corrected. Next we explore how to break this barrier of decodiblity $d/2$

1.1.3 List-Decoding

To decode *error correcting codes* beyond half the minimum distance, the concept of *list decoding* was introduced by Elias [3] and Wozencraft [16]. The objective of list decoding is to output all the codewords within a specified radius around the received word. After the seminal results of Goldreich and Levin [4] and Sudan [14] which gave list decoding algorithms for the Hadamard code and the Reed-Solomon code respectively, there has been tremendous progress in designing list decodable codes. Formally, a code is said to be list decodable when:

Definition 5 *Given $0 \leq \rho \leq 1, L \geq 1$, a code $\mathcal{C} \subseteq \mathbb{F}^n$ is (η, L)-list decodable if for every received word $\mathbf{r} \in \mathbb{F}^n$, we have*

$$|\{\mathbf{c} \in \mathcal{C} : \Delta(\mathbf{r}, \mathbf{c}) \leq \eta n\}| \leq L \tag{3}$$

Now if the fraction of errors that occurred during transmission is at most η then the transmitted codeword is guaranteed to be contained in the output list. Note that in the traditional communication setup, we need to recover the transmitted message. In such a scenario, outputting a list might not be useful. A quick fix to this problem is by declaring a decoding error if list size > 1. The gain over unique decoding comes from the fact that for most error patterns (of weight significantly more than half the distance of the code) the output list size is at most one (provided large enough block length of the code was chosen). In terms of Figure 1, it means that number of vectors in dotted lines is insignificant compared to volume of shaded area (for large enough block length of the code). [10]

Also algorithmically, we would like to have an efficient list-decoding algorithm. Otherwise the decoding problem can become trivial: one can always go through all the codewords in the code and pick the ones that are within η (relative) Hamming distance of the received word. For further details please check out the excellent surveys written by Guruswami [9, 8] and Sudan [15].

1.2 Reed-Muller Codes

Reed-Muller codes (RM codes) were discovered by David E. Muller in 1954 [12] and Irving S. Reed proposed the majority logic decoding for the first time [13]. Although RM codes did not find many applications for traditional communication tasks (except for some limited historical use in deep-space communications), RM codes are very useful in the design of probabilistically checkable proofs in computational complexity theory.

Basically RM codes is a family of linear codes defined over a finite field $\mathbb{F}_q$ of size q. The message space consists of degree $\leq r$ polynomials in m variables over $\mathbb{F}_q$ and the codewords are evaluation of these polynomials on $\mathbb{F}_q^m$. Then RM code $RM_q(m, r)$ is defined as follows:

Definition 6 *Given a field size q, a number m of variables, and a total degree bound r, the* $\mathrm{RM}_q[m, r]$ *code is the linear code over $\mathbb{F}_q$ defined by the encoding map:*

$$\mathsf{Enc} : f(X_1, ..., X_m) \rightarrow \langle f(\alpha) \rangle_{\alpha \in \mathbb{F}_q^m}$$

applies to the domain of all polynomials in $\mathbb{F}_q[X_1, ..., X_m]$ of total degree $\mathsf{deg}(f) \leq r$.

Additionally RM code belongs to the classes of locally testable codes and locally decodable codes.

In this report we restrict ourselves to the binary field. For the binary case, i.e. $q = 2$ the parameters of the RM code are as follows:

- Block length: $n = 2^m$
- Dimension: $k = \sum_{i=0}^{r} \binom{m}{i}$

- Distance: $d = 2^{m-r}, \delta = d/n = 2^{-r}$

In the special case for $r = 1$, RM code boils down to Hadamard code. Among other things, a traditional communication application of Hadamard code was in CDMA (IS-95) as long code for signal acquisition and synchronization due to its excellent distance property. The Hadamard codes are uniquely and locally decodable when the error rate is less that $\eta < 1/4$. Moreover for when error rate $\eta < \frac{1}{2} - \epsilon$, list decoding of Hadamard code is known. This is the Goldreich-Levin Method [4], which is attributed to Rackoff and outputs a list of size $\leq 2m/\epsilon^2$ in time $poly(m, 1/\epsilon)$ when error rate $\eta < \frac{1}{2} - \epsilon$ for some $\epsilon > 0$.

1.2.1 Majority Logic Circuit Decoder

The majority logic circuit decoder is an efficient algorithm for decoding RM codes when the number of errors are less than half the minimum distance, thus falls in the unique decoding regime. It was proposed simultaneously by Muller and Reed [12, 13]. The algorithm is summarized in Algorithm 1, which works when error rate $\eta < 2^{-r-1} - \epsilon$.

Algorithm 1 Majority Logic Circuit Decoder for RM code

Input: A function $R : \mathbb{F}_m^2 \to \mathbb{F}_2$ such that $\exists P \in \mathbb{F}_2[X_1, ..., X_m]_{\leq r}$ with $\delta(R, P) < 2^{-r-1} - \epsilon$
Output: The polynomial $P = \sum_{S \subseteq [m], |S| \leq r} c_S \prod_{i \in S} X_i$

```
1: Initialize $t \to r, F \to R, P \to 0$.
2: while $t \geq 0$ do
3:    for $S \subseteq [m]$ with $|S| = t$ do
4:       $c_S$ = Majority over all $\mathbf{b} \in \mathbb{F}_2^m$ of $\left(\sum_{\mathbf{a} \in \mathbb{F}_2^m, a_S = \mathbf{b}} F(\mathbf{a})\right)$
5:       $P \leftarrow P + c_S \prod_{i \in S} X_i$
6:       $\forall \mathbf{x} \in \mathbb{F}_2^m :\ F(\mathbf{x}) \leftarrow F(\mathbf{x}) - c_S \prod_{i \in S} x_i$
7:    end for
8:    $t \leftarrow t - 1$
9: end while
```

1.2.2 Local Version

In step 3 of Algorithm 1 we are absolutely required to take majority over all possible $\mathbf{b} \in \mathbb{F}_2^{m-r}$. In fact we can be smart and take majority vote over only $O(\frac{\log(m^r/\delta)}{2^{2r}\epsilon^2})$ values of $\mathbf{b}$, we recover c_S correctly with probability $1 - \delta/m^r$. This is because the total number of errors is no more than $2^m(2^{-r-1} - \epsilon) = 2^{m-r}(\frac{1}{2} - 2^r\epsilon)$ by assumption and each error can affect only one of the $\sum_{\mathbf{a}\in\mathbb{F}_2^m, a_S=\mathbf{b}} F(\mathbf{a})$, thus at most $(\frac{1}{2} - 2^r\epsilon)$ fraction of $\sum_{\mathbf{a}\in\mathbb{F}_2^m, a_S=\mathbf{b}} F(\mathbf{a})$ cab be wrong. Then by the union bound, every coefficient c_S is recovered correctly, with probability $1-\delta$. The overall running time is polynomial in m^r and $1/\epsilon$ as desired. This can be summarised as the following result.

Theorem 2 *Majority Logic Decoding solves the Local Unique Decoding problem with probability* $1-\delta$ *in time poly*$(m^r, \epsilon, \log(1/\delta))$.

1.3 Ideal Goals and Results of [6, 7]

The paper by GKZ presents the first local list-decoding algorithm for the r-th order Reed-Muller code RM(r, m) over $\mathbb{F}_2$ for $r \geq 2$. Given an oracle for a received word $R : \mathbb{F}_2^m \to \mathbb{F}_2$, their randomized local list-decoding algorithm produces a list containing all degree r polynomials within relative distance $(2^{-r} - \epsilon)$ from R for any $\epsilon > 0$ in time $poly(m^r, \epsilon^r)$. Since $RM(r, m)$ has relative distance 2^r, their algorithm beats the Johnson bound for $r \geq 2$. Moreover, as the list size could be exponential in m at radius 2^{-r}, their bound is optimal in the local setting.

1.3.1 Beats Johnson Bound!

- Recall Johnson Bound
  - When $\eta < J(\delta) - \epsilon$, then
  - code is list decodable with list size $O(\epsilon^2)$
  - where $J(\delta) = \frac{1}{2}(1 - \sqrt{1-2\delta})$
- For RM codes, we have $\delta = 2^{-r}$

	Johnson Bound	**GKZ List Decoding**
List Size	$O(\epsilon^2)$	$O(\epsilon^2)$
Time	–	$\text{poly}_r(m, 1/\epsilon)$
Max Error	$J(2^{-r}) - \epsilon$	$2^{-r} - \epsilon$
Example ($r = 2$)	0.146	0.25

Table 1: On how the GKZ beats the Johnson bound

1.3.2 Optimality

- Cannot do better as exponentially many codewords at distance of 2^{-r}
- An example:
  - Let $\mathbf{V}_1, ..., \mathbf{V}_t \subset \mathbb{F}_2^m$ such that $\forall i : \mathsf{dim}(\mathbf{V}_i) = m - r$.
  - Each $\mathbf{V}_i$ has a parity check matrix $[\mathbf{H}^{(i)}]_{r \times m}$
  - Consider the polynomials

$$P_i(\mathbf{x}) = \prod_{j=1}^{r}(1 + \langle \mathbf{H}_j^{(i)}, \mathbf{x} \rangle) = \begin{cases} 1 & \text{if } \mathbf{x} \in \mathbf{V}_i \\ 0 & \text{else} \end{cases}$$

  - All P_i's are unique
  - They are valid codewords in RM(m, r) code!
  - If we receive $R = 0$, then all these are at distance 2^{-r}
  - Note $t =$ Number of subspace of dimension $m - r > 2^{r(m-r)}$

1.4 Notations

Before delving into further details, we clearly layout the notations used in this report. A scalar is denoted by lower-case italic letter or greek letter, e.g. n, η. A vector is denoted by a bold lower-case letter, e.g. $\mathbf{r}$ whose i^{th} entry is $\mathbf{v}_i$. Polynomials are denoted by upper-case italic letter, e.g. P. Matrices and linear spaces are denoted by bold upper-case letters, e.g. $\mathbf{A}$ with $(i, j)^{th}$ entry $A_{i,j}$. For a polynomial P, by $P_{\mathbf{A}}$ we denote its restriction to the subspace $\mathbf{A}$, also note that $\mathsf{deg}(P_{\mathbf{A}}) \leq \mathsf{deg}(P)$. Sets are denoted by upper-case calligraphic letters, e.g. $\mathcal{C}$. Table 2 gives a list of common symbols we used.

2 Overview

We first show the main ideas on the Hadamard local-list decoding and then later explain them for RM local list-decoding.

Symbol	**Definition**
m	Number of variables in RM codes
r	Degree of polynomial in RM codes
n	Block length $= 2^m$
$\mathbb{F}$	Finite field, if size q then denote by $\mathbb{F}_q$
$\mathbf{x}, \mathbf{r}, ..$	Vectors in $\mathbb{F}^n$
$\mathbf{A}$	Subspace of $\mathbb{F}^n$
k	Dimension of subspace
$P, Q, ..$	Polynomials in $\mathbb{F}_q[X_1, ..., X_m]$
$P_{\mathbf{A}}$	Restriction of Polynomial P to subspace $\mathbf{A}$
η	Error fraction
$\delta(\mathcal{C})$	Minimum distance of code $\mathcal{C}$
$\Delta(\mathbf{x}, \mathbf{y})$	Hamming distance between codewords x and y
$\mathcal{C}$	Codebook
$\mathcal{L}$	List

Table 2: Symbols and definitions

2.1 Hadamard local list-decoding

Let R be the received word of length $n = 2^m$. We wish to find all the codewords that are within relative distance $\eta = \frac{1}{2} - \epsilon$ from R. Consider any particular codeword P such that $\Delta(R, P) \leq \eta$. We consider two $O(poly(m))$ time algorithms in this section. The first simpler algorithm outputs a list of size $O(poly(\frac{m}{\epsilon}))$ that contains P with high probability. The second algorithm, due to Goldreich and Levin, outputs a shorter list of size $O(poly(\frac{1}{\epsilon}))$ that contains P with a constant probability. Note that we are never concerned about outputting incorrect P's that do not satisfy $\Delta(R, P) \leq \eta$. This is because our algorithm can always test P before outputting and the probability that an algorithm that tests at a random $O(\log\ n)$ locations fails to identify if $P \in \mathcal{L}$ is exponentially small using Chernoff bounds. Thus, if we repeat the second algorithm a few times then with high probability it will also output P with high probability.

The intuition behind the first algorithm is to find $O(m)$ correct evaluations of P so that one can then solve m linear equations for the m bits of the message. One cannot brute force these $O(m)$ evaluations as that would require $2^{O(m)}$ guesses and we are interested in a $poly(m)$ time algorithm. The crucial idea of the algorithm is to try to find the correct evaluation of P at $O(m)$ locations of a subspace $\mathbf{A}$ of dimension O ($\log\ m$). Let $P_{\mathbf{A}}$ denote this correct evaluation. This is easier to find because the algorithm can guess the evaluation of P at the basis elements of $\mathbf{A}$ (only $2^{O(\log\ \frac{m}{\epsilon})} = poly(\frac{m}{\epsilon})$ guesses), and since P is a degree one polynomial, use interpolation to find $P_{\mathbf{A}}$.

The above algorithm works but has the drawback that it only gives a $O(poly(\frac{m}{\epsilon}))$ upper bound on the list size. However, we are interested in an efficient algorithm that gives a $O(poly(\frac{1}{\epsilon}))$ bound on the list size. This is where the GL-algorithm makes the second crucial observation that the dimension of the subspace $\mathbf{A}$ need not be O (log $\frac{m}{\epsilon}$), but can be reduced to only O (log $\frac{1}{\epsilon}$). We still wish to find the correct evaluation of P at $O(m)$ points but because we use a smaller subspace $\mathbf{A}$ the evaluation of P at any point is correct with only a constant probability (i.e. not with high probability as earlier). However, this constant is larger than the unique decoding parameter and we can use a unique decoding algorithm to find $P_{\mathbf{A}}$.

Goldreich-Levin algorithm:

1. Find a random subspace $\mathbf{A}$ of dimension $O(\log\ \epsilon)$.

2. Since P is linear, evaluate $P_{\mathbf{A}}$ by trying every possible evaluation at the basis vectors of $\mathbf{A}$ (polynomial in $\frac{1}{\epsilon}$) and one of them will be correct.

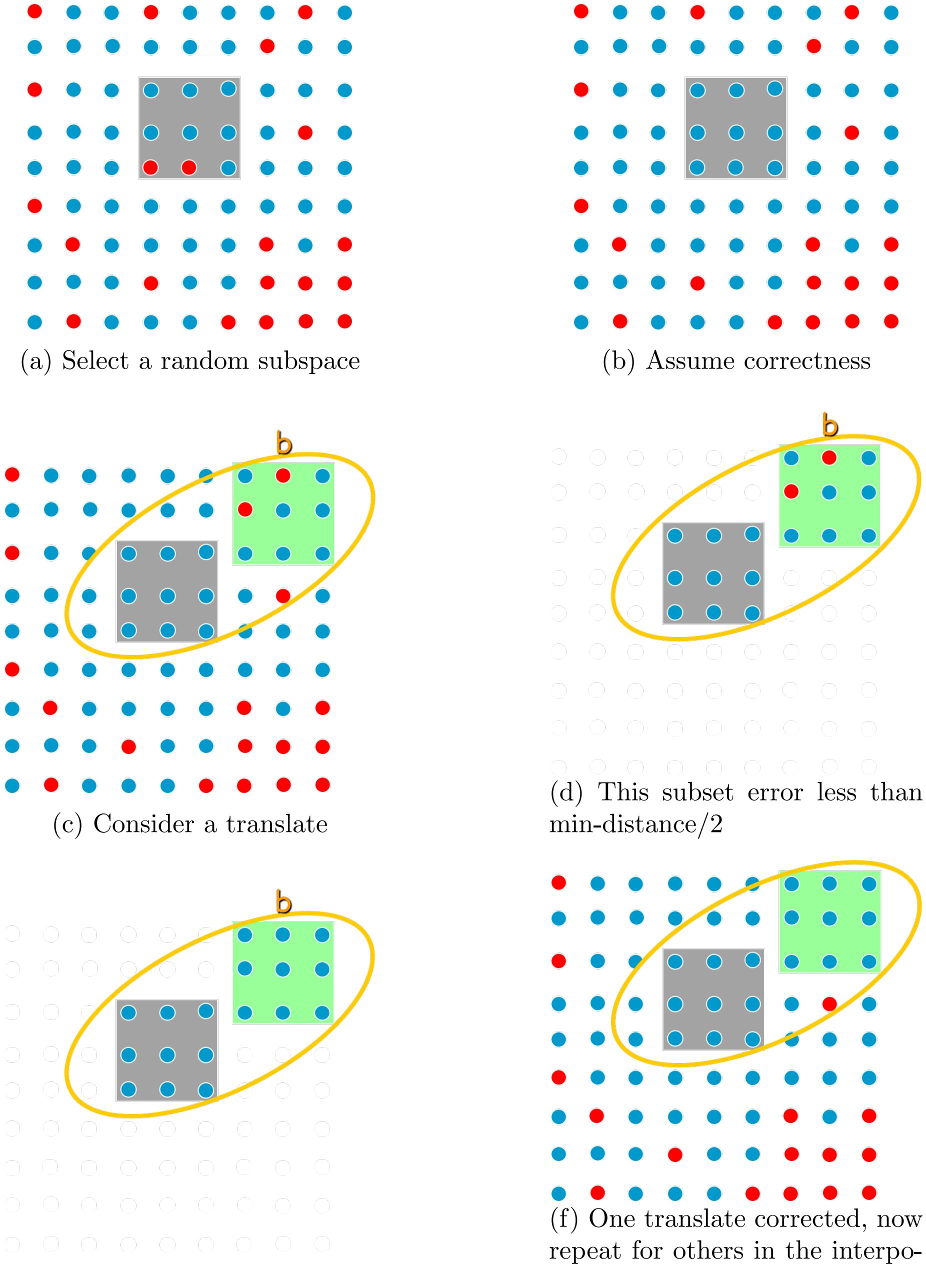
3. Now assuming that we know the restriction $P_{\mathbf{A}}$, consider any point $b \in \mathbb{F}_2^m \backslash \mathbf{A}$. Define $\mathbf{A}' = \mathbf{A} \cup (\mathbf{b} + \mathbf{A})$. They show that for at least 0.9 fraction of $\mathbf{b}$, disagreement between P and $P_{\mathbf{A}} \cup R_{(\mathbf{b}+\mathbf{A})}$ in subspace $\mathbf{A}'$ is at most $\frac{\eta}{2}$. The intuition behind this is that $P_{\mathbf{A}}$ and P agree on $\mathbf{A}$ by assumption, and each element in $\mathbf{b} + \mathbf{A}$ will disagree with probability at most η (using pairwise-independence).

4. For the above 0.9 fraction of $\mathbf{b}$ where $\Delta_{\mathbf{A}'}(Q \cup R_{(\mathbf{b}+\mathbf{A})}, P) \leq \frac{\eta}{2}$, we are within the unique decoding distance in $\mathbf{A}'$. Hence we can just use the majority algorithm to find evaluation of P at $\mathbf{b}$. For the remaining 0.1 fraction of $\mathbf{b}$, the majority algorithm outputs 'garbage'. Let M denote the output of this majority algorithm, which we know agrees with P for at least 0.9 fraction of the points.

5. Now we run the local unique decoding algorithm, which is the majority decoding algorithm over a small random subset of locations, on M to find P.

2.2 RM local list-decoding

Let $R \in \mathbb{F}_2^m$ be the received word. Let $\eta = 2^{-r} - \epsilon$. The RM local list decoding algorithm of Gopalan, Klivans, and Zuckerman [7] also follows the same general ideas of the GL algorithm. Consider a fixed d polynomial P such that $\Delta(P, R) \leq \eta$. We design an algorithm that outputs



(a) Select a random subspace

(b) Assume correctness

(c) Consider a translate

(d) This subset error less than min-distance/2

(e) Unique decode!

(f) One translate corrected, now repeat for others in the interpolating set

Figure 2: Idea of the algorithm

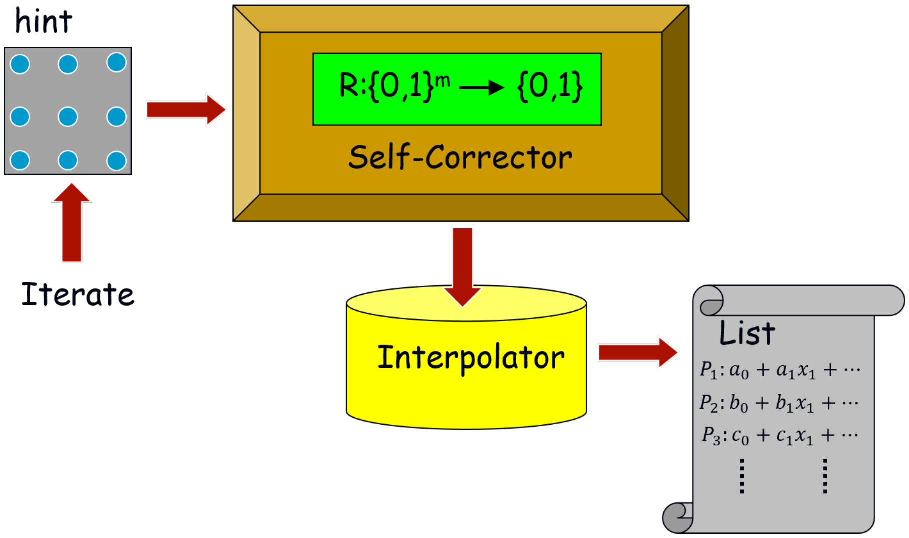

hint
R:{0,1}^m → {0,1}
Self-Corrector
Iterate
Interpolator
List
$P_1: a_0 + a_1x_1 + \cdots$
$P_2: b_0 + b_1x_1 + \cdots$
$P_3: c_0 + c_1x_1 + \cdots$
⋮ ⋮

Figure 3: Overview of the algorithm

P with a constant probability. It starts by finding the correct evaluation of P at a subspace A of dimension $k = O(\log\ \epsilon)$. Next it uses $P_{\mathbf{A}}$ to find $M \in \mathbb{F}_2^m$ that agrees with P on at least a constant fraction of points and M is within the unique decoding distance from P. There are two major new challenges that appear though.

First, we note that finding $P_{\mathbf{A}}$ is not easy because we are no longer dealing with linear polynomials. Thus, even if we guess the evaluation of P at the k bases of $\mathbf{A}$, we cannot linearly interpolate it to the entire subspace $\mathbf{A}$. Instead, for a degree d polynomial, we need to guess at least $O(k^d)$ bits. This is not pleasant as it gives a list size and running time bound of the order of $2^{(\log\ 1/\epsilon)^d}$, which is quasi-polynomial in $\frac{1}{\epsilon}$ for $d \geq 2$. To solve this challenge, the authors in GKZ make an important observation that since $\Delta(P, R) \leq \eta$, with high probability $\Delta(P_{\mathbf{A}}, R_{\mathbf{A}}) \leq \eta + \epsilon$. Hence they find every possible degree d polynomials in the subspace $\mathbf{A}$ that is at relative distance at most η from $R_{\mathbf{A}}$. This is again a list decoding RM codes problem. However, since the dimension k of the subspace $\mathbf{A}$ is way smaller than the original dimension 2^m, we only need a global RM list-decoding algorithm. This is explained in Section 3 in details. We also need to show that the size of the list outputted by the algorithm is not 'very large', which is discussed in Section 4.

The second challenge is that given $M \in \mathbb{F}_2^n$ that agrees with P at atleast η fraction, how to perform the unique decoding step efficiently to find P. The original paper of GKV that appeared in STOC 2008 [6] did not find a 'simple' efficient way of handling this problem and had to consider two different values of k for the analysis. Dvir and Shpilka [2] came up with a clever family of interpolating sets that can be used to locally decode M to find P. In this report we describe another approach that appeared in a later version of the paper by GKZ. It basically shows that Reed's Majority Logic Decoding for unique decoding of RM codes can be also done locally.

We next outline the algorithm. Recollect that we need to show that any fixed degree d polynomial P with $\Delta(P, R) \leq \eta$ will be present in the outputted list with atleast a constant probability.

1. Select a random subspace $\mathbf{A}$ of dimension $k = O\left(\log \frac{1}{\epsilon}\right)$.

2. To find the restriction $P_{\mathbf{A}}$, we note that with high probability it as at a relative distance at most $\eta + \epsilon$ from $R_{\mathbf{A}}$. Since $k = O\left(\log \frac{1}{\epsilon}\right)$, we use the global list decoding algorithm from Section 3 to find list $\mathcal{L}_{\mathbf{A}}$. Next we try every possible element in $\mathcal{L}_{\mathbf{A}}$ and at least one of them will be correct. The fact that $|\mathcal{L}_{\mathbf{A}}| \leq (\frac{1}{\epsilon})^{O(1)}$ follows from Section 4.

3. Now assuming that we know $P_{\mathbf{A}}$, for any $\mathbf{b} \in \mathbb{F}_2^m \backslash \mathbf{A}$, define $\mathbf{A}' = \mathbf{A} \cup (\mathbf{b} + \mathbf{A})$. We show that for at least a constant c fraction of $\mathbf{b}$ (here constant $1 - c$ is at most the unique decoding distance $1 - 2^{-r-1}$) disagreement between $P_{span(\mathbf{A}+\mathbf{b})}$ and $P_{\mathbf{A}} \cup R_{(\underline{+}\mathbf{A})}$ in subspace $\mathbf{A}'$ is at most $\frac{\eta}{2}$. Hence one can use unique decoding over $\mathbf{A}'$ and find the correct value for c fraction of points $\mathbf{b}$. For the remaining $1 - c$ fraction the algorithm outputs 'garbage'. Let M denote this output that agrees with P on at least c fraction of the bits.

4. Now we run the local unique decoding algorithm 1 on M to find P.

3 Algorithm

3.1 Global RM list-decoding

We design a recursive global list decoding algorithm for RM codes. Suppose we receive $R \in \mathbb{F}_2^k$ and we want to find a list $\mathcal{L}$ of all degree d polynomials $Q \in \mathbb{F}_2^k$ such that $\Delta(R, Q) \leq \eta$, where $\eta \leq 2^{-r}$. Let Q be a polynomial on variables $X_1, X_2, \ldots, X_k$. Consider the polynomials $Q_i = Q(X_1, X_2, \ldots, X_{k-1}, i)$ for $i \in \{0, 1\}$. Let η_i denote the fraction of disagreement between Q_i and $R_{X_1,\ldots,X_{k-1},i}$. Assume that we know $\eta_0 \leq \eta_1$ (this can be done because we try both $\eta_0 \leq \eta_1$ and $\eta_1 \leq \eta_0$). Since $\eta = \frac{\eta_0+\eta_1}{2}$, we get $\eta_0 \leq \eta$ and $\eta_1 \leq 2\eta$ (see Fig. 4). Note that we can write

$$Q = Q_0 + X_k Q'(X_1, \ldots, X_{k-1})$$

Algorithm 2 GlobalListDecoding (A, R, k, η)

```
for all $b \in \mathbb{F}_2$ do
    Set $\mathcal{L}^b$ = GlobalListDecoding($\mathbf{A}^b, R_{\mathbf{A}^b}, k, \eta$)
    for all $Q_0 \in \mathcal{L}^b$ do
        Set $\mathcal{L}^{1+b}$ = GlobalListDecoding($\mathbf{A}^{1+b}, R_{\mathbf{A}^{1+b}} + Q_0, k-1, 2\eta$)
        for all $Q' \in \mathcal{L}^{1+b}$ do
            Set $Q(X_1, \ldots, X_k) = Q_0(X_1, \ldots, X_{k-1}) + (X_k + b_k)Q'(X_1, \ldots, X_{k-1})$
            if $\Delta_{\mathbf{A}}(Q, R) \leq \eta$, then add $Q$ to $\mathcal{L}$
        end for
    end for
end for
Return $\mathcal{L}$
```

for some degree $d-1$ polynomial Q'. Since Q_0 has degree at most d and error at most η_0 with respect to $R_{X_1,\ldots,X_{k-1},0}$, we can use our algorithm recursively on Q_0 to find a list $\mathcal{L}_0$. Moreover, Q' is a polynomial of degree at most $d-1$ and its disagreement with $R_{X_1,\ldots,X_{k-1},1}$ is at most 2η, we can again use our algorithm recursively on Q' to find list $\mathcal{L}_1$ as our induction hypothesis gives that the minimum list decoding algorithm works till relative distance 2^{r-1}. Finally the algorithm outputs $\mathcal{L} = \{\mathcal{L}_0, 0\} \cup \{\mathcal{L}_1, 1\}$ where $\{\mathcal{L}_i, i\}$ denotes the list of strings in $\mathcal{L}_i$ appended with i.

The base case of the algorithm is when $r = 1$ or when there is no error. In the former case we can use GL algorithm to output the list and in the later case we just solve a system of linear equations.

Theorem 3 (Theorem 3 in Section 3 of [7]) *Let $l(r, k, \eta)$ denote the list size for degree r, variables k, and error $\eta \leq 2^{-r}$ list decoding. Let $T(r, k, \eta)$ be the time taken by the above recursive global list decoding Algorithm 1. Then, $T(r, k, \eta) \leq 2^{3m} l(r, k, \eta)^r$*

We skip the proof since it follows using elementary algebra and refer interested readers to [7]. In Section 4 we bound $l(r, k, \eta)$ to show that everything can be finished in $O(poly\ m)$ time when k is $O(\frac{1}{\epsilon})$.

Lemma 1 *For any P that satisfies $\Delta(R, P) \leq \eta$, where $\eta = 2^{-r} - \epsilon$, with constant probability the GlobalListDecoding will find $P_{\mathbf{A}}$.*

The above lemma is easy to prove using pairwise-independence of the elements in $\mathbf{A}$ and we refer the reader to Lemma 4 in [7] for details.

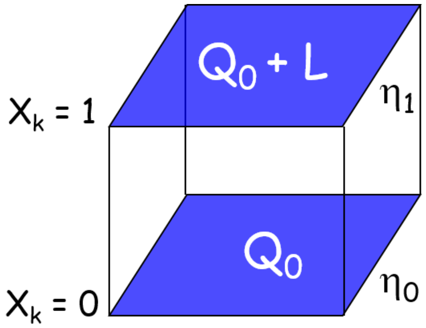


Figure 4: Idea of global list decoding

3.2 Putting it all together

We already know from Lemma 1 that with constant probability GlobalListDecoding finds $P_{\mathbf{A}}$. Thus, whenever we find $P_{\mathbf{A}}$, we can interpolate it to obtain an oracle machine $M_j \in \mathbb{F}_2^n$ that is within the unique decoding distance from P. The following SelfCorrect algorithm describes how to find M_j.

Algorithm 3 SelfCorrect$(R, r, 2^{-r} - \epsilon)$

```
Pick a random $k$-dimensional subspace $\mathbf{A}$ where $2^k \geq c2^{r+3}\epsilon^{-2}$
$\mathcal{L}_{\mathbf{A}} \leftarrow$ GlobalListDecoding$(R_{\mathbf{A}}, r, 2^{-r})$
for all $Q_j \in \mathcal{L}^{\mathbf{A}}$ do {define $M_j$ as follows:}
    Set $R_j(\mathbf{x}) = Q_j(\mathbf{x})$ if $\mathbf{x} \in \mathbf{A}$; and $R_j(\mathbf{x}) = R(\mathbf{x})$ if $\mathbf{x} \notin \mathbf{A}$
    for all $\mathbf{b} \in \mathbb{F}_2^m$ do
        Let $\mathbf{A}' = \mathbf{A} \cup (\mathbf{b} + \mathbf{A})$
        Unique decode $R_j$ on $\mathbf{A}'$ to get $Q' : \mathbf{A}' \to \mathbb{F}_2$ with $deg(Q') \leq r$
        Set $M_j(\mathbf{b} + \mathbf{a}_0) = Q'(\mathbf{b} + \mathbf{a}_0)$
    end for
end for
Return list of oracle machines $\mathcal{M} = \{M_j\}$
```

Since with at least a constant probability M_j is within the unique decoding distance from P, we can use the local unique decoding algorithm from Theorem 2 to obtain P. This is described in the following MainAlgorithm.

Algorithm 4 LocalListDecode($R, r, 2^{-r} - \epsilon$)

```
Use SelfCorrect($R, r, 2^{-r} - \epsilon$) to produce list $\mathcal{M}$ of oracle machines
Run Majority-Logic-Decoding from Theorem 2 on each $M_j^R \in \mathcal{M}$ with $\epsilon = 2^{-r-2}$, $\delta = \frac{1}{\epsilon}$ to get polynomial $P$
$\mathcal{L}_{\mathbf{A}} \leftarrow$ GlobalListDecoding($R_{\mathbf{A}}, r, 2^{-r}$)
Check by random sampling that $P$ satisfies $\Delta(P, R) \leq 2^{-r} - \frac{\epsilon}{2}$. Return the list $\mathcal{L}'$ of all such polynomials
```

4 Bounding the list size

In this section we bound the list size $l(r, m, 2^{-r} - \epsilon)$. Since the size of the list outputted by the global list decoding algorithm is at most $l(r, k, 2^{-r})$ and it contains projection P_A with probability at least $\frac{1}{2}$, which can be uniquely decoded to obtain P, we directly get $l(r, m, 2^{-r} - \epsilon) \leq 2l(r, k, 2^{-r})$.

Lemma 2 *$l(r, m, 2^{-r} - \epsilon) \leq 2l(r, k, 2^{-r})$*

In the remaining section we show that $l(r, k, 2^{-r})$ is $O(\epsilon^{-8r})$.

4.1 The Deletion lemma

Recollect that the Johnson bound tells us that for any code $\mathcal{C}$ the number of codewords C such that $\Delta(R, C) < J(\delta) - \gamma$ is bounded by $c(\gamma) = O(\gamma^{-2}))$, where $J(\alpha) = \frac{1}{2}(1 - \sqrt{1 - 2\alpha})$. Also, the number of codewords C such that $\Delta(R, C) < J(\delta)$ is bounded by $2n$. The deletion lemma is a generalization of the Johnson bound for codes with not too many low weight codewords. Let $A(\alpha)$ denote the number of codewords in a linear code $\mathcal{C}$ of weight less than α.

Lemma 3 (Deletion Lemma [7]) *Let $\mathcal{C} \subseteq \mathbb{F}_2^n$ be a linear code. For any $\alpha \in [0, 1]$ and any $R \in \mathbb{F}_2^n$,*

- *The number of $C \in \mathcal{C}$ such that $\Delta(R, C) < J(\alpha) - \gamma$ is bounded by $A(\alpha)c(\gamma)$.*
- *The number of $C \in \mathcal{C}$ such that $\Delta(R, C) < J(\alpha)$ is bounded by $2A(\alpha)n$.*

Proof 1 *Let $\mathcal{L}$ denote the list of $C \in \mathcal{C}$ such that $\Delta(R, C) < J(\alpha)$. We remove codewords from $\mathcal{L}$ such that there are no two codewords at relative distance less than α. We do this by picking a codeword $C_i \in \mathcal{L}$ and deleting all codewords C_j, $j \neq i$, such that $\Delta(C_i, C_j) < \alpha$*

from $\mathcal{L}$ to obtain a list $\mathcal{L}'$. From the definition of $A(\alpha)$, we know that there are at most $A(\alpha)$ codewords C_j that satisfies this condition. Hence $|\mathcal{L}'| \geq |\mathcal{L}| - A(\alpha)$. Now recursing the last procedure on $\mathcal{L}'$ we obtain list $\mathcal{L}^*$. It is easy to see that this implies $|\mathcal{L}| \leq |\mathcal{L}^*|A(\alpha)$. Now using the Johnson bound on $\mathcal{L}^*$ completes the proof.

4.2 Using Kasami-Tokura theorem to bound the list size

We wish to bound $l(r, k, 2^{-r})$. Let $\alpha = 2^{1-r} - 2^{-2r+1}$. So $J(\alpha) = 2^{-r}$. Now the Deletion lemma (Lemma 3) gives that $l(r, k, 2^{-r}) \leq 2A(\alpha)2^k$. So we basically need to bound $A(\alpha)$ for $\alpha = 2^{1-r} - 2^{-2r+1}$. This is where we use the Kasami-Tokura theorem that helps in understanding the weight distribution of RM codes over $\mathbb{F}_2$.

Theorem 4 (Kasami-Tokura [11]) *Let P be any polynomial with $deg(P) \leq r$, where $r \geq 2$, such that $wt(P) < 2^{1-r}$. Then there exists an invertible affine transformation such that P can be written as either*

$$P(Y_1, \ldots, Y_{r+t}) = Y_1Y_2 \ldots Y_{r-t}(Y_{r-t+1}Y_{r-t+2} \ldots Y_r + Y_{r+1}Y_{r+2} \ldots Y_{r+t})$$

where $3 \leq t \leq r$ and $t + r \leq k$, or

$$P(Y_1, \ldots, Y_{r+2t-2}) = Y_1Y_2 \ldots Y_{r-2}(Y_{r-1}Y_r + Y_{r+1}Y_{r+2} + \ldots + Y_{r+2t-3}Y_{r+2t-2})$$

where $2 \leq 2t \leq k - r + 2$.

Corollary 5 (Corollary 13 in [7]) *For $\alpha = 2^{1-r} - 2^{-2r+1}$, we have $A(\alpha) \leq (2^{-2r+1})^{-2(k+1)}$*

The above corollary follow from Lemma 4 along with some elementary algebra. We refer the interested readers to GKZ [7] for details. Now combining all the results, we have $l(r, m, 2^{-r} - \epsilon) \leq 2l(r, k, 2^{-r}) \leq 4.2^k A(\alpha) = O(2^{(4r-1)(k+1)}) = O(\epsilon^{-8r})$.

5 Conclusions

This report describes the local list decoding algorithm of Reed-Muller codes by Gopalan, Klivans, and Zuckerman. The algorithm generalizes Goldreich-Levin local list decoding algorithm for Hadamard codes. The main theorem shows that any P that is within distance $\eta = 2^{-r} - \epsilon$ from the received word R can obtained with a constant probability. There are two main challenges that appear in this generalization. Firstly, obtaining projection of P on a linear subspace A because we are no longer dealing with linear polynomials and a simple brute force strategy will give an algorithm quasi-polynomial in $\frac{1}{\epsilon}$. In GKZ they tackle this by observing that with high probability P_A is within $\eta + \epsilon$ from R_A. The second challenge is that given M that is within the unique decoding distance from P, how to find P locally. The authors observe that this can be done by transforming Reed's Majority Logic decoding algorithm to a local algorithm.

The paper mainly raises the following conjecture:

Conjecture 6 *For field $\mathbb{F}_q$ and $\epsilon > 0$, $\exists c(q, \epsilon, r)$ independent of n and m such that for all m and r*

$$l_q(r, m, \delta_q(r) - \epsilon) \leq c(q, \epsilon, r)$$

Some progress has been made in the recent years towards proving the conjecture. The original GKZ paper also showed that the above conjecture is true for small q when $q - 1$ divides r. Later, in 2010, Gopalan [5] showed the conjecture to be true for quadratic polynomials for all fields $\mathbb{F}_q$. Recently, Blowmick and Lovett [1] have resolved this conjecture over $\mathbb{F}_p$ for prime p. Their bounds on the list size, although constant for constant ϵ, depend badly on $\frac{1}{\epsilon}$. Improving those bounds is an interesting open problem. In any case, resolving the above conjecture is the main driving force behind this area of work.

References

[1] Abhishek Bhowmick and Shachar Lovett. List decoding reed-muller codes over small fields. *arXiv preprint arXiv:1407.3433*, 2014.

[2] Z. Dvir and A. Shpilka. Noisy interpolating sets for low degree polynomials. In *Computational Complexity, 2008. CCC '08. 23rd Annual IEEE Conference on*, pages 140–148, June 2008.

[3] Peter Elias. List decoding for noisy channels. Technical report, Research Laboratory of Electronics, Massachusetts Institute of Technology, 1957.

[4] Oded Goldreich and Leonid A Levin. A hard-core predicate for all one-way functions. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 25–32. ACM, 1989.

[5] P. Gopalan. A fourier-analytic approach to reed-muller decoding. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 685–694, Oct 2010.

[6] Parikshit Gopalan, Adam R Klivans, and David Zuckerman. List-decoding reed-muller codes over small fields. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 265–274. ACM, 2008.

[7] Parikshit Gopalan, Adam R Klivans, and David Zuckerman. List-decoding reed-muller codes over small fields. Technical report, Jul 2011.

[8] Venkatesan Guruswami. *List decoding of error-correcting codes: winning thesis of the 2002 ACM doctoral dissertation competition*, volume 3282. Springer, 2004.

[9] Venkatesan Guruswami. *Algorithmic results in list decoding.* Now Publishers Inc, 2006.

[10] Venkatesan Guruswami, Atri Rudra, and Madhu Sudan. *Essential Coding Theory.* 2014.

[11] T. Kasami and N. Tokura. On the weight structure of reed-muller codes. *Information Theory, IEEE Transactions on*, 16(6):752–759, Nov 1970.

[12] D.E. Muller. Application of boolean algebra to switching circuit design and to error detection. *Electronic Computers, Transactions of the I.R.E. Professional Group on*, EC-3(3):6–12, Sept 1954.

[13] I. Reed. A class of multiple-error-correcting codes and the decoding scheme. *Information Theory, Transactions of the IRE Professional Group on*, 4(4):38–49, September 1954.

[14] Madhu Sudan. Decoding of reed solomon codes beyond the error-correction bound. *Journal of complexity*, 13(1):180–193, 1997.

[15] Madhu Sudan. List decoding: Algorithms and applications. In *Theoretical Computer Science: Exploring New Frontiers of Theoretical Informatics*, pages 25–41. Springer, 2000.

[16] John M Wozencraft. List decoding. Technical report, Quarterly Progress Report, Research Laboratory of Electronics, Massachusetts Institute of Technology, 1958.