Locally testable and Locally correctable Codes Approaching the Gilbert-Varshamov Bound

Sivakanth Gopi* Swastik Kopparty† Rafael Oliveira* Noga Ron-Zewi‡
Shubhangi Saraf§

August 10, 2016

Abstract

One of the most important open problems in the theory of error-correcting codes is to determine the tradeoff between the rate R and minimum distance δ of a binary code. The best known tradeoff is the Gilbert-Varshamov bound, and says that for every $\delta \in (0, 1/2)$, there are codes with minimum distance δ and rate $R = R_{\mathsf{GV}}(\delta) > 0$ (for a certain simple function $R_{\mathsf{GV}}(\cdot)$). In this paper we show that the Gilbert-Varshamov bound can be achieved by codes which support *local* error-detection and error-correction algorithms.
Specifically, we show the following results.

1. **Local Testing:** For all $\delta \in (0, 1/2)$ and all $R < R_{\mathsf{GV}}(\delta)$, there exist codes with length n, rate R and minimum distance δ that are locally testable with $\mathsf{quasipolylog}(n)$ query complexity.
2. **Local Correction:** For all $\epsilon > 0$, for all $\delta < 1/2$ sufficiently large, and all $R < (1 - \epsilon)R_{\mathsf{GV}}(\delta)$, there exist codes with length n, rate R and minimum distance δ that are locally correctable from $\frac{\delta}{2} - o(1)$ fraction errors with $O(n^{\epsilon})$ query complexity.

Furthermore, these codes have an efficient randomized construction, and the local testing and local correction algorithms can be made to run in time polynomial in the query complexity. Our results on locally correctable codes also immediately give locally decodable codes with the same parameters.

Our local testing result is obtained by combining Thommesen's random concatenation technique and the best known locally testable codes. Our local correction result, which is significantly more involved, also uses random concatenation, along with a number of further ideas: the Guruswami-Sudan-Indyk list decoding strategy for concatenated codes, Alon-Edmonds-Luby distance amplification, and the local list-decodability, local list-recoverability and local testability of Reed-Muller codes. Curiously, our final local correction algorithms go via local list-decoding and local testing algorithms; this seems to be the first time local testability is used in the construction of a locally correctable code.

*Department of computer Science, Princeton University, Princeton NJ 08540, USA. Research supported by NSF grants CCF-1523816 and CCF-1217416. sgopi@cs.princeton.edu and rmo@cs.princeton.edu

†Department of Mathematics & Department of Computer Science, Rutgers University, Piscataway NJ 08854, USA. Supported in part by a Sloan Fellowship and NSF grant CCF-1253886. swastik.kopparty@gmail.com

‡School of Mathematics, Institute for Advanced Study, Princeton, NJ, USA and DIMACS, Rutgers University, Piscataway, NJ, USA. This research was partially supported by NSF grants CCF-1412958 and CCF-1445755 and the Rothschild fellowship. nogazewi@ias.edu

§Department of Mathematics & Department of Computer Science, Rutgers University, Piscataway NJ 08854, USA. Supported in part by NSF grant CCF-1350572. shubhangi.saraf@gmail.com

1 Introduction

In this paper, we show the existence of binary locally testable codes and locally correctable codes with rate-distance tradeoff matching what is known for general error-correcting codes.

Error-correcting codes are combinatorial objects that are used to protect data from errors. Formally, a (binary) error-correcting code C is a subset of $\{0,1\}^n$. There are two parameters of an error-correcting code that measure how good the code is: the rate R and the minimum distance δ. The rate R, which measures how much information can be packed into a codeword, is defined by $R = \frac{\log_2 |C|}{n}$. The minimum distance δ, which measures the error-correcting capability of the code, is defined to be the smallest relative Hamming distance[1] between distinct $x, y \in C$. The error-correction property of codes comes from the simple but important observation that there is at most one codeword within distance $\delta/2$ of any given word $w \in \{0,1\}^n$; finding this codeword given w is the problem of decoding.

One of the main combinatorial problems of coding theory is to determine the best tradeoff between the rate and the minimum distance for binary error-correcting codes. The best tradeoff known today is known as the Gilbert-Varshamov (GV) bound, and states that for every $\delta \in (0, 1/2)$, there exist codes of arbitrarily large length with minimum distance δ and rate $R = R_{\mathsf{GV}}(\delta)$. Here R_{GV} is the function:

$$R_{\mathsf{GV}}(\delta) = 1 - H(\delta),$$

where H is the binary entropy function. There are many known families of codes, including random codes, that achieve the GV bound, and it has been often conjectured that the GV bound is tight.

On the algorithmic side, it is not known how to deterministically construct codes that achieve the GV bound in polynomial time. Nevertheless, efficient deterministic constructions of codes with quite good rate-distance tradeoff are known, and furthermore these codes come equipped with efficient error-detection and correction algorithms. An alternate research direction, which is most relevant for us, has been to show existence of *highly structured* codes achieving the GV bound. Here we mention the beautiful results of Thommesen [Tho83], who gave a randomized construction of codes closely related to Reed-Solomon codes that meet the GV bound, and of Guruswami-Indyk [GI04], who gave a polynomial time algorithm for decoding Thommesen's codes from $\delta/2$-fraction errors (for sufficiently large $\delta < 1/2$). This latter work uses deep results of Guruswami-Sudan [GS99, GS00, GS02] on list-decoding Reed-Solomon codes and concatenated codes.

In recent years, the problem of what algorithmic problems on codes can be solved *locally* has gathered much attention. Locality here refers to algorithms that do not access all of their input, and this often leads to *sublinear time* algorithms. A code C is said to be locally testable if there is a randomized algorithm, which when given access to a received word $w \in \{0,1\}^n$, makes few queries into w and can determine, with high probability, whether w is in C or w is far away from C. A code is said to be locally correctable[2] from α-fraction errors if there is a randomized algorithm, which when given $i \in [n]$ and access to a received word $w \in \{0,1\}^n$, makes few queries into w and can determine c_i with high probability, *where c is the unique codeword that is α-close to w.*

[1] The relative Hamming distance between two strings $x, y \in \Sigma^n$, denoted $\mathrm{dist}(x, y)$, is defined by: $\mathrm{dist}(x, y) = \frac{1}{n} \cdot |\{i \mid x_i \neq y_i\}|$. We will often omit the word "relative".

[2] A closely related notion is that of locally-decodable code in which one requires that, when given a string w that is close to a codeword $c \in C$, and a coordinate i, the randomized algorithm computes the original message bit x_i. Our results for locally correctable codes hold for locally decodable codes as well, see discussion at end of the introduction.

Locally testable codes (LTCs) and locally correctable codes (LCCs) have been extensively studied, and are deeply connected to a number of important notions across theoretical computer science: probabilistic proof systems, hardness amplification, private information retrieval, cryptography and algebraic complexity theory.

Our main result is that there are codes that approach the GV bound, that can be locally tested and locally corrected from $(\delta/2 - o(1))$-fraction errors with sublinear (even polynomially small) query complexity. For binary codes, it was previously known how to do this for codes approaching the Zyablov bound [KMRS16], with the added advantage that the code also had an efficient deterministic construction.

We give the formal statements of our main results next.

Theorem A (Locally testable codes approaching the GV bound)
Let $\delta \in (0, 1/2)$. Let $R < R_{\mathsf{GV}}(\delta)$.

Then there exists an infinite family of codes $\{C_n\}_n$, with $C_n \subseteq \{0,1\}^n$, such that:

- *the minimum distance of C_n is at least δ,*
- *the rate of C_n is at least R,*
- *C_n is locally testable with $(\log n)^{O(\log \log n)}$ queries,*

Furthermore, such a C_n can be constructed by a randomized polynomial time algorithm with high probability, and the local testing algorithm can be implemented to run in time $(\log n)^{O(\log \log n)}$.

Theorem B (Locally correctable codes approaching the GV bound)
Let $\epsilon > 0$, and let $\xi > 0$ be sufficiently small (depending on ϵ). Let $\delta = \frac{1}{2} - \xi$. Let $R < (1-\epsilon) \cdot R_{\mathsf{GV}}(\delta)$.

Then there exists an infinite family of codes $\{C_n\}_n$, with $C_n \subseteq \{0,1\}^n$, such that:

- *the minimum distance of C_n is at least δ,*
- *the rate of C_n is at least R,*
- *C_n is locally correctable from $(\frac{\delta}{2} - o(1))$-fraction errors with $O(n^\epsilon)$ queries.*

Furthermore, such a C_n can be constructed by a randomized polynomial time algorithm with high probability, and the local correction algorithm can be implemented to run in time $\mathrm{poly}(n^\epsilon)$.

Note that our result about local testability allows for codes with rate and distance arbitrarily close to the GV bound for any distance $\delta \in (0, 1/2)$, but our result about local correctability only achieves this for distances sufficiently close to $1/2$ depending on ϵ where $O(n^\epsilon)$ is the query complexity, and with a further $(1-\epsilon)$-factor loss in the rate. These results are the first to show that codes with distance $\delta = 1/2 - \xi$ and rate $\Omega(\xi^2)$ can be locally tested / locally corrected from $(\delta/2 - o(1))$-fraction errors.

We remark that analogous results over large alphabets were only recently obtained [KMRS16]. In this setting, the best tradeoff between R and δ for general codes is completely known. Every code must satisfy $R \leq 1 - \delta$; this bound is known as the Singleton bound. Furthermore, Reed-Solomon codes achieve $R = 1 - \delta$, and they can be decoded from a $\delta/2$-fraction of errors in polynomial time. In [KMRS16], it was shown that there exist explicit locally testable codes and locally correctable codes which satisfy $R = 1 - \delta - \epsilon$ (for all $\epsilon > 0$), and which can further be locally tested and locally corrected from $(\delta/2 - o(1))$-fraction errors in sublinear (and even subpolynomial) time.

1.1 Methods

The starting point for our constructions is the random concatenation technique of Thommesen [Tho83], which he used to show that codes of a particular simple form can achieve the GV bound. Specifically, he showed that if one takes a Reed-Solomon code over a large alphabet as the outer code, and concatenate it with binary linear inner codes chosen uniformly at random and independently for each outer coordinate, then the resulting code C lies on the GV bound with high probability. In fact, the only property of Reed-Solomon codes that is used in this result is that the rate and distance of Reed-Solomon codes lie on the Singleton bound.

Our construction of locally testable codes approaching the GV bound then follows from the result of [KMRS16], which gave constructions of locally testable codes with rate and distance approaching the Singleton bound. We start with such a locally testable code from [KMRS16] as the outer code, and then concatenate it with uniformly random binary linear inner codes (independently for each coordinate of the outer code). The required rate-distance tradeoff of the concatenated code follows from Thommesen's arguments, and the local testability follows easily from the local testability of the outer code.

It is also known how to construct locally correctable codes with rate and distance approaching the Singleton bound [KMRS16]. If we use these codes along with the random concatenation idea, we get locally correctable codes approaching the GV bound. *But Theorem B requires the fraction of errors correctable by the local correction algorithm to approach $\delta/2$.* The natural local correction algorithm for concatenated codes (using the local correction algorithm of the outer code, and decoding inner codes by brute-force whenever an outer coordinate needs to be accessed) turns out to only decode to a much smaller radius (namely half the Zyablov bound); see [KMRS16] for details.

Our proof of Theorem B uses several more ideas. The next ingredient we use is an insight of Guruswami and Indyk [GI04]. They noted that the code C constructed by Thommesen could be decoded from $\delta/2$ fraction errors in polynomial time, provided the distance δ of C is sufficiently large (equivalently, provided the rate R of C is sufficiently small). The main idea is to use the *list-decoding* algorithms for concatenated codes developed by Guruswami and Sudan [GS00, GS02], which for binary codes of distance nearly $1/2$, can list decode from a fraction of errors that is also nearly $1/2$ – in particular, the fraction of errors correctable is far more than half the minimum distance (which is around $1/4$). One first list-decodes each of the inner concatenated codes (by brute-force) to get a list of candidate symbols for each coordinate of the Reed-Solomon code, and then one applies the *list-recovery*[3] algorithm (of Guruswami and Sudan [GS99]) for the outer Reed-Solomon code to get a list of candidate codewords. Finally, by computing the distance between each of these candidate codewords and the given received word, one can identify the one codeword (if any) that lies within distance $\delta/2$ of the received word.

Our local correction algorithm will try to implement this high-level strategy in the local setting. We will choose an outer code C_{out} over a large alphabet with suitable properties, and concatenate it with independently chosen random binary linear inner codes. We describe the properties required of C_{out} next:

- First of all, our choice of C_{out} should be such that the concatenated code approaches the GV bound with high probability. We will achieve this by ensuring that C_{out} has a sufficiently

[3] A list-recovery algorithm is a generalization of a list-decoding algorithm. Here one is given a small list of candidate symbols S_i for each coordinate i of the code, and the goal is to find all codewords $c \in C$ which have the property that for at most α fraction of the coordinates i, the ith coordinate of c does not lie in S_i.

good rate-distance tradeoff[4].

- Next, we would like C_{out} to be *locally* list-recoverable. A local list-recovery algorithm is an algorithm that solves the list-recovery problem (in an implicit manner) using few queries. Instead of outputting all nearby codewords (which is impossible using few queries), the local list-recovery algorithm outputs implicit description of words $w_1, \ldots, w_L$ which is guaranteed to contain all nearby codewords.

- Finally, we would like C_{out} to be *locally testable*. This is in order to identify which of the words w_i are actually codewords. Having done this, we can easily identify, by estimating distances via sampling, the one codeword w_i that is $\delta/2$-close to our original received word.

Summarizing, we want C_{out} to be locally list-recoverable, locally testable and have a decent rate-distance tradeoff. One might have hoped that the recently constructed codes of [KMRS16], which achieve local testability and local correctability with optimal rate-distance tradeoff (on the Singleton bound) would be good candidates for C_{out}. Unfortunately, none of the codes from [KMRS16] are known to achieve *both* local list-recoverability and local testability.

Instead, we go further back in time to the mother of all local codes, Reed-Muller codes. Reed-Muller codes turn out to satisfy the first two of these properties [RS96b, FS95, AS03, STV01], but they fall short on the last property. This brings us to our final ingredient: Alon-Edmonds-Luby (AEL) distance amplification [AEL95]. This distance amplification method improves the rate-distance tradeoff for codes. Furthermore, it was shown in [KMRS16] that this method preserves local testability and local correctability. Here we observe that this distance amplification method also preserves local list-recoverability. Thus, applying AEL distance amplification to Reed-Muller codes gives us a code that is locally list-recoverable, locally testable, and also has a decent rate-distance tradeoff (which turns out to be good enough for our purposes)[5]. This gives us the code C_{out}, and completes the high-level description of our constructions.

1.2 Further remarks

LTCs approaching the GV bound with constant query complexity? Our construction of LTCs approaching the GV bound is based on two ingredients: an LTC approaching the Singleton bound [KMRS16], and Thommesen's random concatenation technique. The result of [KMRS16] is in fact quite general: given any LTC family which can achieve rate arbitrarily close to 1, one can construct another LTC family which approaches the Singleton bound with only a constant factor blowup in the query complexity. Putting everything together: if there exist LTCs with rate arbitrarily close to 1 with query complexity q, then there are LTCs approaching the GV bound with query complexity $O(q)$. It has often been lamented (at least once in print [BSGK$^+$10], see page 2) by researchers in the area that we do not know any lower bounds on the rate-distance tradeoff of LTCs that distinguish them from general codes, and that for all we know, there could be constant query LTCs on the GV bound. Our result shows that such a lower bound would imply something

[4]The rate-distance tradeoff will be quite close to, but not approaching, the Singleton bound. This is the reason for our final locally correctable codes of Theorem B achieving rate that is smaller than $R_{\mathsf{GV}}(\delta)$ by a factor $(1-\epsilon)$.

[5]This description suffices for the existence part of Theorem B. However, to achieve *sublinear time* decoding, we will need one further trick: to concatenate the Reed-Muller codes down to a smaller alphabet before applying the AEL transformation - this smaller alphabet size is needed to let us perform the brute force list decoding of the random inner codes step quickly.

much more qualitative - that there do not exist constant query LTCs with rate arbitrarily close to 1.

Locally decodable codes. An important variant of LCCs are locally decodable codes (LDCs). Those codes are defined similarly to LCCs, with the following difference: Recall that in the definition of LCCs, the decoder gets access to a string w which is close to a codeword c, and is required to decode a coordinate of c. In the definition of LDCs, we view the codeword c as the encoding of some message x, and the decoder is required to decode a coordinate of x. LDCs were studied extensively in the literature, perhaps more so than LCCs. If we restrict ourselves to *linear* codes, then LDCs are a weaker object than LCCs, since every linear LCC can be converted into an LDC by choosing a systematic encoding map. Since the LCCs we construct in this paper are linear, our results apply to LDCs as well.

1.3 Organization of this paper

This paper is structured as follows: in Section 2 we provide some background on error correcting codes and set up the notation that will be used throughout the paper. In Section 3 we show the existence of locally testable codes approaching the GV bound. In Section 4 we show how to convert any code on a large alphabet with (somewhat) good rate and distance into a binary code nearly approaching the GV bound. In Section 5 we show the existence of locally correctable codes approaching the GV bound using the latter transformation. In Sections 6 and 7 we provide further ingredients needed for the construction of our locally correctable codes, namely local list recovery algorithm for Reed-Muller codes (in Section 6), and distance amplification procedure for local list recovery (in Section 7).

2 Preliminaries

We denote by $\mathbb{F}_q$ the finite field of q elements. For any finite alphabet Σ and for any string $x \in \Sigma^n$ the relative weight $\mathrm{wt}(x)$ of x is the fraction of non-zero coordinates of x, that is, $\mathrm{wt}(x) := |\{i \in [n] : x_i \neq 0\}|/n$. For any pair of strings $x, y \in \Sigma^n$, the relative distance between x and y is the fraction of coordinates on which x and y differ, and is denoted by $\mathrm{dist}(x,y) := |\{i \in [n] : x_i \neq y_i\}|/n$. For a positive integer ℓ we denote by $\binom{\Sigma}{\ell}$ the set containing all subsets of Σ of size ℓ, and for any pair of strings $x \in \Sigma^n$ and $S \in \binom{\Sigma}{\ell}^n$ we denote by $\mathrm{dist}(x,S)$ the fraction of coordinates $i \in [n]$ for which $x_i \notin S_i$, that is, $\mathrm{dist}(x,S) := |\{i \in [n] : x_i \notin S_i\}|/n$. Throughout the paper, we use $\exp(n)$ to denote $2^{\Theta(n)}$. Whenever we use log, it is to the base 2.

2.1 Error-correcting codes

Let Σ be an alphabet and let n be a positive integer (the block length). A code is simply a subset $C \subseteq \Sigma^n$. If $\mathbb{F}$ is a finite field and Σ is a vector space over $\mathbb{F}$, we say that a code $C \subseteq \Sigma^n$ is $\mathbb{F}$-linear if it is an $\mathbb{F}$-linear subspace of the $\mathbb{F}$-vector space Σ^n. If $\Sigma = \mathbb{F}$, we simply say that C is linear. The rate of a code is the ratio $\frac{\log|C|}{\log(|\Sigma|^n)}$, which for $\mathbb{F}$-linear codes equals $\frac{\dim_{\mathbb{F}}(C)}{n \cdot \dim_{\mathbb{F}}(\Sigma)}$.

The elements of a code C are called codewords. The relative distance $\mathrm{dist}(C)$ of C is the minimum $\delta > 0$ such that for every pair of distinct codewords $c_1, c_2 \in C$ it holds that $\mathrm{dist}(c_1, c_2) \geq \delta$, which for $\mathbb{F}$-linear codes equals the minimum $\delta > 0$ such that $\mathrm{wt}(c) \geq \delta$ for every $c \in C$. We will use the

notation $\mathrm{dist}(w, C)$ to denote the relative distance of a string $w \in \Sigma^n$ from C, and say that w is ε-close (respectively, ε-far) to C if $\mathrm{dist}(w, C) < \varepsilon$ (respectively, if $\mathrm{dist}(w, C) \geq \varepsilon$).

An encoding map for C is a bijection $E_C : \Sigma^k \to C$, where $|\Sigma|^k = |C|$. For a code $C \subseteq \Sigma^n$ of relative distance δ, a given parameter $\alpha < \delta/2$, and a string $w \in \Sigma^n$, the problem of decoding from α fraction of errors is the task of finding the unique $c \in C$ (if any) which satisfies $\mathrm{dist}(c, w) \leq \alpha$.

List decodable and list recoverable codes. List decoding is a paradigm that allows one to correct more than $\delta/2$ fraction of errors by returning a small list of close-by codewords. More formally, $\alpha \in [0, 1]$ and an integer L we say that a code $C \subseteq \Sigma^n$ is (α, L)-list decodable if for any $w \in \Sigma^n$ there are at most L different codewords $c \in C$ which satisfy that $\mathrm{dist}(c, w) \leq \alpha$. The Johnson bound (see e.g., Corollary 3.2. in [Gur06]) states that any code $C \subseteq \Sigma^n$ of relative distance at least $(1 - \frac{1}{|\Sigma|})\delta$ is (α, L)-list decodable for $\alpha \approx (1 - \frac{1}{|\Sigma|})(1 - \sqrt{1 - \delta})$ and constant L (independent of n).

Theorem 2.1 (Johnson bound)**.** *Let $C \subseteq \Sigma^n$ be a code of relative distance at least $(1 - \frac{1}{|\Sigma|})\delta$. Then C is $\big((1 - \frac{1}{|\Sigma|})\alpha, L\big)$-list decodable for any $\alpha < 1 - \sqrt{1 - \delta}$ with $L = \frac{1}{(1-\alpha)^2 - (1-\delta)}$.*

For decoding concatenated codes it is often useful to consider the notion of list recovery where one is given as input a small list of candidate symbols for each of the coordinates and is required to output a list of codewords that are consistent with the input lists. More specifically, we say that a code $C \subseteq \Sigma^n$ is (α, ℓ, L)-list recoverable if for any $S \in \binom{\Sigma}{\ell}^n$ there are at most L different codewords $c \in C$ which satisfy that $\mathrm{dist}(c, S) \leq \alpha$.

Some useful codes. In what follows we mention several families of codes that will be used in our construction.

Let q be a prime power, let d, n be positive integers such that $d \leq n \leq q$, and let $\alpha_1, \alpha_2, \ldots, \alpha_n$ be n distinct points in $\mathbb{F}_q$. The Reed-Solomon code $RS_n(d, q)$ is the subset of $\mathbb{F}_q^n$ containing all words of the form $(p(\alpha_1), p(\alpha_2), \ldots, p(\alpha_n))$ where $p \in \mathbb{F}_q[x]$ is a univariate polynomial of degree less than d over $\mathbb{F}_q$. It can be verified that $RS_n(d, q)$ has rate d/n and it follows from the Schwartz-Zippel lemma that it has relative distance at least $1 - d/n$. In [Sud97, GS99] it was shown that the Reed-Solomon codes can be efficiently list decoded up to the Johnson bound. We state here a stronger form that applies also to list recovery (see e.g., Theorem 4.11 in [Gur06]).

Theorem 2.2 (List recovery of Reed-Solomon codes)**.** *The following holds for any $\epsilon > 0$, prime power q, and integers d, n, ℓ which satisfy that $\ell d \leq n \leq q$. There exists a deterministic algorithm which given an input string $S \in \binom{\mathbb{F}_q}{\ell}^n$, outputs all codewords $c \in RS_n(d, q)$ such that $\mathrm{dist}(c, S) \leq 1 - \sqrt{\ell \cdot \frac{d}{n}} - \frac{\epsilon}{n}$. The running time of the algorithm is $\mathrm{poly}(q, 1/\epsilon)$.*

For a prime power q and integers $d < q$ and m the Reed-Muller code $RM(m, d, q)$ is the subset of $\mathbb{F}_q^{q^m}$ containing all words of the form $(p(\alpha))_{\alpha \in \mathbb{F}_q^m}$ where $p \in \mathbb{F}_q[x_1, \ldots, x_m]$ is a polynomial of (total) degree less than d in m variables over $\mathbb{F}_q$. Note that $RS_q(d, q) = RM(1, d, q)$ for every d, q. It can also be verified that $RM(m, d, q)$ has rate

$$\frac{\binom{m+d}{m}}{q^m} \geq \left(\frac{d}{mq}\right)^m$$

and relative distance at least $1 - d/q$.

We shall also use the following fact which says that a random binary linear code achieves the Gilbert-Varshamov bound [Gil52, Var57] with high probability. For $x \in [0,1]$ let $H(x) = x \log x + (1-x)\log(1-x)$ denote the binary entropy function.

Fact 2.3 (Gilbert-Varshamov (GV) codes)**.** *For any $\delta \in [0, 1/2)$ and $R \in (0, 1 - H(\delta))$, for sufficiently large n, a random binary linear code of block length n and rate R has relative distance at least δ with probability at least $1 - \exp(-n)$.*

2.2 Locally testable and locally correctable codes

Locally testable codes. Intuitively, a code is said to be locally testable [FS95, RS96a, GS06] if, given a string $w \in \Sigma^n$, it is possible to determine whether w is a codeword of C, or rather far from C, by reading only a small part of w. There are two variants of LTCs in the literature, "weak" LTCs and "strong" LTCs, where the main difference is that weak LTCs are required to reject only words which are of sufficiently large constant relative distance from C, while strong LTCs are required to reject any word w not in C with probability proportional to the relative distance of w from C. From now on, we will work exclusively with strong LTCs, since it is a simpler notion and allows us to state a stronger result.

Definition 2.4 (Locally testable code (LTC))**.** We say that a code $C \subseteq \Sigma^n$ is q-(strongly) locally testable if there exists a randomized algorithm A that satisfies the following requirements:

- **Input:** A gets oracle access to a string $w \in \Sigma^n$.
- **Query complexity:** A makes at most q queries to the oracle w.
- **Completeness:** If w is a codeword of C, then A accepts with probability 1.
- **Soundness:** If w is not a codeword of C, then A rejects with probability at least $\mathrm{dist}(w, C)/4$.

We say that the algorithm A is a local tester of C. Given an infinite family of LTCs $\{C_n\}_n$, a uniform local tester for the family is a randomized oracle algorithm that given n, computes the local tester of C_n. We will often also be interested in the running time of the uniform local tester.

Remark 2.5. It is common to define strong LTCs with an additional parameter ρ, and have the following soundness requirement:

- If w is not a codeword of C, then A rejects with probability at least $\rho \cdot \mathrm{dist}(w, C)$.

Our definition corresponds to the special case where $\rho = \frac{1}{4}$. However, given an LTC with $\rho < \frac{1}{4}$, it is possible to amplify ρ up to $\frac{1}{4}$ at the cost of increasing the query complexity by a multiplicative factor of $1/\rho$. Hence, we chose to fix ρ to $\frac{1}{4}$ in our definition, which somewhat simplifies the presentation. See further discussion in [KMRS15b, Section 2.2.].

Locally correctable codes. Intuitively, a code is said to be locally correctable [BFLS91, STV01, KT00] if, given a codeword $c \in C$ that has been corrupted by some errors, it is possible to decode any coordinate of c by reading only a small part of the corrupted version of c. Formally, it is defined as follows.

Definition 2.6 (Locally correctable code (LCC))**.** We say that a code $C \subseteq \Sigma^n$ is (q, α)-locally correctable if there exists a randomized algorithm A that satisfies the following requirements:

- **Input:** A takes as input a coordinate $i \in [n]$ and also gets oracle access to a string $w \in \Sigma^n$ that is α-close to a codeword $c \in C$.
- **Query complexity:** A makes at most q queries to the oracle w.
- **Output:** A outputs c_i with probability at least $\frac{2}{3}$.

We say that the algorithm A is a local corrector of C. Given an infinite family of LCCs $\{C_n\}_n$, a uniform local corrector for the family is a randomized oracle algorithm that given n, computes the local corrector of C_n. Again, we will often be also interested in the running time of the uniform local corrector.

Remark 2.7. By definition it holds that $\alpha < \mathrm{dist}(C)/2$. The above success probability of $\frac{2}{3}$ can be amplified using sequential repetition, at the cost of increasing the query complexity. Specifically, amplifying the success probability to $1 - e^{-t}$ requires increasing the query complexity by a multiplicative factor of $O(t)$.

Locally list decodable and list recoverable codes. The following definition generalizes the notion of locally correctable codes to the setting of list decoding. In this setting the algorithm A is required to find all the nearby codewords in an implicit sense. Note that our definition below also includes a nonstandard soundness property.

Definition 2.8 (Locally list decodable code)**.** We say that a code $C \subseteq \Sigma^n$ is (q, α, ϵ, L)-locally list decodable if there exists a randomized algorithm A that satisfies the following requirements:

- **Input:** A gets oracle access to a string $w \in \Sigma^n$.
- **Query complexity:** A makes at most q queries to the oracle w.
- **Output:** A outputs L randomized algorithms $A_1, \ldots, A_L$. When algorithm A_j is given as input a coordinate $i \in [n]$, it makes at most q queries to the oracle w and outputs a symbol in Σ.
- **Completeness:** For every codeword $c \in C$ that is α-close to w, with probability at least $1 - \epsilon$ over the randomness of A the following event happens: there exists some $j \in [L]$ such that for all $i \in [n]$,
$$\Pr[A_j(i) = c_i] \geq \frac{2}{3},$$
where the probability is over the internal randomness of A_j.
- **Soundness:** With probability at least $1 - \epsilon$ over the randomness of A, the following event happens: for every $j \in [L]$, there exists some $c \in C$ such that for all $i \in [n]$,
$$\Pr[A_j(i) = c_i] \geq \frac{2}{3},$$
where the probability is over the internal randomness of A_j.

We say that A has running time T if A outputs the description of the algorithms $A_1, \ldots, A_L$ in time at most T and each A_j has running time at most T.

Remark 2.9. (On the soundness property) Typically locally list decodable codes are defined without the soundness property. For us, the soundness property is important to allow us to identify the unique closest codeword to the given received word.

In a later section, we will first construct a locally list decodable code without the soundness property, and then we will achieve soundness via *local testing*.

The definition of locally list decodable codes can also be extended to the setting of list recovery. The same remarks about the soundness property apply to this case also.

Definition 2.10 (Locally list recoverable code)**.** We say that a code $C \subseteq \Sigma^n$ is $(q, \alpha, \epsilon, \ell, L)$-locally list recoverable if there exists a randomized algorithm A that satisfies the following requirements:

- **Input:** A gets oracle access to a string $S \in \binom{\Sigma}{\ell}^n$.
- **Query complexity:** A makes at most q queries to the oracle S.
- **Output:** A outputs L randomized algorithms $A_1, \ldots, A_L$, where each A_j takes as input a coordinate $i \in [n]$, makes at most q queries to the oracle S and outputs a symbol in Σ.
- **Completeness:** For every codeword $c \in C$ for which $\mathrm{dist}(c, S) \leq \alpha$, with probability at least $1 - \epsilon$ over the randomness of A the following event happens: there exists some $j \in [L]$ such that for all $i \in [n]$,
$$\Pr[A_j(i) = c_i] \geq \frac{2}{3},$$
where the probability is over the internal randomness of A_j.
- **Soundness:** With probability at least $1 - \epsilon$ over the randomness of A, the following event happens: for every $j \in [L]$, there exists some $c \in C$ such that for all $i \in [n]$,
$$\Pr[A_j(i) = c_i] \geq \frac{2}{3},$$
where the probability is over the internal randomness of A_j.

As above, we say that A has running time T if A outputs the description of the algorithms $A_1, \ldots, A_L$ in time at most T and each A_j has running time at most T. Note that a code is (q, α, ϵ, L)-locally list decodable if and only if it is $(q, \alpha, \epsilon, 1, L)$-locally list recoverable.

3 LTCs approaching the GV bound

In this section we prove the following theorem which implies Theorem A from the introduction.

Theorem 3.1 (LTCs approaching the GV bound)**.** *For any $\delta \in [0, \frac{1}{2})$ and $\gamma > 0$ there exists an infinite family $\{C'_n\}_n$ of binary linear codes which satisfy the following. The code C'_n has block length n, rate at least $1 - H(\delta) - \gamma$, relative distance at least δ, and is $(\log n)^{O(\log \log n)}$-locally testable.*

Moreover, the code C'_n can be constructed by a randomized polynomial time algorithm that succeeds with probability $1 - \exp(-n)$, and the local testing algorithm can be implemented[6] *to run in time $(\log n)^{O(\log \log n)}$.*

To prove the above theorem we first show in Section 3.1 a transformation which turns any large alphabet code approaching the Singleton bound into a binary code approaching the GV bound. In Section 3.2 we will then use this transformation to obtain LTCs approaching the GV bound.

3.1 Approaching the GV bound via random concatenation

Thommesen [Tho83] used the operation of random concatenation to construct a binary code lying on the GV bound out of a large alphabet code lying on the Singleton bound. The following lemma shows an approximate version of this, replacing "lying on" with "close to". The proof is identical to Thommesen's.

Lemma 3.2. *The following holds for any $\delta \in [0, 1/2)$, $0 < \gamma < 1 - H(\delta)$, $t \geq \frac{4}{\gamma}$, and sufficiently large n. Let $C \subseteq \mathbb{F}_{2^t}^n$ be a linear code of rate $R = 1 - H(\delta) - \gamma$ and relative distance at least $1 - R - \frac{\gamma}{2}$. Let $C' \subseteq \mathbb{F}_2^{t \cdot n}$ be a code obtained from C by applying a (uniformly) random invertible $\mathbb{F}_2$-linear transformation $T_i : \mathbb{F}_{2^t} \to \mathbb{F}_2^t$ on each coordinate $i \in [n]$ of C independently. Then C' has relative distance at least δ with probability at least $1 - \exp(-n)$.*

Proof. The proof follows the arguments of [Tho83].

Fix a codeword $c \in C$ with $\mathrm{wt}(c) = \alpha \geq 1 - R - \gamma/2$ and let $c' \in \mathbb{F}_2^{tn}$ be a word obtained from c by applying a uniformly random *not-necessarily-invertible* $\mathbb{F}_2$-linear transformation $T_i : \mathbb{F}_{2^t} \to \mathbb{F}_2^t$ on each coordinate $i \in [n]$ of c independently. Note that this is a different distribution from what was used in the statement of the lemma, since there is a (good) chance that the uniformly random $\mathbb{F}_2$-linear transformation is not invertible.

Then for each non-zero coordinate i of c it holds that the i-th block of c' of length t is distributed uniformly over $\mathbb{F}_2^t$, and so αtn coordinates of c' are uniformly distributed (while the rest equal zero).

Consequently, we have that

$$\Pr[\mathrm{wt}(c') < \delta] \leq \binom{\alpha tn}{\leq \delta tn} 2^{-\alpha tn} \leq 2^{H(\delta/\alpha)\alpha tn} \cdot 2^{-\alpha tn},$$

where the second inequality follows from the well known fact $\binom{m}{\leq \beta m} \leq 2^{H(\beta) \cdot m}$ for $\beta \leq 1/2$. Now note that the above inequality continues to hold even if the transformations T_i are chosen to be uniformly random *invertible* transformations since this only increases the probability of any non-zero element in the image of T_i by the same multiplicative factor.

Next we apply a union bound over all codewords $c \in C$. For this fix $\alpha > 0$ such that $\alpha \geq 1 - R - \gamma/2$ and $\alpha n \in \mathbb{N}$. Then it holds that the number of codewords in C of relative weight α is at most

$$\binom{n}{\alpha n} \cdot \left(2^t\right)^{\alpha n - (1 - R - \gamma/2)n} \leq 2^n \cdot 2^{(\alpha - (1 - R - \gamma/2))tn},$$

[6] To be precise, in order for the local testing algorithm to run in the claimed time, it needs access to the random choices made during the construction of the code.

where the above bound follows since there are at most $\binom{n}{\alpha n}$ choices for the location of the non-zero coordinates, and for any such choice fixing the value of the first $\alpha n - (1 - R - \gamma/2)n$ non-zero coordinates determines the value of the rest of the non-zero coordinates (since two different codewords cannot differ on less that $(1 - R - \gamma/2)n$ coordinates).

Consequently, we have that

$$\begin{aligned}
\Pr[\mathrm{dist}(C') < \delta] &\leq \sum_{1-R-\gamma/2 \leq \alpha \leq 1,\ \alpha n \in \mathbb{N}} 2^n \cdot 2^{(\alpha-(1-R-\gamma/2))tn} \cdot 2^{H(\delta/\alpha)\alpha tn} \cdot 2^{-\alpha tn} \\
&= \sum_{1-R-\gamma/2 \leq \alpha \leq 1,\ \alpha n \in \mathbb{N}} \exp\left[-tn\left(\left(1 - R - \frac{\gamma}{2}\right) - \alpha \cdot H\left(\frac{\delta}{\alpha}\right) - \frac{1}{t}\right)\right] \\
&\leq \sum_{1-R-\gamma/2 \leq \alpha \leq 1,\ \alpha n \in \mathbb{N}} \exp\left[-tn\left((1 - R - \gamma) - \alpha \cdot H\left(\frac{\delta}{\alpha}\right) + \frac{1}{t}\right)\right],
\end{aligned}$$

where the last inequality follows by assumption that $t \geq 4/\gamma$.

So for $\mathrm{dist}(C') \geq \delta$ to hold with probability at least $1 - \exp(-n)$ it suffices to show that for any $1 - R - \gamma/2 \leq \alpha \leq 1$,

$$1 - R - \gamma \geq \alpha \cdot H\left(\frac{\delta}{\alpha}\right),$$

or equivalently,

$$\alpha \cdot H^{-1}\left(\frac{1 - R - \gamma}{\alpha}\right) \geq \delta.$$

To proceed, we recall an elementary inequality implicit in [Tho83] (see Lemma 3 in [GR10] for an explicit form). Let $H^{-1} : [0, 1] \to [0, \frac{1}{2}]$ be the inverse of the binary entropy function H in the domain $[0, \frac{1}{2}]$.

Fact 3.3. *For any $0 \leq x \leq y \leq 1$ it holds that $\frac{H^{-1}(x)}{x} \leq \frac{H^{-1}(y)}{y}$.*

We now complete the proof of the lemma.

$$\begin{aligned}
\alpha \cdot H^{-1}\left(\frac{1 - R - \gamma}{\alpha}\right) &= (1 - R - \gamma) \cdot \frac{H^{-1}((1 - R - \gamma)/\alpha)}{(1 - R - \gamma)/\alpha} \\
&\geq (1 - R - \gamma) \cdot \frac{H^{-1}(1 - R - \gamma)}{1 - R - \gamma} \\
&= H^{-1}(1 - R - \gamma) \\
&= \delta,
\end{aligned}$$

where the second inequality follows from Fact 3.3. □

3.2 LTCs approaching the GV bound

We now use Lemma 3.2 to show the existence of LTCs approaching the GV bound. To this end we first prove the following lemma which says that if C is locally testable then so is C'.

Lemma 3.4. *Let $C \subseteq \mathbb{F}_{2^t}^n$ be a code and let $C' \subseteq \mathbb{F}_2^{t \cdot n}$ be a code obtained from C by applying an invertible transformation $T_i : \mathbb{F}_{2^t} \to \mathbb{F}_2^t$ on each coordinate $i \in [n]$ of C. Suppose furthermore that C is q-locally testable in time T. Then C' is $O(q \cdot t^2)$-locally testable in time $O(T \cdot \mathrm{poly}(t))$.*

Proof. Let A be the local tester for C. Given oracle access to $w' \in \mathbb{F}_2^{t \cdot n}$ the local tester A' for C' runs A and answers each query i of A by inverting T_i on the i-th block of w' of length t.

The completeness property clearly holds. To show that the soundness property holds as well suppose that $w' \notin C'$ and let $w \in \mathbb{F}_{2^t}^n$ be the word obtained from w' by inverting all the transformations T_i. Then A' rejects w' with probability at least $\frac{1}{4} \cdot \mathrm{dist}(w, C) \geq \frac{1}{4} \cdot \mathrm{dist}(w', C')/t$, and the rejection probability could be amplified to $\frac{1}{4} \cdot \mathrm{dist}(w', C')$ by repeating the test independently $O(t)$ times. Finally, note that the overall query complexity is $O(q \cdot t^2)$ and overall running time is $O(T \cdot \mathrm{poly}(t))$. □

To obtain the final LTCs we shall also use the following theorem from [KMRS16, Theorem 1.2] which states the existence of LTCs approaching the Singleton bound.

Theorem 3.5 (LTCs approaching the Singleton bound)**.** *For any $\gamma > 0$, $0 < R \leq 1 - \gamma$, and $t \geq \mathrm{poly}(1/\gamma)$ there exists an infinite family $\{C_n\}_n$ of linear codes where each $C_n \subseteq \mathbb{F}_{2^t}^n$ has rate R, relative distance at least $1 - R - \gamma$, and is $(\log n)^{O(\log \log n)}$-locally testable.*

Moreover, the code C_n can be constructed by a deterministic polynomial time algorithm, and the local testing algorithm can be implemented to run in time $(\log n)^{O(\log \log n)}$.

Proof of Theorem 3.1. Let $\delta \in [0, \frac{1}{2})$, $0 < \gamma < 1 - H(\delta)$ and $t \in \mathbb{N}$ such that $t \geq \mathrm{poly}(4/\gamma)$ be fixed constants. By Theorem 3.5, there exists a family of codes $\{C_n\}_n$ with rate $R < 1 - H(\delta) - \gamma$ and relative distance at least $1 - R - \gamma/2 = H(\delta) + \frac{\gamma}{2}$ which is $(\log n)^{O(\log \log n)}$-locally testable.

Lemma 3.2 implies that each member of the family of codes $\{C'_n\}_n$ has relative distance at least δ with probability at least $1 - \exp(-n)$. Moreover, Lemma 3.4 implies that each C'_n is $O(t^2 \cdot (\log n)^{O(\log \log n)}) = (\log n)^{O(\log \log n)}$-locally testable in time $(\log n)^{O(\log \log n)}$, as we wanted. □

4 Approaching the GV bound via random concatenation, again

We now begin working towards our LCC constructions. In this section, we revisit the random concatenation operation and show that it can be used to get codes approaching the GV bound with weaker hypotheses on the outer code. In Section 3.1 we showed that given a large alphabet code close to the Singleton bound, we can get a binary code close to the GV bound. We now show that even if we are given a large alphabet code quite far from the Singleton bound (but with some decent rate-distance tradeoff), the random concatenation operation still gives a binary code which is quite close to the GV bound.

More precisely, given any large alphabet code of rate $O(\xi^2)$ and relative distance $1 - O(\gamma\xi)$ for $\xi = O(\gamma)$, the following lemma shows that we can then construct a binary code with rate only a $(1 - \gamma)$ factor away from the GV bound. In Section 5 we will use this lemma to obtain LCCs nearly-approaching the GV bound with some small (but constant) rate.

Lemma 4.1. *There exists an absolute constant c such that the following holds for any $\gamma > 0$, $0 < \xi \leq \gamma/c$, integer t, and sufficiently large n. Let $C \subseteq \mathbb{F}_{2^t}^n$ be a linear code of rate R and relative distance at least $1 - \frac{\gamma}{6} \cdot \xi$. Let $C' \subseteq \mathbb{F}_2^{t \cdot n/r}$ be a code obtained from C by applying a random $\mathbb{F}_2$-linear transformation $T_i : \mathbb{F}_{2^t} \to \mathbb{F}_2^{t/r}$ on each coordinate $i \in [n]$ of C independently.*

Suppose furthermore that

$$r \cdot R \leq \left(1 - H\left(\frac{1}{2} - \xi\right)\right)(1 - \gamma).$$

Then C' has relative distance at least $\frac{1}{2} - \xi$ with probability at least $1 - \exp(-n)$.

For the proof of the above lemma we shall use the Taylor expansion of the binary entropy function at half. The formula follows easily from the Taylor expansion of $\log(1+x)$ at zero.

Fact 4.2. *For any $|x| \leq 1$ it holds that*

$$H\left(\frac{1+x}{2}\right) = 1 - \sum_{k=1}^{\infty} \frac{x^{2k}}{(2k-1)\cdot 2k \cdot \ln 2}.$$

Proof of Lemma 4.1. Fix a codeword $c \in C$, and note that $\mathrm{wt}(c) \geq 1 - \frac{\gamma}{6}\cdot\xi$. Let $c' \in \mathbb{F}_2^{tn/r}$ be a word obtained from c by applying a random linear transformation $T_i : \mathbb{F}_{2^t} \to \mathbb{F}_2^{t/r}$ on each coordinate $i \in [n]$ of c independently. Then for each non-zero coordinate i of c it holds that the i-th block of c' of length t/r is distributed uniformly over $\mathbb{F}_2^{t/r}$, and so at least $(1 - \frac{\gamma}{6}\cdot\xi)\cdot tn/r$ coordinates of c' are uniformly distributed.

Consequently it holds that

$$\begin{aligned}
\Pr\left[\mathrm{wt}(c') < \frac{1}{2} - \xi\right] &\leq \sum_{i=0}^{(1/2-\xi)tn/r} \binom{(1-\xi\gamma/6)tn/r}{i} 2^{-(1-\xi\gamma/6)\cdot tn/r} \\
&\leq 2^{H\left(\frac{1/2-\xi}{1-\xi\gamma/6}\right)\cdot(1-\xi\gamma/6)\cdot tn/r} \cdot 2^{-(1-\xi\gamma/6)\cdot tn/r}.
\end{aligned}$$

By union bound over all codewords $c \in C$, recalling that $|C| = 2^{tRn}$, the above implies in turn that

$$\begin{aligned}
\Pr\left[\mathrm{dist}(C') < \frac{1}{2} - \xi\right] &\leq 2^{tRn} \cdot 2^{H\left(\frac{1/2-\xi}{1-\xi\gamma/6}\right)\cdot(1-\xi\gamma/6)\cdot tn/r} \cdot 2^{-(1-\xi\gamma/6)\cdot tn/r} \\
&= \exp\left[-\frac{tn}{r}\left(\left(1-\frac{\xi\gamma}{6}\right)\cdot\left(1 - H\left(\frac{1/2-\xi}{1-\xi\gamma/6}\right)\right) - r\cdot R\right)\right].
\end{aligned}$$

So for $\mathrm{dist}(C') \geq \frac{1}{2} - \xi$ to hold with probability at least $1 - \exp(-n)$ it suffices to show that

$$\left(1-\frac{\xi\gamma}{6}\right)\cdot\left(1 - H\left(\frac{1/2-\xi}{1-\xi\gamma/6}\right)\right) > r\cdot R.$$

For this we compute

$$\begin{aligned}
1 - H\left(\frac{1/2-\xi}{1-\xi\gamma/6}\right) &\geq 1 - H\Big((1/2-\xi)\cdot(1+\xi\gamma/3)\Big) \\
&\geq 1 - H\left(\frac{1}{2} - \xi\,(1-\gamma/6)\right) \\
&\geq \left(1-\frac{\gamma}{6}\right)^2 \cdot \left(1 - H\left(\frac{1}{2}-\xi\right)\right)\cdot\left(1-\frac{c}{2}\cdot\xi\right),
\end{aligned}$$

where the last inequality follows from Fact 4.2 for some absolute constant c.

Consequently we have that

$$\begin{aligned}\left(1-\frac{\xi\gamma}{6}\right)\cdot\left(1-H\left(\frac{1/2-\xi}{1-\xi\gamma/6}\right)\right) &\geq \left(1-\frac{\xi\gamma}{6}\right)\cdot\left(1-\frac{\gamma}{6}\right)^2\cdot\left(1-\frac{c}{2}\cdot\xi\right)\cdot\left(1-H\left(\frac{1}{2}-\xi\right)\right)\\ &> \left(1-\frac{\gamma}{6}-\frac{\gamma}{3}-\frac{c}{2}\cdot\xi\right)\cdot\left(1-H\left(\frac{1}{2}-\xi\right)\right)\\ &\geq (1-\gamma)\cdot\left(1-H\left(\frac{1}{2}-\xi\right)\right)\\ &\geq r\cdot R,\end{aligned}$$

where the third inequality follows by choice of $\xi \leq \gamma/c$ and the fourth inequality follows by choice of $r \cdot R \leq \left(1-H\left(\frac{1}{2}-\xi\right)\right)(1-\gamma)$. □

5 LCCs approaching the GV bound

In this section we prove the following theorem which implies Theorem B from the introduction.

Theorem 5.1 (LCCs approaching the GV bound)**.** *For any constants $\beta, \gamma > 0$ there exists a constant $\xi_0 = \xi_0(\beta,\gamma)$, such that for any constant $\xi > 0$ which satisfy that $\xi \leq \xi_0$ there exists an infinite family $\{C'_n\}_n$ of binary linear codes which satisfy the following. The code C'_n has block length at least n, rate $\left(1-H\left(\frac{1}{2}-\xi\right)\right)(1-\gamma)$, relative distance at least $\frac{1}{2}-\xi$, and is $\left(n^\beta \cdot \mathrm{poly}(1/\gamma'), \frac{1}{2}\cdot\left(\frac{1}{2}-\xi\right)-\gamma'\right)$-locally correctable for any $\gamma' > 0$.*

Moreover, the code C'_n can be constructed by a randomized polynomial time algorithm that succeeds with probability $1-\exp(-n)$, and the local correction algorithm can be implemented to run in time $\mathrm{poly}(n^\beta, 1/\gamma')$.

5.1 Proof overview and main ingredients

For the proof of the above theorem we shall use Lemma 4.1 as well as the following three lemmas.

The first lemma establishes (an alphabet independent) Johnson bound for list recovery. For a similar (alphabet dependent) statement and a proof sketch, we refer the reader to Theorem 5 in [GS01]. For completeness we provide a simple combinatorial proof of this lemma in Appendix A, based on the proof of the Johnson bound for list decoding given in [Gur06].

Lemma 5.2 (Johnson bound for list recovery)**.** *Let $C \subseteq \Sigma^n$ be a code of relative distance at least δ. Then C is (α,ℓ,L)-list recoverable for any $\alpha < 1-\sqrt{\ell\cdot(1-\delta)}$ with $L = \frac{\delta\ell}{(1-\alpha)^2-\ell(1-\delta)}$.*

The second lemma gives a local list recovery algorithm for Reed-Muller codes. The algorithm is similar to the local list decoding algorithm for Reed-Muller codes from [STV01], with an additional local testing procedure that guarantees the soundness requirement in our definition of locally list recoverable codes, and is given in Section 6.

Lemma 5.3 (Local list recovery of Reed-Muller codes)**.** *There exists an absolute constant c' such that for any $\alpha, \epsilon > 0$ and integers m, d, q, ℓ which satisfy $\alpha < 1 - c'\cdot\sqrt{\frac{\ell d}{q}}$ the Reed-Muller code $RM(m,d,q)$ is $\left(O(q^2\cdot\log(q/\epsilon)), \alpha, \epsilon, \ell, O(q\log(1/\epsilon))\right)$-locally list recoverable. Moreover, the local list recovery algorithm can be implemented to run in $\mathrm{poly}(m,q,\log(1/\epsilon))$ time.*

Finally, we shall use the following lemma which gives a distance amplification procedure for local list recovery. This procedure is similar to the distance amplification procedure for locally correctable codes from [KMRS16], and is given in Section 7.

Lemma 5.4 (Distance amplification for local list recovery)**.** *For any constants $\delta_{out}, \alpha_{out}, \gamma > 0$ there exists an integer $d = d(\delta_{out}, \alpha_{out}, \gamma) \leq poly(1/\delta_{out}, 1/\alpha_{out}, 1/\gamma)$ such that the following holds.*

- *C_{out} be an $\mathbb{F}$-linear code of block length n_{out}, alphabet size σ_{out}, rate r_{out}, and relative distance δ_{out} that is $(q, \alpha_{out}, \epsilon, \ell_{out}, L_{out})$-locally list recoverable.*
- *C_{in} be an $\mathbb{F}$-linear code of block length n_{in}, alphabet size σ_{in}, rate r_{in}, and relative distance δ_{in} that is $(\alpha_{in}, \ell_{in}, L_{in})$-(globally) list recoverable.*
- *additionally, suppose that $n_{in} \geq d$, $\sigma_{out} = \sigma_{in}^{r_{in} \cdot n_{in}}$ and $L_{in} \leq \ell_{out}$.*

Then there exists an $\mathbb{F}$-linear code C of block length n_{out}, alphabet size $\sigma_{in}^{n_{in}}$, rate $r_{in} \cdot r_{out}$ and relative distance at least $\delta_{in} - 2\gamma$ that is $(O(q \cdot n_{in}^2 \cdot \log(n_{in})), \alpha_{in} - \gamma, \epsilon, \ell_{in}, L_{out})$-locally list recoverable.

Moreover, if the running time of the local list recovery algorithm for C_{out} is T_{out} and the running time of the global list recovery algorithm for C_{in} is T_{in} then the running time of the local list recovery algorithm for C is

$$O(T_{out}) + O(q \cdot T_{in}) + \text{poly}(q, n_{in}, \ell_{in}).$$

For the construction of the code $C' := C'_n$ of Theorem 5.1 we first choose the code C_{in} to be a Reed-Solomon code of rate $r_{in} = c_\beta \cdot \Theta(\xi^2)$ and relative distance $\delta_{in} = 1 - c_\beta \cdot \Theta(\xi^2)$ for some constant c_β depending only on β, and use Lemma 5.2 to argue that it can be (globally) list recovered from sufficiently large $\alpha_{in} = \Theta(1)$ fraction of errors. We then use Lemma 5.3 to obtain a Reed-Muller code C_{out} of rate $r_{out} = 1/c_\beta$ and relative distance $\delta_{out} = \Theta(1)$ that is locally list recoverable from sufficiently large $\alpha_{out} = \Theta(1)$ fraction of errors (we will in fact concatenate the Reed-Muller code with another Reed-Solomon code with appropriate parameters to slightly reduce the alphabet size). Next we apply the distance amplification procedure of Lemma 5.4 with outer code C_{out}, inner code C_{in}, and $d = d(\delta_{out}, \alpha_{out}, O(\gamma\xi))$ to obtain a code C of rate $\Theta(\xi^2)$ and relative distance at least $1 - c_\beta \cdot \Theta(\xi^2) - O(\gamma\xi) \geq 1 - O(\gamma\xi)$ (where the inequality will hold for sufficiently small $\xi \leq \xi_0(\beta, \gamma)$) that is locally list recoverable from sufficiently large $\Theta(1)$ fraction of errors. Finally, we apply Lemma 4.1 to the code C to obtain the desired LCC C' that approaches the GV bound. Full details of the construction appear in Section 5.2.

In Section 5.3, we analyze the rate and relative distance of C'. In Section 5.4, we show that the code C' is locally list decodable from $1/4$ fraction of errors. We then use this property in Section 5.5 to show that C' is locally correctable from half the GV bound. This shows that C' satisfies the local correction requirement.

5.2 Construction of C'

In what follows we present the construction of the code $C' := C'_n$. To this end we first set some parameters and then describe the construction of the codes C_{in},C_{out}, C and C'. For better readability, in what follows we will denote each variable v that is set to some absolute constant (independent of $\beta, \gamma, \xi, \gamma'$ and n) by $\hat{v}$. In what follows we will assume that $n^{\beta/4}$ is a sufficiently large power of 2.

Parameter setting. Let $\frac{1}{4} < \hat{\alpha}_0 < \frac{1}{2}$ be an arbitrary constant, and note that by the Johnson bound for list decoding (Theorem 2.1) there exist a constant $\hat{\delta}_0 \in [0, 1/2)$ and an integer $\hat{L}_0$ such that any binary code of relative distance at least $\hat{\delta}_0$ is $(\hat{\alpha}_0, \hat{L}_0)$-list decodable. We will choose the parameters below so that the random binary codes encoded by the T_i's of Lemma 4.1 will have relative distance at least $\hat{\delta}_0$ (with high probability), and the code C will be locally list recoverable from input lists of size $\hat{L}_0$ and sufficiently large constant fraction of errors. It will then follow that the final code C' is locally list decodable from $\frac{1}{4}$ fraction of errors and consequently locally correctable from half the GV bound.

The code C_{in}. Choose an arbitrary constant $0 < \hat{\alpha}_{in} < 1$ such that $\hat{\alpha}_{in} \cdot \hat{\alpha}_0 > \frac{1}{4}$ (such $\hat{\alpha}_{in}$ exists since $\hat{\alpha}_0 > \frac{1}{4}$), and note that by the Johnson bound for list recovery (Lemma 5.2) there exist a constant $\hat{\delta}_{in} \in (0, 1)$ and an integer $\hat{L}_{in}$ such that any code of relative distance at least $\hat{\delta}_{in}$ is $(\hat{\alpha}_{in}, \hat{L}_0, \hat{L}_{in})$-list recoverable.

Let $\sigma_{in} \geq n_{in}$ be growing functions of n such that $\sigma_{in}^{n_{in}} = n^{\beta/4}$ and σ_{in} is a power of 2, and let $\delta_{in} = \delta_{in}(\beta, \gamma, \xi) < 1$ be a constant to be determined later on which satisfies that $\delta_{in} \geq \hat{\delta}_{in}$. Let C_{in} be a Reed-Solomon code of block length n_{in}, alphabet size σ_{in}, relative distance δ_{in}, and rate $r_{in} := 1 - \delta_{in}$. Then by the above discussion the code C_{in} is $(\hat{\alpha}_{in}, \hat{L}_0, \hat{L}_{in})$-(globally) list recoverable in time $\mathrm{poly}(n^\beta)$ (via brute force).

The code C_{out}. The code C_{out} will be a concatenation of an outer Reed-Muller code C'_{out} with an inner Reed-Solomon code C''_{out}. We start by defining the inner Reed-Solomon code C''_{out}.

Choose an arbitrary constant $0 < \hat{\alpha}''_{out} < 1$, and note that by the Johnson bound for list recovery (Lemma 5.2) there exist a constant $\hat{\delta}''_{out} \in (0, 1)$ and an integer $\hat{L}''_{out}$ such that any code of relative distance at least $\hat{\delta}''_{out}$ is $(\hat{\alpha}''_{out}, \hat{L}_{in}, \hat{L}''_{out})$-list recoverable. Let C''_{out} be a Reed-Solomon code of relative distance $\hat{\delta}''_{out}$, rate $\hat{r}''_{out} := 1 - \hat{\delta}''_{out}$, block length $1/(r_{in}\hat{r}''_{out})$ and alphabet size $n^{r_{in} \cdot \beta/4}$. Then by the above the code C''_{out} is $(\hat{\alpha}''_{out}, \hat{L}_{in}, \hat{L}''_{out})$-(globally) list recoverable in time $\mathrm{poly}(n^\beta)$ (via brute force).

Next we define the outer Reed-Muller code C'_{out}. Choose an arbitrary constant $0 < \hat{\alpha}'_{out} < 1$, and note that by Lemma 5.3 there exists a constant $\hat{\delta}'_{out} \in (0, 1)$ such that a Reed-Muller code of relative distance $\hat{\delta}'_{out}$, block length $n \cdot r_{in}\hat{r}''_{out}$ and alphabet size $n^{\beta/4}$ is $\left(n^{\beta/2} \operatorname{polylog} n, \hat{\alpha}'_{out}, \frac{1}{n}, \hat{L}''_{out}, n^{\beta/4} \operatorname{polylog} n\right)$-locally list recoverable in time $\mathrm{poly}(n^\beta)$. Let C'_{out} be a Reed-Muller code of block length $n \cdot r_{in} \cdot \hat{r}''_{out}$, alphabet size $n^{\beta/4}$, relative distance $\hat{\delta}'_{out}$, and rate $r'_{out} = r'_{out}(\beta)$. Then the code C'_{out} is $\left(n^{\beta/2} \operatorname{polylog} n, \hat{\alpha}'_{out}, \frac{1}{n}, \hat{L}''_{out}, n^{\beta/4} \operatorname{polylog} n\right)$-locally list recoverable in time $\mathrm{poly}(n^\beta)$.

Finally, let C_{out} be the code obtained by concatenating the outer Reed-Muller code C'_{out} with the inner Reed-Solomon code C''_{out}. Then it can be verified that C_{out} is an $\mathbb{F}_2$-linear code of block length n, alphabet size $n^{r_{in} \cdot \beta/4}$, relative distance $\hat{\delta}'_{out} \cdot \hat{\delta}''_{out}$, rate $r'_{out} \cdot \hat{r}''_{out}$, and in addition it is $\left(n^{\beta/2} \operatorname{polylog} n, \hat{\alpha}'_{out} \cdot \hat{\alpha}''_{out}, \frac{1}{n}, \hat{L}_{in}, n^{\beta/4} \operatorname{polylog} n\right)$-locally list recoverable in time $\mathrm{poly}(n^\beta)$ by emulating the local list recovery algorithm of C'_{out} in the natural way.

The code C. Let C be the code guaranteed by Lemma 5.4 for the codes C_{out}, C_{in} and $d = d\left(\hat{\delta}'_{out} \cdot \hat{\delta}''_{out}, \hat{\alpha}'_{out} \cdot \hat{\alpha}''_{out}, \frac{\gamma}{24} \cdot \xi\right)$ (note that $n_{in} \geq d$ when n is large enough, $\sigma_{out} = \sigma_{in}^{r_{in} \cdot n_{in}}$, and $L_{in} = \hat{L}_{in} = \ell_{out}$). Then C is an $\mathbb{F}_2$-linear code of block length n, alphabet size $n^{\beta/4}$, relative distance at least $\delta_{in} - \frac{\gamma}{12} \cdot \xi$, rate $r_{in} \cdot r'_{out} \cdot \hat{r}''_{out}$, and is $\left(n^{\beta/2} \operatorname{polylog} n, \hat{\alpha}_{in} - \frac{\gamma}{12} \cdot \xi, \frac{1}{n}, \hat{L}_0, n^{\beta/4} \operatorname{polylog} n\right)$-locally

list recoverable in time poly(n^β).

The code C'. Let $t = \log(n^{\beta/4})$ and let $\hat{r}_0$ be a constant such that a random binary linear code of rate $\hat{r}_0$ and block length t has relative distance at least $\hat{\delta}_0$ with probability at least $1-\exp(-t)$ (such $\hat{r}_0$ exists by Fact 2.3). Considering each symbol of the code C as an element of $\mathbb{F}_{2^t}$, let $C' \subseteq \mathbb{F}_2^{t \cdot n/\hat{r}_0}$ be a code obtained from C by applying a random $\mathbb{F}_2$-linear transformation $T_i : \mathbb{F}_{2^t} \to \mathbb{F}_2^{t/\hat{r}_0}$ on each coordinate $i \in [n]$ of C independently.

Choice of δ_{in} and ξ_0. Finally, we set

$$\delta_{in} = \delta_{in}(\beta, \gamma, \xi) = 1 - \frac{\left(1 - H\left(\frac{1}{2} - \xi\right)\right)(1-\gamma)}{r'_{out} \cdot \hat{r}''_{out} \cdot \hat{r}_0},$$

and

$$\xi_0 := \min\left\{ \frac{1-\hat{\delta}_{in}}{5} \cdot r'_{out} \cdot \hat{r}''_{out} \cdot \hat{r}_0,\ \ \frac{\gamma}{60} \cdot r'_{out} \cdot \hat{r}''_{out} \cdot \hat{r}_0,\ \ \frac{\gamma}{c},\ \ \hat{\alpha}_{in} - \frac{1}{4\hat{\alpha}_0} \right\}, \tag{1}$$

where c is the constant guaranteed by Lemma 4.1. Note that ξ_0 depends only on β and γ (recalling that r'_{out} depends only on β), and that $\delta_{in} \geq \hat{\delta}_{in}$ whenever $\xi \leq \xi_0$ by choice of $\xi_0 \leq \frac{1-\hat{\delta}_{in}}{5} \cdot r'_{out} \cdot \hat{r}''_{out} \cdot \hat{r}_0$.

5.3 Rate and relative distance of C'

We first show that C' has the desired rate and distance.

Claim 5.5. *The code C' is a binary linear code of block length at least n and rate $\left(1 - H\left(\frac{1}{2} - \xi\right)\right)(1-\gamma)$. Moreover, C' has relative distance at least $\frac{1}{2} - \xi$ with probability $1 - \exp(-n)$ over the choice of the T_i's.*

Proof. By construction C' is a binary linear code of block length $tn/\hat{r}_0 \geq n$ and rate

$$r_{in} \cdot r'_{out} \cdot \hat{r}''_{out} \cdot \hat{r}_0 = (1 - \delta_{in}) \cdot r'_{out} \cdot \hat{r}''_{out} \cdot \hat{r}_0 = \left(1 - H\left(\frac{1}{2} - \xi\right)\right)(1-\gamma).$$

To show that C' has the required distance we use Lemma 4.1. To see that the conditions of this lemma hold note first that the relative distance of C is at least

$$\delta_{in} - \frac{\gamma}{12} \cdot \xi = 1 - \frac{\left(1 - H\left(\frac{1}{2} - \xi\right)\right)(1-\gamma)}{r'_{out} \cdot \hat{r}''_{out} \cdot \hat{r}_0} - \frac{\gamma}{12} \cdot \xi \geq 1 - \frac{\gamma}{6} \cdot \xi,$$

where the inequality is by our choice of $\xi_0 \leq \frac{\gamma}{60} \cdot r'_{out} \cdot \hat{r}''_{out} \cdot \hat{r}_0$ in (1). Moreover, by choice of $\xi_0 \leq \gamma/c$ in (1) we have that $\xi \leq \gamma/c$. Thus Lemma 4.1 implies that C' has relative distance at least $\frac{1}{2} - \xi$ with probability at least $1 - \exp(-n)$. □

5.4 Local list decoding of C'

Next we show that C' can be locally list decoded from $1/4$ fraction of errors.

Claim 5.6. *The code C' is $\left(n^{\beta/2} \cdot \operatorname{polylog} n, \frac{1}{4}, \frac{1}{n}, n^{\beta/4} \cdot \operatorname{polylog} n\right)$-locally list decodable with probability $1 - \exp(-n)$ over the choice of the T_i's. Moreover, the local list decoder of C' can be implemented to run in time $\operatorname{poly}(n^\beta)$.*

Proof. By construction the code C is $\left(n^{\beta/2} \cdot \operatorname{polylog} n, \hat{\alpha}_{in} - \frac{\gamma}{12} \cdot \xi, \frac{1}{n}, \hat{L}_0, n^{\beta/4} \cdot \operatorname{polylog} n\right)$-locally list recoverable in time $\operatorname{poly}(n^\beta)$, where $\hat{\alpha}_{in} > \frac{1}{4\hat{\alpha}_0}$, and each T_i is $(\hat{\alpha}_0, \hat{L}_0)$-list decodable with probability at least $1 - \exp(-t) = 1 - o_n(1)$. The local list decoding algorithm A' for C' will run the local list recovery algorithm A for C and answer the queries of A by list decoding the T_i's corresponding to the queries of A. Details follow.

Let A be the algorithm that local list recovers C. On oracle access to $w \in \mathbb{F}_2^{tn/\hat{r}_0}$ the algorithm A' that local list decodes C' runs A and whenever A asks for some coordinate $i \in [n]$, the algorithm A' list decodes the i-th block of w of length $t/\hat{r}_0$ from $\hat{\alpha}_0$ fraction of errors, and feeds the messages corresponding to the first $\hat{L}_0$ codewords in the output list as an answer to the query of A. Let $A_1, \ldots, A_L$ be the resulting output algorithms of A for $L = n^{\beta/4} \cdot \operatorname{polylog} n$. Then A' outputs L algorithms $A'_1, \ldots, A'_L$ where each algorithm A'_j is defined as follows

To decode the k'-th coordinate in the k-th block of C' of length $t/\hat{r}_0$ (that is, a coordinate of the form $kt/\hat{r}_0 + k' \in [tn/\hat{r}_0]$ where $1 \le k \le n$ and $0 \le k' < t/\hat{r}_0$), the algorithm A'_j runs the algorithm A_j on input coordinate k. As above, whenever A_j asks for some coordinate $i \in [n]$, the algorithm A'_j list decodes the i-th block of w of length $t/\hat{r}_0$ from $\hat{\alpha}_0$ fraction of errors, and feeds the messages corresponding to the first $\hat{L}_0$ codewords in the output list as an answer to the query of A_j. Let $\sigma \in \mathbb{F}_{2^t}$ be the output symbol of A_j. Then the algorithm A'_j outputs the k'-th bit of $T_k(\sigma) \in \mathbb{F}_2^{t/\hat{r}_0}$.

The query complexity of A' is at most $n^{\beta/2} \cdot \operatorname{polylog} n \cdot t = n^{\beta/2} \cdot \operatorname{polylog} n$, and the output list size of A' is $n^{\beta/4} \operatorname{polylog} n$. Each block of w of length $t/\hat{r}_0$ can be brute-force list decoded in time $2^{t/\hat{r}_0} = \operatorname{poly}(n^\beta)$ and so the overall running time is $\operatorname{poly}(n^\beta)$. The soundness property clearly holds.

To see that the completeness property holds as well note that if $\operatorname{dist}(w, c') \le \frac{1}{4}$ for some $c' \in C'$, then by Markov's inequality for at most $1/(4\hat{\alpha}_0)$ fraction of $i \in [n]$ it holds that the i-th block of w of length $t/\hat{r}_0$ differs from the i-th block of c' of length $t/\hat{r}_0$ by more than $\hat{\alpha}_0$ fraction of the coordinates. Moreover, since each T_i is $(\hat{\alpha}_0, \hat{L}_0)$-list decodable with probability at least $1 - o_n(1)$, with probability at least $1 - \exp(-n)$ it holds that at most $\xi/2$ fraction of the T_i's do not have this property. This implies in turn that the list decoding of the T_i's fails on at most $\xi/2 + 1/(4\hat{\alpha}_0)$ fraction of the blocks. The completeness then follows since C is locally list recoverable from $\hat{\alpha}_{in} - \frac{\gamma}{12} \cdot \xi > \xi/2 + 1/(4\hat{\alpha}_0)$ fraction of errors (where the inequality holds by choice of $\xi_0 \le \hat{\alpha}_{in} - 1/(4\hat{\alpha}_0)$ in (1)) and input list size $\hat{L}_0$. □

5.5 Local correction of C'

Finally, we show that the code C' is locally correctable from half the GV bound.

Claim 5.7. *For any $\gamma' > 0$ the code C' is $\left(n^\beta \cdot \operatorname{poly}(1/\gamma'), \frac{1}{2} \cdot (\frac{1}{2} - \xi) - \gamma'\right)$-locally correctable with probability $1 - \exp(-n)$ over the choice of the T_i's. Moreover, the local corrector of C' can be implemented to run in time $\operatorname{poly}(n^\beta, 1/\gamma')$.*

Proof. By claims 5.5 and 5.6 we have that C' has relative distance at least $\frac{1}{2}-\xi$ and in addition it is $\left(n^{\beta/2}\cdot\operatorname{polylog} n,\frac{1}{4},\frac{1}{n},n^{\beta/4}\cdot\operatorname{polylog} n\right)$-locally list decodable in time $\operatorname{poly}(n^{\beta})$ with probability at least $1-\exp(-n)$ over the choice of the T_i's. In what follows assume that these two properties hold, we will show that in this case C' is also $\left(n^{\beta}\cdot\operatorname{poly}(1/\gamma'),\frac{1}{2}\cdot(\frac{1}{2}-\xi)-\gamma'\right)$-locally correctable in time $\operatorname{poly}(n^{\beta},1/\gamma')$ for any $\gamma'>0$.

Let A' be the algorithm that local list decodes C'. By increasing the query complexity of A' by a multiplicative factor of $\operatorname{polylog} n$ we may assume that both the completeness and soundness properties of A' hold with success probability $1-\frac{1}{n^2}$ instead of $\frac{2}{3}$. On oracle access to $w\in\mathbb{F}_2^{tn/\hat{r}_0}$ and input coordinate $i\in[tn/\hat{r}_0]$, the algorithm $\tilde{A}$ that local corrects C' first runs A' with oracle access to w, let $A'_1,\ldots,A'_L$ be the output algorithms for $L=n^{\beta/4}\operatorname{polylog} n$. The algorithm $\tilde{A}$ then runs each of the A'_j's on a random subset $S_j\subseteq[tn/\hat{r}_0]$ of coordinates of size $O(\log n/(\gamma')^2)$, and computes the fraction of coordinates δ_j in S_j on which the decoded values differ from the values of w. Finally, the algorithm $\tilde{A}$ finds some A'_j for which $\delta_j\leq\frac{1}{2}\cdot(\frac{1}{2}-\xi)$ (if such A'_j exists), and uses A'_j to decode the input coordinate i.

The query complexity of $\tilde{A}$ is

$$n^{\beta/4}\cdot\operatorname{polylog} n\cdot O(\log n/(\gamma')^2)\cdot n^{\beta/2}\cdot\operatorname{polylog} n\cdot O(\log n),$$

which is at most $n^{\beta}\cdot\operatorname{poly}(1/\gamma')$ for sufficiently large n, and the running time of $\tilde{A}$ is $\operatorname{poly}(n^{\beta},1/\gamma')$. Next we show that $\tilde{A}$ satisfies the required local correction guarantees.

Let $c'\in C'$ be the (unique) codeword which satisfies that $\operatorname{dist}(w,c')\leq\frac{1}{2}\cdot\left(\frac{1}{2}-\xi\right)-\gamma'$. We shall show below that with probability $1-o(1)$, there exists some A'_j that computes c' and satisfies that $\delta_j\leq\frac{1}{2}\cdot\left(\frac{1}{2}-\xi\right)$, and on the other hand, any A'_j which does not compute c' satisfies that $\delta_j>\frac{1}{2}\cdot\left(\frac{1}{2}-\xi\right)$. This will imply in turn that the algorithm $\tilde{A}$ will succeed in decoding the input coordinate with probability $1-o(1)\geq\frac{2}{3}$ as required.

We first show that with probability at least $1-\frac{3}{n}$ there exists some A'_j that computes c' and satisfies that $\delta_j\leq\frac{1}{2}\cdot\left(\frac{1}{2}-\xi\right)$. To see this note that by the completeness property of A' and since $\operatorname{dist}(w,c')\leq\frac{1}{4}$, with probability at least $1-\frac{1}{n}$ over the randomness of A' there exists some A'_j that computes c'. In this case, by union bound with probability at least $1-\frac{1}{n}$ it holds that each decoded coordinate of A'_j in S_j equals to the corresponding coordinate in c'. Furthermore, by Chernoff bound with probability at least $1-\frac{1}{n}$ it holds that w and c' differ on S_j by at most $\frac{1}{2}\cdot\left(\frac{1}{2}-\xi\right)$ fraction of the coordinates. Consequently, with probability at least $1-\frac{3}{n}$ it holds that $\delta_j\leq\frac{1}{2}\cdot\left(\frac{1}{2}-\xi\right)$.

Next we show that with probability at least $1-\frac{3}{n}$, any A'_j which does not compute c' satisfies that $\delta_j>\frac{1}{2}\cdot\left(\frac{1}{2}-\xi\right)$. For this note that by the soundness property of A', with probability at least $1-\frac{1}{n}$ over the randomness of A', for every such A'_j there exists a codeword $\tilde{c}\in C'\setminus\{c'\}$ such that A'_j computes $\tilde{c}$. As above, by union bound this implies in turn that with probability at least $1-\frac{1}{n}$ it holds that each decoded coordinate of A'_j in S_j equals to the corresponding coordinate on $\tilde{c}$. On the other hand, since C' has relative distance at least $\frac{1}{2}-\xi$ and $\operatorname{dist}(w,c')\leq\frac{1}{2}\cdot\left(\frac{1}{2}-\xi\right)-\gamma'$ we have that $\operatorname{dist}(w,\tilde{c})>\frac{1}{2}\cdot\left(\frac{1}{2}-\xi\right)+\gamma'$, and so by Chernoff bound with probability at least $1-\frac{1}{n}$ it holds that w and $\tilde{c}$ differ on S_j by more than $\frac{1}{2}\cdot\left(\frac{1}{2}-\xi\right)$ fraction of the coordinates. Consequently, with probability at least $1-\frac{3}{n}$ it holds that $\delta_j>\frac{1}{2}\cdot\left(\frac{1}{2}-\xi\right)$ for any such A'_j which completes the proof of the claim.

□

6 Local list recovery of Reed-Muller codes

In this section we prove Lemma 5.3 which we recall here.

Lemma 5.3 (Local list recovery of Reed-Muller codes)**.** *There exists an absolute constant c' such that for any $\alpha, \epsilon > 0$ and integers m, d, q, ℓ which satisfy $\alpha < 1 - c' \cdot \sqrt{\frac{\ell d}{q}}$ the Reed-Muller code $RM(m, d, q)$ is $\big(O(q^2 \cdot \log(q/\epsilon)), \alpha, \epsilon, \ell, O(q \log(1/\epsilon))\big)$-locally list recoverable. Moreover, the local list recovery algorithm can be implemented to run in $\mathrm{poly}(m, q, \log(1/\epsilon))$ time.*

For the proof of the above lemma we shall need the following two lemmas. The first lemma from [GS92] gives a local correction procedure for Reed-Muller codes (see. e.g., Proposition 2.6. in [Yek12]).

Lemma 6.1 (Local correction of Reed-Muller codes)**.** *There exists an absolute constant $r_0 > 0$ such that for any integers m, d, q which satisfy that $\frac{d}{q} \leq r_0$ the Reed-Muller code $RM(m, d, q)$ is $\big(q, \frac{1}{4}\big)$-locally correctable.*

In other words, there exists a randomized q-query algorithm Corr *such that given oracle access to a function $f : \mathbb{F}_q^m \to \mathbb{F}_q$ which agrees with a degree d polynomial $p : \mathbb{F}_q^m \to \mathbb{F}_q$ on at least $3/4$ fraction of inputs, and given $x \in \mathbb{F}_q^m$,*

$$\Pr[\mathrm{Corr}^f(x) = p(x)] \geq \frac{2}{3},$$

where the probability is over the internal randomness of Corr*. Moreover* Corr *runs in $\mathrm{poly}(m, q)$ time.*

The second lemma gives a tolerant local testing procedure for Reed-Muller codes. A tolerant local testing algorithm is a local testing algorithm that has the additional property of accepting all words which are sufficiently close to the code. Formally it is defined as follows.

Definition 6.2. Let $0 < \alpha < \beta < 1$. We say that a code $C \subseteq \Sigma^n$ is (q, α, β)-tolerant locally testable if there exists a randomized algorithm A that satisfies the following requirements:

- **Input:** A gets oracle access to a string $w \in \Sigma^n$.
- **Query complexity:** A makes at most q queries to the oracle w.
- **Completeness:** If $\mathrm{dist}(w, C) \leq \alpha$, then A accepts with probability at least $\frac{2}{3}$.
- **Soundness:** If $\mathrm{dist}(w, C) \geq \beta$, then A rejects with probability at least $\frac{2}{3}$.

Lemma 6.3 (Tolerant local testing of Reed-Muller codes)**.** *There exist absolute constants $r_0 > 0$ and $0 < \alpha_0 < 1/4$ such that for any integers m, d, q which satisfy that $\frac{d}{q} \leq r_0$ the Reed-Muller code $RM(m, d, q)$ is $(O(q), \alpha_0, 1/4)$-tolerant locally testable. Moreover the running time of the tester is $\mathrm{poly}(m, q)$.*

The proof of the above lemma is based on the robust local testing procedure for Reed-Muller codes from [FS95], and is deferred to Section 6.2.

6.1 Proof of Lemma 5.3

The proof follows the lines of the algorithm for the local list decoding of Reed-Muller codes from [STV01], we need an additional local testing procedure that guarantees the soundness requirement in our definition of locally list recoverable codes (Definition 2.10). Let $S : \mathbb{F}_q^m \to \binom{\mathbb{F}_q}{\ell}$. We would like to construct a list of oracle algorithms which compute all codewords $p : \mathbb{F}_q^m \to \mathbb{F}_q$ such that $p(x) \in S(x)$ for at least $\beta := 1 - \alpha$ fraction of $x \in \mathbb{F}_q^m$. Moreover every oracle algorithm in the list should compute some codeword. Now we describe an algorithm for this task. We will begin by defining the following (deterministic) sub-algorithm which will be used in the main algorithm.

The algorithm receives as parameters $\beta \in [0, 1]$, $z \in \mathbb{F}_q^m$ and $a \in \mathbb{F}_q$.

Algorithm 1 $\mathcal{M}^S_{z,a,\beta}(x)$

1: Let $\ell_{z,x}(t) = (1-t)z + tx$ denote the line through the points z, x.[6]
2: Find the list $h_1, \cdots, h_r$ that includes all univariate degree d polynomials $p : \mathbb{F}_q \to \mathbb{F}_q$ such that $p(t) \in S(\ell_{z,x}(t))$ for at least $\beta/2$ fraction of $t \in \mathbb{F}_q$.
3: If there exists a unique i such that $h_i(0) = a$, then output $h_i(1)$, else output 'FAIL'.

The parameters z, a in Algorithm 1 must be thought of as advice which tells us that the polynomial takes the value $a \in \mathbb{F}$ at the point $z \in \mathbb{F}_q^m$. The following claim makes this intuition precise.

Claim 6.4. *Let $0 < \tau < 1$, $1 \geq \beta \geq \frac{16}{\tau}\sqrt{\ell d/q}$, and let $p : \mathbb{F}_q^m \to \mathbb{F}_q$ be a degree d polynomial which agrees with S in at least β fraction of inputs, then the following are true.*

1. *$\mathcal{M}^S_{z,a,\beta}$ makes at most q queries to S and runs in* $\mathrm{poly}(m, q)$ *time.*
2. $\Pr_z\left[\Pr_x\left[\mathcal{M}^S_{z,p(z),\beta} \text{ computes } p \text{ at } x\right] \geq 1 - \tau\right] \geq \frac{1}{2}$.

Proof. The number of queries is q since the algorithm only queries points on a line. Also Step 2 of the algorithm, which is the most expensive step, can be implemented in $\mathrm{poly}(m, q)$ time by Theorem 2.2. Now we will prove (2). By Markov inequality,

$$\Pr_z\left[\Pr_x\left[\mathcal{M}^S_{z,p(z),\beta} \text{ does not compute } p \text{ at } x\right] \geq \tau\right] \leq \frac{1}{\tau}\Pr_{z,x}\left[\mathcal{M}^S_{z,p(z),\beta} \text{ does not compute } p \text{ at } x\right].$$

To bound the probability that $\mathcal{M}^S_{z,p(z),\beta}$ does not compute p at x, let us define the following two bad events and bound their probabilities.

Event A: $\nexists i \in [r]$ *s.t.* $h_i = p|_{\ell_{z,x}}$

This will happen only if p does not agree with S on at least $\beta/2$ fraction of points on the line $\ell_{z,x}$. But we know that p has agreement at least β with S on the entire space. Since the set of points on the line are pairwise independent, we can use Chebychev's inequality to bound the probability of this event.

$$\Pr[A] = \Pr[\nexists i \in [r] \text{ s.t. } h_i = p|_{\ell_{z,x}}] \leq \frac{4}{\beta q} \leq \frac{\tau}{4\sqrt{\ell d q}} \leq \frac{\tau}{4}$$

Event B: $\exists i \in [r]$ *s.t.* $h_i \neq p|_{\ell_{z,x}}$ *and* $h_i(0) = p(z)$

Since the list of polynomials $h_1, \cdots, h_r$ depends only on the line through z, x, we can think of the

[6] If $z = x$, choose a random line through z.

random process as first picking a random line ℓ and then picking two random points $t_1, t_2 \in \mathbb{F}_q$ and letting $z = \ell(t_1), x = \ell(t_2)$. If $h_i \neq p|_\ell$, then they agree on at most d points of ℓ, so $\Pr_{t_1}[h_i(t_1) = p(\ell(t_1))] \leq \frac{d}{q}$. By union bound,

$$\Pr[B] = \Pr\left[\exists i \in [r] \text{ s.t. } h_i \neq p|_{\ell_{z,x}} \text{ and } h_i(0) = p(z)\right] \leq \frac{rd}{q}.$$

By applying the Johnson bound for list recovery (Lemma 5.2) to the Reed-Solomon code of degree d on the line ℓ ,

$$r \leq \frac{\ell}{(\beta/2)^2 - \ell d/q} \leq \frac{q/d}{(8/\tau)^2 - 1}.$$

Combining the above bounds we get, $\Pr[B] \leq 1/((8/\tau)^2 - 1) \leq \tau/8$.

Clearly, if events A, B do not happen, then $\mathcal{M}^S_{z,p(z),\beta}$ will compute p at x. Therefore

$$\Pr_{z,x}\left[\mathcal{M}^S_{z,p(z),\beta} \text{ does not compute } p \text{ at } x\right] \leq \Pr[A] + \Pr[B] \leq \frac{\tau}{2}.$$

Therefore,

$$\Pr_z\left[\Pr_x\left[\mathcal{M}^S_{z,p(z),\beta} \text{ does not compute } p \text{ at } x\right] \geq \tau\right] \leq \frac{1}{2}.$$

□

Algorithm 2 Local list recovery algorithm $\mathcal{R}(S, \beta)$

1: Sample $z_1, \cdots, z_t \in \mathbb{F}_q^m$ uniformly at random where $t = \log(2/\epsilon)$.
2: Let $\mathcal{L}$ be the list of all oracle algorithms $\mathcal{M}^S_{z_i,a,\beta}$ for $i \in [t]$ and $a \in \mathbb{F}$.
3: Run the tolerant local tester $\mathcal{T}$ from Lemma 6.3 on each algorithm in $\mathcal{L}$ for $t' = 100\log(2qt/\epsilon)$ times and remove from $\mathcal{L}$ any algorithm which fails a majority of the tests.
4: For every $\mathcal{M} \in \mathcal{L}$, include the oracle algorithm $\text{Corr}^{\mathcal{M}}$ in the output list where Corr is the corrector from Lemma 6.1.

The following claim essentially proves Lemma 5.3.

Claim 6.5. *Let $1 \geq \beta > c'\sqrt{\ell d/q}$ where c' is some sufficiently large absolute constant. Let $\mathcal{L}_{out}$ be the list of oracle algorithms output by $\mathcal{R}(S, \beta)$. Then the following statements are true.*

1. *The size of the list $|\mathcal{L}_{out}| = O(q\log(1/\epsilon))$.*
2. *The algorithm $\mathcal{R}(S, \beta)$ makes at most $O(q^2\log(q/\epsilon))$ queries to oracle S and runs in $\text{poly}(m, q, \log(1/\epsilon))$ time.*
3. *Each algorithm in $\mathcal{L}_{out}$ makes at most q^2 queries to S and runs in $\text{poly}(m, q)$ time.*
4. *Let $p : \mathbb{F}_q^m \to \mathbb{F}_q$ be a degree d polynomial which agrees with S on at least β fraction of inputs, then with probability at least $1 - \epsilon$, there exists $\mathcal{A} \in \mathcal{L}_{out}$ which computes p.*
5. *With probability at least $1 - \epsilon$, every $\mathcal{A} \in \mathcal{L}_{out}$ computes some degree d polynomial.*

Proof. (1) is trivially true since the list only gets smaller after Step 2. To prove (2), note that $\mathcal{R}$ makes queries to S only in Step 3. By Lemma 6.3, the tester $\mathcal{T}$ makes $O(q)$ queries to each algorithm $\mathcal{M}^S_{z_i,a,\beta} \in \mathcal{L}$, and each algorithm $\mathcal{M}^S_{z_i,a,\beta}$ makes at most q queries to S as in Algorithm 1. Since the test is repeated $t' = O(\log(q/\epsilon))$ times, the total queries to S is $O(q^2 \log(q/\epsilon))$. To analyze the running time, note that the tester $\mathcal{T}$ and the algorithms $\mathcal{M}^S_{z_i,a,\beta}$ run in $\text{poly}(m, q)$ time, so the total running time is $\text{poly}(m, q, \log(1/\epsilon))$.

To prove (3), note that every algorithm in $\mathcal{L}_{out}$ looks like $\text{Corr}^{\mathcal{M}}$ for some $\mathcal{M}$ constructed in Step 2. By Lemma 6.1, Corr makes q queries to $\mathcal{M}$, and each $\mathcal{M}$ makes q queries to S as in Algorithm 1. Thus the total queries $\text{Corr}^{\mathcal{M}}$ makes to S on any input is at most q^2. Also both Corr and $\mathcal{M}$ run in $\text{poly}(m, q)$ time, thus $\text{Corr}^{\mathcal{M}}$ also takes $\text{poly}(m, q)$ time.

To prove (4), observe that $\mathcal{M}^S_{z_1,p(z_1)}, \cdots, \mathcal{M}^S_{z_t,p(z_t)}$ are in the list $\mathcal{L}$. Let $0 < \alpha_0 < \frac{1}{4}$ be the constant that appears in Lemma 6.3 and let $c' > \frac{16}{\alpha_0}$. By Claim 6.4, with probability $\geq 1 - 1/2^t = 1 - \epsilon/2$, at least one of these algorithms agree with p on $\geq 1 - \alpha_0$ fraction of inputs, call this algorithm $\mathcal{M}$. Therefore $\mathcal{M}$ will also pass the local testing in Step 3 with probability $1 - \epsilon/2$ by Lemma 6.3 and Chernoff bound. Since $\alpha_0 < \frac{1}{4}$, by Lemma 6.1, $\text{Corr}^{\mathcal{M}}$ will compute p everywhere.

Finally to prove (5), by Lemma 6.3 and Chernoff bound, any $\mathcal{A} \in \mathcal{L}$ that is 1/4 far from any degree d polynomial will remain in the list after Step 3 with probability at most $\frac{\epsilon}{tq}$. By union bound over each algorithm in the list which is of size at most tq, with probability at least $1 - \epsilon$, every algorithm $\mathcal{A}$ in $\mathcal{L}$ that remains after Step 3, will be $\frac{1}{4}$ close to some degree d polynomial p'. So by Lemma 6.1, $\text{Corr}^{\mathcal{A}}$ will compute p' everywhere. Therefore every algorithm in $\mathcal{L}_{out}$ computes some degree d polynomial.

□

6.2 Tolerant local testing of Reed-Muller codes - Proof of Lemma 6.3

For the proof of lemma 6.3 we shall use the following lemma from [FS95, Theorem 7] which gives a robust local testing procedure for Reed-Muller code. A robust local testing algorithm is a local testing algorithm such that its local view on words far from the code is far on average from an accepting view.

Lemma 6.6 (Robust local testing of Reed-Muller codes)**.** *There exists an absolute constant $r_0 > 0$ such that the following holds for any $\alpha > 0$ and integers m, d, q which satisfy that $\frac{d}{q} \leq r_0$. Suppose that $f : \mathbb{F}_q^m \to \mathbb{F}_q$ satisfies that $\text{dist}\,(f, RM(m, d, q)) \geq \alpha$. Then the expected relative distance of f from $RS_q(d, q)$ on a random line is at least $\frac{\alpha}{9}$.*

Proof of Lemma 6.3. Say we are given a function $f : \mathbb{F}_q^m \to \mathbb{F}_q$ and we need to test if it is close to a degree d polynomial. Let $0 < \tau < 1 - \sqrt{d/q}$ be some threshold parameter to be chosen later. The test is to choose a random line ℓ in $\mathbb{F}_q^m$ and find if there is a univariate degree d polynomial which is τ-close to $f|_\ell$. If yes, then accept, else reject. Clearly this test makes only q queries. Also by Theorem 2.2, when $\tau < 1 - \sqrt{d/q}$, this can be implemented in $\text{poly}(m, q)$ time. Now we will show that for an appropriate choice of τ, this is a $(O(q), \alpha_0, 1/4)$ tolerant test for some $\alpha_0 > 0$.

Let $f : \mathbb{F}_q^m \to \mathbb{F}_q$ be some function which is α_0-close to a degree d polynomial p. Since points on a random line are uniform over $\mathbb{F}_q^m$, by Markov inequality, the probability that $f|_\ell$ is τ-far from any univariate degree d polynomial is at most α_0/τ. So the probability that the test rejects f is at most $\beta_0 = \alpha_0/\tau$.

Let $g : \mathbb{F}_q^m \to \mathbb{F}_q$ be some function which is 1/4-far from any degree d polynomial. Then by Lemma 6.6, the expected distance of $g|_\ell$ to $RS_q(d, q)$ is at least 1/36. The probability that $g|_\ell$ is

τ-far from $p|_\ell$ is at least $\beta_1 = \frac{1/36-\tau}{1-\tau}$. When d/q is sufficiently small, we can choose α_0 and τ to be some absolute constants such that $0 < \tau < 1 - \sqrt{d/q}$ and $\beta_0 < \beta_1$.

Finally to get the acceptance and rejection probabilities to $2/3$ as in the definition of tolerant locally testable codes, we repeat the above t times and accept a function if it is accepted in at least $\frac{\beta_0+\beta_1}{2}$ fraction of the tests. When t is large enough (but still some absolute constant), by Chernoff bound, the new test will have the required soundness and completeness.

□

7 Distance amplification for local list recovery

In this section we prove Lemma 5.4 which we recall here.

Lemma 5.4 (Distance amplification for local list recovery)**.** *For any constants $\delta_{out}, \alpha_{out}, \gamma > 0$ there exists an integer $d = d(\delta_{out}, \alpha_{out}, \gamma) \leq poly(1/\delta_{out}, 1/\alpha_{out}, 1/\gamma)$ such that the following holds.*

- *C_{out} be an $\mathbb{F}$-linear code of block length n_{out}, alphabet size σ_{out}, rate r_{out}, and relative distance δ_{out} that is $(q, \alpha_{out}, \epsilon, \ell_{out}, L_{out})$-locally list recoverable.*
- *C_{in} be an $\mathbb{F}$-linear code of block length n_{in}, alphabet size σ_{in}, rate r_{in}, and relative distance δ_{in} that is $(\alpha_{in}, \ell_{in}, L_{in})$-(globally) list recoverable.*
- *additionally, suppose that $n_{in} \geq d$, $\sigma_{out} = \sigma_{in}^{r_{in} \cdot n_{in}}$ and $L_{in} \leq \ell_{out}$.*

Then there exists an $\mathbb{F}$-linear code C of block length n_{out}, alphabet size $\sigma_{in}^{n_{in}}$, rate $r_{in} \cdot r_{out}$ and relative distance at least $\delta_{in} - 2\gamma$ that is $(O(q \cdot n_{in}^2 \cdot \log(n_{in})), \alpha_{in} - \gamma, \epsilon, \ell_{in}, L_{out})$-locally list recoverable.

Moreover, if the running time of the local list recovery algorithm for C_{out} is T_{out} and the running time of the global list recovery algorithm for C_{in} is T_{in} then the running time of the local list recovery algorithm for C is

$$O(T_{out}) + O(q \cdot T_{in}) + \text{poly}(q, n_{in}, \ell_{in}).$$

Proof. The construction and analysis closely follow that of the high rate locally correctable codes from [KMRS16].

One important ingredient in the construction will be a family of bipartite expanders which additionally have the property of being good *samplers*. We define samplers below and state a lemma (very closely related to that from [KMRS16]) showing the existence of the kind of samplers we will need.

For a graph G, a vertex s and a set of vertices T, let $E(s, T)$ denote the set of edges that go from s into T. Roughly speaking, a sampler is a bipartite d-regular graph in which the density of any subset T of right vertices can be approximated by the value of $E(s, T)/d$ for a uniform random left vertex s.

Definition 7.1. Let $G = (U \cup V, E)$ be a bipartite d-regular graph with $|U| = |V| = n$. We say that G is an (α, γ)-sampler if the following holds for every $T \subseteq V$: For at least $1 - \alpha$ fraction of the vertices $s \in U$ it holds that

$$\frac{|E(s, T)|}{d} - \frac{|T|}{n} \leq \gamma.$$

Lemma 7.2. *For every $\alpha, \gamma > 0$, there exists $\hat{d} = \mathrm{poly}(\frac{1}{\alpha\gamma})$ such that for every sufficiently large n and for every $d > \hat{d}$ there exists a bipartite d-regular graph $G_{n,d,\alpha,\gamma} = (U \cup V, E)$ with $|U| = |V| = n$ such that $G_{n,d,\alpha,\gamma}$ is an (α, γ)-sampler. Furthermore, there exists an algorithm that takes as inputs n, d, α, γ and a vertex w of $G_{n,d,\alpha,\gamma}$, and computes the list of the neighbors of w in $G_{n,d,\alpha,\gamma}$ in time $\mathrm{poly}(\frac{\log n \cdot d}{\alpha \cdot \gamma})$.*

The proof follows the outline presented in [KMRS15a, Section 2.4] and we omit the details here. The only difference from [KMRS15a] is that here we require the degree to be any $d > \hat{d}$, whereas in [KMRS15a] it was constructed for specific $d = \mathrm{poly}\left(\frac{1}{\alpha \cdot \gamma}\right)$. However it is easy to see that the proof can be modified to make it work for larger degrees as well, by first constructing an $(\alpha, \gamma/2)$ sampler for a pretty large degree and then adding matchings to the graph to get the required degree and not hurting the sampling property too much.

We now describe the construction of the code C using these samplers.

The code construction We construct the code C by giving a bijection from C_{out} to C. Let $\Sigma_{out}, \Sigma_{in}$ denote the alphabets of C_{out}, C_{in} respectively. Given a codeword $c_{out} \in C_{out}$, one obtains the corresponding codeword $c \in C$ as follows:

- View each codeword symbol in Σ_{out} as a vector of length $r_{in} \cdot n_{in}$ over Σ_{in} and encode it via the code C_{in}. Each codeword symbol gets mapped to a string in $\Sigma_{in}^{n_{in}}$. We denote the resulting string by $c' \in \Sigma_{in}^{n_{in} \cdot n_{out}}$ and the various resulting codewords of C_{in} by $B_1, B_2, \ldots B_{n_{out}} \in \Sigma_{in}^{n_{in}}$.
- Next, we apply a "pseudorandom" permutation to the coordinates of c' as follows: Let $G_{n_{out}}$ be a graph from the infinite family of n_{in}-regular $(\min\{\alpha_{out}, \delta_{out}/2\}, \gamma)$ samplers above and let $U = \{u_1, \ldots, u_{n_{out}}\}$ and $V = \{v_1, \ldots, v_{n_{out}}\}$ be the left and right vertices of $G_{n_{out}}$ respectively. For each $i \in [n_{out}]$ and $j \in [n_{in}]$, we write the j-th symbol of B_i on the j-th edge of u_i. Then, we construct new blocks $D_1, \ldots, D_{n_{out}} \in \Sigma_{in}^{n_{in}}$, by setting the j-th symbol of D_i to be the symbol written on the j-th edge of v_i. We reinterpret each of these blocks to be a symbol of the alphabet $\Sigma \stackrel{\mathrm{def}}{=} \Sigma_{in}^{n_{in}}$.
- Finally, we define the codeword c of $C \subseteq \Sigma^{n_{out}}$ as follows: the i-th coordinate c_i is the block D_i, reinterpreted as a symbol of the alphabet Σ. We choose c to be the codeword in C that corresponds to the codeword c_{out} in C_{out}.

This completes the definition of the bijection. It follows immediately that C is an $\mathbb{F}$-linear code of blocklength n_{out} and alphabet size $\sigma_{in}^{n_{in}}$. The rate of C is

$$\begin{aligned}
\frac{\log |C|}{n_{out} \cdot \log |\Sigma|} &= \frac{\log |C_{out}|}{n_{out} \cdot n_{in} \cdot \log |\Sigma_{in}|} \\
&= \frac{r_{out} \cdot n_{out} \cdot \log |\Sigma_{out}|}{n_{out} \cdot n_{in} \cdot \log |\Sigma_{in}|} \\
&= \frac{r_{out} \cdot \log |\Sigma_{out}|}{n_{in} \cdot \log |\Sigma_{in}|} \\
&= \frac{r_{out} \cdot r_{in} \cdot n_{in} \cdot \log |\Sigma_{in}|}{n_{in} \cdot \log |\Sigma_{in}|} \\
&= r_{out} \cdot r_{in}.
\end{aligned}$$

It remains to show that the relative distance of C is at least $\delta_{in} - 2\gamma$ and that C is $(O(q \cdot n_{in}^2 \cdot \log(n_{in})), \alpha_{in} - \gamma, \epsilon, \ell_{in}, L_{out})$-locally list recoverable.

Once we prove the portion of the theorem that shows that C is $(O(q \cdot n_{in}^2 \cdot \log(n_{in})), \alpha_{in} - \gamma, \epsilon, \ell_{in}, L_{out})$-locally list recoverable, it will follow almost in a blackbox manner that the relative distance of C is at least $\delta_{in} - 2\gamma$ for the following reason. Notice that it will suffice to show that C can be uniquely decoded from $\frac{\delta_{in}}{2} - \gamma$ fraction of errors. Since C_{in} has relative distance at least δ_{in}, C_{in} can be uniquely decoded from $\frac{\delta_{in}}{2}$ fraction of errors and in other words C_{in} is $(\delta_{in}/2, 1, 1)$-(globally) list recoverable. Also C_{out} can be uniquely decoded from $\frac{\delta_{out}}{2}$ fraction of errors and is hence trivially $(n_{out}, \delta_{out}/2, 0, 1, 1)$-locally list recoverable.

Thus by the same construction (i.e same choice of samplers), the code C is $(O(n_{out} \cdot n_{in}^2 \cdot \log(n_{in})), \delta_{in}/2 - \gamma, 0, 1, 1)$-locally list recoverable. In other words C is uniquely decodable from $\delta_{in}/2 - \gamma$ fraction of errors and hence has relative distance at least $\delta_{in} - 2\gamma$.

We now prove that C is $(O(q \cdot n_{in}^2 \cdot \log(n_{in})), \alpha_{in} - \gamma, \epsilon, \ell_{in}, L_{out})$-locally list recoverable.

Local list recoverability We will now describe the $(O(q \cdot n_{in}^2 \cdot \log(n_{in})), \alpha_{in} - \gamma, \epsilon, \ell_{in}, L_{out})$-local list recovery algorithm A for the code C. This is based on the following algorithm $\tilde{A}$ which locally list recovers coordinates of C_{out} (instead of coordinates of C, as required of A).

Lemma 7.3. *There exists a randomized algorithm $\tilde{A}$ that on oracle access to a string $S \in \binom{\Sigma}{\ell_{in}}^{n_{out}}$ makes at most $O(q \cdot n_{in} \cdot \log(n_{in}))$ queries to S and outputs a list of L_{out} randomized algorithms $\tilde{A}_1$, ..., $\tilde{A}_{L_{out}}$ which satisfy the following:*

- *Each $\tilde{A}_j$ takes as input coordinate $i \in [n_{out}]$ and also gets oracle access to the string S. $\tilde{A}_j$ makes at most $O(q \cdot n_{in} \cdot \log(n_{in}))$ queries to S and outputs a symbol $\tilde{A}_j^S(i) \in \Sigma_{out}$.*
- *(Completeness) For each $c_{out} \in C_{out}$ such that the corresponding codeword c of C (as given by the bijection above) satisfies $\mathrm{dist}(c, S) \leq \alpha_{in} - \gamma$, with probability at least $1 - \epsilon$ over the randomness of $\tilde{A}$, there exists some $j \in [L_{out}]$ such that $\Pr_{\tilde{A}_j}\left[\tilde{A}_j^S(i) = c_{out_i}\right] \geq 1 - \frac{1}{3n_{in}}$ for all $i \in [n]$.*
- *(Soundness) With probability at least $1 - \epsilon$ over the randomness of $\tilde{A}$, for every $j \in [L_{out}]$, there exists some $c_{out} \in C_{out}$ such that $\Pr_{\tilde{A}_j}\left[\tilde{A}_j^S(i) = c_{out_i}\right] \geq 1 - \frac{1}{3n_{in}}$ for all $i \in [n]$.*

We first show how to construct the required algorithm A, given such an algorithm $\tilde{A}$ guaranteed by Lemma 7.3. The algorithm A is given oracle access to a string $S \in \binom{\Sigma}{\ell_{in}}^{n_{out}}$, and needs to locally list recover all codewords $c \in C$ that "disagree" with S in at most $\alpha_{in} - \gamma$ fraction of coordinates. A outputs a list of L_{out} randomized algorithms A_1, ..., $A_{L_{out}}$ which work as follows.

Each A_j takes as input a coordinate $i \in [n_{out}]$ and also gets oracle access to the string S. Note that by the above lemma, with probability at least $1 - \epsilon$ over the randomness of $\tilde{A}$, for each $\tilde{A}_j$ there exists some $c_{out} \in C_{out}$ such that $\Pr_{\tilde{A}_j}\left[\tilde{A}_j^S(i) = c_{out_i}\right] \geq 1 - \frac{1}{3n_{in}}$ for all $i \in [n]$. Let the corresponding codeword in C be c. We will use $\tilde{A}_j$ to design A_j that will output the coordinates of c. Let $B_1, \ldots, B_{n_{out}}$ and $D_1, \ldots, D_{n_{out}}$ be the corresponding blocks that arise in the construction of c from c_{out}. In order for $A_j(i)$ to be able to decode the value of c_i, it should be able to correctly decode all the symbols in the block D_i. Let $u_{i_1}, \ldots, u_{i_{n_{in}}}$ be the neighbors of v_i in the graph $G_{n_{out}}$. Each symbol of D_i belongs to one of the blocks $B_{i_1}, \ldots, B_{i_{n_{in}}}$, and therefore it suffices to retrieve these blocks. Each of these blocks B_{i_j} is the encoding of $c_{out_{i_j}}$ (the i_jth symbol of c_{out}) via the

code C_{in}. Thus to recover $B_{i_1}, \ldots, B_{i_{n_{in}}}$, it suffices to recover $c_{out_{i_1}}, \ldots, c_{out_{i_{n_{in}}}}$. The algorithm A_j invokes the algorithm $\tilde{A}_j$ to recover each of $c_{out_{i_1}}, \ldots, c_{out_{i_{n_{in}}}}$, and by the union bound, it recovers all of them correctly with probability at least $1 - n_{in} \cdot \frac{1}{3n_{in}} = 2/3$. Whenever this happens, the algorithm A_j correctly retrieves the blocks $B_{i_1}, \ldots, B_{i_{n_{in}}}$ and hence also D_i and hence c_i.

Clearly the query complexity of A_j is n_{in} times the query complexity of $\tilde{A}_j$, and is hence at most $O(q \cdot n_{in}^2 \cdot \log(n_{in}))$. The completeness and soundness of A follow almost immediately from the completeness and soundness of $\tilde{A}$.

It remains to prove Lemma 7.3.

Proof of Lemma 7.3. Let $\bar{A}$ be the local list recovery algorithm for C_{out}. $\bar{A}$ is a randomized algorithm that on oracle access to a string $\overline{S} \in \binom{\Sigma_{out}}{\ell_{out}}^{n_{out}}$ outputs a list of L_{out} randomized algorithms $\bar{A}_1, \bar{A}_2, \ldots, \bar{A}_{L_{out}}$. By amplification we may assume that for each $i \in [n_{out}]$, $\bar{A}_j(i)$ errs with probability at most $\frac{1}{3 \cdot n_{in}}$, and this incurs a factor of at most $O(\log(n_{in}))$ to its query complexity. Thus the query complexity is at most $O(q \cdot \log(n_{in}))$.

We now describe $\tilde{A}$. Suppose the algorithm $\tilde{A}$ is invoked on a string $S = (S_1, \ldots, S_{n_{out}}) \in \binom{\Sigma}{\ell_{out}}^{n_{out}}$, the algorithm $\tilde{A}$ invokes the algorithm $\bar{A}$ and emulates $\bar{A}$ in the natural way. Recall that $\bar{A}$ expects to be given a string $\bar{S} \in \binom{\Sigma_{out}}{\ell_{out}}^{n_{out}}$. On input coordinate i, $\bar{A}$ makes queries to this sequence and outputs a value $\bar{A}(i)$. For any $k \in [n_{out}]$, whenever $\bar{A}$ queries the kth element of the sequence $\bar{S}_1, \ldots, \bar{S}_{n_{out}} \in \binom{\Sigma_{out}}{\ell_{out}}$, the algorithm $\tilde{A}$ performs the following steps.

1. In the first step, for each coordinate $r \in [n_{in}]$ of B_k, $\tilde{A}$ will find a list $S^{(k,r)} \in \binom{\Sigma_{in}}{\ell_{in}}$ and associate that list with the rth coordinate of B_k. The list $S^{(k,r)}$ is defined as follows: Suppose that v_{k_r} is the rth neighbor of the vertex u_k in $G_{n_{out}}$. Suppose that u_k is the $\hat{r}$th neighbor of the vertex v_{k_r}. Then in the construction of the codeword c from c_{out}, the value of the rth coordinate of B_k is stored in the $\hat{r}$th coordinate of D_{k_r}. Now $S_{k_r} \in \binom{\Sigma}{\ell_{in}} = \binom{\Sigma_{in}^{n_{in}}}{\ell_{in}}$ is the input list associated with the k_rth coordinate. Note that each element $s \in S_{k_r}$ can be viewed as an n_{in}-tuple of elements from Σ_{in}. Let the $\hat{r}$th element of this tuple be $s^{(\hat{r})}$. Then $S^{(k,r)}$ is defined to be the set in $\binom{\Sigma_{in}}{\ell_{in}}$ obtained by taking the $\hat{r}$th element of each member of the set S_{k_r}. $\tilde{A}$ can find this set by making a single query to the k_rth element of S to obtain S_{k_r}, and from it find $S^{(k,r)}$.

2. $\tilde{A}$ then invokes the global-list recovery algorithm for C_{in} with the lists $S^{(k,r)}$ for each $r \in [n_{in}]$. The output of this algorithm is a list of size at most L_{in} with elements from $\Sigma_{in}^{n_{in}}$. We denote by $\bar{S}_k$ the set of messages in $\Sigma_{in}^{r_{in} n_{in}} = \Sigma_{out}$ corresponding to the codewords in this list. This is what $\tilde{A}$ feeds to $\bar{A}$.

It is not hard to see that the query complexity of $\tilde{A}$ is at most n_{in} times the query complexity of $\bar{A}$, and hence it is at most $O(q \cdot n_{in} \cdot \log(n_{in}))$. It remains to show that $\tilde{A}$ satisfies the completeness and soundness requirements. We first show the completeness.

Completeness: Let $c_{out} \in C_{out}$ be such that the corresponding codeword c of C (as given by the bijection above) satisfies $\mathrm{dist}(c, S) \leq \alpha_{in} - \gamma$.

Claim 7.4. *The string $\bar{S} := (\bar{S}_1, \bar{S}_2, \ldots, \bar{S}_{n_{out}})$ as defined above satisfies $\mathrm{dist}(c_{out}, \bar{S}) \leq \alpha_{out}$.*

Once we have the claim, the completeness of $\tilde{A}$ will follow immediately from the completeness of $\bar{A}$. We now prove the claim.

Proof of Claim 7.4. Let $T = \{k \in [n_{out}] \mid c_k \notin S_k\}$. Then we know that $|T| \leq (\alpha_{in} - \gamma)n_{out}$. Let G be the set of all $i \in [n_{out}]$ such that in the graph $G_{n_{out}}$, u_i has at most α_{in} fraction of its neighbors v_j with $j \in T$. By the sampling property of $G_{n_{out}}$, it holds that $|G| \geq (1 - \alpha_{out}) \cdot n_{out}$.

We will now show that for all $k \in G$, $c_{out_k} \in \bar{S}_k$. Since $|G| \geq (1 - \alpha_{out}) \cdot n_{out}$, this shows that $\mathrm{dist}(c_{out}, \bar{S}) \leq \alpha_{out}$ and thus proves the claim.

Let $k \in G$. For each $r \in [n_{in}]$, let $S^{(k,r)} \in \binom{\Sigma_{in}}{\ell_{in}}$ be the set assigned to the rth coordinate of B_k (as described above). To show that $c_{out_k} \in \bar{S}_k$, it suffices to show that the encoding of c_{out_k} via the code C_{in} (which we call B_k) agrees with various $S^{(k,r)}$ for at least $1 - \alpha_{in}$ fraction of coordinates $r \in [n_{in}]$, since then the global list recovery algorithm of C_{in} succeeds in outputting c_{out_k}.

Now let r be any coordinate such that the rth neighbor of u_k in $G_{n_{out}}$ is a vertex v_{k_r} where $k_r \notin T$. Thus $c_{k_r} \in S_{k_r}$. Hence, by the definition of $S^{(k,r)}$, it holds that the rth coordinate of B_k agrees with $S^{(k,r)}$. Since at most α_{in} fraction of the r's could have been such that $k_r \in T$, thus for at least $1 - \alpha_{in}$ fraction of coordinates $r \in [n_{in}]$, B_k agrees with $S^{(k,r)}$, and hence $c_{out_k} \in \bar{S}_k$.

□

Soundness: The soundness of $\tilde{A}$ follows immediately from the soundness of $\bar{A}$.

□

It can be verified that the local list recovery algorithms $\tilde{A}$ and A can be implemented efficiently as required by the "moreover" part of the lemma.

□

References

[AEL95] Noga Alon, Jeff Edmonds, and Michael Luby. Linear time erasure codes with nearly optimal recovery. In *proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 512–519. IEEE Computer Society, 1995.

[AS03] Sanjeev Arora and Madhu Sudan. Improved low-degree testing and its applications. *Combinatorica*, 23(3):365–426, 2003.

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 21–31. ACM Press, 1991.

[BSGK+10] Eli Ben-Sasson, Venkatesan Guruswami, Tali Kaufman, Madhu Sudan, and Michael Viderman. Locally testable codes require redundant testers. *SIAM J. Comput*, 39(7):3230–3247, 2010.

[FS95] Katalin Friedl and Madhu Sudan. Some improvements to total degree tests. In *proceedings of the 3rd Israel Symposium on the Theory of Computing and Systems (ISTCS)*, pages 190–198. IEEE Computer Society, 1995.

[GI04] Venkatesan Guruswami and Piotr Indyk. Efficiently decodable codes meeting gilbert-varshamov bound for low rates. In J. Ian Munro, editor, *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004*, pages 756–757. SIAM, 2004.

[Gil52] Edgar N. Gilbert. A comparision of signalling alphabets. *Bell System Technical Journal*, 31:504–522, 1952.

[GR10] Venkatesan Guruswami and Atri Rudra. The existence of concatenated codes list-decodable up to the hamming bound. *IEEE Trans. Information Theory*, 56(10):5195–5206, 2010.

[GS92] Peter Gemmell and Madhu Sudan. Highly resilient correctors for polynomials. *Information Processing Letters*, 43(4):169–174, 28 September 1992.

[GS99] Venkatesan Guruswami and Madhu Sudan. Improved decoding of reed-solomon and algebraic-geometry codes. *IEEE Trans. Information Theory*, 45(6):1757–1767, 1999.

[GS00] Venkatesan Guruswami and Madhu Sudan. List decoding algorithms for certain concatenated codes. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 181–190. ACM, 2000.

[GS01] Venkatesan Guruswami and Madhu Sudan. Extensions to the johnson bound. 2001.

[GS02] Venkatesan Guruswami and Madhu Sudan. Decoding concatenated codes using soft information. In *IEEE Conference on Computational Complexity*, pages 148–157. IEEE Computer Society, 2002.

[GS06] Oded Goldreich and Madhu Sudan. Locally testable codes and PCPs of almost linear length. *Journal of ACM*, 53(4):558–655, 2006.

[Gur06] Venkatesan Guruswami. Algorithmic results in list decoding. *Foundations and Trends in Theoretical Computer Science*, 2(2), 2006.

[KMRS15a] Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally-correctable and locally-testable codes with sub-polynomial query complexity. *Electronic Colloquium on Computational Complexity (ECCC)*, 2015.

[KMRS15b] Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally-testable codes with quasi-polylogarithmic query complexity. *Electronic Colloquium on Computational Complexity (ECCC)*, 2015.

[KMRS16] Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High rate locally-correctable and locally-testable codes with sub-polynomial query complexity. In *proceedings of the 48th Annual Symposium on Theory of Computing (STOC)*, pages 25–32. ACM Press, 2016.

[KT00] Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 80–86. ACM Press, 2000.

[RS96a] Ronitt Rubinfeld and Madhu Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal of Computing*, 25(2):252–271, 1996.

[RS96b] Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM J. Comput.*, 25(2):252–271, 1996.

[STV01] Madhu Sudan, Luca Trevisan, and Salil P. Vadhan. Pseudorandom generators without the xor lemma. *J. Comput. Syst. Sci.*, 62(2):236–266, 2001.

[Sud97] Madhu Sudan. Decoding of reed solomon codes beyond the error-correction bound. *J. Complexity*, 13(1):180–193, 1997.

[Tho83] Christian Thommesen. The existence of binary linear concatenated codes with reed - solomon outer codes which asymptotically meet the gilbert- varshamov bound. *IEEE Trans. Information Theory*, 29(6):850–853, 1983.

[Var57] R. R. Varshamov. Estimate of the number of signals in error correcting codes. *Doklady Akadamii Nauk*, pages 739–741, 1957.

[Yek12] Sergey Yekhanin. Locally decodable codes. *Foundations and Trends in Theoretical Computer Science*, 6(3):139–255, 2012.

A Johnson Bound for List Recovery

In this section we prove Lemma 5.2 restated below.

Lemma 5.2 (Johnson bound for list recovery)**.** *Let $C \subseteq \Sigma^n$ be a code of relative distance at least δ. Then C is (α, ℓ, L)-list recoverable for any $\alpha < 1 - \sqrt{\ell \cdot (1-\delta)}$ with $L = \frac{\delta\ell}{(1-\alpha)^2 - \ell(1-\delta)}$.*

Proof. The proof is a simple adaptation of the proof of the Johnson bound for list decoding from [Gur06, Theorem 3.3].

Let $|\Sigma| = q$, let $S \in \binom{\Sigma}{\ell}^n$ be a string, and let $\mathcal{N} := \{c \in C \mid \mathrm{dist}(c, S) \leq \alpha\}$. Our goal will be to show that $L = |\mathcal{N}| \leq \frac{\delta\ell}{(1-\alpha)^2 - \ell(1-\delta)}$. As the minimum relative distance of the code C is δ and each $c \in \mathcal{N}$ has relative distance at most α from the string S, we have

$$\delta \leq \mathop{\mathbb{E}}_{\{\mathbf{x},\mathbf{y}\} \sim \binom{\mathcal{N}}{2}} \left[\frac{\Delta(\mathbf{x}, \mathbf{y})}{n}\right] \quad \text{and} \quad \alpha \geq \epsilon := \mathop{\mathbb{E}}_{\substack{\mathbf{x} \sim \mathcal{N} \\ i \sim [n]}} [1_{x_i \notin S_i}]\,. \tag{2}$$

Let $\mathbf{x}, \mathbf{y}$ be two distinct words in $\mathcal{N}$, chosen uniformly at random. We will obtain a lower bound on the expected fraction of coordinates where $\mathbf{x}$ and $\mathbf{y}$ agree (in terms of L, α and ℓ). We know that this expectation is at most $1 - \delta$. The theorem will follow by comparing these two quantities.

For $i \in [n]$, and $z \in \Sigma$, let $k_i(z) = |\{\mathbf{x} \in \mathcal{N} \mid x_i = z\}|$. Then we have that

$$\begin{aligned}
\mathop{\mathbf{Pr}}_{\{\mathbf{x},\mathbf{y}\} \sim \binom{\mathcal{N}}{2}} [x_i = y_i] &= \binom{L}{2}^{-1} \cdot \sum_{z \in \Sigma} \binom{k_i(z)}{2} \\
&= \binom{L}{2}^{-1} \cdot \left[\sum_{z \in S_i} \binom{k_i(z)}{2} + \sum_{z \in \Sigma \setminus S_i} \binom{k_i(z)}{2}\right] \\
&\geq \binom{L}{2}^{-1} \cdot \left[\ell \cdot \binom{k_i}{2} + (q - \ell)\binom{\frac{L - \ell k_i}{q - \ell}}{2}\right]
\end{aligned}$$

where $k_i = \dfrac{1}{\ell} \cdot \sum_{z \in S_i} k_i(z)$ and we used Jensen's inequality.

Hence, the expected fraction of coordinates where $\mathbf{x}$ and $\mathbf{y}$ agree is bounded by

$$\begin{aligned}\frac{1}{n}\cdot\sum_{i=1}^{n}\underset{\{\mathbf{x},\mathbf{y}\}\sim\binom{\mathcal{N}}{2}}{\mathbf{Pr}}[x_i=y_i] &\geq \frac{1}{n}\cdot\binom{L}{2}^{-1}\cdot\sum_{i=1}^{n}\left[\ell\cdot\binom{k_i}{2}+(q-\ell)\binom{\frac{L-\ell k_i}{q-\ell}}{2}\right]\\ &\geq \binom{L}{2}^{-1}\cdot\left[\ell\cdot\binom{t}{2}+(q-\ell)\binom{\frac{L-t\ell}{q-\ell}}{2}\right]\end{aligned}$$

where $t=\frac{1}{n}\cdot\sum_{i=1}^{n}k_i$ and we again used Jensen's inequality. Since the left hand side is bounded from above by $1-\delta$, after some rearrangement, we have:

$$(1-\delta)\cdot\binom{L}{2}\geq\ell\cdot\binom{t}{2}+(q-\ell)\binom{\frac{L-t\ell}{q-\ell}}{2} \tag{3}$$

Since ϵ is the expected fraction of disagreement between the words in $\mathcal{N}$ and S, we have that $Ln\epsilon$ is the total amount of disagreement between $\mathcal{N}$ and S. We can also count the amount of disagreement in the following way: $\ell k_i=\sum_{z\in S_i}k_i(z)$ is the amount of agreement between the words of $\mathcal{N}$ and the input list S_i. Hence, the total agreement between the words of $\mathcal{N}$ and S is $\sum_{i=1}^{n}\ell k_i=t\ell n$. This implies that the total disagreement is $Ln-t\ell n=Ln\epsilon$. Thus, we obtain that $\frac{t\ell}{L}=1-\epsilon$.

Substituting $\frac{t\ell}{L}=1-\epsilon$ in equation (3), and rearranging terms, we have

$$\begin{aligned}(1-\delta)\cdot\frac{L(L-1)}{2} &\geq \ell\cdot\frac{t(t-1)}{2}+\frac{(L-t\ell)(L-t\ell-q+\ell)}{2(q-\ell)}\\ &= \frac{\ell t^2}{2}-\frac{\ell t}{2}+\frac{(L-t\ell)^2}{2(q-\ell)}-\frac{L-t\ell}{2}\\ &= \frac{(1-\epsilon)^2L^2}{2\ell}-\frac{(1-\epsilon)L}{2}+\frac{(L\epsilon)^2}{2(q-\ell)}-\frac{L\epsilon}{2},\end{aligned}$$

which gives

$$\begin{aligned}(1-\delta)\cdot(L-1) &\geq \frac{(1-\epsilon)^2}{\ell}L-(1-\epsilon)+\epsilon\left(\frac{\epsilon L}{q-\ell}-1\right)\\ &= \frac{(1-\epsilon)^2}{\ell}L+\frac{\epsilon^2L}{q-\ell}-1.\end{aligned}$$

By grouping the terms with L and rearranging the inequality above, we get that

$$L\leq\frac{\delta}{\frac{(1-\epsilon)^2}{\ell}+\frac{\epsilon^2}{q-\ell}-(1-\delta)}.$$

By looking at the denominator of the equation above, we have

$$\begin{aligned}
\frac{q\ell}{q-\ell}\cdot\left(\frac{(1-\epsilon)^2}{\ell}+\frac{\epsilon^2}{q-\ell}-(1-\delta)\right) &= \frac{q}{q-\ell}-\frac{2q\epsilon}{q-\ell}+\frac{q\epsilon^2}{q-\ell}+\frac{q\ell\epsilon^2}{(q-\ell)^2}-\frac{q\ell(1-\delta)}{q-\ell}\\
&= \frac{q}{q-\ell}-\frac{2q\epsilon}{q-\ell}+\frac{q^2\epsilon^2}{(q-\ell)^2}-\frac{q\ell(1-\delta)}{q-\ell}\\
&\geq \frac{q}{q-\ell}-\frac{2q\epsilon}{q-\ell}+\frac{q\epsilon^2}{q-\ell}-\frac{q\ell(1-\delta)}{q-\ell}\\
&= \frac{q}{q-\ell}\cdot\big[(1-\epsilon)^2-\ell(1-\delta)\big].
\end{aligned}$$

Hence, we obtain

$$\begin{aligned}
L &\leq \frac{\delta}{\frac{(1-\epsilon)^2}{\ell}+\frac{\epsilon^2}{q-\ell}-(1-\delta)}\\
&= \frac{\frac{q\ell}{q-\ell}\cdot\delta}{\frac{q\ell}{q-\ell}\cdot\left(\frac{(1-\epsilon)^2}{\ell}+\frac{\epsilon^2}{q-\ell}-(1-\delta)\right)}\\
&\leq \frac{\frac{q\ell}{q-\ell}\cdot\delta}{\frac{q}{q-\ell}\cdot[(1-\epsilon)^2-\ell(1-\delta)]}\\
&= \frac{\delta\ell}{(1-\epsilon)^2-\ell(1-\delta)}\\
&\leq \frac{\delta\ell}{(1-\alpha)^2-\ell(1-\delta)},
\end{aligned}$$

where the last inequality follows since $\epsilon \geq \alpha$. □