Asymptotically-Good RLCCs with
$(\log n)^{2+o(1)}$ Queries

Gil Cohen∗  Tal Yankovitz†

July 25, 2023

Abstract

Recently, Kumar and Mon reached a significant milestone by constructing asymptotically good relaxed locally correctable codes (RLCCs) with poly-logarithmic query complexity. Specifically, they constructed $n$-bit RLCCs with $O(\log^{69} n)$ queries. This significant advancement relies on a clever reduction to locally testable codes (LTCs), capitalizing on recent breakthrough works in LTCs.

With regards to lower bounds, Gur and Lachish (SICOMP 2021) proved that any asymptotically-good RLCC must make at least $\tilde{\Omega}(\sqrt{\log n})$ queries. Hence emerges the intriguing question regarding the identity of the least value $\frac{1}{2} \leq e \leq 69$ for which asymptotically-good RLCCs with query complexity $(\log n)^{e+o(1)}$ exist.

In this work, we make substantial progress in narrowing the gap by devising asymptotically-good RLCCs with a query complexity of $(\log n)^{2+o(1)}$. The key insight driving our work lies in recognizing that the strong guarantee of local testability overshoots the requirements for the Kumar-Mon reduction. Consequently, we can replace the LTCs by “vanilla” expander codes which indeed have the necessary property: local testability in the vicinity of the code.

∗Tel Aviv University.gil@tauex.tau.ac.il. Supported by ERC starting grant 949499 and by the Israel Science Foundation grant 1569/18.

†Tel Aviv University.talyankovitz@mail.tau.ac.il. Supported by ERC starting grant 949499.
## Contents

1 Introduction 1
   1.1 Our result ......................................................... 2

2 Proof Overview 3
   2.1 Expander codes are locally testable in their vicinity ............ 4
      2.1.1 Expander codes .................................................. 4
      2.1.2 Expander codes are locally testable in their vicinity ........ 5
   2.2 RLCCs from expander codes ........................................ 5
      2.2.1 The construction ............................................... 5
      2.2.2 The tester and its analysis ..................................... 6
      2.2.3 Improving the correction radius ................................. 7
      2.2.4 Explicitness .................................................... 9

3 Preliminaries 9
   3.1 Notations and conventions ......................................... 9
   3.2 Error correcting codes ............................................ 10
   3.3 Relaxed locally correctable codes ................................ 10
   3.4 Expanders and expander codes ..................................... 11

4 Vicinity Locally Testable Codes 11

5 RLCCs from VLTCs 13
1 Introduction

Error correcting codes with “local guarantees” play a pivotal role in modern coding theory, and their study is highly motivated by applications to theoretical computer science. Of particular interest are locally decodable codes (LDCs), introduced by Katz and Trevisan [KT00], and locally correctable codes (LCCs) that originated in works on program checking [BK95, Lip90]. These are codes that admit highly efficient procedures for recovering a single data symbol. LDCs allow one to decode a specific symbol of the message while querying only a small number of symbols of the received, possibly corrupted, codeword. On the other hand, LCCs offer a method to recover any desired symbol of the codeword using only a few queries.

In their influential work, Ben-Sasson, Goldreich, Harsha, Sudan and Vadhan [BSGH06] introduced a natural relaxation of LDCs dubbed relaxed locally decodable codes (RLDCs). In essence, RLDCs allow the decoder to abort in the face of corruption, while still being required to always succeed when provided access to a codeword. The natural counterpart to LCCs, known as relaxed locally correctable codes (RLCCs), was later introduced by Gur, Ramnarayan, and Rothblum [GRR20]. For linear codes, RLCCs directly induce RLDCs, and so in this case it can be easily seen that RLCCs are stronger objects. 1 LDCs, LCCs and their relaxed counterparts have attracted significant attention. The reader is referred to [GI05, Yek08, DGY11, Efr12, GKS13, KSY14, Mei14, HOW15, GKO18, GRR20, DGGW20, GL21, CY21, CY22a, DGL21, BBC23, Gol23a, Gol23b] and references therein.

For simplicity, in this introductory part we focus on binary codes. Formally, a $(q, δ, ε)$ RLCC is an error correcting code $C ⊆ \{0, 1\}^n$ that is equipped with a randomized “correction procedure” $\text{Cor} : \{0, 1\}^n \times [n] \rightarrow \{0, 1\} \cup \{\perp\}$ that makes at most $q$ queries to its $n$-bit input, and have the following guarantees:

1. For every codeword $c \in C$, $\text{Cor}(c, i) = c_i$ for every $i \in [n]$, with certainty.

2. For every $w \in \{0, 1\}^n$ of distance at most $δn$ from some codeword $c \in C$, and for every $i \in [n]$, $\text{Cor}(w, i) \in \{c_i, \perp\}$ with probability at least $1 - ε$.

We designate $δ$ as the correction radius of the RLCC, emphasizing that, as a direct implication, $δ$ serves as a lower bound on the code’s relative-distance. In this paper we consider asymptotically-good RLCCs, by which we mean RLCCs with a constant rate and a constant correction radius. The reader may consult [CGS20], and references therein, to learn more about the constant query regime.

1For the case of non-linear codes, see [BGT16], Theorem A.6.
In their work, Gur, Ranmarayan and Rothblum [GRR20] constructed asymptotically-good RLCCs and RLDCs with query complexity \((\log n)^{O(\log \log n)}\). This offers a significant saving over the query complexity of the state-of-the-art LCCs and LDCs, \(q = 2^{\tilde{O}(\sqrt{\log n})}\), obtained by Kopparty, Meir, Ron-Zewi, and Saraf [KMRS17]. Interestingly, the RLCC of [GRR20] draws inspiration from the ideas presented in the construction of locally testable codes (LTCs) that appears in [KMRS17], rather than building on the LCC construction from the same paper. The construction is based on a repeated application of tensoring and distance amplification.

Continuing along a similar framework, but employing a more rate-efficient ingredient instead of tensoring, Cohen and Yankovitz [CY22b] obtained asymptotically-good linear RLCCs, hence also RLDCs, with query complexity \((\log n)^{O(\log \log \log n)}\). This somewhat unnatural looking function, also taking into account the \(\tilde{\Omega}(\sqrt{\log n})\) lower bound on the query complexity of asymptotically-good RLCCs [GL21] gave some hope that a query complexity of \((\log n)^{O(1)}\) is achievable.

Indeed, this hope was realized in an exciting recent work by Kumar and Mon [KM23] who obtained RLCCs with query complexity \(O(\log^{69} n)\). Their proof builds on a reduction to LTCs, cementing the intuitive connection between RLCCs and LTCs, as hinted in the work of [GRR20], and building on the recent breakthrough in LTCs construction by Dinur et al. [DEL+22].

1.1 Our result

The works of Kumar and Mon [KM23] and Gur and Lachish [GL21] leave us with the fundamental question regarding the identity of the least value \(\frac{1}{2} \leq e \leq 69\) for which asymptotically-good RLCCs with \((\log n)^{e+o(1)}\) queries exist. In this work, we make significant progress in narrowing the gap by proving that \(e \leq 2\).

**Theorem 1.1** (Main result). For every \(\delta < 1\) and for infinitely many \(n\)-s there exists an explicit binary asymptotically-good linear RLCC (hence also RLDC) of block-length \(n\) having correction radius \(\delta\), rate \(1 - \delta^{1-o(1)} - o_n(1)\), and query complexity

\[
q = (\log n)^{2+o(1)}.
\]

Although Kumar and Mon did not explicitly focus on optimizing the exponent in their query complexity, it appears that achieving an exponent as low as 2 is not feasible

\footnote{In the case of non-adaptive RLDCs, a slightly stronger lower bound of \(\Omega(\sqrt{\log n})\) is known [Gol23b]. By combining this result with [Gol23a], the strengthened lower bound of \(\Omega(\sqrt{\log n})\) can be extended to encompass all linear RLDCs as well.}

\footnote{Kumar and Mon require LTCs with rate approaching 1, hence they could not use the independently discovered LTCs by Panteleev and Kalachev [PK22].}
using existing LTCs within their framework. We believe that the realization that a more “economical” primitive, substituting the LTCs employed by Kumar and Mon, can be employed, plays a pivotal role in achieving such a low query complexity. On the flip side, we believe that new ideas are required to go below \( \log^2 n \) queries, if at all possible.

The exact asymptotic behavior of the query complexity \( q \) which is hidden, by design, under the \( o_n(1) \)-notation is \( q = (\log n)^{2+\varepsilon(n)} \), where \( \varepsilon(n) = \frac{(\log \log \log n)^3}{\log \log n} \). Similarly, the precise asymptotic behavior underlying the term \( \delta^{1-o(1)} \) that appears in the bound on the rate is \( \delta \cdot 2^{O((\log \log 1/\delta)^3)} \). These expressions are derived from the parameters of the lossless expander utilized in our work [CRVW02]. While it is possible that slight improvements could be achieved by employing newer constructions of randomness extractors in place of the ingredients used within [CRVW02], we have not made any specific attempts to optimize the \( o(1) \) terms. At any rate, the reader is referred to Theorem 5.2 for the formal statement.

We emphasize that even from an information theoretic standpoint, the question of the lowest achievable query complexity for an asymptotically-good RLCC is intriguing. Explicitly aside, we can obtain a slightly reduced query complexity, \( q = O(\log^2 n \cdot \log \log n) \). Moreover, in such case the rate comes quite close to the Gilbert-Varshamov bound, \( \rho = 1 - O(\delta \log 1/\delta) - o(1) \). In fact, we can construct RLCCs with these parameters in quasi-polynomial time, \( 2^{(\log n)^{O(1)}} \) by instantiating our construction with another expander construction from [CRVW02].

## 2 Proof Overview

An LTC (Locally Testable Code) is a type of error correcting code that incorporates a local tester—an algorithm that performs a limited number of queries on the received word \( w \in \{0,1\}^n \) and rejects it with a probability proportional to its distance from the code. Importantly, a tester never rejects a valid codeword. LTCs with such a guarantee are occasionally referred to as strong LTCs in the literature to differentiate them from an alternative, weaker definition, which only requires the tester to reject words that are sufficiently distant from the code. It is important to recognize that LTCs must in particular handle words that are very far from the code, which constitute the vast majority, “unstructured” portion of \( \{0,1\}^n \). For a more comprehensive exploration of LTCs, we recommend referring to Goldreich’s lecture notes [Gol16].

The key insight driving our work lies in recognizing that the strong guarantee of local testability overshoots the requirements for the Kumar-Mon reduction. Expander codes, although provably not full-fledged LTCs in general, satisfy the required property,
namely, all expander codes are locally testable in their vicinity. We make this more precise in Section 2.1 below where we also recall the definition of expander codes. Then, in Section 2.2, we explain how to obtain our RLCCs by instantiating the Kumar-Mon reduction with expander codes instead of with LTCs.

The fact that expander codes are locally testable in the vicinity of the code can be derived as a consequence of the analysis of the sequential decoding algorithm for expander codes. The reader is referred to Section 2.3.1 in Spielman’s PhD Thesis [Spi95] and to the discussion in Chapter 5. Interestingly, in his lecture notes, Goldreich [Gol16] discusses offhand a variant of what we call local testability in the vicinity of the code (see Definition 10 in the notes), remarks that this definition may potentially be useful despite being highly non-intuitive in the context of PCPs, and refers to the abovementioned discussion in Spielman’s thesis.

For the sake of completeness, we provide a simple proof for the testability of expander codes in their vicinity without relying on a full decoding argument. This streamlined approach helps clarify the concept and establishes the essential property of local testability which is necessary for the reduction.

2.1 Expander codes are locally testable in their vicinity

2.1.1 Expander codes

Let us begin by revisiting the notion of expander codes, introduced by Sipser and Spielman [SS96]. Let $G = (L, R, E)$ be a bipartite $d$-left-regular graph. Denote $|L| = n$ and $|R| = \tau n$. The graph $G$ is said to be a $(\gamma, (1-\varepsilon)d)$-lossless expander if for every $S \subseteq L$ of size $|S| \leq \gamma n$, the set of neighbors of $S$, denoted $\Gamma(S)$, is of size at least $(1-\varepsilon)d|S|$. Additionally, we define $\Gamma_u(S)$ as the set of unique neighbors of $S$ which consists of all vertices $v \in R$ such that $|\Gamma(v) \cap S| = 1$. It is easy to prove that

$$|\Gamma(S)| \geq (1-\varepsilon)d|S| \implies |\Gamma_u(S)| \geq (1-2\varepsilon)d|S|.$$ 

Moving forward, we will assume that $\varepsilon$ is a small enough constant such that the right-hand side of the aforementioned equation remains nontrivial. For instance, we can take $\varepsilon = \frac{1}{4}$ as one possible choice. Accordingly, we will refer to the graph $G$ satisfying the condition for this chosen value of $\varepsilon$ as a $\gamma$-lossless expander for brevity.

By employing the probabilistic method, it is possible to prove the existence of $\gamma$-lossless expanders for every desired sizes $|L| = n$, $|R| = \tau n$, where the left-degree $d = O\left(\log \frac{1}{\gamma}\right)$, and $\gamma = O\left(\frac{1}{d}\right)$. For the sake of simplicity and convenience, we shall use such an expander throughout this informal section. In Section 2.2.4, we will briefly discuss the modifications in parameters if we choose to work with the explicit expander from the work of [CRVW02].
With the expander $G$, we associate a binary code $\text{EC}(G)$ on block-length $n$, dubbed the expander code associated with $G$ as follows. Every vertex $v \in R$ is thought of as a constraint, namely, for $x \in \{0, 1\}^n$ to be a codeword, we require that for every $v \in R$, the parity of the bits $\{x_u \mid u \in \Gamma(v)\}$ equals 0 (where we identify the set $L$ with the index set $[n]$). It readily follows that $\text{EC}(G)$ has rate at least $1 - \tau$, and it is not hard to show that the code has relative-distance at least $\gamma$.\footnote{In fact, stronger bounds on the relative-distance are known though they will not be necessary for our purposes.}

2.1.2 Expander codes are locally testable in their vicinity

We turn to show that $\text{EC}(G)$ is locally testable in its vicinity. Let $w \in \{0, 1\}^n$ be word of distance exactly $\gamma' n$ from $\text{EC}(G)$ and let $S \subseteq L$ be the set that corresponds to $w$, $S = \{i \mid w_i = 1\}$. We assume that $\gamma' \leq \gamma$, reflecting the fact that we are in the vicinity of the code. Our tester will simply sample a right vertex $v$ at random and rejects if the constraint associated with $v$ is unsatisfied.

Note that the tester will reject whenever $v$ is sampled from $\Gamma_u(S)$. Thus, the probability of rejection is bounded below by

$$\frac{|\Gamma_u(S)|}{|R|} \geq \frac{d|S|}{2\tau n} = \frac{d\gamma'}{2\tau}.$$ 

Plugging the parameters of the non-explicit expander above, we get that the rejection probability is bounded below by $\Omega(\frac{\gamma'}{\gamma})$. In particular, if $w$ is at the “outskirts” of the expander code, namely, $\gamma' \leq \gamma$ yet $\gamma' = \Omega(\gamma)$, then the rejection probability is constant.

Of course, the rejection probability can be amplified to $1 - 2^{-t}$ by repeating the process for $O(t)$ times.

As for the query complexity, for simplicity assume that $G$ is also $c$-right regular. Then, the query complexity required for obtaining a constant rejection probability is

$$c = \frac{d}{\tau} = O \left( \frac{1}{\tau \log \frac{1}{\tau}} \right).$$

2.2 RLCCs from expander codes

2.2.1 The construction

The key distinction between RLCCs and LTCs, whether they are full-fledged LTCs or only guaranteed to work in their vicinity, lies in the fact that RLCCs are also provided with an index $i \in [n]$ indicating the specific bit to be corrected. To bridge this gap, following
Kumar and Mon [KM23], we define our RLCC using a binary tree of expander codes so as to make sure that any index \( i \) participates in expander codes of increasing size. This allows one to “zoom in” on the \( i \)-th bit using expander codes. We elaborate on this next.

Assume for simplicity that \( n = 2^m \). We take a sequence of \( m \) expander codes \( C_0, C_1, \ldots, C_{m-1} \) on block-lengths \( n, \frac{n}{2}, \frac{n}{4}, \ldots \), respectively. All these expander codes share the same parameters as in Section 2.1, namely, all expanders have the same left and right degrees \( d, c \), hence the same \( \tau \), as well as the same parameter \( \gamma \).

Our RLCC, denoted \( C' \), is obtained by intersecting the code \( C_0 \) on the index set \([n]\) with the code \( C_1 \) on both the index set \([n/2]\) and \([n/2]+[n/2] \). Put differently, we impose the linear constraints of \( C_1 \) on both the first half and second half of the bits. The linear constraints of the code \( C_2 \) are enforced onto the four blocks \([n/4], [n/4]+[n/4], [n/4]+[n/4], \) and \([3n/4]+[n/4] \), and so forth in a binary tree fashion. It is evident that the rate of the resulting code, \( C' \), is at least \( 1 - m\tau \), which implies that we need to select \( \tau < \frac{1}{m} = \frac{1}{\log n} \) to satisfy the rate constraint.

2.2.2 The tester and its analysis

Our claim is that \( C' \) is an RLCC with correction radius \( \frac{\gamma}{2} = \Omega(\frac{1}{\log n \cdot \log \log n}) \) using the corrector we describe and analyze next. In Section 2.2.3 we explain how to modify the construction slightly so as to obtain any desired correction radius. Before we begin, we remark that it is readily seen that the corrector described below never aborts and always outputs the correct bit given oracle access to a codeword of \( C' \). Therefore, we focus on the scenario where a word \( w \in \{0,1\}^n \setminus C' \) is given, with a distance at most \( \frac{\gamma}{2} \cdot n \) from the code \( C' \). In this case, our objective is to either abort or output the \( i \)-th bit of the unique codeword closest to \( w \). Indeed, as \( C' \subseteq C_0 \), and since \( C_0 \) has relative-distance at least \( \gamma \), there exists a unique codeword \( c \in C' \) that is of distance at most \( \frac{\gamma}{2} \cdot n \) from \( w \).

With this in mind, let us consider a specific index \( i \in [n] \), and let \( B \) be either \([n/2]\) or \([n/2]+[n/2] \), depending on which of these blocks contains \( i \). We define \( \varepsilon \) such that \( \varepsilon \cdot \frac{n}{2} \) is the distance between \( w_B \) and \( c_B \) - the projections of \( w, c \) onto block \( B \), respectively. We know that \( \varepsilon \leq \gamma \) as in the worst case all \( \frac{\gamma}{2} \cdot n = \gamma \cdot |B| \) errors could fall into \( B \). We consider the two possible cases based on whether the ratio of errors deteriorates or not when moving to block \( B \), i.e., whether \( \varepsilon \leq \frac{\gamma}{2} \) or not.

Assume that \( \varepsilon > \frac{\gamma}{2} \). As we also know that \( \varepsilon \leq \gamma \), namely, \( w_B \) is in the vicinity of the
code $C_1$, we may invoke $C_1$-s tester, and by making

$$O \left( t \cdot \frac{1}{\tau} \log \frac{1}{\tau} \right) = O(t \cdot \log n \cdot \log \log n)$$

queries to $w_B$, reject with probability $1 - 2^{-t}$. Hence, if the tester ended up not aborting, we may assume that we are in the case $\varepsilon \leq \frac{\gamma}{2}$, and our assumption will be wrong with probability at most $2^{-t}$. Thus, unless the tester aborted, we can safely recurse to $B$. In more detail, since $w_B$ is of distance at most $\frac{\gamma}{2} \cdot |B|$ from $c_B$, and since $C_1$ is a code with relative-distance $\gamma$ on the index set $B$ that participates in the intersection defining $C'$, we know that $c_B$ is the unique codeword of $C_1$ that is $\frac{\gamma}{2} \cdot |B|$-close to $w_B$. This is precisely the same guarantee we started with and, importantly, we maintain the invariant that the projection of $c$ to the block is the closest codeword to $w$’s projection to that block, with respect to the suitable code. Hence, if and when the time comes to return the $i$-th bit, it will be that bit of $c$ that is returned rather than the bit of another codeword. This invariant allows us to recurse to $B$.

If in any of the $m = \log n$ levels of recursion the tester aborts, the corrector succeeds. Otherwise, the code $C_m$ is invoked and returns the correct bit except in case where the corrector should have aborted. By a union bound over the $m$ levels, this event occurs with probability at most $2^{-t} m$. Setting $t = O(\log m) = O(\log \log n)$, the total number of queries made is

$$O \left( mt \cdot \log n \cdot \log \log n \right) = O \left( (\log n \cdot \log \log n)^2 \right).$$

We remark that the factor of $t = O(\log \log n)$ can be removed as the union bound can be avoided with some care.

### 2.2.3 Improving the correction radius

To achieve any desired correction radius $\delta_0 < 1$, we can easily modify the construction. Simply take the expander code $C_0$ to have relative-distance $\gamma_0 = 2\delta_0$ and rate $1 - \tau_0 = 1 - O(\delta_0 \log \frac{1}{\delta_0})$, while keeping the parameters of the remaining codes $C_1, \ldots, C_m$ unchanged. The rate of the resulting code, denoted $C''$, is given by $1 - (\tau_0 + (m - 1)\tau)$, point being that we can afford taking $C_0$ to be a high-rate code as we only “pay” $\tau_0$ once rather than $m$ times.

In the modified construction, the corrector remains unchanged with the exception of an initial phase. In this initial phase, we invoke $C_0$-s tester (which, as the perceptive reader may have noted, has not been used in Section 2.2.2) to check whether the number of errors is less than $\frac{\gamma}{2} \cdot |B|$. The probability to catch an unsatisfied constraint is no longer
constant as before; instead, it becomes
\[ \Omega \left( \frac{\gamma}{\gamma_0} \right) = \Omega \left( \frac{1}{\log n \cdot \log \log n} \right). \]

To ensure a constant rejection probability, we need to sample not just one but \( \Theta \left( \frac{n_0}{\gamma} \right) \) right vertices and query their neighbors. If we denote the right-degree of the expander underlying \( C_0 \) by \( c_0 \), this will result in a total number of
\[ O \left( \frac{n_0}{\gamma} \cdot c_0 \right) = O \left( \frac{1}{\gamma} \right) = O \left( \log n \cdot \log \log n \right) \]
queries. Note that we have used the fact that in the probabilistic construction, \( c_0 \gamma_0 = O(1) \).

If \( C_0 \)-s corrector does not reject, we maintain the same guarantee we had before regarding the number of errors in \( B \), and we can proceed with the same strategy as previously describe. Hence, with the same query complexity of \( O(\log^2 n \cdot \log \log n) \), it is possible to obtain any distance \( \delta_0 \) and rate \( 1 - O(\delta_0 \log \frac{1}{\delta_0}) - o(1) \).

**A remark regarding the bi-regularity assumption.** We wish to draw attention to an issue that might be easily overlooked regarding the initial phase discussed above in the absence of bi-regularity. Throughout this informal proof overview, we are working under the premise that the expander that is underlying the expander code is bi-regular. This can be assumed to be the case for the probabilistic construction though not necessarily for the expander that we are using for our RLCC construction [CRVW02].

In the absence of bi-regularity, one can proceed by defining the tester as follows: When sampling a right vertex, query its neighbors only if its degree is at most \( \kappa c \), where \( \kappa \) serves as a cutoff parameter and \( c \) now stands for the average right degree. That is, if the degree exceeds this threshold, the vertex is ignored for the purpose of testing. As a result, the “heavy” constraints are embedded in the code’s definition, yet they are not utilized by the tester. This seemingly minor technicality has a rather surprising impact on the parameters: the query complexity of the tester in the initial phase alone now becomes \( (\log n)^{2+o(1)} \). However, this increase is affordable, given that it applies only to the initial phase. As we progress through the remaining \( \log n \) levels, the query complexity for each level remains at \( (\log n)^{1+o(1)} \).
2.2.4 Explicitness

Capalbo, Reingold, Vadhan and Wigderson [CRVW02] constructed explicit \( \gamma \)-lossless expanders with near-optimal parameters. \(^7\) Quantitatively, following the notation in Section 2.1.1, their construction has degree

\[
d = 2^{O\left((\log \log \frac{1}{\tau})^3\right)} = \left(\frac{1}{\tau}\right)^{o(1)},
\]

which should be compared with \( d = O\left(\log \frac{1}{\tau}\right) \) obtained using the probabilistic construction, while maintaining \( \gamma = O\left(\frac{\tau}{d}\right) \). As before, the probability of the expander code’s tester to reject a word from the outskirts of the code is constant. Hence, the query complexity is, again, the right degree,

\[
c = \frac{d}{\tau} = \frac{1}{\tau} \cdot 2^{O\left((\log \log \frac{1}{\tau})^3\right)} = \left(\frac{1}{\tau}\right)^{1+o(1)}.
\]

Recall that, due to rate considerations, \( \tau \) is taken to be \( \frac{1}{\log n} \), thus the query complexity of the expander code’s tester is \((\log n)^{1+o(1)}\). The overall query complexity of the resulted RLCC’s corrector is then \( m \cdot (\log n)^{1+o(1)} = (\log n)^{2+o(1)} \).

3 Preliminaries

3.1 Notations and conventions

Unless stated otherwise, all logarithms in this paper are taken to the base 2. The set of natural numbers is \( \mathbb{N} = \{0, 1, 2, \ldots\} \). For \( n \in \mathbb{N}, \ n \geq 1 \), we use \([n]\) to denote the set \( \{1, \ldots, n\}\). This does not affect the stated results. For \( q \in \mathbb{N}, \ q \geq 2 \), we use \( H_q \) to denote the \( q \)-ary entropy function, and \( H = H_2 \) to denote the binary entropy function.

For a finite set \( N \), we refer to a function \( v \in \mathbb{F}^N \) as a vector and we say that it is indexed by \( N \). For a vector \( v \in \mathbb{F}^N \) and \( i \in N \) we use \( v_i \) as a shorthand for \( v(i) \). For a vector \( v \in \mathbb{F}^N \) and a set \( N' \subseteq N \) we denote by \( v_{N'} \) the vector \( v' \in \mathbb{F}^{N'} \) such that \( v'_i = v_i \) for every \( i \in N' \). For two vectors \( u, v \in \mathbb{F}^N \), their (absolute) hamming distance is \( |\{i \in N \mid u_i \neq v_i\}| \), which we denote by \( \text{Dist}(u, v) \), and their relative hamming distance is \( \frac{\text{Dist}(u, v)}{|N|} \), which we denote by \( \text{RelDist}(u, v) \).

\(^7\)A lot of work has been done, much of it very recently, on simplifying the [CRVW02] construction and on obtaining different variants of lossless expanders such as unique neighbor expanders, however, none of these works seem to be sufficient for our needs. The reader may consult [AD23, CRTS23, Gol23c, HMMP23] and references therein.
3.2 Error correcting codes

We start by recalling the definition of an error correcting code. In this work we only consider linear codes. The definition below is standard, however, for our purposes we find it convenient to work with an arbitrary index set rather than the usual set $[n]$, and so the reader may benefit from glancing over the definition.

**Definition 3.1.** For a finite set $N$ of size $|N| = n$ and a field $F$, a code is a linear subspace $C \subseteq F^N$. We say that the code $C$ is indexed by $N$ and that it is over $F$. The length of the code is $n$. The dimension of the code, usually denoted by $k$, is the dimension of $C$ over $F$. The (non-local) distance of the code, denoted by $d$, is $\min_{c,c' \in C, c \neq c'} \text{Dist}(c, c')$. The rate of the code, typically denoted by $\rho$, is $\frac{k}{n}$. The (non-local) relative-distance of the code is defined to be $\frac{d}{n}$. The elements of $C$ are called codewords.

3.3 Relaxed locally correctable codes

We turn to recall the definition of relaxed locally correctable codes as put forth by Gur, Ramnarayan and Rothblum [GRR20].

**Definition 3.2.** A code $C \subseteq F^N$ is called a $(q, \delta, \varepsilon)$-RLCC (relaxed locally correctable code, abbreviated) if there exists a randomized procedure $\text{Cor} : F^N \times N \to F \cup \{\bot\}$ with the following guarantees:

- For every $i \in N$, $c \in C$ and $w \in F^N$, satisfying $\text{RelDist}(w, c) \leq \delta$, $\text{Cor}(w, i) \in \{c_i, \bot\}$ with probability at least $1 - \varepsilon$.
- $\text{Cor}(c, i) = c_i$ with probability one on any $c \in C$ and $i \in N$.
- $\text{Cor}(w, i)$ always makes at most $q$ queries to $w$.

We refer to $\text{Cor}$ as the local corrector (or the corrector). The parameter $\delta$ is called the correction radius, and the parameter $q$ is called the query complexity.

The error parameter of an RLCC can be easily amplified at low cost to the query complexity, as stated in the following claim (for a simple proof see, e.g., [CY22b]).

**Claim 3.3.** Let $C \subseteq F^N$ be a $(q, \delta, \varepsilon)$-RLCC. Then, for any $h \in N$, $C$ is also an $(hq, \delta, \varepsilon^h)$-RLCC.
3.4 Expanders and expander codes

We set some standard notation. Let \( G = (V, E) \) be an undirected graph. For \( v \in V \) we define \( \Gamma(v) \) as the set of neighbors of \( v \) in \( G \), and let \( \text{deg}(v) \) be the degree of \( v \). For a set of vertices \( S \subseteq V \), we let \( \Gamma(S) = \bigcup_{v \in S} \Gamma(v) \), and define

\[
\Gamma_u(S) = \{ v \in V \mid v \text{ is adjacent to exactly one } u \in S \}.
\]

**Definition 3.4 (Unique-neighbor expanders).** A left-\( d \)-regular bipartite graph \( G = (L, R, E) \) is a \((\gamma, \alpha)\)-unique-neighbor expander if for every \( S \subseteq U \) such that \(|S| \leq \gamma |U|\), it holds that \(|\Gamma_u(S)| \geq \alpha d |S|\).

The following theorem readily follows by the construction of lossless conductors as given by Theorem 7.3 in \[CRVW02\].

**Theorem 3.5 \([CRVW02]\).** There exist universal constants \( c_0 \geq 1 \) and \( \beta \leq 1 \) such that the following holds. For every \( n \) and \( m \leq n \), there exists an explicit \((\gamma, \alpha)\)-unique-neighbor expander \( G = (L, R, E) \) with \(|L| = 2^n, |R| = 2^m\), having left degree

\[
d \leq 2^{c_0 \log^3(n-m)},
\]

where \( \alpha = \Omega(1) \), and \( \gamma = \beta \cdot \frac{2^{m-n}}{d} \).

**Definition 3.6 (Expander codes).** Let \( G = (L, R, E) \) be a bipartite graph and let \( \mathbb{F} \) be a field. The expander code associated with \( G \) is defined by

\[
\text{EC}_\mathbb{F}(G) = \left\{ w \in \mathbb{F}^L \mid \forall v \in R \sum_{u \in \Gamma(v)} w_u = 0 \right\}.
\]

We usually omit the subscript \( \mathbb{F} \) when the field is clear from context.

It is easy to see that the rate of \( \text{EC}_\mathbb{F}(G) \) is at least \( 1 - \frac{|R|}{|L|} \).

4 Vicinity Locally Testable Codes

In this section we give the formal definition of local testability in the vicinity of the code and prove that expander codes have this property.

**Definition 4.1 (VLTCs).** A code \( C \subseteq \mathbb{F}^N \) is called a \((q, \delta, \kappa, \sigma)\)-VLTC (vicinity locally testable code, abbreviated) if there exists a randomized procedure \( \text{Tes}: \mathbb{F}^N \to \{\circ, \perp\} \), with the following guarantees:
• For every $c \in C$ and $w \in \mathbb{F}^N$, satisfying $\text{RelDist}(w, c) \leq \delta$,

$$\Pr[\text{Tes}(w) = \perp] \geq \kappa \cdot \text{RelDist}(w, c) - \sigma;$$

• $\text{Tes}(c) = \circ$ with probability one on any $c \in C$.

• $\text{Tes}(w)$ always makes at most $q$ queries to $w$.

We call $\text{Tes}$ a local tester (or tester for short). The parameter $q$ is referred to as the query complexity.

We move to show that expander codes constructed from unique-neighbor expanders are VLTCs.

Lemma 4.2. Let $G = (L, R, E)$ be a $d$-left-regular $(\gamma, \alpha)$-unique-neighbor expander with average right-degree $\bar{c}$. Then, for every $b > 1$, $\text{EC}(G)$ is a $(b\bar{c}, \gamma, \alpha\bar{c}, \frac{1}{b})$-VLTC.

Proof. Define

$$R' = \{v \in R \mid \deg(v) \leq b\bar{c}\}.$$

By an averaging argument, $|R'| \geq (1 - \frac{1}{b})|R|$. The tester for $\text{EC}(G)$, given oracle access to $w \in \mathbb{F}^L$, proceeds as follows:

1. Sample $v \in R'$ uniformly at random.
2. Query $w$ on $\Gamma(v)$.
3. Output $\circ$ if $\sum_{u \in \Gamma(v)} w_u = 0$; and $\perp$ otherwise.

As the sampled vertex $v$ is in $R'$, the query complexity of the tester is indeed bounded above by $b\bar{c}$. Further, when $w \in \text{EC}(G)$, the tester outputs $\circ$ with certainty.

Consider then a word $w \in \mathbb{F}^L$ such that $\text{RelDist}(w, c) \leq \gamma$ for some codeword $c \in \text{EC}(G)$. Let $S = \{v \in L \mid w_v \neq c_v\}$. As $|S| \leq \gamma|L|$ we have that $|\Gamma_u(S)| \geq ad|S|$. Notice that if the vertex $v$ that is sampled in Step 1 lies in $\Gamma_u(S)$ then the tester outputs $\perp$. Therefore, the probability of the tester to output $\perp$ is at least

$$\frac{|\Gamma_u(S)| - |R \setminus R'|}{|R|} \geq \frac{ad|S|}{|R|} - \frac{1}{b}$$

$$= \alpha \cdot \frac{d|L|}{|R|} \cdot \frac{|S|}{|L|} - \frac{1}{b}$$

$$= \alpha\bar{c} \cdot \text{RelDist}(w, c) - \frac{1}{b},$$

which concludes the proof. \qed
5 RLCCs from VLTCs

Following a similar argument to the one underlying the Kumar-Mon reduction, the following proposition states that a sequence of VLTCs can be used to construct an RLCC.

Proposition 5.1. Let $C_1 \subseteq \mathbb{F}^{N_1}, \ldots, C_m \subseteq \mathbb{F}^{N_m}$ be codes with rates $\rho_1, \ldots, \rho_m$, respectively, such that for every $i \in [m-1]$, $C_i$ is a $(q, \delta', \kappa, \sigma)$-VLTC, and $C_m$ is a $(q, \delta, \kappa, \sigma)$-VLTC. Further assume that $|N_1| \leq \frac{1}{2^\delta'}, |N_m| = n$, and for every $1 < i \leq m$, $|N_i| = 2|N_{i-1}|$. Then, there exists an $(mq + 1, \delta, \varepsilon)$-RLCC $C \subseteq \mathbb{F}^n$ with rate

$$\rho \geq 1 - \sum_{i=1}^{m} (1 - \rho_i),$$

where

$$\varepsilon \leq 1 + \sigma - \frac{\kappa \delta'}{2}.$$ 

Moreover, if the codes $C_1, \ldots, C_m$ are explicit, then so is the resulting code $C$.

Proof. We start by describing how the code $C$ is constructed.

The code construction. Let $P_1, \ldots, P_m$ be an arbitrary fixed sequence of partitions of $[n]$, satisfying that for every $i \in [m]$, $P_i$ has $2^{m-i}$ parts denoted $\{B_{1_i}^m, \ldots, B_{2^{m-i}}^m\}$, and that for every $1 < i \leq m$, $P_{i-1}$ is a sub-partition of $P_i$ (that is, for every $B \in P_{i-1}$, there exists $B' \in P_i$ such that $B \subseteq B'$). For every $i \in [m]$ and $B \in P_i$ let $f_{i,B} : B \rightarrow N_i$ be an arbitrary bijection, and define $C_{i,B} = \{c \circ f_{i,B} \mid c \in C_i\}$. Finally, define the code

$$C = \{w \in \mathbb{F}^n \mid \forall i \in [m], B \in P_i : w_B \in C_{i,B}\}.$$ 

The moreover part of the proof readily follows. The efficiency of the corrector will be self evident as well once the corrector is presented.

Rate analysis. For every $i \in [m]$ and $B \in P_i$, the number of linear constraints required to impose that $w_B \in C_{i,B}$ is at most $(1 - \rho_i)|N_i|$. Therefore the total number of constraints in the definition of the code $C$ is bounded above by

$$\sum_{i=1}^{m} |P_i|(1 - \rho_i)|N_i| = \sum_{i=1}^{m} n_i(1 - \rho_i),$$

which establishes the lower bound on the rate of $C$. 

13
The corrector. We turn to describe a corrector $\text{Cor} : \mathbb{F}^n \times [n] \to \mathbb{F} \cup \{ \bot \}$ for $C$. As for every $i \in [m]$, $C_i$ is a VLTC, it is immediate that so is $C_{i,B}$ for every $B \in P_i$, with the same parameters as $C_i$. The local tester for $C_{i,B}$ that is induced in the natural way from the local tester for $C_i$ is denoted $\text{Tes}_{i,B} : \mathbb{F}^B \to \{ 0, \bot \}$.

Let $w \in \mathbb{F}^n$ and $j \in [n]$. Let $r_1 = r_1(j), \ldots, r_m = r_m(j)$ be the indices of blocks within the partitions $P_1, \ldots, P_m$ such that $j \in B_{r_1}^1 \subseteq \cdots \subseteq B_{r_m}^m$. The corrector $\text{Cor}(w, j)$ proceeds as follows:

1. For $i = 1, \ldots, m$, simulate $\text{Tes}_{i,B_{r_i}^i}(w_{B_{r_i}^i})$. If any of the simulations outputted $\bot$, halt and output $\bot$.
2. Output $w_j$.

Query analysis. As $\text{Cor}$ simulates $m$ testers, each with query complexity $q$, the query complexity is $qm + 1$, accounting also for querying $w_j$.

Correctness. Clearly, if $w$ is a codeword of $C$ then $w_{B_{r_i}^i} \in C_{i,B_{r_i}^i}$, and so $\text{Tes}_{i,B_{r_i}^i}(w_{B_{r_i}^i}) = \circ$ with certainty. Therefore, $\text{Cor}(w, j) = w_j$ with certainty, as required. Assume that $w \in \mathbb{F}^n$ is such that $\text{Dist}(w, c) \leq \delta n$ for $c \in C$. Since $\text{Cor}(w, j)$ always either outputs $\bot$ or $w_j$, it suffices to show that if $w_j \neq c_j$ then the corrector outputs $\bot$ with high enough probability. Towards this end, assume $w_j \neq c_j$, and hence $w_{B_{r_i}^i} \neq c_{B_{r_i}^i}$, and further note that $w_{B_{r_m}^m} = w$ and $c_{B_{r_m}^m} = c$. For every $i \in [m - 1]$ define $\delta_i = \delta'$, and let $\delta_m = \delta$. Since, per our assumption, $|N_1| \leq \frac{1}{\delta'}$, we have that

$$\text{RelDist} \left( w_{B_{r_1}^1}, c_{B_{r_1}^1} \right) \geq \delta' = \delta_1,$$

whereas

$$\text{RelDist} \left( w_{B_{r_m}^m}, c_{B_{r_m}^m} \right) \leq \delta = \delta_m.$$

Let $\iota \in \{ 2, 3, \ldots, m \}$ be any index satisfying that

$$\text{RelDist} \left( w_{B_{r_{\iota - 1}}^{\iota - 1}}, c_{B_{r_{\iota - 1}}^{\iota - 1}} \right) \geq \delta_{\iota - 1}$$

$$\text{RelDist} \left( w_{B_{r_{\iota}}^{\iota}}, c_{B_{r_{\iota}}^{\iota}} \right) \leq \delta_{\iota}.$$ 

By the above account, $\iota$ is well-defined. Since $B_{r_{\iota - 1}}^{\iota - 1} \subseteq B_{r_{\iota}}^{\iota}$ and $|B_{r_{\iota - 1}}^{\iota - 1}| = \frac{1}{2}|B_{r_{\iota}}^{\iota}|$, we have that

$$\frac{\delta_{\iota - 1}}{2} \leq \text{RelDist} \left( w_{B_{r_{\iota}}^{\iota}}, c_{B_{r_{\iota}}^{\iota}} \right) \leq \delta_{\iota}.$$ 

As $\text{Tes}_{i,B_{r_i}^i}$ is a local tester for the $(q, \delta_i, \kappa, \sigma)$-VLTC $C_{i,B_{r_i}^i}$ and since $c_{B_{r_i}^i} \in C_{i,B_{r_i}^i}$, it holds that $\text{Tes}_{i,B_{r_i}^i}(w_{B_{r_i}^i})$ outputs $\bot$ with probability at least

$$\frac{\kappa \delta_{\iota - 1}}{2} - \sigma = \frac{\kappa \delta'}{2} - \sigma.$$ 

14
Thus, Cor\((w, j)\) outputs \(\bot\) with probability at least \(\frac{\kappa \delta'}{2} - \sigma\), as required. \(\square\)

We are now ready to prove our main theorem.

**Theorem 5.2.** For every finite field \(\mathbb{F}\), \(n\) which is a power of two, and \(\delta > 0\), there exists an explicit \((q, \delta, \frac{1}{3})\)-RLCC \(C \subseteq \mathbb{F}[n]\) with query complexity

\[
q = (\log n)^{2+o(1)},
\]

and rate

\[
\rho = 1 - \delta \cdot 2^{O\left(\left(\log \log \frac{1}{\delta}\right)^2\right)} - o(1).
\]

**Proof.** Write \(n = 2^r\), and let \(\ell = r \log r = \log n \cdot \log \log n\). Due to the claimed tradeoff between the rate and the correction radius, we may as well assume that \(\delta \geq \frac{1}{2^{|\log \ell|}}\). We proceed to describe the sequence of expander codes that we will use, which consists of \(s = r - \lceil \log \ell \rceil + 1\) codes. For every \(i \in [s]\), the block-length of the \(i\)-th code is

\[
n_i = 2^{[\log \ell] + i - 1}.
\]

Note that, in particular, \(n_s = n\). Further, the number of linear constraints defining the \(i\)-th code is \(m_i = 2^{i-1}\) for \(i \in [s-1]\), whereas for the \(s\)-th code,

\[
m_s = 2^{r - \left[\log \frac{1}{\delta} + \log \beta - c_0 \log^3 (\log \frac{1}{\delta})\right]},
\]

where \(\beta\) is the constant from Theorem 3.5.

Invoking Theorem 3.5, for every \(i \in [s]\) let \(G_i = (L_i, R_i, E_i)\) be a \(d_i\)-left-regular bipartite graph with \(|L_i| = n_i\) and \(|R_i| = m_i\) which is a \((\delta_i, \alpha)\)-unique-neighbor expander for \(\alpha = \Omega(1)\), such for \(i \in [s-1]\),

\[
d_i \leq 2^{c_0 \log^4 \left(\left(\log \ell\right)\right)} \leq d, \quad \delta_i \geq \frac{\beta}{d} \cdot 2^{\left[\log \ell\right]} \geq \delta',
\]

and

\[
d_s \leq 2^{c_0 \log^4 \left(\frac{1}{\delta}\right)} \leq d, \quad \delta_s \geq \frac{\beta}{d_s 2^{\left[\log (1/\delta) + \log (\beta) - c_0 \log^3 \left(\log (1/\delta)\right)\right]}} \geq \delta.
\]

The sequence of codes is defined by setting, for every \(i \in [s]\), \(C_i = \text{EC}(G_i)\).

We turn to address the VLTC-ness of \(C_1, \ldots, C_s\). Set \(b = \frac{4d}{\alpha \beta}\). By Lemma 4.2, for every \(i \in [s-1]\), \(C_i\) is a

\[
\left(\frac{bd_i n_i}{m_i} \leq bd 2^{\left[\log \ell\right]}, \delta', \alpha 2^{\left[\log \ell\right]}, \frac{1}{b}\right)\)-VLTC,
and $C_s$ is a

$$\left( \frac{bd_sn_s}{m_s} \leq bd2^{\lceil \log \ell \rceil}, \delta, \alpha 2^{\lceil \log \ell \rceil}, \frac{1}{b} \right) \text{-VLTC}.$$ 

We can now invoke Proposition 5.1 (indeed, the proposition’s prerequisites are met, i.e., $n_1 = 2^{\lceil \log \ell \rceil} < \frac{1}{\delta}$, $n_i = 2n_{i-1}$, and $n_s = n$) to obtain a code $C \subseteq F^n$ which is an

$$\left( sbd2^{\lceil \log \ell \rceil} + 1, \delta, 1 + \frac{1}{b} - \frac{1}{2} \alpha 2^{\lceil \log \ell \rceil} \beta' \right) \text{-RLCC}.$$ 

By our choice of parameters, the error guarantee of the corrector for $C$ is

$$1 + \frac{1}{b} - \frac{1}{2} \alpha 2^{\lceil \log \ell \rceil} \beta' = 1 + \frac{1}{b} - \frac{\alpha \beta}{2d}$$

$$= 1 + \frac{\alpha \beta}{4d} - \frac{\alpha \beta}{2d}$$

$$= 1 - \frac{\alpha \beta}{4d}.$$ 

To decrease the error to $\frac{1}{3}$, we apply Claim 3.3 with $h = O\left(\frac{4d}{\alpha \beta} \right) = O(d)$, and get that $C$ is also a $(q, \delta, \frac{1}{3})$-RLCC for

$$q = O\left(sbd^2\ell\right) = O\left(sd^2\ell\right).$$

Recall that $s \leq \log n$, $d = 2^{O((\log \log n)^3)} = 2^{O((\log \log \log n)^3)}$ and $\ell = O(\log n \cdot \log \log n)$. Therefore,

$$q = \log^2 n \cdot 2^{O((\log \log \log n)^3)} = (\log n)^{2+o(1)}.$$ 

Lastly, as the rate $\rho_i$ of every code $C_i$ in the sequence is at least $1 - \frac{m_i}{n_i}$, Proposition 5.1 implies that the rate $\rho$ of $C$ is lower bounded by

$$\rho \geq 1 - \sum_{i=1}^{s} \left( \frac{m_i}{n_i} \right)$$

$$= 1 - (s - 1) \frac{1}{2^{\lceil \log \ell \rceil}} - \frac{1}{2^{\lceil \log \frac{1}{\delta} + \log \beta - c_0 \log^2(\log \frac{1}{\delta}) \rceil}}$$

$$= 1 - O\left(\frac{s}{\ell}\right) - \delta \cdot 2^{O(\log^3(\log \frac{1}{\delta}))}$$

$$= 1 - O\left(\frac{r}{\ell}\right) - \delta \cdot 2^{O(\log^3(\log \frac{1}{\delta}))}$$

$$= 1 - \delta \cdot 2^{O(\log^3(\log \frac{1}{\delta}))} - o(1).$$

This concludes the proof. 

\[ \square \]
References


[CRTS23] Itay Cohen, Roy Roth, and Amnon Ta-Shma. HDX condensers. In Electronic Colloquium on Computational Complexity (ECCC), number 090, 2023.


