Electronic Colloquium on Computational Complexity, Report No. 208 (2025)

Relaxed vs. Full Local Decodability with Few Queries: Equivalence and Separations for Linear Codes

Elena Grigorescu*
elena-g@uwaterloo.ca
University of Waterloo

Vinayak M. Kumar†
vmkumar@cs.utexas.edu
University of Texas at Austin

Peter Manohar‡
pmanohar@ias.edu
The Institute for Advanced Study

Geoffrey Mon§
gmon@cs.utexas.edu
University of Texas at Austin

November 25, 2025

Abstract

A locally decodable code (LDC) $C\colon \{0,1\}^k \to \{0,1\}^n$ is an error-correcting code that allows one to recover any bit of the original message with good probability while only reading a small number of bits from a corrupted codeword. A relaxed locally decodable code (RLDC) is a weaker notion where the decoder is additionally allowed to abort and output a special symbol $\perp$ if it detects an error. For a large constant number of queries q, there is a large gap between the blocklength n of the best q-query LDC and the best q-query RLDC. Existing constructions of RLDCs achieve polynomial length $n = k^{1+O(1/q)}$ [BGH+04, GRR18, CGS20, AS21, Gol24b], while the best-known q-LDCs only achieve subexponential length $n = 2^{k^{o(1)}}$ [Yek08, Efr09]. On the other hand, for $q = 2$, it is known that RLDCs and LDCs are equivalent [BBC+23]. We thus ask the question: what is the smallest q such that there exists a q-RLDC that is not a q-LDC?

In this work, we show that any linear 3-query RLDC is in fact a 3-LDC, i.e., linear RLDCs and LDCs are equivalent at 3 queries. More generally, we show for any constant q, there is a soundness error threshold $s(q)$ such that any linear q-RLDC with soundness error below this threshold must be a q-LDC. This implies that linear RLDCs cannot have "strong soundness" — a stricter condition satisfied by linear LDCs that says the soundness error is proportional to the fraction of errors in the corrupted codeword — unless they are simply LDCs.

In addition, we give simple constructions of linear 15-query RLDCs that are not q-LDCs for any constant q, showing that for $q = 15$, linear RLDCs and LDCs are not equivalent.

We also prove nearly identical results for locally correctable codes and their corresponding relaxed counterpart.

*Supported in part by NSF Grant CCF-2228814 while at Purdue University.

†Supported by NSF Grant CCF-2312573, a Simons Investigator Award (#409864, David Zuckerman), a Jane Street Fellowship, and a UT Austin Dean's Prestigious Fellowship Supplement.

‡Supported by NSF Grant DMS-2424441.

§Supported by NSF Grant CCF-2312573, an NSF Graduate Research Fellowship (DGE-2137420), and a UT Austin Dean's Prestigious Fellowship Supplement.

ISSN 1433-8092

Contents

1 Introduction

A binary locally decodable code (LDC) $C\colon \{0,1\}^k \to \{0,1\}^n$ is an error-correcting code that admits a local decoding algorithm with the following guarantee: when given access to a corrupted codeword y obtained by corrupting $x = C(b)$ in a constant fraction of coordinates, the local decoder is able to recover any bit b_i of the chosen message b with good probability while only reading a small (say, constant) number of bits from y. Although they were first formally defined in [KT00], locally decodable codes were implicitly used in the proof of the PCP theorem [ALM+98, AS98], and have since found numerous applications to, e.g., private information retrieval, hardness amplification, probabilistically checkable proofs, self-correction, fault-tolerant circuits and data structures (e.g., [BFLS91, LFKN92, BLR93, BK95, CKGS98, IK99, BIW10, CGW13, ALRW17]). We refer the reader to the surveys of [Tre04, Dvi12, Yek12] for more details.

The central question in the study of LDCs is to understand the length n of the best locally decodable code that can tolerate a small constant fraction of errors as a function of k, the length of the message, and q, the number of queries of the decoder. Following [KT00], there has been a long line of work on both constructing and proving lower bounds for locally decodable codes, with a particular focus on the constant query regime when $q = O(1)$.

A simple observation (see [KT00, Section 3.2]) shows that it is not possible to construct 1-query locally decodable codes. For $q = 2$, the Hadamard code gives a 2-LDC of blocklength $n = 2^k$, and this is optimal up to constant factors in the exponent: the works of [KW04, GKST06] show that $n \geq 2^{\Omega(k)}$ for any 2-LDC. More generally, the Reed–Muller code — a generalization of the Hadamard code (which are evaluations of linear functions) to polynomials of larger degree — gives a construction of a q-LDC of length $n = 2^{O(k^{1/(q-1)})}$. However, for $q \geq 3$, this can be improved: the matching vector codes of [Yek08, Efr09, DGY11] give constructions of q-LDCs of subexponential, but still superpolynomial, blocklength $n = 2^{k^{o(1)}}$ for any constant $q \geq 3$.[1] But, unlike the case of $q = 2$, we are far from understanding whether these codes are optimal or not: our best lower bound is a polynomial lower bound of $n \geq \tilde{\Omega}(k^{q/(q-2)})$ for any $q \geq 3$ [KW04, AGKM23, HKM+24, BHKL24, JM24].

This gap between subexponential constructions and polynomial lower bounds for LDCs has led to the definition of a class of codes with weaker local decoding properties called relaxed locally decodable codes (RLDCs), for which much better constructions are known. Introduced in [BGH+04], a relaxed LDC is no longer required to output the correct message bit b_i with good probability, and instead may output a special symbol $\perp$ to signify that the decoder has detected an error.[2] Unlike standard locally decodable codes, for large enough constant q one can construct q-query RLDCs with a polynomial blocklength of $n = k^{1+O(1/q)}$ [BGH+04, GRR18, CGS20, AS21, Gol24b], and the works of [GL20, DGL21, Gol23] prove a near-matching lower bound of $n \geq k^{1+\Omega(1/q^2)}$. However, somewhat curiously, for the specific case of $q = 2$ the work of [BBC+23] proves an exponential lower bound of $n \geq 2^{\Omega(k)}$. That is, up to constant factors in the exponent, the best 2-RLDC is a 2-LDC, namely the Hadamard code.

[1] More precisely, the length of these codes is $n = 2^{2^{(\log k)^{\varepsilon(q)}}}$ for some constant $\varepsilon(q) \approx 1/\log_2 q < 1$ that depends on q.

[2] To prevent the trivial decoder that always outputs $\perp$ from satisfying the definition, one requires that the decoder outputs the correct bit b_i with good probability when given access to an uncorrupted codeword $x = C(b)$.

To summarize: for RLDCs, there exists a large constant q such that there are q-RLDCs of length $n = \text{poly}(k)$, whereas the best 2-RLDC has length $n = 2^{\Omega(k)}$. The work of [BBC+23] thus raises the following interesting question (mentioned explicitly in [BBC+23, Section 2]):

Question 1.1 ([BBC+23])**.** What is the threshold q where the optimal blocklength of a q-RLDC "transitions" from superpolynomial in k to polynomial in k?

In this work, we investigate Question 1.1. However, there is an obvious barrier to answering Question 1.1, coming from our lack of understanding of the analogous question for LDCs. For example, if it is the case that the "transition threshold" is above $q = 3$, then to prove this one would need to prove that there are no 3-RLDCs of polynomial length. In particular, this would also imply that there are no 3-LDCs (unrelaxed) of polynomial length as well. However, the best 3-LDC lower bound is only $n \geq \tilde{\Omega}(k^3)$ [AGKM23], and improving this (or showing that it is tight) is a well-studied and challenging open question.

To avoid such issues, we instead reinterpret the result of [BBC+23] as follows. Not only do they prove that any 2-RLDC has $n \geq 2^{\Omega(k)}$, they in fact prove this result by showing that any 2-RLDC *is* a 2-LDC, and then they apply the exponential lower bound of [KW04, GKST06] for 2-LDCs. Thus, for $q = 2$, RLDCs and LDCs are equivalent. On the other hand, for large constant q there is a large gap between the best constructions of q-RLDCs (which have length $n = k^{1+O(1/q)}$ [BGH+04, AS21, Gol24b]) and q-LDCs (which have length $n = 2^{k^{o(1)}}$ [Yek08, Efr09, DGY11]), giving evidence that for a large enough constant q, RLDCs and LDCs are not equivalent. We thus pose the following question:

Question 1.2. What is the smallest r where (1) for every $q < r$, every q-RLDC is a q-LDC, and (2) there is an r-RLDC that is *not* an r-LDC?

Our main results show that the threshold r in Question 1.2 is between 4 and 15 for the case of linear codes. In fact, we show this for a "stronger" version of Question 1.2 where the r-RLDC in Item (2) is not only not an r-LDC, but also not a t-LDC for any constant t.

Locally correctable and relaxed locally correctable codes. The above discussion is for locally decodable codes and their relaxed variant. One may ask the same questions for locally correctable codes (LCCs) and their relaxed variant, as LCCs are a closely related notion to LDCs that are defined in a near-identical way. The difference between an LCC and an LDC is that a locally correctable code requires that the decoder is able to self-correct bits of the *codeword*, whereas LDCs only need to correct bits of the *message*. One can show that any LCC is in fact an LDC as well, so LCCs are a stronger notion.[3] And, similarly to LDCs, one can define a relaxed notion of LCCs where the decoder is allowed to output a special error symbol $\perp$.

For LCCs, the state-of-the-art constructions and lower bounds are quite different compared to LDCs. For $q = 2$, these notions are equivalent, and as shown by [KW04, GKST06], the Hadamard

[3]This is fairly straightforward to show for linear codes, as without loss of generality, by changing bases one can make any linear code systematic, i.e., the first k bits of any codeword $x = C(b)$ is simply the message b. For nonlinear codes, this can also be done with more effort, see [BGT17, Appendix A].

code with $n = 2^k$ is also an optimal 2-LCC. For $q \geq 3$, however, the best construction of q-LCCs remains the folklore construction from Reed–Muller codes, which achieves a length of $n = 2^{O(k^{1/(q-1)})}$. This is unlike the case of LDCs, where the constructions of subexponential length coming from matching vector codes are much better than Reed–Muller codes when $q \geq 3$.

We also have stronger lower bounds for LCCs as compared to LDCs. Namely, for $q = 3$, the work of [KM24a] and follow-up works of [AG24, Yan24, KM24b] prove exponential lower bounds for 3-LCCs: the current best results are $n \geq 2^{\Omega(\sqrt{k/\log k})}$ for linear codes [AG24] and $n \geq 2^{\Omega(k^{1/5})}$ for nonlinear codes [KM24b]. For comparison, recall that the best 3-LDC lower bound remains $n \geq \tilde{\Omega}(k^3)$. Furthermore, for any odd constant $q \geq 5$, one can show a lower bound of $n \geq \tilde{\Omega}(k^{(q-1)/(q-3)})$ for linear q-LCCs [AG24], which is better than the best q-LDC lower bound by a small polynomial factor in k.

For relaxed LCCs (RLCCs), the current best results are identical to RLDCs. Namely, for large enough constant q, the work of [AS21] constructs q-query RLCCs of length[4] $n = k^{1+O(1/q)}$ and the lower bounds of [GL20, DGL21, Gol23] again prove a near-matching lower bound of $n \geq k^{1+\Omega(1/q^2)}$. And, for $q = 2$, [BBC+23] proves an exponential lower bound for 2-RLCCs of $n \geq 2^{\Omega(k)}$.[5]

We can thus also ask Question 1.2 for LCCs and RLCCs.

Question 1.3 (Question 1.2 for LCCs)**.** What is the smallest r where (1) for every $q < r$, every q-RLCC is a q-LCC, and (2) there is an r-RLCC that *not* an r-LCC?

We also prove results for LCCs/RLCCs similar to the case of LDCs/RLDCs. In particular, we show that the threshold r in Question 1.3 is between 4 and 41 for the case of linear codes.

1.1 Our results

Before we state our results, let us formally define LDCs and RLDCs (LCCs and RLCCs are defined analogously, see Definitions 3.7 and 3.12). For two strings $x, y \in \{0,1\}^n$, we let $\Delta(x, y)$ denote the Hamming distance between x and y, i.e., the number of indices $j \in [n]$ where $x_j \neq y_j$.

Definition 1.4 (Binary locally decodable codes; see Definition 3.6)**.** A code $C \colon \{0,1\}^k \to \{0,1\}^n$ is a (q, δ, c, s)-LDC if there exists a randomized decoding algorithm $\mathrm{Dec}(\cdot)$ with the following properties. The algorithm $\mathrm{Dec}(\cdot)$ is given oracle access to a string $y \in \{0,1\}^n$, takes an index $i \in [k]$ as input, and outputs a bit in $\{0,1\}$ with the following guarantees:

(1) (q-queries) for any y and i, the algorithm $\mathrm{Dec}^y(i)$ reads at most q indices of y,

(2) (c-completeness) for all $b \in \{0,1\}^k$ and $i \in [k]$, $\Pr[\mathrm{Dec}^{C(b)}(i) = b_i] \geq c$, and

(3) ((δ, s)-soundness error) for all $b \in \{0,1\}^k$, $i \in [k]$, and all $y \in \{0,1\}^n$ with $\Delta(y, C(b)) \leq \delta n$, $\Pr[\mathrm{Dec}^y(i) \neq b_i] \leq s$.

[4] The works of [BGH+04, Gol24b] construct q-RLDCs with the same parameters, but their codes are not RLCCs.

[5] Unlike the case of RLDCs/LDCs, the work of [BBC+23] does not prove a 2-RLCC to 2-LCC reduction. However, since any 2-RLCC is a 2-RLDC, the exponential lower bound still applies.

Definition 1.5 (Binary relaxed locally decodable codes; see Definition 3.11)**.** A code $C\colon \{0,1\}^k \to \{0,1\}^n$ is a (q,δ,c,s)-RLDC if there exists a randomized decoding algorithm $\mathrm{Dec}(\cdot)$ with the following properties. The algorithm $\mathrm{Dec}(\cdot)$ is given oracle access to a string $y \in \{0,1\}^n$, takes an index $i \in [k]$ as input, and outputs either a bit in $\{0,1\}$ or a special symbol $\perp$ with the following guarantees:

(1) (q-queries) for any y and i, the algorithm $\mathrm{Dec}^y(i)$ reads at most q indices of y,

(2) (c-completeness) for all $b \in \{0,1\}^k$ and $i \in [k]$, $\Pr[\mathrm{Dec}^{C(b)}(i) = b_i] \geq c$, and

(3) ((δ,s)-relaxed soundness error) for all $b \in \{0,1\}^k$, $i \in [k]$, and all $y \in \{0,1\}^n$ with $\Delta(y, C(b)) \leq \delta n$, $\Pr[\mathrm{Dec}^y(i) \notin \{b_i, \perp\}] \leq s$.

The standard definition of an RLDC sets parameters as follows: $\delta = O(1)$ (constant fraction of errors), $c = 1$ (perfect completeness), and $s = 1/3$. That is, in the presence of a constant fraction of errors, the RLDC decoder outputs either the correct bit or a special error symbol $\perp$ with probability at least $2/3$. In the standard definition of an LDC, it is also common to set $s = 1/2 - \varepsilon$ for a small constant ε, meaning that in the presence of a constant fraction of errors, the LDC decoder outputs the correct bit with probability at least $1/2 + \varepsilon$.

Our main results. Our first result shows that any linear 3-RLDC with soundness error $1/2 - \eta$ is a 3-LDC, and thus the threshold in Question 1.2 must be at least 4 for linear codes.

Theorem 1. *Let $C\colon \{0,1\}^k \to \{0,1\}^n$ be a linear $(3,\delta,1,\frac{1}{2}-\eta)$-RLDC with a possibly adaptive decoder. Then, for any $\varepsilon > 0$, C is a linear $(3, 2\eta\delta\varepsilon/3, 1, \varepsilon)$-LDC. In particular, if C is a linear $(3,\Theta(1),1,\frac{1}{3})$-RLDC, then C is a linear $(3,\Theta(1),1,\frac{1}{3})$-LDC. Furthermore, the same statement holds for RLCCs/LCCs.*

By combining Theorem 1 with the best known lower bounds for linear 3-LDCs [AGKM23, HKM+24, BHKL24, JM24] and linear 3-LCCs [AG24], we obtain the following new lower bounds for 3-RLDCs and 3-RLCCs.

Corollary 1.6. *Let $C\colon \{0,1\}^k \to \{0,1\}^n$ be a linear code. Then, the following hold:*

(1) If C is a $(3,\delta,1,\frac{1}{2}-\varepsilon)$-RLDC with a possibly adaptive decoder, then $n \geq \Omega_{\delta,\varepsilon}((k/\log k)^3)$.

(2) If C is a $(3,\delta,1,\frac{1}{2}-\varepsilon)$-RLCC with a possibly adaptive decoder, then $n \geq 2^{\Omega_{\delta,\varepsilon}(\sqrt{k/\log k})}$.

For comparison, the prior best lower bound for 3-RLDCs or 3-RLCCs came from the works [GL20, DGL21, Gol23], which prove a lower bound of $n \geq k^{1+\Omega(1/q^2)}$ for any q-RLDC/RLCC, where the constant in the $\Omega(\cdot)$ is smaller than $1/10$. Thus, Corollary 1.6 gives a stronger lower bound for 3-RLDCs, and a substantially stronger lower bound for 3-RLCCs.

The lower bounds in Corollary 1.6 require the RLDC/RLCC to have a soundness error of $1/2 - \varepsilon$. However, for RLDCs/RLCCs (unlike LDCs/LCCs), any soundness error of $1 - \varepsilon$ can be amplified to $1/2 - \varepsilon$ via sequential repetition (see Observation 1.9), although this does increase the number of queries in the RLDC by a constant factor. In this way, one could view the requirement on the

soundness error in Corollary 1.6 as a mild limitation, although we note that the lower bound for 2-RLDCs in [BBC+23] also has the same requirement.

Theorem 1 is a corollary of the following more general statement that we show. There is a soundness threshold $s(q)$, a function of q, such that *any* linear q-RLDC with soundness error $s \leq (1-\alpha)s(q)$ must be a linear q-LDC. That is, if a linear q-RLDC has soundness error better than $s(q)$, then the "reason" is that the q-RLDC is in fact a q-LDC. For 3-RLDCs, this threshold $s(q)$ is $1/2$, which implies Theorem 1.

Theorem 2 (Soundness error threshold for q-RLDCs; binary case of Theorem 4.1)**.** *Let $s(q) := 2^{-\lfloor q/2 \rfloor}$. Let $C\colon \{0,1\}^k \to \{0,1\}^n$ be a linear $(q, \delta, 1, s)$-RLDC with a possibly adaptive decoder, where $s \leq (1-\alpha)s(q)$. Then, for any $\varepsilon > 0$, C is a linear $(q, \alpha\delta\varepsilon/q, 1, \varepsilon)$-LDC. Furthermore, the same result holds for RLCCs/LCCs.*

Theorem 2 shows that the answer to Question 1.2 depends on the soundness error s that we choose in Definition 1.5.[6] The standard definition of RLDCs typically chooses $s = 1/3$ (see, e.g., [BGH+04, GL20]), which is a constant smaller than $\frac{1}{2}$. If one wishes to have soundness error strictly less than, say, $s = 1/8$, then Theorem 2 implies that any q-RLDC that is additionally not a q-LDC must have $q \geq 8$.

More generally, Theorem 2 shows a qualitative difference between LDCs and RLDCs (that are not LDCs): the only way for a linear RLDC to have *strong soundness*, a property satisfied by constant query linear LDCs, is for it to already be an LDC. Strong soundness is a natural property, typically desired in local testing or decoding algorithms, that intuitively says that corrupted codewords y with fewer errors are decoded more successfully. More formally, strong soundness replaces Item (3) in Definitions 1.4 and 1.5 with the following corresponding condition:

Definition 1.7 (Strong soundness for LDCs and RLDCs)**.**

(1) (δ'-strong LDC soundness) for all $b \in \{0,1\}^k$, $i \in [k]$, and all $y \in \{0,1\}^n$, $\Pr[\mathrm{Dec}^y(i) \neq b_i] \leq \Delta(y, C(b))/\delta'$.

(2) (δ'-strong RLDC soundness) for all $b \in \{0,1\}^k$, $i \in [k]$, and all $y \in \{0,1\}^n$, $\Pr[\mathrm{Dec}^y(i) \notin \{b_i, \perp\}] \leq \Delta(y, C(b))/\delta'$.

Definition 1.7 says that the probability that the decoder is incorrect is proportional to the number of errors in y. The connection established in [KT00] between linear constant q-LDCs and "smooth decoders" implies that any $(q, \delta, 1, 1/2 - \varepsilon)$-LDC (Definition 1.4) satisfies Definition 1.7 for $\delta' = \delta/q$ (see Appendix A). On the other hand, Theorem 2 implies that no such analogous statement can hold for RLDCs unless all RLDCs are LDCs, which is known to be false (see, e.g., Theorem 4). Formally, we have the following corollary of Theorem 2.

Corollary 1.8 (RLDCs with Strong Soundness are LDCs)**.** *Let $C\colon \{0,1\}^k \to \{0,1\}^n$ be a linear q-RLDC with perfect completeness and δ'-strong soundness (Definition 1.7). Then for any $\varepsilon > 0$, C is a linear $(q, s(q)\varepsilon\delta'/4q, 1, \varepsilon)$-LDC. Furthermore, the same result holds for RLCCs/LCCs.*

[6] Technically, to deduce that this depends on the soundness error one also needs to exhibit a q-RLDC (with soundness error $> s(q)$) that is also not a q-LDC. Existing constructions of RLDCs ([BGH+04, GRR18, CGS20, AS21, Gol24b]) should satisfy this condition, and in this paper we give an RLDC with this property (Theorem 4).

At first glance, the exponential dependence on q in $s(q)$ in Theorem 2 may appear to be weak. However, a simple sequential repetition of the decoder, combined with a constant query RLDC that is not an q-LDC for any constant q, shows that the exponential dependence is necessary.

Observation 1.9. Let $C\colon \{0,1\}^k \to \{0,1\}^n$ be a $(q,\delta,1,\varepsilon)$-RLDC with a possibly adaptive decoder. Then, for any integer $t \geq 1$, C is a $(qt,\delta,1,\varepsilon^t)$-RLDC. The same statement holds for RLCCs.

Observation 1.9 implies that if the threshold $s(q)$ in Theorem 2 did not decay exponentially with q, we could take any $(q,\delta,1,\varepsilon)$-RLDC (where q,δ,ε are constant) and then choose t to be a constant so that $\varepsilon^t < s(qt)$ (which exists if $s(qt)$ does not decay exponentially), and this would show that the code is also a constant query LDC. However, as we will show (Theorem 4), there is a code that is an $O(1)$-RLDC but not a q-LDC for any constant q. We thus conclude that $s(q)$ must decay exponentially in q.

The proof of Observation 1.9 is simple, and is included in Section 3.4.

Extensions of Theorem 2. Theorem 2 makes two assumptions on the code: it assumes that C is linear, and that it has perfect completeness. As we will discuss in Section 2.2, our proof makes heavy use of the linearity of C, and it is not clear how to remove this assumption from Theorem 2. However, we can extend Theorem 2 to codes with imperfect completeness via our next theorem, which shows how to convert any linear RLDC with imperfect completeness to a linear RLDC with perfect completeness.

Theorem 3 (Binary case of Theorem 5.2)**.** *Let $C\colon \{0,1\}^k \to \{0,1\}^n$ be a linear $(q,\delta,1-\varepsilon,s)$-RLDC with a possibly adaptive decoder. Then, C is a linear $(q,\delta,1,s+3\varepsilon)$-RLDC with a nonadaptive decoder. Furthermore, the same result holds for RLCCs/LCCs.*

Combining Theorems 2 and 3, we obtain the following corollary, which extends Theorem 2 to linear RLDCs with imperfect completeness.

Corollary 1.10. *Let $s(q) := 2^{-\lfloor q/2 \rfloor}$. Let $C\colon \{0,1\}^k \to \{0,1\}^n$ be a linear $(q,\delta,1-\eta,s)$-RLDC with a possibly adaptive decoder, where $s \leq (1-\alpha)s(q)-3\eta$. Then, for any $\varepsilon > 0$, C is a linear $(q,\alpha\delta\varepsilon/q,1,\varepsilon)$-LDC. Furthermore, the same result holds for RLCCs/LCCs.*

Theorem 3 is a query-preserving version of a recent theorem of [Gol24a], which shows a similar result but requires an extra query. Namely, [Gol24a] shows how to convert any linear q-RLDC with an adaptive decoder and imperfect completeness into a $(q+1)$-RLDC with a nonadaptive decoder and perfect completeness. This extra query made by [Gol24a] is significant for us, as our soundness threshold $s(q) = 2^{-\lfloor q/2 \rfloor}$ satisfies $s(3) = 1/2$ but $s(4) = 1/4$, and so the result of [Gol24a] only allows us to show that a 3-RLDC (RLCC) with imperfect completeness and soundness error $1/4$ (which is smaller than the standard setting of $1/3$) is a 4-LDC (LCC). For the case of RLCCs, this extra query is very significant, as the best lower bound for linear 4-LCCs is only $n \geq \tilde{\Omega}(k^2)$ [KW04], whereas the best lower bound for (binary) linear 3-LCCs is $n \geq 2^{\Omega(\sqrt{k/\log k})}$ [AG24].

Larger fields. It is straightforward to extend Theorem 2 to larger fields $\mathbb{F}$ by replacing the function $s(q)$ with $s_{\mathbb{F}}(q) := |\mathbb{F}|^{-\lfloor q/2 \rfloor}$, and we do this when we prove Theorem 2 in Section 4. In Section 7, we extend Theorem 2 to larger fields *without* any field-dependent loss in $s(q)$ for the case of $q = 2$.

We prove Theorems 1 and 2 in Section 4 and Theorem 3 in Section 5.

Constructions of RLDCs/RLCCs with small queries that are not LDCs. To complement Theorem 2 and provide a partial answer to Questions 1.2 and 1.3, we give a simple construction of RLDCs and RLCCs with small queries that are not constant query LDCs for any constant. To our knowledge, these are the first RLDC constructions to achieve constant queries for an explicit small constant.

Theorem 4 (Constructions of constant query RLDCs/RLCCs that are not LDCs)**.** *For every k, there is a linear code $C\colon \{0,1\}^k \to \{0,1\}^n$ where $n = k^{O(\log\log k)}$ such that C is a $(15, \delta, 1, 1/3)$-RLDC and a $(41, \delta, 1, 1/2-\varepsilon)$-RLCC for a small constant $\varepsilon > 0$. Furthermore, C is not a $(O(\log k), O(n^{-1/3}), 1, 1/2-\varepsilon)$-LDC for any $\varepsilon > 0$ (including subconstant ε).*

The blocklength of Theorem 4 is worse than that of prior constructions, as it is slightly superpolynomial in k rather than $\mathrm{poly}(k)$. However, the importance of Theorem 4 is that (1) the number of queries made by the code is an explicit small constant, and (2) we can prove that the code is not a constant query LDC. Thus, when combined with Corollary 1.10, Theorem 4 shows that the threshold in Question 1.2 is somewhere between 4 and 15 for linear RLDCs with perfect completeness and soundness $1/2 - \varepsilon$ for any constant $\varepsilon > 0$: Corollary 1.10 implies that linear 3-RLDCs with soundness $1/2 - \varepsilon$ are 3-LDCs, and Theorem 4 gives a 15-RLDC with soundness $1/2 - \varepsilon$ that is not an q-query LDC for any constant q.

While Theorem 4 gives a construction of a q-RLDC with an explicit constant $q = 15$, one may wonder what the implicit constant q is in previous works. We attempted to determine the number of queries q needed for the RLDCs of [BGH+04, GRR18] to achieve soundness error $< 1/2$, and to the best of our knowledge, this q is at least 10^7 for both constructions. Intuitively, the reason the query complexity is so high is that these constructions make use of "proof composition" style results with robust probabilistically checkable proofs of proximity (PCPPs), and the soundness gap (defined as 1 − soundness error) deteriorates multiplicatively with each composition step. This causes the soundness gap to deteriorate quickly even when just a few composition steps are used, and as a result the final soundness gap is smaller than 10^{-7}. This means that the soundness error is at least $1 - 10^{-7}$, and so one must repeat the decoder many times (see Observation 1.9) to amplify the soundness error back down to $1/2$, which makes the query complexity large. In hindsight, the fact that these RLDCs use a large number of queries is perhaps unsurprising given that the constants in their proofs are likely not intentionally optimized.

We prove Theorem 4 in Section 6.

2 Techniques

In this section, we give a proof overview of our main result, Theorem 2. We will primarily focus on the case of $q = 2$ and $q = 3$ for RLDCs, though we shall also explain how to generalize the proof to arbitrary q and also to RLCCs. For simplicity, we will assume that the decoder is nonadaptive; Theorem 3 handles the case of adaptive decoders.

2.1 The proof strategy for Theorem 2

Let $C : \{0,1\}^k \to \{0,1\}^n$ be a linear $(q, \delta, 1, s)$-RLDC with a nonadaptive decoder $\mathrm{Dec}(\cdot)$ satisfying the properties in Definition 1.5 with $s \le (1-\alpha)s(q)$. Our goal is to show that C is a $(q, \delta', 1, s')$-LDC for some δ', s'. To do this, we need to construct a (possibly different) LDC decoder $\mathrm{Dec}'(\cdot)$ using the RLDC decoder $\mathrm{Dec}(\cdot)$. We will do this by "opening up" $\mathrm{Dec}(\cdot)$, i.e., we will crucially *not* use $\mathrm{Dec}(\cdot)$ in a black-box manner.

LDCs and smooth decoders. The starting point of our proof is the following observation of [KT00].

Observation 2.1. Suppose that C admits a q-query *smooth decoder* $\mathrm{Dec}_{\mathrm{Smooth}}(\cdot)$, which is a nonadaptive local decoder with the following two properties:

(1) (perfect completeness) for every $b \in \{0,1\}^k$ and $i \in [k]$, $\Pr[\mathrm{Dec}_{\mathrm{Smooth}}^{C(b)}(i) = b_i] = 1$, and

(2) (η-smoothness) for every $i \in [k]$ and $j \in [n]$, $\Pr[\mathrm{Dec}_{\mathrm{Smooth}}(i) \text{ queries } j] \le \frac{1}{\eta n}$.

Then, for any $\varepsilon > 0$, C is a $(q, \eta\varepsilon, 1, \varepsilon)$-LDC.

Indeed, Observation 2.1 follows by a simple union bound, as if there are at most $\eta\varepsilon n$ errors, then the probability that at least one of the q queries made by the decoder is corrupted is at most $\frac{1}{\eta n} \cdot \eta\varepsilon n = \varepsilon$, and if all queries are uncorrupted then the decoder outputs b_i, by perfect completeness. Thus, to prove Theorem 2, it suffices to extract a smooth decoder from $\mathrm{Dec}(\cdot)$.

A canonical RLDC decoder. Let us now consider the behavior of $\mathrm{Dec}^y(i)$ for a fixed $i \in [k]$ and some $y \in \{0,1\}^n$. Because $\mathrm{Dec}^y(i)$ is nonadaptive, we can view its operation as a two step process. First, $\mathrm{Dec}^y(i)$ samples a set of queries Q of size $\le q$ (for simplicity, let us assume that $|Q| = q$) from a query distribution $\boldsymbol{Q}_i$ over $\binom{[n]}{\le q}$ that does not depend on y. Then, it reads the values y_j for each $j \in Q$, and outputs the value in $\{0,1,\perp\}$ of a (possibly randomized) function $f_Q(\{y_j\}_{j\in Q})$ that depends on the set of queries Q and their answers $\{y_j\}_{j\in Q}$. We can now make the following simple observation: since the original decoder $\mathrm{Dec}(\cdot)$ has perfect completeness, there is a canonical choice of the decoding function f_Q for each set $Q \subseteq [n]$. If the values of y on Q read by the decoder are consistent with some codeword $x = C(b)$, i.e., $y|_Q = x|_Q$, then the decoder must output b_i, the i-th bit of the underlying message in x, by perfect completeness. And, if the values of y on Q are inconsistent with *all* codewords x, then y must have an error in the set Q and so the decoder can safely output $\perp$, as this can only decrease the soundness error of the decoder.

Decomposing the canonical RLDC decoder into smooth and nonsmooth parts. The above observation thus implies that we can view the decoder $\mathrm{Dec}^y(i)$ as being solely determined by the query distribution $\boldsymbol{Q}_i$. Given a distribution $\boldsymbol{Q}_i$, we can define the set of "nonsmooth" or "heavy" queries as $H_i := \{j \in [n] : \Pr_{Q\leftarrow\boldsymbol{Q}_i}[j \in Q] > \frac{q}{\delta n}\}$. Note that for a (δ/q)-smooth decoder, the set H_i is empty. We clearly have $|H_i| \le \delta n$, as

$$q = \sum_{j\in[n]} \Pr_{Q\leftarrow\boldsymbol{Q}_i}[j \in Q] \ge \sum_{j\in H_i} \Pr_{Q\leftarrow\boldsymbol{Q}_i}[j \in Q] \ge \frac{q|H_i|}{\delta n}.$$

In particular, this means that for any $b \in \{0,1\}^k$, the RLDC decoder $\mathrm{Dec}(i)$ on input i must satisfy $\Pr[\mathrm{Dec}^y(i) \notin \{b_i, \perp\}] \le s$ for any y where $y_j = C(b)_j$ for all $j \notin H_i$. That is, if we only introduce

errors in the "heavy set" H_i, we have introduced a small enough fraction of errors so that the soundness condition of the RLDC decoder still applies.

Call a set $Q \in \binom{[n]}{q}$ "i-smoothable" if one can recover b_i from $x|_{Q \setminus H_i}$ for any codeword $x = C(b)$, and let $p_{i,\text{good}} = \Pr_{Q \leftarrow \mathcal{Q}_i}[Q \text{ is } i\text{-smoothable}]$. Let $\mathcal{Q}_{i,\text{good}}$ be the distribution $\mathcal{Q}_i$ conditioned on the output Q being "i-smoothable", and let $\mathcal{Q}_{i,\text{bad}}$ be $\mathcal{Q}_i$ conditioned on Q being not "i-smoothable". Recall that because there is an optimal canonical decoder, a (nonadaptive) decoder is determined completely by its query distribution. Thus, the set of query distributions $\mathcal{Q}_{i,\text{good}}$ for each $i \in [k]$ defines a decoder that we call Dec_{LDC}, and similarly $\{\mathcal{Q}_{i,\text{bad}}\}_{i \in [k]}$ defines a decoder Dec_{RLDC}. Furthermore, for any $y \in \{0,1\}^n$ and any $i \in [k]$, for every output $\sigma \in \{0, 1, \perp\}$, it holds that

$$\Pr[\text{Dec}^y(i) = \sigma] = p_{i,\text{good}} \Pr[\text{Dec}^y_{\text{LDC}}(i) = \sigma] + (1 - p_{i,\text{good}}) \Pr[\text{Dec}^y_{\text{RLDC}}(i) = \sigma] .$$

In other words, we can view the behavior of $\text{Dec}^y(i)$ as follows: with probability $p_{i,\text{good}}$, it runs $\text{Dec}^y_{\text{LDC}}(i)$, and with probability $1 - p_{i,\text{good}}$ it runs $\text{Dec}^y_{\text{RLDC}}(i)$. The two decoders Dec_{LDC} and Dec_{RLDC} have their names chosen to indicate that Dec_{LDC} is the "LDC part" or "smooth part" of the original decoder Dec, and Dec_{RLDC} is the "pure RLDC part" or "nonsmooth part" of the original decoder.

Showing that the "smooth part" is nontrivial by breaking soundness of the "nonsmooth" part. To finish proving that C is an LDC, it remains to argue that $p_{i,\text{good}} \geq \Omega(1)$ for each $i \in [k]$, i.e., the "smooth part" of the RLDC decoder is nontrivially large. This is because the decoder Dec_{LDC} is $\delta p_{i,\text{good}}/q$-smooth[7] for each $i \in [k]$, and so by Observation 2.1 it is an LDC decoder that can tolerate $\delta p/q$ errors, where $p = \min_{i \in [k]} p_{i,\text{good}}$. We thus want $p_{i,\text{good}} \geq \Omega(1)$ for each $i \in [k]$, so that $\delta p/q = \Omega(\delta/q) = \Omega(1)$ is at least a constant.

To argue that $p_{i,\text{good}} \geq \Omega(1)$ for each $i \in [k]$, we will fix $i \in [k]$ and we show that any decoder that makes at least one "nonsmooth" query has soundness error at least $s(q) = 2^{-\lfloor q/2 \rfloor}$. More formally, we show that for each $b \in \{0,1\}^k$, there is a $y \in \{0,1\}^n$ where $y_v = C(b)_v$ for $v \notin H_i$ such that $\Pr[\text{Dec}^y_{\text{RLDC}}(i) \notin \{b_i, \perp\}] \geq s(q)$. That is, y agrees with a codeword on the "smooth queries" and only disagrees on some of the "nonsmooth queries", which also implies that $\Delta(y, C(b)) \leq |H_i| \leq \delta n$. Note that by soundness of the original decoder Dec, this implies

$$\begin{aligned}
&(1 - \alpha)s(q) \geq s \geq \Pr[\text{Dec}^y(i) \notin \{b_i, \perp\}] \\
&= p_{i,\text{good}} \Pr[\text{Dec}^y_{\text{LDC}}(i) \notin \{b_i, \perp\}] + (1 - p_{i,\text{good}}) \Pr[\text{Dec}^y_{\text{RLDC}}(i) \notin \{b_i, \perp\}] \\
&\geq s(q)(1 - p_{i,\text{good}}) ,
\end{aligned}$$

and so $p_{i,\text{good}} \geq \alpha$.

We explain how to break soundness of Dec_{RLDC} in the next section.

2.2 Analyzing the "nonsmooth" linear RLDC decoder

It remains to show that Dec_{RLDC} has soundness error at least $s(q) = 2^{-\lfloor q/2 \rfloor}$. As explained above, we will show that for each $b \in \{0,1\}^k$, there exists $y \in \{0,1\}^n$ that agrees with $x = C(b)$ on all

[7] Here, we sample Q from $\mathcal{Q}_{i,\text{good}}$ and only query $Q \setminus H_i$. Note that because Q is i-smoothable, we can still decode b_i from $Q \setminus H_i$, and so we have preserved perfect completeness while making the decoder smooth.

coordinates in $[n] \setminus H_i$ such that $\Pr[\mathrm{Dec}^y_{\mathrm{RLDC}}(i) = 1 - b_i] \geq s(q)$. Because the code is linear, without loss of generality we may assume that $b = 0^k$, so that $x = 0^n$, and our goal is to show the existence of such a $y \in \{0,1\}^n$ where $\Pr[\mathrm{Dec}^y_{\mathrm{RLDC}}(i) = 1] \geq s(q)$.

Fooling a fixed set Q. Let us now explain how to construct such a y. Fix a set Q in the support of the query distribution of $\mathrm{Dec}_{\mathrm{RLDC}}$. Because we have an optimal canonical decoder, when $\mathrm{Dec}_{\mathrm{RLDC}}$ queries Q, it simply checks if $y|_Q = x|_Q$ for some codeword x, and if so, then it must output the bit b_i where $x = C(b)$.

As a first attempt, suppose we choose y so that $y|_{H_i}$ is random and otherwise y agrees with 0^n, the original uncorrupted codeword. Then, with probability at least 2^{-q}, it holds that $y|_Q = x|_Q$ for some codeword $x = C(b)$ with $b_i = 1$. Indeed, because Q is not "i-smoothable", the message bit b_i cannot be recovered from $x|_{Q \setminus H_i}$ for a codeword $x = C(b)$, and so there must exist a codeword $x = C(b)$ with $b_i = 1$ such that $x|_{Q \setminus H_i}$ agrees with the codeword 0^n. And, since $y|_{H_i}$ is random, we have that $y|_{Q \cap H_i} = x|_{Q \cap H_i}$ with probability at least $2^{-|Q \cap H_i|} \geq 2^{-q}$. We note that this observation is sufficient to prove Theorem 2 with $s(q) = 2^{-q}$.

To prove Theorem 2 with $s(q) = 2^{-\lfloor q/2 \rfloor}$, however, we require a more sophisticated analysis that will end up using the linearity of C quite strongly. For a linear code C, $y|_Q$ agrees with a codeword x on Q if and only if $y|_Q$ satisfies a certain system of linear equations that are the "local parity check constraints on Q". For example, it could be the case that $Q = \{j_1, j_2, j_3\}$ where any codeword x satisfies $x_{j_1} + x_{j_2} + x_{j_3} = 0$, and there are no other local constraints. In this case, if y satisfies $y_{j_1} + y_{j_2} + y_{j_3} = 0$, then there exists some codeword x where $x_{j_1} = y_{j_1}$, $x_{j_2} = y_{j_2}$, and $x_{j_3} = y_{j_3}$.

However, the above observation is not enough for us, as we additionally need to make $\mathrm{Dec}_{\mathrm{RLDC}}$ output 1. To make this happen, we will again make use of the linear structure of C. Because $C \colon \{0,1\}^k \to \{0,1\}^n$ is a linear map, for every $j \in [n]$ there exists $v_j \in \{0,1\}^k$ such that for every $b \in \{0,1\}^k$, $C(b)_j := \langle b, v_j \rangle$; the vector v_j is simply the j-th row of the generator matrix of C. Note that "local constraints" above correspond to linear dependencies among the rows of the generator matrix. That is, $x_{j_1} + x_{j_2} + x_{j_3} = 0$ for all codewords x if and only if $v_{j_1} + v_{j_2} + v_{j_3} = 0^k$.

The fact that $\mathrm{Dec}_{\mathrm{RLDC}}$ has perfect completeness implies that one can recover b_i exactly from $x = C(b)$ restricted to Q. In linear algebraic terms, this means that the i-th standard basis vector e_i is in $\mathrm{span}(\{v_j : j \in Q\})$. We have assumed that Q is not "i-smoothable", meaning that one *cannot* recover b_i from $x|_{Q \setminus H_i}$, as otherwise Q is a set that "belongs to" $\mathrm{Dec}_{\mathrm{LDC}}$. So, e_i is not in the span of $\mathrm{span}(\{v_j : j \in Q \setminus H_i\})$.

Our main technical lemma (Lemma 4.3) shows that if we pick y such that: (1) $y_j = C(0^k)_j = 0$ for $j \notin H_i$, and (2) $y_j = C(b)_j$ for $j \in H_i$ where $b \in \{0,1\}^k$ is *random* with $b_i = 1$, then with probability at least $s(q) = 2^{-\lfloor q/2 \rfloor}$, $y|_Q = x|_Q$ for some codeword $x = C(b')$ where $b'_i = 1$. That is, $y|_Q$ satisfies all local checks in Q and is consistent with a codeword that has i-th message bit equal to 1, and thus when this occurs $\mathrm{Dec}_{\mathrm{RLDC}}$ outputs 1.

Remark 2.2. The intuition for our main technical lemma (and indeed the main intuition behind Theorem 2) is as follows. Firstly, in order for the RLDC decoder to be "not LDC-like", it needs to make queries to the nonsmooth set H_i. Secondly, in order for the RLDC decoder to be hard to fool, it must use the queries to $Q \setminus H_i$ to check consistency with the queries $Q \cap H_i$; if there are no consistency checks, then we can freely choose the values of y on $Q \cap H_i$ and easily fool the decoder.

Finally, when q is small (say $q = 2$ or 3), the decoder cannot simultaneously make many queries to both H_i and $[n] \setminus H_i$, i.e., one of $Q \cap H_i$ or $Q \setminus H_i$ must be small, and this is an inherent weakness that allows us to fool $\mathrm{Dec}_{\mathrm{RLDC}}$ with reasonable probability.

Casework for $q = 2$. Let us now explain a simple argument to prove our main technical lemma when $q = 2$. Let $Q = \{j_1, j_2\}$. We split the analysis into 3 cases, depending on $|Q \cap H_i|$, i.e., the number of "nonsmooth queries" made in Q. As we shall see, the case of $q = 2$ is simple and also somewhat degenerate, in that it is not possible to have a nontrivial "local check". In fact, this is the reason that we can show a very good lower bound for 2-RLDCs over large fields (see Section 7).

(1) Case 1: $|Q \cap H_i| = 0$. In this case, $Q \cap H_i = \emptyset$, and therefore e_i is in $\mathrm{span}(\{v_j\}_{j \in Q \setminus H_i})$, contradicting the fact that Q is not i-smoothable. Thus, this case cannot occur.

(2) Case 2: $|Q \cap H_i| = 1$. In this case, $Q = \{j_1, j_2\}$ where $j_1 \in H_i$ and $j_2 \notin H_i$. We have that $e_i \notin \mathrm{span}(v_{j_2})$, which implies that $v_{j_2} \neq e_i$. We also have $e_i \in \mathrm{span}(v_{j_1}, v_{j_2})$, which implies that either $e_i = v_{j_1}$, or $e_i = v_{j_1} + v_{j_2}$.

We also have at most one "local check": either $v_{j_1} + v_{j_2} = 0^k$, or else there are no local check constraints. However, if there is a local check constraint, then if $v_{j_1} = e_i$, this implies that $v_{j_2} = e_i$, and so $e_i \in \mathrm{span}(v_{j_2})$ (a contradiction), and if $v_{j_1} + v_{j_2} = e_i$, then the local check constraint implies $e_i = 0^k$ (a contradiction). Thus, we cannot have a local check constraint.

Because $j_2 \notin H_i$, we have $y_{j_2} = 0$, and so we fool the decoder if and only if $y_{j_1} = 1$. This clearly happens with probability at least $1/2$, as either $v_{j_1} = e_i$, in which case $y_{j_1} = 1$ with probability 1, or else $v_{j_2} \neq e_i$ and is nonzero, in which case $y_{j_1} = 1$ with probability $1/2$.

(3) Case 3: $|Q \cap H_i| = 2$. In this case, $y|_Q = x|_Q$ for some $x = C(b')$ where $b'_i = 1$, by definition. Thus, the canonical decoder outputs 1 with probability 1.

In all cases, we see that the decoder outputs 1 with probability at least $s(2) = 1/2$ over the random choice of y, as required.

Generalizing to higher q. The above casework proves the desired claim for $q = 2$. One can repeat a similar case-by-case analysis for $q = 3$, but the analysis quickly becomes unwieldy as q increases. For example, when $q = 3$ one can have $Q = \{j_1, j_2, j_3\}$ where $j_1, j_2 \in H_i$, $j_3 \notin H_i$, and $v_{j_1} + v_{j_2} = e_i$ and $v_{j_2} + v_{j_3} = 0$. That is, we can have both a "decoding constraint" that sums to e_i and a "check constraint" that sums to 0. Again, one can verify that our choice of random y will satisfy both these constraints with probability $1/2$: this is because the two constraints imply $v_{j_1} + v_{j_2} = e_i$, and this constraint is satisfied with probability 1 (because y is consistent with $x = C(b')$ with $b'_i = 1$ on indices in H_i), and the constraint $v_{j_2} + v_{j_3} = 0$ is satisfied with probability $1/2$, independently of the first constraint. We note that it is easy to show how to set the values y_j for $j \in H_i$ to fool the decoder for a particular Q, but the key difficulty is we need to find a single global y that fools an $s(q)$-fraction of the Q's simultaneously.

To find a single global y, we show that if we sample y from the distribution as above, i.e., where $y|_{H_i}$ is chosen to be $C(b)|_{H_i}$ where b is uniformly random with $b_i = 1$, then for any set Q that is not i-smoothable, with probability at least $s(q)$ over the draw of y, y fools the decoder when it

queries Q. Recall that y fools the decoder if and only if $y|_Q \in C'|_Q$, where C' is the affine subspace $\{C(b)|_{Q\cap H_i} : b_i = 1\}$. This requires checking that (1) $y|_{Q\cap H_i}$ is in $C'|_{Q\cap H_i}$, (2) $y|_{Q\setminus H_i}$ is in $C'|_{Q\setminus H_i}$, and (3) $y|_{Q\cap H_i}$ is consistent with $0^{Q\setminus H_i}$ and the constraint that $b_i = 1$. Notice that y satisfies all of the type (1) constraints with probability 1, since $y|_{Q\cap H_i}$ is drawn uniformly over that set, and y satisfies all of the type (2) constraints by definition because Q is not i-smoothable. We can view the type (3) constraints as imposing a system of inhomogeneous linear constraints on the vector $y|_{Q\cap H_i}$, so that y fools the decoder if and only if $y|_{Q\cap H_i}$ lies in a particular affine subspace.

Formally, we show that for each Q and each codeword $C(b)$, there is a vector $z \in \{0,1\}^{Q\cap H_i}$ and linear subspaces $\mathcal{V}$ and $\mathcal{W}$ in $\{0,1\}^{Q\cap H_i}$ with $\mathcal{V} \subseteq \mathcal{W}$ and $\dim(\mathcal{W}/\mathcal{V}) \le \min(|Q \cap H_i|, |Q \setminus H_i|)$ such that $y|_Q$ fools the decoder if and only if $y|_{Q\cap H_i} - z$ lies in $\mathcal{W}^\perp$. Intuitively, the vector z shifts the affine subspaces to be linear subspaces, and then the subspace $\mathcal{W}$ is the set of "local parity check constraints" on $Q \cap H_i$ satisfied by all codewords $C(b')$ with $b'_i = 0$ that are also 0 on $Q \setminus H_i$ (type (1) and type (3) constraints), and the subspace $\mathcal{V}$ is the set of constraints on $Q \cap H_i$ satisfied by all codewords $C(b')$ with $b'_i = 0$ (type (1) constraints only). The distribution of $y|_{Q\cap H_i} - z$ is then uniform over the larger subspace $\mathcal{V}^\perp$ in $\{0,1\}^{Q\cap H_i}$ that contains $\mathcal{W}^\perp$. Hence, y lies in the desired affine subspace with probability at least $2^{-\dim(\mathcal{W}/\mathcal{V})}$ and this is at least $2^{-\lfloor q/2\rfloor}$, as one can show that $\dim(\mathcal{W}/\mathcal{V}) \le \min(|Q \cap H_i|, |Q \setminus H_i|) \le \lfloor q/2\rfloor$. This is the "inherent weakness" mentioned in Remark 2.2: one of $|Q \cap H_i|$ or $|Q \setminus H_i|$ must have size $\le \lfloor q/2\rfloor$, and they control the dimension of the "extra local checks" that $y|_{Q\cap H_i}$ must satisfy in order to fool the decoder.

The full proof of Theorem 2 is in Section 4.

Generalizing to RLCCs. The above analysis considers the behavior of $\mathrm{Dec}^y(i)$ on each input i separately, and shows how to convert it to a smooth decoder. Thus, it seamlessly generalizes to RLCCs, as we can consider the behavior of the RLCC decoder $\mathrm{Dec}^y(u)$ on each input u separately as well, and convert $\mathrm{Dec}^y(u)$ to a smooth decoder for each u to obtain an LCC.

3 Preliminaries

3.1 Basic notation

We let $[n]$ denote the set $\{1, \dots, n\}$. For a natural number $t \in \mathbb{N}$, we let $\binom{[n]}{t}$ be the collection of subsets of $[n]$ of size exactly t.

Given a string $x \in \Sigma^n$ and a set $S \subseteq [n]$, we define $x|_S$ to be the restriction of x to the indices in S. Similarly, for a set of strings $X \subseteq \Sigma^n$, we define $X|_S := \{x|_S : x \in X\}$. If Σ is an alphabet with a distinguished element $0 \in \Sigma$, we also define $X_{\subseteq S} := \{x \in X : \mathrm{supp}(x) \subseteq S\}$, where $\mathrm{supp}(x) := \{i \in S : x_i \neq 0\}$.

Given a finite field $\mathbb{F}$ and $x, y \in \mathbb{F}^n$, we let $\langle x, y\rangle = \sum_{i=1}^n x_i y_i$ denote their inner product.

Definition 3.1 (Hamming distance)**.** For strings $x, y \in \Sigma^n$, the (absolute) *Hamming distance* $\Delta(x, y)$ is the number of indices where x differs from y. The *relative Hamming distance* $\bar{\Delta}(x, y) := \Delta(x, y)/n$ is the fraction of indices where x differs from y.

$$\Delta(x, y) := \#\{i : x_i \neq y_i\}, \qquad \bar{\Delta}(x, y) := \Delta(x, y)/n\,.$$

If $Y \subseteq \Sigma^n$, then $\Delta(x, Y)$ denotes the minimum Hamming distance between x and an element $y \in Y$:

$$\Delta(x, Y) := \min_{y \in Y} \Delta(x, y)\,.$$

3.2 Linear codes

Definition 3.2 (Linear codes)**.** A *linear code* C over a finite field $\mathbb{F}$ is an injective $\mathbb{F}$-linear map $C\colon \mathbb{F}^k \to \mathbb{F}^n$. The integer k is the length of the message of the code, and the integer n is the blocklength of the code. Because C is injective, $k = \dim(\mathrm{im}(C))$ is the dimension of the code.

We will sometimes specify C as a dimension k linear subspace of $\mathbb{F}^n$, rather than by its encoding map, i.e., we have $C \subseteq \mathbb{F}^n$, and we write $x \in C$ to indicate that x is in the image of the (implicitly defined) linear map from $\mathbb{F}^k \to C \subseteq \mathbb{F}^n$.

A linear code C can be described by both a "generator matrix" and a "parity check matrix."

- (Generator matrix) Given a linear code $C\colon \mathbb{F}^k \to \mathbb{F}^n$, there is a matrix $M \in \mathbb{F}^{n \times k}$ such that $C(b) = Ab$ for each $b \in \mathbb{F}^k$. The matrix A is called the "generator matrix" of the code.
- (Parity check matrix) Given a linear code $C\colon \mathbb{F}^k \to \mathbb{F}^n$, there is a matrix $B \in \mathbb{F}^{(n-k)\times n}$ such that $x \in \mathrm{im}(C)$ if and only if $Bx = 0^{n-k}$. The matrix B is called a "parity check matrix" for C.

We say that a family of linear codes is *explicit* if there is a uniform efficient algorithm for computing generator matrices or parity check matrices for the family.

Definition 3.3 (Dual code)**.** Let $C\colon \mathbb{F}^k \to \mathbb{F}^n$. Its *dual code*, denoted by $C^\perp$ is the linear subspace of $\mathbb{F}^n$ given by $C^\perp = \{y \in \mathbb{F}^n : \langle x, y\rangle = 0 \ \forall x \in C\}$. Here, $\langle x, y\rangle = \sum_{i=1}^n x_i y_i$ is the standard inner product.

The following fact relates the dual code of $C|_S$ to the dual code of C.

Fact 3.4. *Let $C \subseteq \mathbb{F}^n$ be a linear code, and let $S \subseteq [n]$. Then, $(C|_S)^\perp := C^\perp_{\subseteq S}$, where $C^\perp_{\subseteq S} = \{y \in C^\perp : \mathrm{supp}(y) := \{i : y_i \neq 0\} \subseteq S\}$.*

Proof. We have that $z \in (C|_S)^\perp \subseteq \mathbb{F}_2^S$ if and only if

$$\forall x \in C,\ 0 = \langle x|_S, z\rangle = \sum_{i \in S} x_i z_i = \sum_{i \in S} x_i z_i + \sum_{i \notin S} x_i \cdot 0 = \langle x, y\rangle\,,$$

where $y_i = z_i$ for $i \in S$ and $y_i = 0$ for $i \notin S$, and the latter statement is true if and only if $y \in C^\perp_{\subseteq S}$, as desired. □

The following fact describes exactly when a linear combination $\langle v, b\rangle$ of the message b is determined by $C(b)|_Q$ for a set Q.

Fact 3.5. *Let $C\colon \mathbb{F}^k \to \mathbb{F}^n$ be a linear code. For each $j \in [n]$, let v_j denote the j-th row of the generator matrix, so that $\langle v_j, b\rangle = C(b)_j$ for all $b \in \mathbb{F}^k$.*

Let $Q \subseteq [n]$ and let $x^* = C(b^*)$. Let $v \in \mathbb{F}^k$. Then, every $x = C(b)$ with $x|_Q = x^*|_Q$ satisfies $\langle v, b\rangle = \langle v, b^*\rangle$ if and only if $v \in \mathrm{span}(\{v_j\}_{j\in Q})$. Furthermore, if $v \notin \mathrm{span}(\{v_j\}_{j\in Q})$, then for every $\sigma \in \mathbb{F}$, there exists $b \in \mathbb{F}^k$ such that $x|_Q = x^*|_Q$ and $\langle v, b\rangle = \sigma$.

That is, if we are given the values of a codeword x^* restricted to Q, it "fixes" some linear combination of the message symbols if and only if the linear combination is in the span of rows of the generator matrix corresponding to the set Q, and otherwise it is "free".

3.3 Locally decodable/correctable codes and their relaxed notions

Below, we define LDCs/RLDCs and LCCs/RLCCs.

Definition 3.6 (Locally decodable codes)**.** A code $C\colon \Sigma^k \to \Sigma^n$ is a (q, δ, c, s)-LDC if there exists a randomized decoding algorithm $\mathrm{Dec}(\cdot)$ with the following properties. The algorithm $\mathrm{Dec}(\cdot)$ is given oracle access to a string $y \in \Sigma^n$, takes an index $i \in [k]$ as input, and outputs a symbol in Σ with the following guarantees:

(1) (q-queries) for any y and i, the algorithm $\mathrm{Dec}^y(i)$ reads at most q indices of y,

(2) (c-completeness) for all $b \in \Sigma^k$ and $i \in [k]$, $\Pr[\mathrm{Dec}^{C(b)}(i) = b_i] \geq c$, and

(3) ((δ, s)-soundness error) for all $b \in \Sigma^k$, $i \in [k]$, and all $y \in \Sigma^n$ with $\Delta(y, C(b)) \leq \delta n$, $\Pr[\mathrm{Dec}^y(i) \neq b_i] \leq s$.

Definition 3.7 (Locally correctable codes)**.** A code $C\colon \Sigma^k \to \Sigma^n$ is a (q, δ, c, s)-LCC if there exists a randomized decoding algorithm $\mathrm{Dec}(\cdot)$ with the following properties. The algorithm $\mathrm{Dec}(\cdot)$ is given oracle access to a string $y \in \Sigma^n$, takes an index $u \in [n]$ as input, and outputs a symbol in Σ with the following guarantees:

(1) (q-queries) for any y and u, the algorithm $\mathrm{Dec}^y(u)$ reads at most q indices of y,

(2) (c-completeness) for all $b \in \Sigma^k$ and $u \in [n]$, $\Pr[\mathrm{Dec}^{C(b)}(u) = C(b)_u] \geq c$, and

(3) ((δ, s)-soundness error) for all $b \in \Sigma^k$, $u \in [n]$, and all $y \in \Sigma^n$ with $\Delta(y, C(b)) \leq \delta n$, $\Pr[\mathrm{Dec}^y(u) \neq C(b)_u] \leq s$.

We will now define smooth decoders/correctors. These are decoders which only need to work on valid codewords, but must not favor querying any one index heavily.

Definition 3.8 (Smooth decoder)**.** An *η-smooth decoder* of a code $C\colon \Sigma^k \to \Sigma^n$ is a decoder $\mathrm{Dec}(\cdot)$ such that

(1) (q-queries) for any y and $i \in [k]$, the algorithm $\mathrm{Dec}^y(i)$ reads at most q indices of y,

(2) (perfect completeness) for every $b \in \Sigma^k$ and $i \in [k]$, $\Pr[\mathrm{Dec}^{C(b)}(i) = b_i] = 1$, and

(3) (η-smoothness) for every $i \in [k]$ and $j \in [n]$, $\Pr[\mathrm{Dec}(i) \text{ queries } j] \leq \frac{1}{\eta n}$.

Definition 3.9 (Smooth corrector)**.** A *η-smooth corrector* of a code $C \subset \Sigma^n$ is a decoder $\mathrm{Dec}(\cdot)$ such that

(1) (q-queries) for any y and $u \in [n]$, the algorithm $\mathrm{Dec}^y(u)$ reads at most q indices of y,

(2) (perfect completeness) for every $b \in \Sigma^k$ and $u \in [n]$, $\Pr[\mathrm{Dec}^{C(b)}(u) = C(b)_u] = 1$, and

(3) (δ-smoothness) for every $u, j \in [n]$, $\Pr[\mathrm{Dec}(u) \text{ queries } j] \leq \frac{1}{\eta n}$.

The notion of smooth decoders was introduced in [KT00] because of their equivalence to locally decodable codes. For this paper, we will need the following simple fact.

Fact 3.10 (Smooth decoder implies local decoding)**.** *Let C be a code with a q-query η-smooth decoder (corrector). Then, C is a $(q, \eta\varepsilon, 1, \varepsilon)$-LDC (LCC).*

Proof. We will prove the smooth decoder is indeed the desired local decoder (the corrector case follows analogously). Perfect completeness follows by definition in Definition 3.8, so it remains to show that the soundness error is at most ε. Consider a received word y such that $\Delta(y, C(b)) \leq \eta\varepsilon n$ for some $b \in \Sigma^k$. Let $S \subset [n]$ be the indices where y differs from $C(b)$. By η-smoothness, we have that for any $v \in S$, $\Pr[\mathrm{Dec}(i) \text{ queries } v] \leq \frac{1}{\eta n}$. Union bounding over all $v \in S$, it follows

$$\Pr[\mathrm{Dec}(i) \text{ does not query } S] \geq 1 - \frac{1}{\eta n} \cdot |S| \geq 1 - \varepsilon\,.$$

Now if $\mathrm{Dec}^y(i)$ never queries any index in S, its local view is consistent with the codeword $C(b)$. Consequently, the decoder must output b_i with probability 1 by perfect completeness. Hence, $\mathrm{Dec}(\cdot)$ has soundness error $\leq \varepsilon$ as desired. □

Definition 3.11 (Relaxed locally decodable codes)**.** A code $C\colon \Sigma^k \to \Sigma^n$ is a (q, δ, c, s)-RLDC if there exists a randomized decoding algorithm $\mathrm{Dec}(\cdot)$ with the following properties. The algorithm $\mathrm{Dec}(\cdot)$ is given oracle access to a string $y \in \Sigma^n$, takes an index $i \in [k]$ as input, and outputs either a symbol in Σ or a special symbol $\perp$ with the following guarantees:

(1) (q-queries) for any y and i, the algorithm $\mathrm{Dec}^y(i)$ reads at most q indices of y,

(2) (c-completeness) for all $b \in \Sigma^k$ and $i \in [k]$, $\Pr[\mathrm{Dec}^{C(b)}(i) = b_i] \geq c$, and

(3) ((δ, s)-relaxed soundness error) for all $b \in \Sigma^k$, $i \in [k]$, and all $y \in \Sigma^n$ with $\Delta(y, C(b)) \leq \delta n$, $\Pr[\mathrm{Dec}^y(i) \notin \{b_i, \perp\}] \leq s$.

Definition 3.12 (Relaxed locally correctable codes)**.** A code $C\colon \Sigma^k \to \Sigma^n$ is a (q, δ, c, s)-RLDC if there exists a randomized decoding algorithm $\mathrm{Dec}(\cdot)$ with the following properties. The algorithm $\mathrm{Dec}(\cdot)$ is given oracle access to a string $y \in \Sigma^n$, takes an index $u \in [n]$ as input, and outputs either a symbol in Σ or a special symbol $\perp$ with the following guarantees:

(1) (q-queries) for any y and u, the algorithm $\mathrm{Dec}^y(u)$ reads at most q indices of y

(2) (c-completeness) for all $b \in \Sigma^k$ and $u \in [n]$, $\Pr[\mathrm{Dec}^{C(b)}(u) = C(b)_u] \geq c$,

(3) ((δ, s)-relaxed soundness error) for all $b \in \Sigma^k$, $u \in [n]$, and all $y \in \Sigma^n$ with $\Delta(y, C(b)) \leq \delta n$, $\Pr[\mathrm{Dec}^y(u) \notin \{C(b)_u, \perp\}] \leq s$.

For RLDCs/RLCCs with perfect completeness, we can assume that the decoder behaves in a certain "canonical" way.

Fact 3.13 (Canonical behavior of a local decoder)**.** *Let $C\colon \{0,1\}^k \to \{0,1\}^n$ be a $(q, \delta, 1, s)$-RLDC with decoder Dec_1. Then, there is a decoder Dec_2 for C with perfect completeness such that (1) C is a $(q, \delta, 1, s)$-RLDC using Dec_2, and (2) whenever $\mathrm{Dec}_2^y(i)$ queries a set Q, its behavior is as follows:*

(1) Find $x = C(b)$ such that $x|_Q = y|_Q$. If there is no such x, output $\perp$.

(2) Otherwise, output b_i.

Furthermore, if Dec_1 is nonadaptive, then so is Dec_2, and the analogous statement also holds for RLCCs.

We say that the RLDC decoder of C is "canonical" if it follows the operation of Dec_2.

Proof. We define Dec_2 by (1) running Dec_1 until it has finished making all of its queries, and then (2) following the above decoding behavior. Note that Dec_2 has perfect completeness by definition, as Dec_1 has perfect completeness. The only difference between Dec_1 and Dec_2 is that Dec_2 may output $\perp$ when Dec_1 outputs some other symbol. But, this can only decrease the soundness error, which finishes the proof. □

In the case of linear codes, the canonical decoder described in Fact 3.13 has a nice linear algebraic structure, coming from the dual code.

Fact 3.14. *Let $C\colon \mathbb{F}^k \to \mathbb{F}^n$ be a linear $(q, \delta, 1, s)$-RLDC with a canonical decoder* Dec*. Let $Q \subseteq [n]$ be a subset, and let $z \in \mathbb{F}^Q$. The behavior of the* Dec*, when it queries the set Q and sees the local view z, is described as follows:*

(1) Check if $z \in C|_Q$, and output $\perp$ otherwise;

(2) Let $j_1, \ldots, j_t \in Q$ be such that $x_{j_1} + \cdots + x_{j_t} = b_i$ for all $x = C(b)$ (such a set must exist by perfect completeness). Output $z_{j_1} + \cdots + z_{j_t}$.

Furthermore, condition (1) can be checked by verifying that $Mz = 0$ for some matrix $M \in \mathbb{F}^{t \times Q}$, where $t \leq |Q|$. We call these constraints the "testing constraints" and the constraint in Item (2) the "decoding constraint".

Proof. Item (1) exactly matches the behavior of the decoder in Fact 3.13. To see Item (2), we use the generator matrix definition of the map C. That is, for each $j \in [n]$, there exists $v_j \in \mathbb{F}^k$ such that $x_j = \langle v_j, b \rangle$ when $x = C(b)$. With this perspective, we can recover b_i from $\{x_j\}_{j \in Q}$ if and only if $e_i \in \mathrm{span}(\{v_j\}_{j \in Q})$. Hence, by perfect completeness, e_i must be in the span, and so there exist $j_1, \ldots, j_t \in Q$ such that $v_{j_1} + \cdots + v_{j_t} = e_i$.

To prove the "furthermore", we observe that since $C|_Q$ is a linear subspace, we can check membership via a system of homogeneous linear equations, which yields the matrix M. □

3.4 Proof of Observation 1.9

We prove Observation 1.9, which is restated below.

Observation 3.15. Let $C\colon \{0,1\}^k \to \{0,1\}^n$ be a $(q,\delta,1,\varepsilon)$-RLDC with a possibly adaptive decoder. Then, for any integer $t \geq 1$, C is a $(qt,\delta,1,\varepsilon^t)$-RLDC. The same statement holds for RLCCs.

Proof. Consider the new RLDC decoder that runs the original decoder t times independently, and outputs a bit $\sigma \in \{0,1\}$ if all invocations of the decoder output σ, and otherwise the decoder outputs $\perp$. This decoder clearly satisfies perfect completeness, and has soundness error at most ε^t because the t invocations of the decoder are independent. □

3.5 Linearity testing

We recall the well-known result for linearity testing over $\mathbb{F}_2$.

Fact 3.16 (Linearity Test [BLR93, BCH+95])**.** *Let $G\colon \mathbb{F}_2^n \to \mathbb{F}_2$ be an arbitrary function, and let $\bar{\Delta}(G,\mathrm{LIN})$ be the minimum, over linear functions $F\colon \mathbb{F}_2^n \to \mathbb{F}_2$, of $\mathbb{E}_{x\in\mathbb{F}_2^n}[G(x) \neq F(x)]$. Then, $\Pr_{x,y\in\mathbb{F}_2^n}[G(x)+G(y)+G(x+y)=0] \leq 1-\bar{\Delta}(G)$. That is, if the linearity test passes with probability at least $1-\varepsilon$, then $\bar{\Delta}(G,\mathrm{LIN}) \leq \varepsilon$.*

We also recall that one can self-correct functions that are close to linear.

Fact 3.17 (Self-correction of near-linear functions)**.** *Let $G\colon \mathbb{F}_2^n \to \mathbb{F}_2$ be an arbitrary function, and let $F\colon \mathbb{F}_2^n \to \mathbb{F}_2$ be a linear function. Let $\bar{\Delta}(F,G) = \mathbb{E}_{x\in\mathbb{F}_2^n}[G(x) \neq F(x)]$. Then, for any $x \in \mathbb{F}_2^n$, $\Pr_{y\in\mathbb{F}_2^n}[G(x+y)+G(y)=F(x)] \geq 1-2\bar{\Delta}(F,G)$.*

4 Relaxed Locally Decodable Codes Cannot Have Strong Soundness

In this section, we prove Theorem 2, which shows a qualitative difference between linear RLDCs and LDCs: an LDC has strong soundness (soundness error can be made arbitrarily low by adjusting the decoding radius; see Appendix A), while an RLDC does not unless it is also an LDC. We will in fact prove the following theorem, which is a generalization of Theorem 2 to any finite field for the case of nonadaptive decoders. The case of adaptive decoders and imperfect completeness is handled generically by Theorem 5.2, which we prove in Section 5.

Theorem 4.1 (Theorem 2 for nonadaptive decoders and any field)**.** *Let $s_{\mathbb{F}}(q) := |\mathbb{F}|^{-\lfloor q/2 \rfloor}$. Let $C\colon \mathbb{F}^k \to \mathbb{F}^n$ be a linear $(q,\delta,1,s)$-RLDC with a nonadaptive decoder, where $s \leq (1-\alpha)s_{\mathbb{F}}(q)$. Then, for any $\varepsilon > 0$, C is a linear $(q,\alpha\delta\varepsilon/q,1,\varepsilon)$-LDC. Furthermore, the same result holds for RLCCs/LCCs.*

Proof. Let $C\colon \mathbb{F}^k \to \mathbb{F}^n$ be a linear $(q,\delta,1,s)$-RLDC with $s \leq (1-\alpha)s_{\mathbb{F}}(q)$. We will follow the proof outline from Section 2.

Let $\mathrm{Dec}(\cdot)$ be the nonadaptive RLDC decoder of C. Because $\mathrm{Dec}(\cdot)$ is nonadaptive, we may assume (Fact 3.13) that it behaves as follows. For each i, there is a distribution Q_i over subsets of $[n]$ of size at most q, and the decoder $\mathrm{Dec}^y(i)$ on input i simply samples $Q \leftarrow Q_i$, reads y_v for each $v \in Q$, and then follows the behavior of the canonical decoder in Fact 3.13.

Defining the smooth decoder $\mathbf{Dec_{LDC}(\cdot)}$. Let $i \in [k]$ be a message index. For each $i \in [k]$, we partition the codeword indices $[n]$ into heavy ("nonsmooth") and light ("smooth") indices as follows.

$$H_i := \{j \in [n] : \Pr_{Q \leftarrow \boldsymbol{Q}_i}[j \in Q] > q/\delta n\}$$
$$L_i := \{j \in [n] : \Pr_{Q \leftarrow \boldsymbol{Q}_i}[j \in Q] \leq q/\delta n\}$$

We then have

$$q \geq \mathbb{E}_{Q \leftarrow \boldsymbol{Q}^{(i)}}\left[\sum_{j \in [n]} \mathbf{1}(j \in Q)\right] \geq \sum_{j \in H_i} \mathbb{E}_{Q \leftarrow \boldsymbol{Q}^{(i)}}\left[\mathbf{1}(j \in Q)\right] \geq \frac{q|H_i|}{\delta n}.$$

Hence, $|H_i| \leq \delta n$.

We now split $\boldsymbol{Q}_i$ into the "smooth part" and the "nonsmooth part". Recall from the definition of a linear code (Definition 3.2) that for each $j \in [n]$, there exists $v_j \in \mathbb{F}^k$ such that for every $b \in \mathbb{F}^k$, $C(b)_j = \langle v_j, b \rangle$. We now define the notion of a "smoothable" query set, which are sets Q where one can recover b_i from only the light indices, i.e., $Q \cap L_i$.

Definition 4.2. For a set Q in the support of $\boldsymbol{Q}_i$, we call Q *i-smoothable* if $e_i \in \operatorname{span}\{v_j : j \in Q \cap L_i\}$.

Let us now define the following query distribution $\tilde{\boldsymbol{Q}}_i$. In the distribution $\tilde{\boldsymbol{Q}}_i$, we sample $Q \leftarrow \boldsymbol{Q}_i$ conditioned on Q being i-smoothable (we will show that this probability is nonzero, so this is well-defined), and then we output $\tilde{Q} := Q \cap L_i$. Note that $\tilde{Q}$ is i-smoothable since Q is.

Given the distribution $\tilde{\boldsymbol{Q}}_i$ for each $i \in [k]$, we define a decoder $\mathrm{Dec}_{\mathrm{LDC}}(\cdot)$ as follows. On input i, $\mathrm{Dec}^y_{\mathrm{LDC}}(i)$ draws $\tilde{Q} \leftarrow \tilde{\boldsymbol{Q}}$, and then uses the "decoding constraint" to decode b_i. That is, because $\tilde{Q}$ is i-smoothable, there exists a subset $T \subseteq \tilde{Q}$ such that $\sum_{j \in T} v_j = e_i$, and the decoder outputs $\sum_{j \in T} y_j$.

Arguing smoothness of $\mathbf{Dec_{LDC}(\cdot)}$. We will now show that $\mathrm{Dec}_{\mathrm{LDC}}(\cdot)$ is $(\alpha\delta/q)$-smooth (Definition 3.8). For each $i \in [k]$, let $p_{i,\mathrm{good}}$ be the probability that $Q \leftarrow \boldsymbol{Q}_i$ is i-smoothable. Let $\mathrm{Dec}_{\mathrm{RLDC}}(\cdot)$ denote the decoder that (1) samples $Q \leftarrow \boldsymbol{Q}_i$ conditioned on Q being *not* i-smoothable, and then (2) decodes using the canonical decoder. Observe that we can view the original decoder as simply calling $\mathrm{Dec}_{\mathrm{RLDC}}(\cdot)$ with probability $1 - p_{i,\mathrm{good}}$, where $i \in [k]$ is the input index (and, with probability $p_{i,\mathrm{good}}$, the decoder does something else). We then have that for any $b \in \mathbb{F}^k$ and $y \in \mathbb{F}^n$ with $\Delta(y, C(b)) \leq \delta n$ and any $i \in [k]$, it holds that

$$(1-\alpha)s_{\mathbb{F}}(q) \geq s \geq \Pr[\mathrm{Dec}^y(i) = \sigma \in \mathbb{F} \setminus \{b_i\}] \geq (1 - p_{i,\mathrm{good}}) \Pr[\mathrm{Dec}^y_{\mathrm{RLDC}}(i) = \sigma \in \mathbb{F} \setminus \{b_i\}]. \quad (1)$$

Our main technical lemma is the following lemma, which we will use to lower bound the soundness error of $\mathrm{Dec}_{\mathrm{RLDC}}$.

Lemma 4.3 (Fooling a set Q)**.** *Let $C : \mathbb{F}^k \to \mathbb{F}^n$ be a linear code. Let $Q = H \cup L$ where H and L are disjoint. Let $v^* \in \mathbb{F}^k$, and suppose that $v^* \notin \operatorname{span}(\{v_j : j \in L\})$. Let $b \in \mathbb{F}^k$, and let $x = C(b)$. Fix $i \in [k]$, and let $\sigma \in \mathbb{F}$.*

Let $b' \leftarrow \mathbb{F}^k$ be sampled uniformly at random with $\langle v^*, b' \rangle = \sigma$, and let $x' = C(b')$. Define $z \in \mathbb{F}^Q$ to be $z_j = x_j$ for $j \in L$ and $z_j = x'_j$ for $j \in H$. Then, with probability at least $|\mathbb{F}|^{-\min(|H|,|L|)}$ over the choice of b', there exists $b'' \in \mathbb{F}^k$ such that $C(b'')|_Q = z$ and $\langle v^*, b'' \rangle = \sigma$.

Lemma 4.3 says the following. Suppose we are given a query set Q and a codeword $x = C(b)$, and we want to corrupt $x|_Q$ by only modifying its values on H_i so that the corrupted version of $x|_Q$ is now equal to $x''|_Q$ for a different codeword x''. Then, if we corrupt $x|_Q$ by replacing $x|_{Q\cap H_i}$ with a uniformly random codeword, then it will satisfy this condition with some nontrivial probability. Furthermore, if we additionally wish the codeword x'' to be equal to $C(b'')$ for some b'' satisfying a particular inhomogeneous linear constraint $\langle v^*, b''\rangle = \sigma$, then this is possible provided that $v^* \notin \mathrm{span}(\{v_j : j \in Q \cap L_i\})$ and $v^* \in \mathrm{span}(\{v_j : j \in Q\})$.

We postpone the proof of Lemma 4.3 to Section 4.1, and for now use it to finish the proof of Theorem 4.1. As we shall shortly see, Lemma 4.3 implies that the soundness error of $\mathrm{Dec}_{\mathrm{RLDC}}$ is at least $s_{\mathbb{F}}(q)$.

Indeed, let $v^* = e_i$, and let Q be any set in the support of $\boldsymbol{Q}_i$ that is not i-smoothable. Let $b \in \mathbb{F}^k$ and let $x = C(b)$. Suppose that we define (a distribution over) $y \in \mathbb{F}^n$ with $\Delta(y, C(b)) \le \delta n$ by (1) sampling $b' \leftarrow \mathbb{F}^k$ with $b'_i \neq b_i$, and (2) setting y to be $y|_{H_i} = C(b')|_{H_i}$, and $y|_{L_i} = C(b)|_{L_i}$. We clearly have that $\Delta(y, C(b)) \le |H_i| \le \delta n$ holds with probability 1. On the other hand, by Lemma 4.3, with probability at least $|\mathbb{F}|^{-\min(|Q\cap H_i|,|Q\cap L_i|)} \ge |\mathbb{F}|^{-\lfloor q/2\rfloor} = s_{\mathbb{F}}(q)$, it holds that $y|_Q = x''|_Q$ for some $x'' = C(b'')$ where $b''_i = b'_i \neq b_i$. Because $\mathrm{Dec}_{\mathrm{RLDC}}$ behaves as the canonical decoder, it must therefore output $b''_i \neq b_i$.

The above shows that for any $i \in [k]$, there is a distribution over y with $\Delta(y, C(b)) \le \delta n$ such that $\mathbb{E}_y[\Pr[\mathrm{Dec}^y_{\mathrm{RLDC}}(i) \neq b_i]] \ge s_{\mathbb{F}}(q)$. Hence, by averaging, there exists y such that this holds. This implies that the soundness error of $\mathrm{Dec}^y_{\mathrm{RLDC}}(i)$ is at least $s_{\mathbb{F}}(q)$, and so by Eq. (1), we conclude that $(1-\alpha)s_{\mathbb{F}}(q) \ge (1-p_{i,\mathrm{good}})s_{\mathbb{F}}(q)$, which implies that $p_{i,\mathrm{good}} \ge \alpha$, and this holds for all $i \in [k]$.

With this lower bound on $p_{i,\mathrm{good}}$ in hand, let us now argue smoothness of $\mathrm{Dec}_{\mathrm{LDC}}(\cdot)$. Indeed, for any $j \in [n]$, we have that

$$\Pr[\mathrm{Dec}^y_{\mathrm{LDC}}(i) \text{ queries } j] = \Pr_{\tilde{Q}\leftarrow\tilde{\boldsymbol{Q}}_i}[j \in \tilde{Q}] = \Pr_{Q\leftarrow\boldsymbol{Q}_i}[j \in Q \cap L_i \mid Q \text{ is } i\text{-smoothable}]\,.$$

Notice that this probability is 0 if $j \in H_i$, and otherwise for $j \in L_i$ we have

$$\begin{aligned}
\Pr[\mathrm{Dec}^y_{\mathrm{LDC}}(i) \text{ queries } j] &= \Pr_{Q\leftarrow\boldsymbol{Q}_i}[j \in Q \mid Q \text{ is } i\text{-smoothable}] \\
&= \frac{\Pr_{Q\leftarrow\boldsymbol{Q}_i}[j \in Q \wedge Q \text{ is } i\text{-smoothable}]}{\Pr_{Q\leftarrow\boldsymbol{Q}_i}[Q \text{ is } i\text{-smoothable}]} \\
&= \frac{\Pr_{Q\leftarrow\boldsymbol{Q}_i}[j \in Q \wedge Q \text{ is } i\text{-smoothable}]}{p_{i,\mathrm{good}}} \\
&\le \frac{\Pr_{Q\leftarrow\boldsymbol{Q}_i}[j \in Q]}{p_{i,\mathrm{good}}} \\
&\le \frac{q}{\alpha\delta n}\,,
\end{aligned}$$

where we use that $j \in L_i$. Hence, $\mathrm{Dec}_{\mathrm{LDC}}$ is $(\alpha\delta/q)$-smooth. By Fact 3.10, this implies that C is a $(q, \alpha\delta\varepsilon/q, 1, \varepsilon)$-LDC with decoder $\mathrm{Dec}_{\mathrm{LDC}}$, which proves Theorem 4.1 for the case of RLDCs.

Extension to the RLCC case. Let us now briefly explain how to extend the above proof to the case of RLCCs. As we are proceeding by analyzing the behavior of $\mathrm{Dec}(\cdot)$ on each input, this generalizes

seamlessly to RLCCs. The only difference in the proof is that in Lemma 4.3, we take $v^* = v_j$, where v_j is the j-th row of the generator matrix of C and $j \in [n]$ is the input to $\mathrm{Dec}(\cdot)$. That is, v^* is no longer necessarily a standard basis vector e_i. All the remaining steps of the proof are unchanged. □

4.1 Proof of Lemma 4.3

In this subsection, we prove Lemma 4.3.

Proof of Lemma 4.3. Let $Q \subseteq [n]$, and let $Q = H \cup L$ where $H \cap L = \emptyset$. Let $b \in \mathbb{F}^k$, and let $x = C(b)$. Fix $i \in [k]$, and let $\sigma \in \mathbb{F}$. Our goal is to show that, if we choose $b' \leftarrow \mathbb{F}^k$ uniformly at random with $\langle v^*, b' \rangle = \sigma$, then with probability at least $|\mathbb{F}|^{-\min(|H|,|L|)}$, the string $z \in \mathbb{F}^Q$ defined as $z_j = x_j$ for $j \in L$ and $z_j = x'_j$ for $j \in H$ is consistent with some other codeword $C(b'')$ that also satisfies $\langle v^*, b'' \rangle = \sigma$.

Because C is linear, without loss of generality we may assume that $b = 0^k$, so that $z_j = 0$ for all $j \in L$. We wish to argue that $\Pr_z[\exists b'' \text{ s.t. } z = C(b'')|_Q \wedge \langle v^*, b'' \rangle = \sigma] \geq |\mathbb{F}|^{-\min(|H|,|L|)}$, when z is drawn from the distribution defined in Lemma 4.3.

Let us first argue that such a z exists. This implies that the probability is at least $|\mathbb{F}|^{-|Q|}$, which is already sufficent to prove Theorem 4.1 with $s_{\mathbb{F}}(q) = |\mathbb{F}|^{-q}$.

Indeed, because $v^* \notin \mathrm{span}\{v_j : j \in L\}$ and there is a codeword that is identically 0 on all $j \in L$, it follows by Fact 3.5 that such a z exists. Let $z^* = C(b^*)|_Q$ be any such solution, with corresponding codeword b^*.

Because there exists such a b^*, by the linearity of C, it suffices to show that $\Pr_{z \leftarrow \mathcal{D}}[\exists b'' \text{ s.t. } z = C(b'')|_Q \wedge \langle v^*, b'' \rangle = 0] \geq |\mathbb{F}|^{-\min(|H|,|L|)}$, where $\mathcal{D}$ is the distribution over z given by (1) sampling $b' \leftarrow \mathbb{F}^k$ uniformly at random with $\langle v^*, b' \rangle = 0$, and (2) outputting $z \in \mathbb{F}^Q$ where $z_j = 0$ for $j \in L$ and $z_j = C(b')_j$ for $j \in H$. This is because any z satisfies the above condition if and only if $z + z^*$ satisfies the conditions in Lemma 4.3.

Let us now define the following vector spaces: $\mathcal{V}_0 = \{C(b') : \langle v^*, b' \rangle = 0\}$, $\mathcal{W}_0 = \{C(b') : \langle v^*, b' \rangle = 0, C(b')_j = 0 \; \forall j \in L\}$. Note that $\mathcal{W}_0 \subseteq \mathcal{V}_0$, and these are both linear subspaces. We let $\mathcal{V}_0^\perp$ and $\mathcal{W}_0^\perp$ denote their corresponding dual subspaces (Definition 3.3).

By Fact 3.4, $(\mathcal{V}_0^\perp)_{\subseteq H}$ are the set of local constraints defining $\mathcal{V}_0|_H$, and similarly for $(\mathcal{W}_0^\perp)_{\subseteq H}$ and $\mathcal{W}_0|_H$. We will refer to these subspaces often, and so we let $\mathcal{V} := (\mathcal{V}_0^\perp)_{\subseteq H}$ and $\mathcal{W} := (\mathcal{W}_0^\perp)_{\subseteq H}$.

A key fact that we will use is that $((\mathcal{V}_0^\perp)_{\subseteq Q})|_H = \mathcal{W}$. This is immediate by definition, as $(\mathcal{V}_0^\perp)_{\subseteq Q}$ is the set of local constraints that defines $\mathcal{V}_0|_Q$, and when we enforce $C(b')_j = 0$ for all $j \in L$, the restriction of any constraint in $\mathcal{V}_0|_Q$ to the set H is a local constraint in $\mathcal{W}_0$ on the subset $H = Q \setminus L$. For notational convenience, we will let $\mathcal{U} = (\mathcal{V}_0^\perp)_{\subseteq Q}$, so that $\mathcal{U}|_H = \mathcal{W}$, and $\mathcal{U}_{\subseteq H} = \mathcal{V}$.

We now show the following two claims, which together imply Lemma 4.3.

Claim 4.4. $\Pr_{z \leftarrow \mathcal{D}}[\exists b'' \text{ s.t. } z = C(b'')|_Q \wedge \langle v^*, b'' \rangle = 0] \geq |\mathbb{F}|^{-\dim(\mathcal{W}/\mathcal{V})}$.

Claim 4.5. $\dim(\mathcal{W}/\mathcal{V}) \leq \min(|H|, |L|)$.

Indeed, Claims 4.4 and 4.5 imply that $\Pr_{z \leftarrow \mathcal{D}}[\exists b'' \text{ s.t. } z = C(b'')|_Q \wedge \langle v^*, b'' \rangle = 0] \geq |\mathbb{F}|^{-\min(|H|,|L|)}$, which we have already shown suffices to prove Lemma 4.3. □

Proof of Claim 4.4. By definition of the subspace $\mathcal{V}_0$, we have that $w := z|_H$ is uniformly distributed over $\mathcal{V}_0|_H$. By definition of the subspace $\mathcal{V}_0$, we have that z satisfies the desired condition if and only if $w \in \mathcal{W}_0$. Thus, $\Pr_{w \leftarrow \mathcal{V}_0|_H}[w \in \mathcal{W}_0|_H] \geq |\mathbb{F}|^{-t}$ where t is the number of "independent checks" in $\mathcal{W}$ that are not in $\mathcal{V}$. We have that $t = \dim(\mathcal{W}/\mathcal{V})$, which gives us the claim.

More formally, let $r^{(1)}, \ldots, r^{(d)}$ be vectors in $\mathcal{W}$ that are linearly independent in $\mathcal{W}/\mathcal{V}$ (and hence also linearly independent in $\mathcal{W}$). By definition of $\mathcal{W}/\mathcal{V}$, any $r \in \mathcal{W}$ can be expressed as $s + \sum_{j \in T} r^{(j)}$, where $s \in \mathcal{V}$. Now, w satisfies $\langle w, s \rangle = 0$, and hence if $\langle w, r^{(j)} \rangle = 0$ holds for all $j \in [d]$, then $\langle w, r \rangle = 0$ for all $r \in \mathcal{W}$. Finally, observe that because $r^{(1)}, \ldots, r^{(t)}$ are linearly independent in $\mathcal{W}/\mathcal{V}$, the elements $\langle w, r^{(j)} \rangle$ are independent and uniformly random from $\mathbb{F}$ when $w \leftarrow V_0$. Hence, the probability that $\langle w, r^{(j)} \rangle = 0$ for all $j \in [t]$ is at least $|\mathbb{F}|^{-t}$. □

Proof of Claim 4.5. Because $\mathcal{V}, \mathcal{W}$ are subspaces in $\mathbb{F}^H$, it follows that $\dim(\mathcal{W}/\mathcal{V}) \leq \dim(\mathcal{W}) \leq |H|$. Thus, it remains to prove that $\dim(\mathcal{W}/\mathcal{V}) \leq |L|$, which is the nontrivial case.

This proof uses the following key fact that we established earlier: $\mathcal{U}|_H = \mathcal{W}$ and $\mathcal{U}_{\subseteq H} = \mathcal{V}$, where $\mathcal{U} \subseteq \mathbb{F}^Q$ is a subspace.

Suppose that $\dim(\mathcal{W}/\mathcal{V}) \geq |L| + 1$. Let $d = |L| + 1$, and let $r^{(1)}, \ldots, r^{(d)}$ be elements of $\mathcal{W}$ that are linearly independent in $\mathcal{W}/\mathcal{V}$. Because $\mathcal{W} = \mathcal{U}|_H$, for each $j \in [t]$, there exist $s^{(1)}, \ldots, s^{(d)} \in \mathbb{F}^Q$ with $\mathrm{supp}(s^{(j)}) \subseteq L$ such that $r^{(j)} + s^{(j)} \in \mathcal{U}$ and $\mathrm{supp}(r^{(j)} + s^{(j)}) \subseteq H$. Now, because $s^{(1)}, \ldots, s^{(d)}$ are in $\mathbb{F}^Q$ and have support contained in L, they lie in a subspace of dimension at most $|L|$, and since $d = |L| + 1$, they must be linearly dependent. Hence, there exist $\alpha_1, \ldots \alpha_d \in \mathbb{F}$, not all zero, such that $\sum_{j=1}^{d} \alpha_j s^{(j)} = 0$ in $\mathbb{F}^Q$. We then have that

$$\sum_{j=1}^{d} \alpha_j r^{(j)} = \sum_{j=1}^{d} \alpha_j (r^{(j)} + s^{(j)}) - \sum_{j=1}^{d} \alpha_j s^{(j)} = \sum_{j=1}^{d} \alpha_j (r^{(j)} + s^{(j)}) \,.$$

Now, because $\mathrm{supp}(r^{(j)} + s^{(j)}) \subseteq H$ for all j, this implies that $\mathrm{supp}(\sum_{j=1}^{d} \alpha_j (r^{(j)} + s^{(j)})) \subseteq H$ also, and therefore $\sum_{j=1}^{t} \alpha_j (r^{(j)} + s^{(j)})$ is an element of $\mathcal{U}$ whose support is contained in H, i.e., an element of $\mathcal{U}_{\subseteq H} = \mathcal{V}$. Hence, $\sum_{j=1}^{d} \alpha_j (r^{(j)} + s^{(j)}) \in \mathcal{V}$. It thus follows that $\sum_{j=1}^{d} \alpha_j (r^{(j)} + s^{(j)})$ is 0 in $\mathcal{W}/\mathcal{V}$, and hence $\sum_{j=1}^{d} \alpha_j r^{(j)}$ is also 0 in $\mathcal{W}/\mathcal{V}$. Therefore, $r^{(1)}, \ldots, r^{(d)}$ are linearly independent in $\mathcal{W}/\mathcal{V}$, which proves the claim. □

5 Query-Preserving Goldberg Transformation

Up to this point, we have assumed that our RLDC or RLCC has a local decoder/corrector which has *perfect completeness* (always returns the right answer for a valid codeword) and which is *nonadaptive* (the local view is sampled before any queries have been made). We may assume that such a decoder has a *canonical* behavior, as we showed in Fact 3.13, and we rely on this structure in our proofs. However, what if we begin with an RLDC or RLCC with a local decoder that has imperfect completeness, adaptivity, or both? Goldberg's transformation [Gol24a] shows that such a decoder can be transformed (potentially inefficiently) into a nonadaptive decoder with perfect completeness:

Theorem 5.1 ([Gol24a]). *Every linear RLDC or RLCC*[8] *has a nonadaptive decoder with perfect completeness:*

1. *If* $C\colon \mathbb{F}^k \to \mathbb{F}^n$ *is a linear systematic* $(q, \delta, 1-\varepsilon, s)$*-RLDC, then* C *is also a* $(q+1, \delta, 1, s+\varepsilon)$*-RLDC with a nonadaptive decoder.*
2. *If* $C\colon \mathbb{F}^k \to \mathbb{F}^n$ *is a linear* $(q, \delta, 1-\varepsilon, s)$*-RLCC, then* C *is also a* $(q+1, \delta, 1, s+\varepsilon)$*-RLCC with a nonadaptive decoder.*

We can use this theorem to lift our lower bound to general linear RLDCs and RLCCs. However, the extra query (from q to $q+1$) is very costly for our purposes. In particular, it would imply that the lower bound in Corollary 1.6 only applies to *2-RLDCs* with imperfect completeness, rather than 3-RLDCs. As discussed in Section 1, this extra query has a substantial impact on our results. To avoid losing this query, we give an modified analysis of the main result of [Gol24a] that does not lose this additional query. In doing so, we will lose slightly in the soundness error. We also do not need C to be systematic, which was required in [Gol24a].

Theorem 5.2 (General form of Theorem 3). *If* $C\colon \mathbb{F}^k \to \mathbb{F}^n$ *is a linear* $(q, \delta, 1-\varepsilon, s)$*-RLDC, then* C *is also a* $(q, \delta, 1, s + (2 + 1/(|\mathbb{F}|-1))\varepsilon)$*-RLDC with a nonadaptive decoder. The same result holds for RLCCs.*

We will rely on the lemmas and proofs from [Gol24a] and point out where we differ. To begin, we can model any adaptive randomized relaxed local decoder[9] $\mathrm{Dec}(i)$ as a distribution over decision trees $\Gamma \leftarrow \mathcal{D}(i)$, where vertices are labeled with query indices from $[n]$ and edges are labeled with the symbols read from those query indices. Then, each leaf ℓ in each tree Γ corresponds to an ordered query tuple Q and a corresponding string $\sigma \in \mathbb{F}^Q$. Every leaf ℓ is labeled with a symbol from $\mathbb{F} \cup \{\perp\}$, which is the value returned by the decoder. With this in mind, the Goldberg transformation is based around relabeling leaves and rerandomizing the input.

5.1 Relabeling leaves

We first modify the decoder for each $i \in [k]$ to shift the input by a uniformly chosen codeword $C(\tilde{b})$ (and then subtract $\tilde{b}_i$ from the answer if it is not $\perp$); this gives a new decoder Dec_R with the same parameters as the initial decoder. Next, every decision tree leaf which contributes to the completeness error needs to be relabeled. For each $i \in [k]$, these are the leaves in trees $\Gamma \in \mathrm{supp}(\mathcal{D}(i))$ which are reached by some uncorrupted codeword $C(b)$ but which are *not* labeled with b_i. Goldberg shows that changing the label of this leaf to b_i trades completeness error for soundness error, and we can iterate this process to get an adaptive relaxed local decoder with perfect completeness. The input rerandomization is crucial to make this lemma work.

Lemma 5.3 (leaf relabeling, [Gol24a, Claim 18]). *Let* $\mathrm{Dec}_R(i)$ *be the rerandomized decoder for index* i*, and let* $\mathrm{Dec}'_R(i)$ *be the same decoder with a single leaf of a single decision tree relabeled as described. Then, if* $\mathrm{Dec}_R(i)$ *has completeness error* ε *and soundness error* s*, and* $\mathrm{Dec}'_R(i)$ *has completeness error* ε' *and soundness error* s'*, then*

$$s' - s \le \varepsilon - \varepsilon'.$$

[8] [Gol24a] only proves this theorem for binary codes, but the proof easily extends to all finite fields.

[9] This entire section will be written in terms of RLDCs, but the same proof works for RLCCs.

There is a critical aspect to this relabeling step that we have so far overlooked. What if, for some $i \in [k]$, there is a leaf ℓ corresponding to queried values (Q, σ) where the i-th message index is *linearly independent* of the indices in Q (Fact 3.5)? This means that the decoder receives absolutely no information about the i-th message index from its queries, *even* if all of the queries match an uncorrupted codeword. Thus, there is no fixed label we can give to this leaf to reduce completeness error. Call these leaves *toxic*.

Definition 5.4 (Toxic leaves)**.** A leaf (Q, σ) of a decision tree Γ in the support of $\mathcal{D}(i)$ is *toxic* if there exist codewords $C(b), C(b')$ such that $C(b)|_Q = C(b')|_Q = \sigma$, but $b_i \neq b'_i$. By Fact 3.5, the local view Q does not give any information on the i-th index — for any local view and any codeword, there are $|\mathbb{F}| - 1$ other codewords, one for each possible symbol, which look identical on the local view yet differ on the i-th message index. On the other hand, if a leaf is non-toxic, then by Fact 3.5 the i-th message index for any codeword is completely determined by its restriction to Q.

5.2 Isolating toxic leaves

Goldberg disambiguates toxic leaves by adding one extra query to retrieve the value of the desired symbol (this is also why the RLDC must be systematic, so that b_i is part of $C(b)$). This is the query that we wish to save in Theorem 5.2. To avoid losing this extra query, we will isolate the toxic leaves by first relabeling all non-toxic leaves, so that all of the remaining completeness error is caused only by toxic leaves. Then, the following lemma shows that toxic leaves are rarely chosen:

Lemma 5.5. *Let $\mathrm{Dec}_R(i)$ be the rerandomized decoder for index i after all non-toxic leaves have been relabeled. Suppose it has completeness error ε which is caused only by toxic leaves. Then, for all $b \in \mathbb{F}^k$,*

$$\Pr[\mathrm{Dec}_R^{C(b)}(i) \text{ ends on a toxic leaf}] \leq \frac{|\mathbb{F}|}{|\mathbb{F}| - 1} \cdot \varepsilon \,.$$

Proof. Intuitively, the best possible behavior on a toxic leaf is to guess a random symbol from $\mathbb{F}$, in order to minimize the worst-case completeness error across all codewords. The probability of guessing correctly is $1/|\mathbb{F}|$, and so we should get a $|\mathbb{F}|/(|\mathbb{F}| - 1)$ factor relative to ε.

Formally, pick $|\mathbb{F}|$ messages $b^1, \ldots, b^{|\mathbb{F}|}$ where b^j has i-th symbol $j \in \mathbb{F}$. All of these codewords have the same completeness error because $\mathrm{Dec}_R(i)$ rerandomizes the input. We then have

$$\varepsilon = \Pr[\mathrm{Dec}_R^{C(b)}(i) \neq b_i] = \sum_{\text{toxic leaves } \ell} \Pr[\text{leaf } \ell \text{ chosen}] \cdot \Pr[\mathrm{Dec}_R^{C(b)}(i) \neq b_i \mid \text{leaf } \ell \text{ chosen}]$$

Because Dec_R rerandomizes over the input and all of the b^j are codewords, the probability of picking each particular leaf is the same for all of the different messages. Hence,

$$\begin{aligned} |\mathbb{F}|\varepsilon &= \sum_j \Pr[\mathrm{Dec}_R^{C(b^j)}(i) \neq b_i^j] \\ &= \sum_{\text{toxic leaves } \ell} \Pr[\text{leaf } \ell \text{ chosen}] \cdot \left(\sum_j \Pr[\mathrm{Dec}_R^{C(b^j)}(i) \neq b_i^j \mid \text{leaf } \ell \text{ chosen}] \right) \end{aligned}$$

$$= \sum_{\text{toxic leaves } \ell} \Pr[\text{leaf } \ell \text{ chosen}] \cdot \#\{\alpha \in \mathbb{F} : \ell \text{ is } \textit{not} \text{ labeled with } \alpha\}$$
$$\geq (|\mathbb{F}| - 1) \cdot \Pr[\text{toxic leaf chosen}]\,. \qquad \square$$

We now know how often toxic leaves are selected by the decoder, which will help us remove them from the query distribution later on. For now, however, we can relabel them to make additional queries and achieve perfect completeness. When the decoder ends up on a toxic leaf, run a *global decoding* subroutine which queries the entire input and decodes the i-th message index; this shifts the remaining completeness error to soundness error using Lemma 5.3. We use global decoding to avoid requiring that the RLDC is systematic, which was necessary in the original proof of Goldberg. In summary, we have shown the following transformation:

Proposition 5.6. *Let $C\colon \mathbb{F}^k \to \mathbb{F}^n$ be a linear $(q, \delta, 1-\varepsilon, s)$-RLDC (or RLCC). Then, C has an adaptive relaxed local decoder (or corrector) with perfect completeness and soundness error at most $s + \varepsilon$. This decoder, when given a valid codeword as input, selects a non-toxic leaf (and makes q queries) with probability $\geq 1 - (|\mathbb{F}|\varepsilon)/(|\mathbb{F}| - 1)$, and selects a toxic leaf (and makes n queries) otherwise.*

5.3 Removing adaptivity and pruning toxic leaves

Lastly, Goldberg removes adaptivity by showing that we can evaluate the decision tree distribution on a uniformly random codeword to determine the index set to query. Now that the queries are nonadaptive, we can condition on not picking a toxic leaf, which adds a bit more soundness error.

Lemma 5.7 ([Gol24a, Lemma 20])**.** *If C has an adaptive relaxed local decoder* Dec *with perfect completeness and soundness error s, then C has a* nonadaptive *and* canonical *relaxed local decoder with the same completeness and soundness, by selecting a uniformly random codeword c and simulating* Dec *on c to determine its query set Q.*

Proof of Theorem 5.2. Using Proposition 5.6, we can start with C which is a $(q, \delta, 1-\varepsilon, s)$-RLDC and get a relaxed local decoder with perfect completeness and soundness error $s + \varepsilon$, and which selects a relabeled toxic leaf with probability at most $p = (|\mathbb{F}|\varepsilon)/(|\mathbb{F}| - 1)$. Then, use Lemma 5.7 to get a nonadaptive decoder with the same completeness and soundness. Finally, modify the query distribution to condition on never selecting a toxic leaf; the perfect completeness is unharmed but the soundness error will increase from $s + \varepsilon$ to $(s + \varepsilon)/(1 - p) \leq s + \varepsilon + p$, which is $s + \varepsilon(2|\mathbb{F}| - 1)/(|\mathbb{F}| - 1)$. Now, we are also guaranteed that the decoder makes q queries, which finishes the proof. $\square$

6 Constructions of Small Query RLDCs/RLCCs That Are Not LDCs

In this section, we give a simple family of explicit codes that are q-RLDCs/RLCCs where q is a small, explicit constant, and are not q-LDCs for any constant q.

Theorem 6.1 (Formal Theorem 4)**.** *There is a linear code $C\colon \mathbb{F}_2^k \to \mathbb{F}_2^N$ where $N = k^{O(\log\log k)}$ such that for some constant $\delta > 0$, C is: (1) a $(15, \delta, 1, 1/3)$-RLDC; (2) a $(41, \delta, 1, 1/2 - \varepsilon)$-RLCC for a small constant $\varepsilon > 0$; (3) a $(58, \delta, 1, 1/3)$-RLCC; (4) not a $(O(\log k), O(N^{-1/3}), 1, 1/2 - \varepsilon)$-LDC for any (including subconstant) $\varepsilon > 0$.*

We will prove Theorem 6.1 in the next 3 subsections. In Section 6.1, we will define our code. Then, in Section 6.2, we will prove that it is an RLDC, and in Section 6.3, we will prove that is an RLCC. Finally, in Section 6.4 we will prove that it is not an LDC.

6.1 The construction of the code

In this section, we will define the code C in Theorem 6.1. Let $t \in \mathbb{N}$ and let $\mathbb{F}_{2^t}$ be the finite field with 2^t elements. We recall the following basic facts about finite fields.

Fact 6.2 (Finite field notation)**.** *Let $\mathbb{F}_{2^t}$ be the finite field with 2^t elements. The field $\mathbb{F}_{2^t}$ is an $\mathbb{F}_2$-vector space of dimension t, and therefore there is an $\mathbb{F}_2$-linear isomorphism $\pi\colon \mathbb{F}_{2^t} \to \mathbb{F}_2^t$. The map π depends on the choice of basis for $\mathbb{F}_{2^t}$, which we will view as fixed in advance.*

For any $\alpha \in \mathbb{F}_{2^t}$ and $i \in [t]$, we let $\pi(\alpha)_i$ denote the i-th coordinate of α. Because multiplication by α is an invertible $\mathbb{F}_2$-linear transformation in $\mathbb{F}_{2^t}$, there exists a $t \times t$ matrix $M_\alpha \in \mathbb{F}_2^{t\times t}$ such that for any $\beta \in \mathbb{F}_{2^t}$, $\pi^{-1}(M_\alpha \pi(\beta)) = \alpha\beta$. In particular, for each $\alpha \in \mathbb{F}_{2^t}$ and $i \in [t]$, there exists $v \in \mathbb{F}_2^t$ such that for any $\beta \in \mathbb{F}_{2^t}$, $\langle v, \pi(\beta)\rangle = \pi(\alpha\beta)_i$.

The code C is defined formally via its encoding map. We will first define the linear subspace of codewords, and then explain how to define the encoding map.

Let $n, d \in \mathbb{N}$ be parameters with $d < 2^t$, and let $k = \binom{n}{\le d} := \sum_{i=0}^{d}\binom{n}{i}$. Let $f\colon \mathbb{F}_{2^t}^n \to \mathbb{F}_{2^t}$ be a polynomial of degree at most d in n variables $x_1, \dots, x_n$. For each line L in $\mathbb{F}_{2^t}^n$, let S_L be an arbitrary (ordered) subset of L of size $d+1$. For a collection of field elements $(\alpha_1, \dots, \alpha_{d+1}) \in \mathbb{F}_{2^t}^{d+1}$, we let $\mathrm{Had}(\alpha_1, \dots, \alpha_{d+1})$ be the encoding of these field elements using the Hadamard code over $\mathbb{F}_2$ and the map π. That is, $\mathrm{Had}(\alpha_1, \dots, \alpha_{d+1})$ is a vector of length $2^{t(d+1)}$, where entries are indexed by $v = (v_1, \dots, v_{d+1}) \in \mathbb{F}_2^{t(d+1)}$, and the v-th entry is $\sum_{i=1}^{d+1}\langle v_i, \pi(\alpha_i)\rangle$.

With the above setup, we can now specify the set of codewords. For each polynomial $f\colon \mathbb{F}_{2^t}^n \to \mathbb{F}_{2^t}$ of degree at most d, we obtain a codeword by encoding the function f via the bit-wise concatenation of $\mathrm{Had}(f(S_L))$ for each line L. That is, the codeword corresponding to f has, for each line L, a block of $2^{t(d+1)}$ bits that is $\mathrm{Had}(f(S_L))$, so that we can view the codeword as a collection $\{\mathrm{Had}(f(S_L))\}_L$ indexed by all lines L in $\mathbb{F}_{2^t}^n$.

Setting parameters. The set of codewords is clearly an $\mathbb{F}_2$-linear subspace, and it has dimension tk, where $k = \binom{n}{\le d}$. Moreover, it is a linear subspace in $\mathbb{F}_2^{\#L\cdot 2^{t(d+1)}}$, where $\#L = 2^{tn}(2^{tn}-1)/(2^t(2^t-1))$ is the number of lines in $\mathbb{F}_{2^t}^n$. To minimize the blocklength as a function of k, we take $d = O(n)$ (which forces $t = \Theta(\log d) = \Theta(\log n)$, as we need $2^t > d$), so that $k = 2^{O(n)}$ and the blocklength is $2^{O(n\log n)}$. Hence, the blocklength is $k^{O(\log\log k)}$.

Choosing an encoding map. To finish defining the code C, we need to specify the encoding map. To do this, we first choose an arbitrary $\mathbb{F}_{2^t}$-linear map A from $\mathbb{F}_{2^t}^k$ to the set of degree $\le d$ polynomials f with the property that there are elements $x^{(1)}, \dots, x^{(k)} \in \mathbb{F}_{2^t}$ such that if $f = Ab$ for $b \in \mathbb{F}_{2^t}$, then $f(x^{(i)}) = b_i$ for all i. We can then extend A to a map from $\mathbb{F}_2^{tk}$ to the set of degree $\le d$ polynomials by splitting the input into k blocks of size t, applying the linear map π^{-1} (Fact 6.2) on each block, applying A to the output, and then applying π to each field element in the evaluation table of the resulting polynomial f.

As we will show, our RLDC decoder, when given access to a corrupted version of the codeword corresponding to the polynomial f, will be able to recover $\pi(\alpha f(x))_i$ for every $\alpha \in \mathbb{F}_{2^t}$, $x \in \mathbb{F}_{2^t}^n$, and any $i \in [t]$. As a result, the specific choice of encoding map A specified above is not important.

Comparison to [AS21]. Our construction shares some similarities to the work of [AS21]. Similar to our construction, they first use a Reed–Muller code to encode a message as a polynomial f, and then they encode each "local view" of f in some way. Their "local views", however, are a special set of planes $\mathcal{P}$ that have "directions" in $\mathbb{H}^n$, where $\mathbb{H}$ is a subfield of $\mathbb{F}$. They then encode $f|_{\mathcal{P}}$ for each plane $\mathcal{P}$ using a canonical correctable Probabilistically Checkable Proof of Proximity (ccPCPP). Their decoder then takes certain random walk of length $[\mathbb{F} : \mathbb{H}] + 1$ on the special planes $\mathcal{P}$, and uses the ccPCPPs to decode values of f and do consistency checks.

In contrast, we encode f by encoding its "local view" $f|_L$ for every line L using the Hadamard code. As we shall see, our RLDC decoder takes a random walk of length 2 on the lines, and our RLCC decoder takes a random walk of length 3.

An observation about C. Finally, we make some observations about C, which will be useful in the proofs.

Observation 6.3. Fix a line L. For any $z \in L$, any $\alpha \in \mathbb{F}_{2^t}$, and any $i \in [t]$, there exists a point $v^{L,z,\alpha,i} \in \mathbb{F}_2^{t(d+1)}$ such that $\mathrm{Had}(f(S_L))(v^{L,z,\alpha,i}) = \pi(\alpha f(z))_i$. That is, for any $i \in [t]$, one can recover the i-th bit of $\alpha \cdot f(z)$ using the Hadamard encoding of (only) $d+1$ points S_L on the line L.

The point $v^{L,z,\alpha,i}$ should be interpreted as: to recover the i-th bit of $\alpha f(z)$ using the Hadamard encoding of $f|_L$, we query the Hadamard encoding of $f|_L$ at the point $v^{L,z,\alpha,i}$.

Proof. Let $z_1, \ldots, z_{d+1}$ be the points in S_L. Because f is a degree d polynomial and $d < 2^t = |\mathbb{F}_{2^t}|$, by polynomial interpolation, there exist coefficients $\alpha_1, \ldots, \alpha_{d+1} \in \mathbb{F}_{2^t}$ such that $f(z) = \sum_{j=1}^{d+1} \alpha_j f(z_j)$. Therefore, $\pi(\alpha f(z))_i = \sum_{j=1}^{d+1} \pi(\alpha\alpha_j f(z_j))_i$. For each j, let v^j be the i-th row of $M_{\alpha\alpha_j}$, the matrix defined in Fact 6.2. The vector v^j then has the property that $\langle v^j, \pi(\beta)\rangle = \pi(\alpha\alpha_j\beta)_i$ for all $\beta \in \mathbb{F}_{2^t}$, and therefore $\pi(\alpha\alpha_j f(z_j))_i = \langle v^j, \pi(f(z_j))\rangle$. Concatenating the vectors v^j together yields the vector $v^{L,z,\alpha,i}$. □

6.2 The RLDC decoder and its analysis

In this subsection, we prove Item (1) in Theorem 6.1, where the code is defined in Section 6.1.

Proof of Item (1) in Theorem 6.1. To show that the code is an RLDC, we will analyze the decoder defined below.

Algorithm 6.4.

Given: A collection of functions $\{G_L\}_L$ where each $G_L : \mathbb{F}_2^{t(d+1)} \to \mathbb{F}_2$ is an arbitrary function, along with a point $x^* \in \mathbb{F}_{2^t}^n$, a field element $\alpha^* \in \mathbb{F}_{2^t}$, and an index $i^* \in [t]$. The collection $\{G_L\}_L$ is supposed to be equal to $\{\mathrm{Had}(f(S_L))\}_L$ for some polynomial f of degree $\leq d$.

Output: A symbol in $\{0, 1, \perp\}$, hopefully equal to $\pi(\alpha^* f(x^*))_{i^*}$.

Operation:

(1) Choose a line L^* containing x uniformly at random.

(2) **Run r_1 linearity tests on G_{L^*}:** pick random $v^{(1)}, v^{(2)}, v^{(3)} \in \mathbb{F}_2^{t(d+1)}$, and check that $G_{L^*}(v^{(1)}) + G_{L^*}(v^{(2)}) + G_{L^*}(v^{(3)}) = 0$. If the check fails, output $\perp$. Repeat r_1 times.

(3) **Run r_2 consistency tests:**

(a) Pick $y \in L^*$ uniformly at random, $\alpha \in \mathbb{F}_{2^t}$ uniformly at random, and $i \in [t]$.

(b) **Locally decode $\pi(\alpha f(y))_i$ from G_{L^*}:** let $v^{L,y,\alpha,i}$ be the vector from Observation 6.3. Let $v^{(4)} \in \mathbb{F}_2^{t(d+1)}$ be chosen uniformly at random. Let $a_{L^*} = G_{L^*}(v^{L^*,y,\alpha,i} + v^{(4)}) + G_{L^*}(v^{(4)})$.

(c) **Locally decode $\pi(\alpha f(y))_i$ from $G_{L'}$:** choose L' to be a uniformly random line containing y. Let $v^{L',y,\alpha,i}$ be the vector from Observation 6.3. Let $v^{(5)} \in \mathbb{F}_2^{t(d+1)}$ be chosen uniformly at random. Let $a_{L'} = G_{L'}(v^{L',y,\alpha,i} + v^{(5)}) + G_{L'}(v^{(5)})$.

(d) **Consistency check:** check that $a_{L^*} = a_{L'}$ and output $\perp$ if the check fails.

(e) Repeat r_2 times.

(4) **Locally decode $\pi(\alpha^* f(x^*))_{i^*}$ from G_{L^*}:** let v^{L^*,x^*,α^*,i^*} be the vector from Observation 6.3. Let $v^{(6)} \in \mathbb{F}_2^{t(d+1)}$ be chosen uniformly at random. Output $G_{L^*}(v^{L^*,x^*,\alpha^*,i^*} + v^{(6)}) + G_{L^*}(v^{(6)})$.

In the above algorithm, r_1 and r_2 are positive integers, which we will choose later.

First, we observe that by Observation 6.3, Algorithm 6.4 returns $\pi(\alpha^* f(x^*))_{i^*}$ for any input (x^*, α^*, i^*) if the collection of functions is indeed $\{\mathrm{Had}(f(S_L))\}_L$ for some polynomial f of degree $\leq d$. We note that our choice of encoding map, the message bits correspond to evaluations of f on specific points, and so this allows us to recover any bit of the message. Thus, Algorithm 6.4 has perfect completeness. We also note that Algorithm 6.4 makes $3r_1 + 4r_2 + 2$ queries, though we will explain how to slightly reduce the query complexity later.

We now analyze the soundness error of Algorithm 6.4. For each line L, let H_L be the closest linear function to G_L. Given a linear function H_L, there is a unique univariate degree d polynomial h_L on the line L such that for each $z \in L$, $\alpha \in \mathbb{F}_{2^t}$, and $i \in [t]$, $\pi(\alpha h_L(z))_i = H_L(v^{L,z,\alpha,i})$. The polynomial h_L is simply defined by "extracting" the values of h_L on S_L using Observation 6.3 and then defining h_L on the rest of the line using polynomial interpolation. We let F_L be the linear function $\mathrm{Had}(f(S_L))$, and we define f_L to be the polynomial $f|_L$. We note that the process used to obtain h_L from H_L yields f_L if $H_L = F_L$. For two Boolean functions F_L and G_L, we let $\bar{\Delta}(F_L, G_L)$ denote the (relative) Hamming distance over $\mathbb{F}_2$, and for two polynomials f_L and g_L, we let $\bar{\Delta}(f_L, g_L)$ denote the (relative) Hamming distance over $\mathbb{F}_{2^t}$, i.e., the fraction of $x \in L$ such that $f_L(x) \neq g_L(x)$.

We will split our analysis into the following disjoint cases, and bound the probability that decoder errs in each case.

(1) The random line L^* through x^* is "bad", in that $\mathbb{E}_{L':|L'\cap L^*|\geq 1, x^*\notin L'}[\bar{\Delta}(G_{L'}, F_{L'})] \geq \delta^{2/3}$.

(2) $f_{L^*} \neq h_{L^*}$ and $\mathbb{E}_{L':|L'\cap L^*|\geq 1, x^*\notin L'}[\bar{\Delta}(G_{L'}, F_{L'})] \leq \delta^{2/3}$, but both the repeated linearity test and the repeated consistency test pass.

(3) $f_{L^*} = h_{L^*}$, but the repeated linearity test passes and the output is $1 - \pi(\alpha^* f(x^*))_{i^*}$ (i.e., not in $\{\pi(\alpha^* f(x^*))_{i^*}, \perp\}$).

Analysis of Case 1. Recall that the collection $\{G_L\}_L$ is δ-close to $\{F_L\}_L$. Equivalently, this means that $\mathbb{E}_L[\bar{\Delta}(G_L, F_L)] \leq \delta$. Since L^* is a uniformly random line passing through x^*, and L' is a uniformly random line passing through a random point $y \neq x^*$ on L^*, it follows that L' is distributed as a uniformly random line. Therefore, $\mathbb{E}_L[\mathbb{E}_{L':|L'\cap L|\geq 1, x^*\notin L'}[\bar{\Delta}_L(G_L, F_L)]] = \delta$. It follows by Markov's inequality that the probability over L that $\mathbb{E}_{L':|L'\cap L|\geq 1, x^*\notin L'}[\bar{\Delta}_L(G_L, F_L)] \geq \delta^{2/3}$ is at most $\delta^{1/3}$.

Analysis of Case 2. Let $\varepsilon = \bar{\Delta}(G_{L^*}, H_{L^*})$. Recall by Fact 3.16, the probability that one iteration of the linearity test passes is at most $1 - \bar{\Delta}(G_{L^*}, H_{L^*}) = 1 - \varepsilon$. Hence, the probability that all r_1 repetitions pass is at most $(1-\varepsilon)^{r_1}$.

Let us now analyze one iteration of the consistency test. Call the line L' chosen "bad" if $\bar{\Delta}(G_{L'}, F_{L'}) \geq \delta^{1/3}$. Since $\mathbb{E}_{L':|L'\cap L^*|\geq 1, x^*\notin L'}[\bar{\Delta}(G_{L'}, F_{L'})] \leq \delta^{2/3}$, it follows that the probability that L' is "bad" is at most $\delta^{1/3}$.

Let us proceed assuming that L' is not "bad". Then, since $F_{L'}$ is a linear function, by Fact 3.17 and Observation 6.3, it follows that $a_{L'}$ is equal to $\pi(\alpha f_{L^*}(y))_i$ (which is $\pi(\alpha f(y))_i$) with probability at least $1 - 2\delta^{1/3}$. Similarly, because $\bar{\Delta}(G_{L^*}, H_{L^*}) = \varepsilon$, it follows that a_{L^*} is equal to $\pi(\alpha h_{L^*}(y))_i$ with probability at least $1 - 2\varepsilon$. Moreover, this holds regardless of the choice of α and i.

Finally, because $f_{L^*} \neq h_{L^*}$ and these are different degree d polynomials, the probability that $f_{L^*}(y) \neq h_{L^*}(y)$ is at least $1 - d/n$. And, if such a y is chosen, the probability that $\pi(\alpha f_{L^*}(y))_i \neq \pi(\alpha h_{L^*}(y))_i$ is $1/2$, as $\alpha(f_{L^*}(y) - h_{L^*}(y))$ is a random element of $\mathbb{F}_{2^t}$, and so its i-th bit is 1 with probability $1/2$.

Thus, we can conclude that each round of the consistency test passes with probability at most $\delta^{1/3} + d/n + (1 - d/n)(\frac{1}{2} + \frac{1}{2}(2\delta^{1/3} + 2\varepsilon))$. Since the tests are independent, we conclude that the probability that all rounds of both tests pass is at most $(1-\varepsilon)^{r_1}\left(\delta^{1/3} + d/n + (1 - d/n)(\frac{1}{2} + \delta^{1/3} + \varepsilon)\right)^{r_2}$.

Analysis of Case 3. By the previous analysis, the probability that all linearity tests pass is at most $(1-\varepsilon)^{r_1}$, where $\varepsilon = \bar{\Delta}(G_{L^*}, H_{L^*})$. By Fact 3.17, the output is $\pi(h_{L^*}(x^*))_{i^*}$ with probability at least $1 - 2\varepsilon$, which is $\pi(f_{L^*}(x^*))_{i^*} = \pi(f(x^*))_{i^*}$ since $h_{L^*} = f_{L^*}$. Thus, the probability that the decoder outputs an incorrect answer is at most $(1-\varepsilon)^{r_1} \cdot 2\varepsilon$ in this case.

In total, we conclude that the probability that the output of the decoder is not in $\{\pi(f(x^*))_i, \perp\}$ is at most

$$\eta(\varepsilon) = \max\{\delta^{1/3}, (1-\varepsilon)^{r_1}\left(\delta^{1/3} + d/n + (1 - d/n)(1/2 + \delta^{1/3} + \varepsilon)\right)^{r_2}, (1-\varepsilon)^{r_1} \cdot 2\varepsilon\},$$

where $\varepsilon = \bar{\Delta}(G_{L^*}, H_{L^*})$. Setting $r_1 = r_2 = 2$, δ and d/n to be sufficiently small constants, and taking the maximum over all ε shows that $\eta \leq \frac{1}{2} - \varepsilon'$ for some constant ε'. In fact, by taking δ and d/n to be sufficiently small constants, we can make $\eta = (3/4)^4 + \varepsilon'' \leq 1/3$ for a small constant ε''.

The query complexity is $3r_1 + 4r_2 + 2 = 16$. Below, we shall explain how to save one query.

Saving a query. Observe that in the above analysis, we analyzed Cases (2) and (3) separately. This allows us to "recycle" our queries across the different cases. Namely, we observe that in Item (3b) in Algorithm 6.4, it holds that $v^{(4)}$ is chosen uniformly at random. Because we do not

use the consistency test to analyze Case (3) above, we can reuse this query by taking $v^{(6)} = v^{(4)}$ and the analysis does not change. This allows us to make the query complexity slightly smaller: $3r_1 + 4r_2 + 1 = 15$. □

6.3 The RLCC decoder and its analysis

In this subsection, we prove Items (2) and (3) in Theorem 6.1. The key difference between the RLCC decoder and RLDC decoder is that the RLCC decoder may be asked to decode an arbitrary point $v \in \mathbb{F}_2^{t(d+1)}$ on the Hadamard encoding of some line L^*. Unlike in the case of the RLDC decoder, where the point v corresponded to decoding $\pi(f(x))_i$ for some point x and some index i, here the point v might not correspond to any evaluation point of the polynomial f.

Proof of Items (2) and (3) in Theorem 6.1. To show that the code is an RLCC, we will analyze the decoder defined below.

Algorithm 6.5.

Given: A collection of functions $\{G_L\}_L$ where each $G_L \colon \mathbb{F}_2^{t(d+1)} \to \mathbb{F}_2$ is an arbitrary function, along with a point $v^* \in \mathbb{F}_2^{t(d+1)}$ and a line L^*. The collection $\{G_L\}_L$ is supposed to be equal to $\{\mathrm{Had}(f(S_L))\}_L$ for some polynomial f of degree $\leq d$.

Output: A symbol in $\{0, 1, \perp\}$, hopefully equal to $\mathrm{Had}(f(S_{L^*}))(v^*)$.

Operation:

(1) **Run r_1 linearity tests on G_{L^*}:** pick random $v^{(1)}, v^{(2)}, v^{(3)} \in \mathbb{F}_2^{t(d+1)}$, and check that $G_{L^*}(v^{(1)}) + G_{L^*}(v^{(2)}) + G_{L^*}(v^{(3)}) = 0$. If the check fails, output $\perp$. Repeat r_1 times.

(2) **Run r_2 consistency tests times:**

  (1) Pick $y \in L^*$ uniformly at random, $\alpha \in \mathbb{F}_{2^t}$ uniformly at random, and $i \in [t]$.

  (2) **Locally decode $\pi(\alpha f(y))_i$ from G_{L^*}:** let $v^{L,y,\alpha,i}$ be the vector from Observation 6.3. Let $v^{(4)} \in \mathbb{F}_2^{t(d+1)}$ be chosen uniformly at random. Let $a_{L^*} = G_{L^*}(v^{L^*,y,\alpha,i} + v^{(4)}) + G_{L^*}(v^{(4)})$.

  (3) **Locally decode $\pi(\alpha f(y))_i$ using the RLDC decoder:** Run the RLDC decoder in Algorithm 6.4 to recover $\pi(\alpha f(y))_i$. If the RLDC decoder outputs $\perp$, then abort and output $\perp$. Else, let $a_{\mathrm{RLDC}} \in \{0, 1\}$ be the output of the RLDC decoder.

  (4) **Consistency check:** check that $a_{L^*} = a_{\mathrm{RLDC}}$ and output $\perp$ if the check fails.

  (5) Repeat r_2 times.

(3) **Locally decode $\mathrm{Had}(f(S_{L^*}))(v^*)$ from G_{L^*}:** Let $v^{(6)} \in \mathbb{F}_2^{t(d+1)}$ be chosen uniformly at random. Output $G_{L^*}(v^* + v^{(6)}) + G_{L^*}(v^{(6)})$.

The analysis of Algorithm 6.5 is similar to the analysis of Algorithm 6.4. Below, we will reuse the same notation as done in Section 6.2. The fact that the decoder has perfect completeness is straightforward, and so we proceed with analyzing the soundness error.

As before, will split our analysis into disjoint cases, and bound the probability that decoder errs in each case.

(1) $f_{L^*} \neq h_{L^*}$, but both the repeated linearity test and repeated consistency test pass.

(2) $f_{L^*} = h_{L^*}$, but the repeated linearity test passes and the output is $1 - \mathrm{Had}(f(S_{L^*}))(v^*)$ (i.e., not in $\{\mathrm{Had}(f(S_{L^*}))(v^*), \perp\}$).

Analysis of Case 1. The probability that all r_1 repetitions of the linearity pass when $\bar{\Delta}(G_{L^*}, H_{L^*}) = \varepsilon$ is at most $(1-\varepsilon)^{r_1}$ by Fact 3.16.

Let us now analyze one iteration of the consistency test. Because $f_{L^*} \neq h_{L^*}$ and these are different degree d polynomials, the probability that $f_{L^*}(y) \neq h_{L^*}(y)$ is at least $1 - d/n$. Suppose that such a y is chosen. Then, the probability that $\pi(\alpha f_{L^*}(y))_i \neq \pi(\alpha h_{L^*}(y))_i$ is $1/2$, as $\alpha(f_{L^*}(y) - h_{L^*}(y))$ is a random element of $\mathbb{F}_{2^t}$, and so its i-th bit is 1 with probability $1/2$.

Let us assume that y, α and i are "good", meaning that $\pi(\alpha f_{L^*}(y))_i \neq \pi(\alpha h_{L^*}(y))_i$. Because $\bar{\Delta}(G_{L^*}, H_{L^*}) \leq \varepsilon$, it follows that a_{L^*} is equal to $\pi(\alpha h_{L^*}(y))_i$ with probability at least $1 - 2\varepsilon$. Moreover, this holds regardless of the choice of α and i.

We now invoke the soundness of the RLDC decoder. The RLDC decoder either outputs the correct bit $\pi(\alpha f_{L^*}(y))_i$, or else it outputs $\perp$, with probability at least $1 - \eta_{\mathrm{RLDC}}$ for some constant η_{RLDC}. Hence, the probability that the RLDC decoder outputs the wrong bit is at most η_{RLDC}.

Thus, the probability that one round of the consistency test passes is at most $d/n + \frac{1}{2}(1 - d/n) + \frac{1}{2}(1 - d/n)(\eta_{\mathrm{RLDC}} + (1 - \eta_{\mathrm{RLDC}})2\varepsilon)$. Because the consistency test and linearity tests are independent, the total probability of all tests passing is at most

$$(1-\varepsilon)^{r_1}\left(d/n + \frac{1}{2}(1-d/n)(1 + \eta_{\mathrm{RLDC}} + (1-\eta_{\mathrm{RLDC}})2\varepsilon)\right)^{r_2}.$$

Analysis of Case 2. The probability that all r_1 repetitions of the linearity pass when $\bar{\Delta}(G_{L^*}, H_{L^*}) = \varepsilon$ is at most $(1-\varepsilon)^{r_1}$ by Fact 3.16. By Fact 3.17, the output is $h_{L^*}(v^*)$ with probability at least $1 - 2\varepsilon$, which is $\mathrm{Had}(f(S_{L^*}))(v^*)$ since $h_{L^*} = f_{L^*}$. As these are independent, the probability that the decoder outputs an incorrect answer is at most $(1-\varepsilon)^{r_1} \cdot 2\varepsilon$ in this case.

In total, we conclude that the probability that output of the decoder is incorrect is at most

$$\eta(\varepsilon) = \max\{(1-\varepsilon)^{r_1}\left(d/n + \frac{1}{2}(1-d/n)(1 + \eta_{\mathrm{RLDC}} + (1-\eta_{\mathrm{RLDC}})2\varepsilon)\right)^{r_2}, (1-\varepsilon)^{r_1} \cdot 2\varepsilon\},$$

where $\varepsilon = \bar{\Delta}(G_{L^*}, H_{L^*})$. Recall that $\eta_{\mathrm{RLDC}} < 1/3$ provided that δ is a sufficiently small constant. Hence, setting $r_1 = r_2 = 2$, d/n to be a sufficiently small constant, and maximizing $\eta(\varepsilon)$ over all ε, we see that the maximum is at most $\frac{4}{9} + \varepsilon'' < 1/2$ for some sufficiently small constant ε''.

Setting $r_1 = 2$ and $r_2 = 3$, we get that $\eta \leq (2/3)^3 + \varepsilon'' < 1/3$ for a sufficiently small constant ε''.

The query complexity of the decoder is $3r_1 + (2 + q_{\mathrm{RLDC}})r_2 + 1$ (using the trick to reduce the query complexity by 1 from Section 6.2). Since $q_{\mathrm{RLDC}} = 15$, this yields 41 queries for the case of soundness error below $1/2$, and 58 queries for the case of soundness error below $1/3$. □

6.4 The code is not an LDC

In this section, we prove Item (4) in Theorem 6.1, i.e., that the code constructed in Section 6.1 is not an LDC.

Proof of Item (4) in Theorem 6.1. Recall that the code is defined as follows. For each degree $\leq d$ polynomial $f : \mathbb{F}_{2^t}^n \to \mathbb{F}_{2^t}$ in n variables, we encode f by writing down $\mathrm{Had}(f(S_L))$ for each line L in $\mathbb{F}_{2^t}^n$, where we have set $d = \Theta(n)$, $t = \Theta(\log d) = \Theta(\log n)$, and we have $k = t\binom{n}{\leq d}$. The blocklength of the code is $N = \#L \cdot 2^{t(d+1)}$, where $\#L = 2^{tn}(2^{tn} - 1)/(2^t(2^t - 1))$ is the number of lines in $\mathbb{F}_{2^t}^n$. We will show that C is not a $(q, \delta, 1, 1/2 - \varepsilon)$-LDC for any $\varepsilon > 0$ (that need not be constant) for $q = d$ and $\delta = 2^{-n(t-1)}$. Note that $q = d \leq O(\log k)$ and $\delta = 2^{-n(t-1)} \leq O(N^{-1/3})$.

Setup. To show that the code is not a $(d, 2^{-n(t-1)}, 1, 1/2 - \varepsilon)$-LDC for any $\varepsilon > 0$, it suffices to show that for any d-query LDC decoder Dec and any choice of $\alpha^* \in \mathbb{F}_{2^t}$, $x^* \in \mathbb{F}_{2^t}^n$, and $i^* \in [t]$, there is a collection of "local encodings" $\{h_L\}_L$ that is $2^{-n(t-1)}$-close to a codeword $\{\mathrm{Had}(f(S_L))\}_L$ such that $\Pr[\mathrm{Dec}^{\{h_L\}_L}(\alpha^*, x^*, i^*)] = \frac{1}{2}$.

First, we observe that it suffices to argue this in the *erasure* error model, as opposed to the standard Hamming error model. In the erasure model, we corrupt a codeword by replacing a bit in the encoding with an error symbol $\perp$ to signify that the bit has been erased. That is, the outputs of the "local encodings" h_L can only agree with $\mathrm{Had}(f(S_L))$ or be $\perp$. Clearly, if a decoder works in the standard Hamming error model, then it also works in the erasure error model, as we can simulate the Hamming error model by replacing any $\perp$ symbol by a 0. Hence, it suffices to argue that there is no decoder in the erasure error model. In fact, it suffices to argue this even in the following weaker "line erasure error model", where we are only allowed to erase entire lines. That is, for each line L, we either have $h_L \equiv \mathrm{Had}(f(S_L))$, i.e., they are the same function, or $h_L \equiv \perp$, i.e., the entire function outputs $\perp$.

Now that we are working in the line erasure error model, we will strengthen the decoder by allowing it to make "line queries". That is, instead of allowing the decoder to make one query to retrieve an evaluation of h_L for a line L and evaluation point of its choice, we will instead allow the decoder to make "line queries" where the decoder reads the entire evaluation table of a function h_L with one "line query L". Clearly, any q-query decoder in the standard query model can be simulated in the line query model with $\leq q$ queries, as any query to a specific evaluation of a local function h_L can simply be replaced by a query to the entire line L. Notice that since the decoder makes line queries and each local function $\mathrm{Had}(f(S_L))$ is either completely replaced with $\perp$ or is untouched, we may further assume that the decoder receives $f|_L$ when it makes the line query L, as opposed to $\mathrm{Had}(f(S_L))$, as it can simply compute $\mathrm{Had}(f(S_L))$ from $f|_L$. Thus, we will view the decoder as having "line query" access to the polynomial f, where for every line it either receives $f|_L$ (the entire line) or $\perp$ (if the line has been erased). For a set of *erased* lines $\mathcal{L}$, we will use the notation $\mathrm{Dec}^{f,\mathcal{L}}$ to indicate that the decoder has line query access to the function f on all lines *except* those in $\mathcal{L}$. Namely, if the decoder queries a line L and $L \in \mathcal{L}$, then it receives $\perp$, and otherwise it receives $f|_L$.

Lower bound against "line query" decoders in the erasure model. We will now show the following. For any "line query" decoder Dec making at most d queries and for any $\alpha^* \in \mathbb{F}_{2^t}$, $x^* \in \mathbb{F}_{2^t}^n$, and

$i^* \in [t]$, there is a set of δ-fraction of lines $\mathcal{L}$ that can be erased such that $\mathbb{E}_f[\Pr[\mathrm{Dec}^{f,\mathcal{L}}(\alpha^*, x^*, i^*) = \pi(\alpha^* f(x^*))_{i^*}]] = \frac{1}{2}$, where the outer expectation is over the random choice of the polynomial f and the inner probability is over the randomness of the decoder. In particular, this implies that there exists a polynomial f of degree $\leq d$ such that $\Pr[\mathrm{Dec}^{f,\mathcal{L}}(\alpha^*, x^*, i^*) = \pi(\alpha^* f(x^*))_{i^*}]] \leq \frac{1}{2}$, which will finish the proof.

Let us now argue the above claim. First, the set $\mathcal{L}$ of lines will be all lines L with $x^* \in L$, which we denote by $\mathcal{L}_{x^*}$. We observe that the number of such lines is $(2^{tn}-1)/(2^t-1)$, which is a $\delta = 2^{-(t-1)n}$-fraction of all lines. As we permit the decoder to be adaptive, the queries made by the decoder can be described as a depth $d+1$ decision tree (with d layers of internal "decision" or "query" nodes and one layer of leaves or output nodes), where each decision node has a line L that will be queried as well as a child for each possible evaluation table $f|_L$, and the output of the decoder is given by the leaves. Note that we may assume without loss of generality that the decoder makes exactly d queries, so that the tree is a complete tree with depth exactly $d+1$. Furthermore, the decision tree is determined by the input (α^*, x^*, i^*) to the decoder and the choice r of the decoder randomness. That is, for each input (α^*, x^*, i^*) and randomness r, there is a tree $T_{(\alpha^*, x^*, i^*, r)}$ that determines the queries that can be made when the decoder randomness is r. We use the notation $T_{(\alpha^*, x^*, i^*, r)}(f, \mathcal{L})$ to denote the output of the tree $T_{(\alpha^*, x^*, i^*, r)}$ when given access to f on all lines not in $\mathcal{L}$, which we note is equal to $\mathrm{Dec}^{f,\mathcal{L}}(\alpha^*, x^*, i^*; r)$, the output of the decoder when the randomness is r. We thus have

$$\begin{aligned}&\mathbb{E}_f[\Pr[\mathrm{Dec}^{f,\mathcal{L}_{x^*}}(\alpha^*, x^*, i^*) = \pi(\alpha^* f(x^*))_{i^*}]] = \mathbb{E}_f[\mathbb{E}_r[\mathbf{1}(T_{(\alpha^*, x^*, i^*, r)}(f, \mathcal{L}) = \pi(\alpha^* f(x^*))_{i^*})]] \\ &= \mathbb{E}_r[\mathbb{E}_f[\mathbf{1}(T_{(\alpha^*, x^*, i^*, r)}(f, \mathcal{L}) = \pi(\alpha^* f(x^*))_{i^*})]]\,,\end{aligned}$$

using linearity of expectation, where the expectation over f is over a random degree $\leq d$ polynomial and the expectation over r is over the randomness r of the decoder.

We will now argue that for any decision tree T of depth $d+1$ that does not query lines in $\mathcal{L}_{x^*}$, it holds that $\mathbb{E}_f[\mathbf{1}(T(f) = \pi(\alpha^* f(x^*))_{i^*})] = \frac{1}{2}$. (Note that we can replace $T_{(\alpha^*, x^*, i^*, r)}$ with an equivalent decision tree that never queries lines in $\mathcal{L}_{x^*}$.) This follows from the following observation, which we will prove at the end of the subsection.

Claim 6.6. Let $x \in \mathbb{F}_{2^t}^n$ and let $L_1, \ldots, L_q$ be lines in $\mathbb{F}_{2^t}^n$ that do not contain x. Then, there is a degree-q polynomial $g\colon \mathbb{F}_{2^t}^n \to \mathbb{F}_{2^t}$ such that $g(y) = 0$ for all $y \in \cup_{i=1}^q L_i$ and $g(x) = 1$.

For a set of d lines $L_1, \ldots, L_d$, let $g_{x^*, L_1, \ldots, L_d}$ be the polynomial in Claim 6.6 (which may not be unique, but we choose an arbitrary fixed one for each choice of lines). Now, consider the following distribution: (1) sample f uniformly at random, (2) let $L_1^{(f)}, \ldots, L_d^{(f)}$ be the lines queried by $T(f)$ (which are not in $\mathcal{L}_{x^*}$), and (3) output $f + \beta g_{x^*, L_1^{(f)}, \ldots, L_d^{(f)}}$ where $\beta \leftarrow \mathbb{F}_{2^t}$ uniformly at random. Note that this is well-defined since $x \notin L_i^{(f)}$ for all $i \in [d]$, as $L_i^{(f)} \notin \mathcal{L}_{x^*}$. We claim that this distribution is simply uniform on degree $\leq d$ polynomials, i.e., it is the same as choosing f uniformly at random. To see this, let $S_f = \{f + \beta g_{x^*, L_1^{(f)}, \ldots, L_d^{(f)}}\}_{\beta \in \mathbb{F}_{2^t}}$. Observe that for any $h \in S_f$, we have that $S_h = S_f$. Indeed, this follows because $g_{x^*, L_1^{(f)}, \ldots, L_d^{(f)}}(y) = 0$ for all $y \in \cup_{i=1}^d L_i^{(f)}$, and so it follows that h agrees with f on all lines $L_1^{(f)}, \ldots, L_d^{(f)}$, and hence the decision tree must follow the same root-to-leaf path

for both h and f. In particular, this also implies that $T(f) = T(h)$ for all $h \in S_f$. It thus follows that the S_f's partition the set of degree $\leq d$ polynomials into sets of size 2^t. Hence, the above distribution simply first chooses a set S_f in the partition uniformly at random, and then chooses a random polynomial in S_f, which is the same as choosing a uniformly random degree $\leq d$ polynomial.

Now, to finish the argument, we have that

$$\mathbb{E}_f[\mathbf{1}(T(f) = \pi(\alpha^* f(x^*))_{i^*})] = \mathbb{E}_f[\mathbb{E}_{h \in S_f}[\mathbb{E}_\beta[\mathbf{1}(T(h) = \pi(\alpha^* h(x^*))_{i^*})]]]$$
$$= \mathbb{E}_f[\mathbb{E}_{h \in S_f}[\mathbb{E}_\beta[\mathbf{1}(T(f) = \pi(\alpha^* h(x^*))_{i^*})]]] = \frac{1}{2},$$

where the third equality uses that $T(f) = T(h)$ for all $h \in S_f$ and the fourth inequality uses that $g_{x^*, L_1^{(f)}, \ldots, L_d^{(f)}}(x^*) = 1$, so that $\pi(\alpha^* h(x^*))_{i^*})$ is distributed as a uniformly random bit when $h \leftarrow S_f$.

Because the above argument holds for any decision tree T that does not query lines in $\mathcal{L}_{x^*}$, it follows that

$$\mathbb{E}_f[\Pr[\mathrm{Dec}^{f, \mathcal{L}_{x^*}}(\alpha^*, x^*, i^*) = \pi(\alpha^* f(x^*))_{i^*}]] = \mathbb{E}_r[\mathbb{E}_f[\mathbf{1}(T_{(\alpha^*, x^*, i^*, r)}(f, \mathcal{L}_{x^*}) = \pi(\alpha^* f(x^*))_{i^*})]] = \frac{1}{2},$$

as $T(\cdot, \mathcal{L}_{x^*})$ is a decision tree that cannot query lines in $\mathcal{L}_{x^*}$. This shows that the decoder's success probability cannot exceed $1/2$ in expectation over a random codeword, and hence there exists a codeword where the success probability is $\leq 1/2$. This finishes the proof up to the proof of Claim 6.6, which we do below. □

Proof of Claim 6.6. We will first show that for any line L and any point $x \notin L$, there is a degree-1 polynomial $g_{x,L}$ satisfying $g_{x,L}(x) = 1$ and $g_{x,L}(y) = 0$ for all $y \in L$. Let $L = \{a + \lambda b\}$ where $a \in \mathbb{F}_{2^t}^n$, $b \in \mathbb{F}_{2^t}^n \setminus \{0^n\}$, and $\lambda \in \mathbb{F}_{2^t}$. Since $x \notin L$, there exists $v \in \mathbb{F}_{2^t}^n$ such that $\langle x - a, v\rangle \neq 0$ and $\langle b, v\rangle = 0$. Indeed, this follows because $x \notin L$ implies that $x - a$ and b are linearly independent, in which case the subspaces $\mathrm{span}\{b\}$ and $\mathrm{span}\{x - a, b\}$ are distinct subspaces of dimension 1 and 2, respectively. The vector v is simply any vector in $(\mathrm{span}\{b\})^\perp \setminus (\mathrm{span}\{x - a, b\})^\perp$, which is nonempty since the two subspaces are distinct.

With the vector v, we now let $g_{x,L}(z) := \langle z - a, v\rangle / \langle x - a, v\rangle$, which is a degree-1 polynomial that is well-defined since $\langle x - a, v\rangle \neq 0$. We have that $g_{x,L}(x) = 1$, and for any $y \in L$, i.e., $y = a + \lambda b$, we have that $g_{x,L}(a + \lambda b) = \lambda \langle b, v\rangle / \langle x - a, v\rangle = 0$ since $\langle b, v\rangle = 0$.

Finally, to finish the proof, we simply take $g_{x,L_1,\ldots,L_q}$ to be $\prod_{i=1}^q g_{x,L_i}$. □

7 Lower Bound for Linear 2-RLDCs Over Any Finite Field

In this section, we extend Theorem 2 to any finite field $\mathbb{F}$, for the specific case of $q = 2$.

Theorem 7.1. *Let $C \colon \mathbb{F}^k \to \mathbb{F}^n$ be a linear $(2, \delta, 1, s)$-RLDC with a nonadaptive decoder. Then, there exists a linear $(2, \delta/2, 1, s)$-LDC $C' \colon \mathbb{F}^{k'} \to \mathbb{F}^n$ where $k' \geq k - \lfloor 2/\delta \rfloor$ (and analogously for 2-RLCCs implying 2-LCCs). By [GKST06, Theorem 1.4], this implies that $n \geq 2^{\Omega_{\delta,s}(k) - \log_2(|\mathbb{F}|)}$.*

Theorem 7.1 thus extends the exponential lower bound of [BBC+23] for 2-query binary RLDCs to arbitrary finite fields. It also extends the proof ideas to get the corresponding result that 2-RLCCs

give 2-LCCs. With Theorem 5.2, we can extend this lower bound to 2-RLDCs and 2-RLCCs with imperfect completeness and adaptive decoders.

Corollary 7.2. *Let $C\colon \mathbb{F}^k \to \mathbb{F}^n$ be a linear $(2, \delta, 1-\varepsilon, s)$-RLDC. Then, there exists a linear $(2, \delta, 1, s + (2 + 1/(|\mathbb{F}| - 1))\varepsilon)$-LDC $C'\colon \mathbb{F}^{k'} \to \mathbb{F}^n$ where $k' \geq k - \lfloor 2/\delta \rfloor$ (and analogously for 2-RLCCs implying 2-LCCs). By [GKST06, Theorem 1.4], this implies that $n \geq 2^{\Omega_{\delta,s,\varepsilon,|\mathbb{F}|}(k) - \log_2(|\mathbb{F}|)}$.*

To prove Theorem 7.1, we adapt the approach of [BBC+23] to show that every linear 2-RLDC (or 2-RLCC) over *any* finite field must also be a 2-LDC (or 2-LCC).[10] We will show that the relaxed local decoder has two "modes," one of which behaves like a non-relaxed local decoder, and the other which necessarily has poor soundness. Thus, if the 2-RLDC or 2-RLCC has decent soundness, we can eliminate the latter case to build a 2-LDC or 2-LCC.

At a high level, we can use the structure of a linear code from Fact 3.13 to argue that every local view of the decoder either looks like a Hadamard code ("smooth" case) or a repetition code ("nonsmooth" case). Then, we can observe that the latter case must have high soundness error, and so our 2-RLDC must essentially be a Hadamard code, which is a 2-LDC.

7.1 Smooth (Hadamard code) vs. nonsmooth (repetition code) cases

Let $\mathrm{Dec}(\cdot)$ be the relaxed local decoder (or corrector) for C satisfying the assumed decoding radius, completeness, and soundness parameters. Assuming that the decoder is nonadaptive, we can assume that $\mathrm{Dec}^y(i)$ behaves as follows:

1. Sample a query pair $(j, \ell) \leftarrow Q_i$ with a corresponding decoding function $f^i_{j,\ell}\colon \mathbb{F}^2 \to \mathbb{F} \cup \{\perp\}$.
2. Query y at indices j, ℓ and return the evaluation of $f^i_{j,\ell}(y_j, y_\ell)$.

Consider each query pair (j, ℓ) from the support of Q_i.

- Suppose C is an RLDC. Then we say that j is *fixed* by i if there is some $\alpha \in \mathbb{F}$ such that $C(b)_j = \alpha b_i$ for all b.
- If C is an RLCC, then we say that j is *fixed* by i if there is some $\alpha \in \mathbb{F}$ such that $C(b)_j = \alpha C(b)_i$ for all b.

Let $S_i \subseteq [n]$ be the set of codeword indices fixed by i. Note that because the encoding is linear, every $j \in [n]$ is either unfixed, or fixed by exactly one index. Then, we can consider each case:

1. If both j and ℓ are not fixed by i, then the decoding function $f^i_{j,\ell}$ cannot return $\perp$, because $C|_{\{j,\ell\}}$ has dimension 2 and any local view can be completed to a valid codeword due to Fact 3.4. Thus, $f^i_{j,\ell}$ always returns a linear combination of its two inputs, which behaves like the local decoder of a *Hadamard code*. This is the "smooth" case described in Section 2.1.

[10] In fact, assuming linear structure eliminates much of the casework necessary in [BBC+23]. It would be interesting to extend this proof to nonlinear codes over large alphabets.

2. If both j and ℓ are fixed by i, then $C|_{\{j,\ell\}}$ has dimension 1. The truth table of the decoding function $f^i_{j,\ell}$ has exactly $|\mathbb{F}|^2 - |\mathbb{F}|$ entries that are $\perp$. Because both entries are multiples of the i-th message symbol, this local view looks like a *repetition code*. This is necessarily the "nonsmooth" case because these codeword indices only contain information on a single message index.

3. If j is fixed by i but ℓ is not fixed by i (or vice versa), then $C|_{\{j,\ell\}}$ has dimension 2 and so $f^i_{j,\ell}$ never returns $\perp$. In addition, by perfect completeness, the decoding function must take the form $f^i_{j,\ell}(y_j, y_\ell) = \alpha^{-1} y_j$, and so it does not depend on the ℓ-th symbol at all. Hence, we can treat (j, ℓ) as if it queried only j, which is a repetition code case.

Thus, without loss of generality, either both queries are fixed by i (and the decoder essentially tests equality between two copies of i) or both are not fixed by i (and the decoder cannot test and must always return a linear combination of the two symbols). We can show that as long as $|S_i|$ is not too large, the repetition code case can be fooled with certainty (compare to the higher query case in Lemma 4.3, where the tests can be more sophisticated). This implies that the decoder has good soundness when conditioned on choosing a Hadamard-like query case, where it never returns $\perp$.

Claim 7.3. Suppose C is a 2-RLDC satisfying the premise of Theorem 7.1 and suppose $i \in [k]$ such that $|S_i| \le \delta n/2$. Then, for all y such that $\Delta(y, C(b)) \le \delta n/2$,

$$\Pr[\mathrm{Dec}^y(i) = b_i \mid \text{Hadamard-like query case}] \ge 1 - s\,.$$

Analogously for a 2-RLCC C and for $i \in [n]$ satisfying the same conditions,

$$\Pr[\mathrm{Dec}^y(i) = C(b)_i \mid \text{Hadamard-like query case}] \ge 1 - s\,.$$

Proof. Let $E := j \in S_i \wedge k \in S_i$ be the event of the "repetition code" case. We will bound the behavior of the decoder conditioned on $\neg E$ (i.e., conditioned on the Hadamard case) by tampering with y and introducing errors designed to completely fool the repetition code case. We mildly reduce the decoding radius from δ to $\delta/2$ to give us the breathing room to introduce these errors, which are only used for the analysis.

Begin by finding a codeword c that differs in the desired symbol:

- If C is a 2-RLDC, then let $c := C(b + e_i)$, so that it differs in the i-th message symbol.
- If C is a 2-RLCC, let c be any arbitrary codeword such that $c_i \neq C(b)_i$; one must exist unless $C|_{\{i\}} = \{0^n\}$.

Then, let y' be the string formed by taking y and then setting $y'|_{S_i} = c|_{S_i}$, i.e., replace the symbols at S_i with those from c. Using the triangle inequality, $\Delta(y', C(b)) \le \delta n$, and so y' is subject to the soundness error property of C:

- If C is a 2-RLDC, then

$$1 - s \le \Pr[E] \cdot \Pr[\mathrm{Dec}^{y'}(i) \in \{b_i, \perp\} \mid E] + \Pr[\neg E] \cdot \Pr[\mathrm{Dec}^{y'}(i) \in \{b_i, \perp\} \mid \neg E]\,.$$

- If C is a 2-RLCC, then

$$1 - s \leq \Pr[E] \cdot \Pr[\mathrm{Dec}^{y'}(i) \in \{C(b)_i, \perp\} \mid E] + \Pr[\neg E] \cdot \Pr[\mathrm{Dec}^{y'}(i) \in \{C(b)_i, \perp\} \mid \neg E]\,.$$

However, conditioned on event E, our decoder Dec makes two queries to indices in S_i, and hence sees a local view indistinguishable from c, which we purposely chose to trick our decoder. By the perfect completeness property, the decoder must return $b_i + 1 \neq b_i$ (or $c_i \neq C(b)_i$) with probability 1. This gives a lower bound on the conditional probability of the decoder returning the right answer:

$$1 - s \leq \Pr[\neg E] \cdot \Pr[\mathrm{Dec}^{y'}(i) \in \{b_i, \perp\} \mid \neg E] \leq \Pr[\mathrm{Dec}^{y}(i) = b_i \mid \neg E]\,,$$

or for a 2-RLCC, $\Pr[\mathrm{Dec}^{y}(i) = C(b)_i \mid \neg E] \geq 1 - s$. We use two key observations here:

1. Recall that $y|_{[n]\setminus S_i} = y'|_{[n]\setminus S_i}$. Conditioned on on $\neg E$, both queries are outside of S_i, and since $y'|_{S_i} = y|_{S_i}$, we can replace y' with y on the right hand side.
2. In addition, the decoder cannot return $\perp$ because every possible pair of queried values will agree with some codeword. Hence, $\mathrm{Dec}(\cdot)$ must return b_i (or $C(b)_i$) to satisfy perfect completeness. □

7.2 Repetition case is rare

If $|S_i| \leq \delta n/2$ for every $i \in [k]$ (or every $i \in [n]$), then Claim 7.3 would prove that C is a $(2, \delta/2, 1, s)$-LDC (or LCC), because we can simply condition our decoder's query distribution to pick a Hadamard-like local view and get the required soundness error property. This may not be true for every i, but we can use a simple counting argument to show that it holds for nearly all of the message indices i. Thus, C itself is not necessarily a 2-LDC (or 2-LCC), but it does "contain" a 2-LDC (or 2-LCC) with nearly the same parameters.

Claim 7.4. Let $X = \{i : |S_i| > \delta n/2\}$. Then, $|X| \leq 2/\delta$.

Proof. Recall that a codeword index j cannot be fixed by two different (message for RLDC or codeword for RLCC) indices. Thus, the sets S_i are mutually disjoint, so

$$\delta |X| n/2 < \sum_{i \in X} |S_i| \leq n \implies |X| < 2/\delta\,.$$

Note that the sets S_i may not be mutually disjoint if the code is nonlinear. Indeed, in this case a codeword index could be fixed by many different message indices, which (for the binary case) requires a careful analysis in [BBC+23]. □

Now, we have the tools to prove Theorem 7.1. We will remove the indices in X from the code, either by fixing them to zero for the RLDC case or by adding linear constraints setting them to zero for the RLCC case. Then, every remaining message or codeword index either satisfies Claim 7.3 or has its value fixed.

Proof of Theorem 7.1. Start by defining $k' := k - \lfloor 2/\delta \rfloor$. For C which is an RLDC, without loss of generality, $X = \{k'+1, k'+2, \ldots, k\}$. Let $C' \colon \mathbb{F}^{k'} \to \mathbb{F}^n$ be defined as $C'(b) = C((b, 0^{k-k'}))$, i.e., append $k - k' = |X|$ zeroes to the message and encode using C. Then, every $i \in [k']$ satisfies $|S_i| \leq \delta n/2$, and so Claim 7.3 gives us a local decoder for all $i \in [k']$ which never returns $\perp$. This gives the required $(\delta/2, s)$-soundness error property. The query complexity and perfect completeness properties are inherited from the original decoder, showing that C' is a $(2, \delta/2, 1, s)$-LDC. At last, apply the 2-LDC lower bound of [GKST06, Theorem 1.4] which gives $n \geq 2^{(1/2-s)\delta k'/16 - 1 - \log_2 |\mathbb{F}|}$.

For C which is an RLCC, take an arbitrary parity check matrix $B \in \mathbb{F}^{(n-k)\times n}$ for C, and then add each $e_i \in \mathbb{F}^n$ for $i \in X$ as rows to B. This defines a new code $C' \subseteq C$ with dimension at least k' where the indices at X are always zero, and we can pick an arbitrary basis to get the encoding map $C' \colon \mathbb{F}^{k'} \to \mathbb{F}^n$. If $i \in [n]$ is not in X, then it satisfies Claim 7.3 yielding a local corrector for i that never returns $\perp$; if $i \in [n]$ is in X, we can now use a trivial local corrector that always returns zero. Thus, C' is a $(2, \delta/2, 1, s)$-LCC. □

References

[AG24] Omar Alrabiah and Venkatesan Guruswami. Near-tight bounds for 3-query locally correctable binary linear codes via rainbow cycles. In *65th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2024, Chicago, IL, USA, October 27-30, 2024*. IEEE, 2024.

[AGKM23] Omar Alrabiah, Venkatesan Guruswami, Pravesh K. Kothari, and Peter Manohar. A near-cubic lower bound for 3-query locally decodable codes from semirandom CSP refutation. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 1438–1448. ACM, 2023.

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM (JACM)*, 45(3):501–555, 1998.

[ALRW17] Alexandr Andoni, Thijs Laarhoven, Ilya P. Razenshteyn, and Erik Waingarten. Optimal hashing-based time-space trade-offs for approximate near neighbors. In *SODA*, pages 47–66, 2017.

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM (JACM)*, 45(1):70–122, 1998.

[AS21] Vahid R Asadi and Igor Shinkar. Relaxed locally correctable codes with improved parameters. In *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[BBC+23] Alexander R. Block, Jeremiah Blocki, Kuan Cheng, Elena Grigorescu, Xin Li, Yu Zheng, and Minshen Zhu. On relaxed locally decodable codes for Hamming and insertion-deletion errors. In *38th Computational Complexity Conference, CCC 2023, July 17-20, 2023,*

Warwick, UK, volume 264 of *LIPIcs*, pages 14:1–14:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[BCH+95] Mihir Bellare, Don Coppersmith, Johan Håstad, Marcos A. Kiwi, and Madhu Sudan. Linearity testing in characteristic two. In *36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, USA, 23-25 October 1995*, pages 432–441. IEEE Computer Society, 1995.

[BFLS91] László Babai, Lance Fortnow, Leonid A Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 21–32, 1991.

[BGH+04] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust PCPs of proximity, shorter PCPs and applications to coding. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 1–10. ACM, 2004.

[BGT17] Arnab Bhattacharyya, Sivakanth Gopi, and Avishay Tal. Lower bounds for 2-query LCCs over large alphabet. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[BHKL24] Arpon Basu, Jun-Ting Hsieh, Pravesh K. Kothari, and Andrew D. Lin. Improved lower bounds for all odd-query locally decodable codes. *CoRR*, abs/2411.14361, 2024.

[BIW10] Omer Barkol, Yuval Ishai, and Enav Weinreb. On locally decodable codes, self-correctable codes, and t-private PIR. *Algorithmica*, 58(4):831–859, 2010.

[BK95] Manuel Blum and Sampath Kannan. Designing programs that check their work. *Journal of the ACM (JACM)*, 42(1):269–291, 1995.

[BLR93] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of computer and system sciences*, 47(3):549–595, 1993.

[CGS20] Alessandro Chiesa, Tom Gur, and Igor Shinkar. Relaxed locally correctable codes with nearly-linear block length and constant query complexity. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1395–1411. SIAM, 2020.

[CGW13] Victor Chen, Elena Grigorescu, and Ronald de Wolf. Error-correcting data structures. *SIAM J. Comput.*, 42(1):84–111, 2013.

[CKGS98] Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *J. ACM*, 45(6):965–981, 1998.

[DGL21] Marcel de Sena Dall'Agnol, Tom Gur, and Oded Lachish. A structural theorem for local algorithms with applications to coding, testing, and privacy. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1651–1665. SIAM, 2021.

[DGY11] Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching vector codes. *SIAM Journal on Computing*, 40(4):1154–1178, 2011.

[Dvi12] Zeev Dvir. Incidence theorems and their applications. *CoRR*, abs/1208.5073, 2012.

[Dvi16a] Zeev Dvir. Lecture notes on linear locally decodable codes. https://www.cs.princeton.edu/~zdvir/LDCnotes/LDC1.pdf, Fall 2016.

[Dvi16b] Zeev Dvir. Lecture notes on linear locally decodable codes. https://www.cs.princeton.edu/~zdvir/LDCnotes/LDC8.pdf, Fall 2016.

[Efr09] Klim Efremenko. 3-query locally decodable codes of subexponential length. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 39–44. ACM, 2009.

[GKST06] Oded Goldreich, Howard Karloff, Leonard J Schulman, and Luca Trevisan. Lower bounds for linear locally decodable codes and private information retrieval. *Computational Complexity*, 15(3):263–296, 2006.

[GL20] Tom Gur and Oded Lachish. On the power of relaxed local decoding algorithms. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1377–1394. SIAM, 2020.

[Gol23] Oded Goldreich. On the lower bound on the length of relaxed locally decodable codes. *Electron. Colloquium Comput. Complex.*, TR23-064, 2023.

[Gol24a] Guy Goldberg. Linear relaxed locally decodable and correctable codes do not need adaptivity and two-sided error. In *51st International Colloquium on Automata, Languages, and Programming (ICALP 2024)*, volume 297 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 74:1–74:20, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[Gol24b] Oded Goldreich. On the relaxed LDC of BGHSV: a survey that corrects the record. *Electron. Colloquium Comput. Complex.*, TR24-078, 2024.

[GRR18] Tom Gur, Govind Ramnarayan, and Ron D. Rothblum. Relaxed locally correctable codes. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, volume 94 of *LIPIcs*, pages 27:1–27:11. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[HKM+24] Jun-Ting Hsieh, Pravesh K. Kothari, Sidhanth Mohanty, David Munhá Correia, and Benny Sudakov. Small even covers, locally decodable codes and restricted subgraphs of edge-colored Kikuchi graphs. *CoRR*, abs/2401.11590, 2024.

[IK99] Yuval Ishai and Eyal Kushilevitz. Improved upper bounds on information-theoretic private information retrieval (extended abstract). In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, May 1-4, 1999, Atlanta, Georgia, USA*, pages 79–88. ACM, 1999.

[JM24] Oliver Janzer and Peter Manohar. A $k^{q/q-2}$ lower bound for odd query locally decodable codes from bipartite Kikuchi graphs. *CoRR*, abs/2411.14276, 2024.

[KM24a] Pravesh K. Kothari and Peter Manohar. An exponential lower bound for linear 3-query locally correctable codes. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 776–787. ACM, 2024.

[KM24b] Pravesh K. Kothari and Peter Manohar. Exponential lower bounds for smooth 3-LCCs and sharp bounds for designs. In *65th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2024, Chicago, IL, USA, October 27-30, 2024*. IEEE, 2024.

[KT00] Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 80–86, 2000.

[KW04] Iordanis Kerenidis and Ronald de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. *Journal of Computer and System Sciences*, 69(3):395–420, 2004.

[LFKN92] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992.

[Tre04] Luca Trevisan. Some applications of coding theory in computational complexity. *arXiv preprint cs/0409044*, 2004.

[Yan24] Tal Yankovitz. A stronger bound for linear 3-LCC. In *65th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2024, Chicago, IL, USA, October 27-30, 2024*. IEEE, 2024.

[Yek08] Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *Journal of the ACM (JACM)*, 55(1):1–16, 2008.

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

A Linear LDCs and LCCs Have Strong Soundness

In this appendix, we show that any linear LDC or LCC satisfying Definitions 3.6 and 3.7 additionally has strong soundness (Definition 1.7). More formally, we will show the following lemma.

Lemma A.1. *Let $C\colon \mathbb{F}^k \to \mathbb{F}^n$ be a linear (q,δ,c,s)-LDC (LCC) with $s < 1 - \frac{1}{|\mathbb{F}|}$. Then, there is a decoder $\mathrm{Dec}(\cdot)$ satisfying the following conditions:*

(1) (q-queries) for any y and i, the algorithm $\mathrm{Dec}^y(i)$ reads at most q indices of y

(2) (perfect completeness) for all $b \in \mathbb{F}^k$ and $i \in [k]$, $\Pr[\mathrm{Dec}^{C(b)}(i) = b_i] = 1$,

(3) ((δ/q)-strong soundness) for all $b \in \mathbb{F}^k$, $i \in [k]$, and all $y \in \mathbb{F}^n$ with $\Delta(y, C(b)) \le \delta n/q$, $\Pr[\mathrm{Dec}^y(i) \neq b_i] \le \frac{q\Delta(y,C(b))}{\delta n}$.

Proof. By [Dvi16a], given any (q,δ,c,s)-LDC, there exist collections $\mathcal{H}_1,\dots,\mathcal{H}_k$ where (1) each $\mathcal{H}_i$ is a set of $|\mathcal{H}_i| \ge \delta n/q$ q-sparse vectors in $\mathbb{F}^n$ with disjoint support, i.e., any $v, v' \in \mathcal{H}_i$, $\mathrm{supp}(v) \cap \mathrm{supp}(v') = \emptyset$, and (2) for every $b \in \mathbb{F}^k$, $i \in [k]$, and $v \in \mathcal{H}_i$, it holds that $b_i = \sum_{j=1}^n v_j x_j$ where $x = C(b)$. Note that the latter sum is actually over at most q indices, as $|\mathrm{supp}(v)| \le q$.

Consider the decoder $\mathrm{Dec}^y(i)$ that behaves as follows: on input i, sample $v \leftarrow \mathcal{H}_i$ uniformly at random, read y_j for each $j \in \mathrm{supp}(v)$, and output $\sum_{j\in\mathrm{supp}(v)} v_j y_j$. If $y_j = x_j$ for all $j \in \mathrm{supp}(v)$, where $x = C(b)$, then by the above, the decoder outputs b_i.

Finally, $b \in \mathbb{F}^k$, $x = C(b)$, and let $y \in \mathbb{F}^n$ with $\Delta(y, C(b)) \le \delta n/q$. Let S denote the set of coordinates $j \in [n]$ where $y_j \neq C(b)_j$. Because $\mathrm{supp}(v)$'s are disjoint for $v \in \mathcal{H}_i$, it follows that

$$\Pr_{v\leftarrow\mathcal{H}_i}[\mathrm{supp}(v)\cap S \neq \emptyset] \le \mathbb{E}_{v\leftarrow\mathcal{H}_i}[|\mathrm{supp}(v)\cap S|] = \frac{1}{|\mathcal{H}_i|}\sum_{v\in\mathcal{H}_i}|\mathrm{supp}(v)\cap S| = \frac{|(\cup_{v\in\mathcal{H}_i}\mathrm{supp}(v))\cap S|}{|\mathcal{H}_i|} \le \frac{|S|}{|\mathcal{H}_i|}.$$

As $|\mathcal{H}_i| \ge \delta n/q$, it follows that the probability the decoder outputs b_i is at least $1 - \frac{q|S|}{\delta n}$, as required.

To prove the statement for LCCs, we use [Dvi16b] instead of [Dvi16a], and the rest of the proof follows immediately. □

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il