Electronic Colloquium on Computational Complexity, Revision 2 of Report No. 56 (2023)

Approximate Locally Decodable Codes with Constant Query Complexity and Nearly Optimal Rate

Geoffrey Mon, Dana Moshkovitz, and Justin Oh
Department of Computer Science
University of Texas at Austin
{gmon, danama, sjo}@cs.utexas.edu

Abstract

We present simple constructions of good approximate locally decodable codes (ALDCs) in the presence of a δ-fraction of errors for $\delta < 1/2$. In a standard locally decodable code $C\colon \Sigma_1^k \to \Sigma_2^n$, there is a decoder M that on input $i \in [k]$ correctly outputs the i-th symbol of a message x (with high probability) using only q queries to a given string w that is δ-close to $C(x)$. In an ALDC, the decoder M only needs to be correct on a $1-\varepsilon$ fraction of $i \in [k]$ for ε much smaller than δ. We present a construction of explicit ALDCs for all constants $1/2 > \delta > \varepsilon$ with a constant number of queries q and with constant, near-optimal rate. Standard LDCs with constant number of queries and *any* constant rate are known to be impossible.

We additionally explore what is the lowest error probability ε one can achieve for fixed δ and q. We show that for any ALDC, $\varepsilon = \Omega(\delta^{\lceil q/2 \rceil})$. We then show that there exist explicit constant rate ALDCs for any constant q that achieve $\varepsilon = O(\delta^{\lceil q/2 \rceil})$. In particular, for $q = 3$, we have a constant rate ALDC with error probability $\varepsilon = O(\delta^2)$.

1 Introduction

Locally decodable codes (LDCs) are a useful and pervasive tool in both application and theory, and there has been intense study towards constructing such codes with optimal parameters. By the seminal work of Katz and Trevisan [KT00], asymptotically good LDCs with the "dream" parameters of constant rate, distance, and query complexity cannot exist. In this work, we show that simple constructions *can* achieve such ideal parameters for the relaxed notion of *approximate* locally decodable codes (ALDCs), which are natural codes that, like LDCs, often emerge in complexity theory.

In a standard (q, δ, ε)-LDC $C\colon \Sigma_1^k \to \Sigma_2^n$, there exists a randomized decoding algorithm M that takes as input $i \in [k]$ and has query access to a string w that is δ-close to a codeword $C(x)$. This sublinear-time decoder must correctly output $M^w(i, r) = x_i$ with probability at least $1 - \varepsilon$ over its internal randomness r for any $i \in [k]$, using at most q (non-adaptive) queries to w. In contrast, an approximate locally decodable code only requires that M successfully decodes *most* of the message coordinates with few queries. That is, for a (q, δ, ε)-approximate locally decodable code $C\colon \Sigma_1^k \to \Sigma_2^n$ there again exists a randomized algorithm M that takes as input $i \in [k]$ and makes at most q queries to a string w that is δ-close to a codeword $C(x)$. This time, the decoder must correctly output $M^w(i, r) = x_i$ with probability $1 - \varepsilon$ over its internal randomness r *on average* over all coordinates $i \in [k]$ of the message.[1] The identity code is a trivial $(1, \delta, \varepsilon)$-ALDCs for $\delta = \varepsilon$, so $\varepsilon \ll \delta$ is the only parameter regime where ALDCs make sense.

[1]The decoder does not necessarily know which indices i are decoded correctly. A different relaxed definition, known as *relaxed locally decodable codes* [BGH+06], requires the decoder to report which indices are corrupted.

ISSN 1433-8092

To make sense of this definition, suppose we would like to amplify the decoding radius of a code $C\colon \Sigma_0^k \to \Sigma_1^n$ from ε to δ. It is natural to seek a map $C'\colon \Sigma_1^n \to \Sigma_2^m$ equipped with a decoder which can, given any input which is δ-close to some $C'(x)$, return a string x' which is ε-close to x. Then, $C' \circ C$ is a new code with a decoding radius of δ: run the decoder of C' on any input which is δ-close to $C' \circ C$ to get x' which is ε-close to some codeword of C, and then run the decoder of C on x' to uniquely recover the message. If the decoder of C' features locality, then C' is precisely an ALDC which can be used to amplify the decoding radius of an LDC [BET10].

Historically, the notion of ALDCs is motivated by topics in computational complexity theory such as hardness amplification. Informally, suppose there is a function f with a truth table that differs from the truth table of every possible efficiently computable function on at least an ε fraction of inputs—that is, no efficient algorithm can compute f with accuracy better than $1 - \varepsilon$. Then, we can encode the truth table of f with an ALDC to get the truth table of a new, *harder*, function which cannot be computed by any efficient algorithm with accuracy better than $1 - \delta < 1 - \varepsilon$: if there existed such an algorithm, then we could combine it with an algorithm that computes the ALDC decoder to build an algorithm computing f with accuracy better than $1 - \varepsilon$, which is a contradiction. Explicit ALDCs considered in this context are the *XOR code* and the *direct product code* as well as their derandomized counterparts [IW97, Tre03, IJKW10].

Versions of ALDCs also appear in the context of probalistically checkable proofs, which are proofs that can be verified with very few queries and randomness. In this research area, certain constructions allow for symbols of the witness to be decoded from the proof using few queries, a concept called a *PCP of proximity* [BGH$^+$06] or *decoding PCP* [MR10, DH13]. Such constructions typically only guarantee that most (instead of all) symbols are decoded correctly.

1.1 Our Results

We give both constructions and lower bounds, demonstrating the possibility and limits of ALDCs in terms of rate and locality. The first constructive result gives constant query ALDCs with near optimal rate and arbitrary error reduction.

1.1.1 Constant Query ALDCs with Nearly Optimal Rate

In the case of unique decoding, the Singleton bound tells us that a code that can be decoded from δ fraction errors must have rate at most $1 - 2\delta + o(1)$, because a decoding radius of δ implies a distance of at least 2δ. Notably, the Reed–Solomon code is a uniquely (non-local) decodable code that achieves this bound. Kopparty, Meir, Ron-Zewi, and Saraf [KMRS17] use Reed–Solomon as the base code for Alon–Edmonds–Luby distance amplification [AEL95, AL96] to construct ALDCs with rate approaching the Singleton bound; this construction is crucial to their construction of asymptotically-good LDCs with subpolynomial query complexity:

Theorem 1.1 (due to [KMRS17, Lemma 3.2])**.** *For any $\delta, \varepsilon > 0$ and any parameter $\alpha > 0$, there is an explicit (q, δ, ε)-ALDC with rate $1 - 2\delta - 2\alpha$ and query complexity $q = \mathsf{poly}(1/\varepsilon\alpha)$.*

This theorem shows a quite striking contrast. One cannot hope to have an LDC with both constant rate and constant number of queries. Nevertheless, it is possible to have an ALDC that reduces any sufficiently small constant error rate δ, to any arbitrarily smaller constant error ε, with constant query complexity and with arbitrarily high rate.

Then, a natural question to ask is: what is the optimal rate for a (q, δ, ε)-ALDC? In particular, the Singleton rate upper bound of $1 - 2\delta + o(1)$ does *not* apply to (q, δ, ε)-ALDCs, because they can have distance less than 2δ: an ALDC's decoder is only required to return most of the message

correctly, so two messages can map to the same ALDC codeword as long as these messages are ε-close to each other. We take advantage of this fact to get an ALDC construction with higher rate slightly exceeding the Singleton bound:

Theorem 1.2 (See Corollary 3.9)**.** *For any constants $\delta, \varepsilon > 0$, any sufficiently small constant parameter α, any constant sized finite field Σ_1, and any sufficiently large k, there is an explicit $C\colon \Sigma_1^k \to \Sigma_2^n$ which is a (q, δ, ε)-ALDC with $q = \mathsf{poly}(1/\varepsilon\alpha)$ with $|\Sigma_2| \leq |\Sigma_1|^{q2^{O(q)}}$. The rate of the code is*

$$\frac{1 - 2\delta - 2\alpha}{1 - .99H_{|\Sigma_1|}(\varepsilon/2)} - o(1).$$

To achieve this, we refine the Alon–Edmonds–Luby technique [AEL95, AL96] as used by Kopparty, Meir, Ron-Zewi, and Saraf [KMRS17]. This technique builds a code that divides the message into small constant-size blocks, encodes each block with a base code, and then permutes all of the symbols according to a *sampler* graph to form the codeword. The sampler graph guarantees that no matter which δ fraction of codeword symbols are corrupted, at least a $1 - \varepsilon$ fraction of the blocks will "see" an approximately δ fraction of corruption. Because the sampler graph permutes symbols but does not duplicate them, the rate of this code is the same as the rate of the base code. In addition, each message symbol can be decoded by querying all of the (constant many) symbols in its corresponding block. If Reed–Solomon is used as the base code, then this yields an ALDC with constant query complexity and rate approaching the Singleton bound [KMRS17]. The $1 - \varepsilon$ fraction of blocks have bounded distance from the Reed–Solomon base code and can be uniquely decoded, while the remaining ε fraction of blocks which have too much corruption are written off.

We improve the rate by using a (δ, ε)-approximate code[2] as the base code for the AEL construction instead of Reed–Solomon. This base code is weaker than a uniquely decodable code such as Reed–Solomon, and hence can have higher rate. Then, in each of the blocks with bounded corruption, a $1 - \varepsilon$ fraction of message symbols can be recovered. Since $1 - \varepsilon$ of the blocks have at most $O(\delta)$ corruption, a $1 - O(\varepsilon)$ fraction of the entire message can be decoded. Hence, we can spread the message corruption across all of the blocks, instead of concentrating it in the ε fraction of blocks written off by the sampler, in order to relax the base code and get a higher rate.

The final step is to show that an approximate code with higher rate exists. This code will be used on constant-sized blocks, so we can afford to construct it by brute force. Because this code only needs to preserve $1 - \varepsilon$ of the message coordinates, we can pick a subset D (known as a covering code) of the message space Σ^k such that every message is ε-close to some string in D. Then, we can view D as a new, smaller set of messages $\Sigma^{k'}$ and encode it using Reed–Solomon with distance 2δ. To approximately decode, we can use the Reed–Solomon decoder to fully recover an element of D which is guaranteed to be ε-close to our true original codeword. The rate of this code is $k/k' \cdot (1 - 2\delta)$, which "exceeds" the Singleton bound.

1.1.2 Rate Upper Bound

Theorem 1.2 demonstrates that an ALDC can have rate slightly exceeding a uniquely decodable code of the same decoding radius. We show that this rate is in fact nearly optimal for approximate codes in general, even without locality. Simply observe that composing an approximate code with a uniquely decodable code yields a uniquely decodable code, which is then governed by traditional coding theory rate bounds. Hence, adding locality incurs almost no cost on rate.

[2] A (δ, ε)-*approximate code* is an ALDC with no requirement of locality: there is some (potentially global) decoder that reduces error from δ to ε.

Theorem 1.3 (See Theorem 3.7). *A (δ, ε)-approximate code $C\colon \Sigma_1^k \to \Sigma_2^n$ must have rate*

$$R \leq \frac{1 - 2\delta + o(1)}{1 - H_{|\Sigma_1|}(2\varepsilon) - o(1)}.$$

1.1.3 Constant Query Approximate Local Weak List Decodable Codes with High Rate

We can adapt the construction in Theorem 1.2 to admit a local weak list decoding algorithm with a constant number of queries in the presence of a large fraction of errors.

We consider the following definition of local list decoding: A $(q, \delta, \varepsilon, \ell)$-approximate local weak list decodable code $C\colon \Sigma_1^k \to \Sigma_2^n$ is equipped with a decoder M that on input $i \in [k]$, makes at most q queries to a string $w \in \Sigma_2^n$ and outputs a list of symbols $M(i) \subset \Sigma_1$ such that $|M(i)| \leq \ell$. For any codeword $C(x)$ that is δ-close to w we have $x_i \in M(i)$ for at least $1 - \varepsilon$ fraction of $i \in [k]$. We generally aim to construct such codes with $\ell \ll |\Sigma_1|$. This definition is similar to the definition of list decoding for locally decode or reject codes [MR10], but is weaker than the list decoding definition for ALDC that appears in [IJKW10]. In the latter, the decoder outputs ℓ circuits $A^1, \ldots, A^\ell$ such that for any $C(x)$ that is δ-close to w, there is $j \in [\ell]$ that on input $i \in [k]$ decodes $A^j(i) = x_i$ for at least $1 - \varepsilon$ fraction of the $i \in [k]$. This stronger definition can be easily obtained in the case of polynomially small rate, but we do not know how to achieve it in the case of constant rate and a constant number of queries.

The notion of approximate weak list decoding (without locality) was used implicitly before [GI02, GI04] as an ingredient for constructing (standard) list decodable codes. A standard list decodable code of radius δ and list size ℓ is a code $C\colon \Sigma_1^k \to \Sigma_2^n$ such that every string $w \in \Sigma_2^n$ has at most ℓ codewords that are δ-close. Such works also utilize the distance amplification technique of [AEL95] as we do, and [GKO$^+$16, HRW20] later observed that such a construction also has locality.

We use the same AEL technique as these previous works, and demonstrate that our technique of slightly improving the rate by first applying a covering code can also be applied to approximate weak list decoding. Specifically, we show that there are explicit approximate local weak list decodable codes that slightly exceed the probabilistic bound on the rate of standard list decodable codes. A probabilistic argument shows that there exist list decodable codes $C\colon \Sigma_1^k \to \Sigma_2^n$ for any radius $\delta < 1 - \frac{1}{|\Sigma_2|}$, and any list size ℓ with rate $1 - H_{|\Sigma_2|}(\delta) - \frac{1}{\ell+1} - o(1)$. Moreover, this rate is essentially optimal for list decodable codes, up to the dependence on ℓ (see for example: [Vad12, Theorem 5.8]).

Theorem 1.4 (See Theorem 4.4). *Let Σ_1 be an arbitrary constant-sized alphabet of size ≥ 3. For parameters $0 < \alpha < \varepsilon < 1/2$, $1/2 < r < 1 - 1/|\Sigma_1|$, and $\ell \geq 1$, there exists an explicit $(q, r, \varepsilon, \ell)$-approximate local weak list decodable code $C\colon \Sigma_1^k \to \Sigma_2^n$ with $|\Sigma_2| \leq |\Sigma_1|^{q^{2^{O(q)}}}$. The rate of the code is*

$$\geq \frac{1 - H_{|\Sigma_1|}(r + \alpha) - \frac{1}{\ell+1} - o(1)}{1 - 0.99 H_{|\Sigma_1|}(\varepsilon)}$$

and the query complexity is $q \leq O\left(\frac{1}{\alpha^2} \log \frac{1}{\varepsilon}\right)$.

To obtain a weak approximate locally list decodable code we can simply replace the Reed Solomon code used in the constant sized approximate code above with an optimal list decodable code found via brute force. This will yield a constant sized *approximate list decodable code*, where there is a decoder M that on input any string w, outputs a small list of possible messages $M(w)$ such that for every codeword $C(x)$ that δ-close to w, at least one message from $M(w)$ is ε-close to

x. We will use this approximate list decodable code in the same sampler-based construction above. Again, the rate in the final construction will inherit the rate of the constant sized approximate list decodable code. For the weak list decoding property, we observe that the list of strings returned by M above for a given small block of the message also naturally induces a small list of symbols for every coordinate in the block.

1.1.4 3-Query ALDCs and Optimal Error Reduction

In the most extreme setting of local decoding, one may ask what is possible with only 3 queries. Indeed, this has been the subject of extended study in the case of LDCs with both upper bounds [Yek08, Efr12, DGY11] and lower bounds [Woo07, AGKM23, KM23, Yan24, AG24], and as discussed, it is impossible to achieve constant rate in that case.

Evidently, relaxing the goal of the decoder to error reduction rather than "error elimination" drastically improves the state of what is feasible in the constant rate regime. Thus it is natural to ask, what is the best error reduction possible given q queries for a constant rate ALDC? We show that for 3 queries, one cannot hope for better than a $(3, \delta, \Theta(\delta^2))$-ALDC for *any* rate. Intuitively, a randomized decoder makes three uniformly random queries to the coordinates of a codeword with a δ-fraction of corruptions. When at least two of these queries read a corrupted symbol, there is no hope the decoder will output the correct message symbol. In fact, we show that any q-query decoder cannot succeed (with probability over both its randomness and a uniform choice of message coordinate) with probability better than the probability that the majority of its queries land on uncorrupted symbols.

Theorem 1.5 (See Theorem 5.1)**.** *Let $C\colon \Sigma_1^k \to \Sigma_2^n$ be a (q, δ, ε)-ALDC with query complexity $q = O(1)$ and decoding radius $\delta < 1/2$. Then $\varepsilon = \Omega(\delta^{\lceil q/2 \rceil})$.*

In addition, we show that it is indeed possible to construct a constant rate 3-query ALDC with this optimal error reduction. In fact we show that it is possible to obtain optimal error reduction for any given constant number of queries, by adapting a different but related distance amplification technique due to Alon–Bruck–Naor–Naor–Roth [ABN+92].

Theorem 1.6 (See Corollary 5.4)**.** *Let $q > 1$ and $0 < \delta < 1/2$ be constants. Let Σ_1 be any alphabet. There are explicit (q, δ, ε)-ALDCs $C\colon \Sigma_1^k \to \Sigma_2^n$ with constant rate, and $\Sigma_2 = \Sigma_1^{D_R}$, where $\varepsilon = O(\delta^{\lceil q/2 \rceil})$, $D_L = O(\frac{q^2}{\delta^2} \log \frac{q}{\delta^{\lceil q/2 \rceil}})$, and $D_R = 2^{D_L 2^{O(D_L)}}$.*

This construction is similar to the near-optimal rate ALDC construction. We can use a sampler graph with the repetition code for each message symbol, and decode by taking the majority over q uniformly random copies of the desired message symbol.

1.1.5 ALDCs with Small Alphabet

All of our ALDC constructions feature codeword alphabets of constant size which depends on the parameters q and δ; when compared to the identity code, one might wonder whether binary ALDCs can achieve nontrivial error amplification. We can get explicit binary ALDCs achieving similar error amplification to the construction from Theorem 1.6, by concatenating that construction with the Hadamard code.

Theorem 1.7 (See Theorem 5.5)**.** *Let $q > 1$ and $0 < \delta < 1/2$ be constants. There are explicit (q, δ, ε)-ALDCs $C\colon \{0,1\}^k \to \{0,1\}^n$ with constant rate, where $\varepsilon = O((2\delta)^{\lceil q/2 \rceil})$.*

1.2 Related work

Distance amplification of LDCs. The work of [KMRS17] was the first to apply the Alon–Edmonds–Luby distance amplification technique [AEL95, AL96] in order to amplify the decoding radius of a locally decodable code from ε to δ. By breaking the message into constant-size blocks and encoding with a Reed–Solomon code, they are able to implicitly construct an ALDC that reduces error from δ to ε with rate approaching the Singleton bound $1 - 2\delta - 2\alpha$ for an arbitrarily small parameter α. In their construction, for any given corruption of a δ fraction of codeword symbols, the sampler ensures that a $\geq 1 - \varepsilon$ fraction of constant-size Reed–Solomon blocks are "good": they see few enough errors to be exactly decoded, so that $\geq 1 - \varepsilon$ symbols of the entire message can be recovered. Our construction is able to increase this rate to $\frac{1-2\delta-2\alpha}{1-.99H(\varepsilon/2)}$, because we allow each of the good constant-size blocks to get $O(\varepsilon)$ of their message symbols incorrect. Distributing this allowable error into the constant-size codes allows them to have higher rate, which improves the overall rate of the ALDC.

The works of [CY21, CY22] build on [KMRS17] by constructing distance amplification procedures for LDCs with subconstant decoding radius. This gives them a general transformation converting LDCs with constant rate and very small distance to LDCs with constant rate and constant distance, with a smaller overhead in query complexity compared to [KMRS17]. Their techniques make use of properties specific to decoders for LDCs. We also directly construct a distance amplification transformation (in the form of an ALDC) with constant decoding radius, rate, and query complexity using a similar technique, but with particular emphasis on obtaining near-optimal rate. In addition, the only property required to use an ALDC for distance amplification is to have a decoder that handles an ε decoding radius.

Relaxed LDCs. ALDCs are not the only relaxation of LDCs. The work of [BGH$^+$06] introduced the notion of *relaxed* locally decodable codes. A relaxed locally decodable code $C\colon \Sigma_1^k \to \Sigma_2^n$ is also equipped with a local randomized decoder M that takes as input $i \in [k]$ and makes a small number of q queries to a string w that is δ-close to a codeword $C(x)$. For every $i \in [k]$, the decoder must either correctly output x_i or an error symbol $\perp$ with high probability over its randomness. Moreover, the decoder is not allowed to return an error symbol when the string is uncorrupted. Because most local views are uncorrupted, there must be at least $1 - \varepsilon$ of coordinates i for which $M(i, r) = x_i$ with high probability. In words, for every coordinate, the decoder must either output the correct message symbol for most coordinates, and indicate that an error has been detected for the rest.

Thus relaxed LDCs are a weaker notion than LDCs, and indeed [BGH$^+$06] show that there exist relaxed LDCs using a constant number of queries and block length $k^{1+\gamma}$ for any small constant $\gamma > 0$, a vast improvement to the state of the art of standard LDCs. Nevertheless, [GL21, DGL21] show that relaxed LDCs using a constant number of queries cannot have constant rate. In contrast, our work shows that the even more relaxed notion of ALDCs can have nearly optimal constant rate using a constant number of queries.

Average-case smooth codes. *Smooth* codes have local decoders which only need to successfully decode every message index $i \in [k]$ when given an uncorrupted codeword, but with the additional requirement of *smoothness*: for every $i \in [k]$, the decoder queries any index of the input string w with probability $O(1/n)$, i.e., the queries to the codeword are almost uniform. The work of [KT00] showed that LDCs are equivalent to smooth codes, and reasoned with smooth codes to prove the first rate upper bounds for LDCs. In particular, an LDC can be made smooth. Let S_i be the set of codeword indices that the LDC decoder queries with probability $> q/\delta n$ on input $i \in [k]$. This

set cannot be larger than δn, or else the decoder makes more than q queries. Therefore, we can build a smooth decoder that black boxes the LDC decoder, such that for each i, queries to S_i made by the LDC decoder are answered with 0 instead of with an actual query to the codeword. Then, every codeword index is queried with probability $\leq q/\delta n$, and the simulated LDC decoder sees a codeword with at most δ fraction of errors, so it will still be able to successfully decode the ith symbol of the message.

While studying pathways to nonexplicit LDCs, [BDG19] shows that an *average-case* smooth code, which successfully decodes on average over both the message index and over all possible messages, implies a smooth code that works over all message indices and all messages, with a constant factor loss in rate. This notion is remarkably similar to the idea of ALDCs, and a similar generalization has also been used to show lower bounds on LDCs [Woo07]. Critically however, average-case smooth codes are still smooth while ALDCs may not be. An ALDC cannot be made smooth using the technique from the previous paragraph. The sets S_i can differ for distinct values of i. However, the ALDC guarantee only states that *most* message symbols can be decoded when queries to S_i are ignored. Therefore, i may not be one of those successfully decoded message indices, and the decoder may fail on most or *all* of the message symbols. Indeed, the constructions we give are far from smooth, because for each $i \in [k]$ the support of the query distribution for the decoder for i is constant sized.

Local self-correction for Reed–Muller codes. A *local self-corrector* for a code C is an algorithm that, given access to a string w which is δ-close to some codeword in C, is able to make few queries to w in order to return an *entire* message m such that $C(m)$ is δ'-close to w, where δ' can be (much) larger than δ. This is possible if the block length n of C is much longer (say, exponential) than the message length k, such that a decoder that makes e.g. $\mathsf{poly}(k)$ queries is still local because it reads a negligible fraction of the codeword. Local self-correctors have also been called *approximate local decoders* [HT18], but this definition significantly differs from our definition of ALDCs, as well as LDCs and LCCs. LDCs and LCCs have algorithms that return an *individual* index of the unique message or codeword that is close to w, and ALDCs return an individual index of a string which is ε-close to the original message with ε being significantly smaller than the original error rate δ. In contrast, a local self-corrector must return an entire message m, whose codeword $C(m)$ is within a (potentially larger) radius from w. This definition is related to local list decoding for Reed–Muller codes [GL89, STV01] and has been studied for constant-degree Reed–Muller codes [TW14, HT18, KLT23].

2 Preliminaries

For an alphabet Σ, we will make use of the $|\Sigma|$-ary entropy function $H_{|\Sigma|}(\varepsilon) = \varepsilon \log_{|\Sigma|}(|\Sigma| - 1) - \varepsilon \log_{|\Sigma|} \varepsilon - (1-\varepsilon)\log_{|\Sigma|}(1-\varepsilon)$.

2.1 Error-Correcting Codes

Definition 2.1. For some alphabet Σ, let $x, y \in \Sigma^n$. The *relative (Hamming) distance* of x, y, denoted $\delta(x, y)$, is the fraction of coordinates on which x, y differ.

Say that a string a is δ-close to a string b if the relative distance of a and b is $\leq \delta$. We first recall the definition of a (standard) code. We will also use $\mathrm{Vol}_{|\Sigma|}(\delta, k)$ to denote the number of strings $y \in \Sigma^k$ that are δ-close to any given $x \in \Sigma^k$.

Definition 2.2. $C\colon \Sigma_1^k \to \Sigma_2^n$ is a code with distance δ if for any two $x \neq x' \in \Sigma^k\colon \delta(C(x), C(x')) \geq \delta$.

We will also refer to $\delta/2$ as the decoding radius of the code, because it characterizes the maximum error rate from which unique decoding is still possible. The rate of a code is the ratio $k \log \Sigma_1/(n \log \Sigma_2)$. We will utilize two well known bounds on codes, the Singleton bound and the Gilbert–Varshamov bound.

Theorem 2.3 (Singleton bound)**.** *Any code with block length n and radius δ must have rate*

$$R \leq 1 - 2\delta + 1/n$$

Theorem 2.4 (Gilbert-Varshamov bound)**.** *For any alphabet $|\Sigma| \geq 2$ and any radius $\delta < \frac{1}{2} - \frac{1}{2|\Sigma|}$, there exists a code $C\colon \Sigma^k \to \Sigma^n$ with rate*

$$R \geq 1 - H_{|\Sigma|}(2\delta) - o(1)$$

Lemma 2.5. *For any positive integer k, constant size field $\mathbb{F}$, and $\delta \in (0, 1/2)$, there exists an explicit code $\mathrm{RS}_{\mathbb{F},k,\delta}\colon \mathbb{F}^k \to \Sigma^n$ where $|\Sigma| \leq O(n)$, with distance 2δ and rate $\geq 1 - 2\delta$.*

Proof. For any $b \leq n \leq t$ where t is a power of $|\mathbb{F}|$, the Reed–Solomon code $\Sigma^b \to \Sigma^n$ has rate $b/n = 1 - 2\delta + 1/n$, distance 2δ, and alphabet Σ which is an extension of field of $\mathbb{F}$ such that $|\Sigma| = t \leq |\mathbb{F}|n$. Therefore, to encode k-bit strings, we need to lift our string to the larger alphabet Σ by partitioning into contiguous substrings of length $\log t$; if k is not a multiple of $\log t$, we can pad by $< \log t$ additional bits. Then, the rate is

$$\frac{k}{n \log t} = \frac{k/\log t}{n} \geq \frac{b-1}{n} = 1 - 2\delta$$

The distance of the code is the same as the distance of the unmodified Reed–Solomon code, which is 2δ, because any two bit strings will map to different strings in Σ^b. □

2.1.1 Uniquely Decodable Codes

Before discussing the approximate variants of codes, we first define (standard) locally decodable codes. We first recall the definition of a standard locally decodable code.

Definition 2.6. $C\colon \Sigma_1^k \to \Sigma_2^n$ is a (q, δ, ε)-*locally decodable code* (LDC) if there exists a (randomized) decoder $M(i, r)$, where r is the randomness used, such that

1. M makes $\leq q$ queries
2. for every $i \in [k]$ and every w which is δ-close to some codeword $C(x)$,

$$\Pr_r[M^w(i, r) = x_i] \geq 1 - \varepsilon$$

The most famous example of an LDC is the Hadamard code, which encodes a message $m \in \{0,1\}^k$ as a string containing one bit for the evaluation of each of the 2^k possible linear functions (mod 2) on m.

Lemma 2.7. *The Hadamard code is a $(2, \delta, 2\delta)$-LDC.*

Proof. For input i, the decoder can pick a uniformly random linear function f, and then query two bits: the bit corresponding to $f(m)$ and the bit corresponding to $f(m) + m_i$. Each query is uniformly distributed and has probability δ of being corrupt, so by union bound both queries will be uncorrupted with probability $\geq 1 - 2\delta$ and the correct value of m_i will be returned. □

As an analogue of a standard code, we can define an approximate code.

Definition 2.8. $C\colon \Sigma_1^k \to \Sigma_2^n$ is an (δ, ε)-*approximate code* if there exists some deterministic decoding algorithm M such that for every $w \in \Sigma_2^n$ which is δ-close to some $c \in C$, $M(w)$ returns a string z such that $\delta(z, c) \leq \varepsilon$.

We often refer to δ as the *decoding radius* and ε as the *error* of the code. Note that it is entirely possible for an approximate code to have distance 0. Indeed, C may map two messages that are ε-close to the same codeword. This will be the case in our constructions. One may ask how such codes could still be useful. The observation is that when composed with a standard code, the messages encoded by the approximate code are themselves codewords that should be ε-far. It is natural to ask whether the decoding procedures of an approximate code can be local.

Definition 2.9. $C\colon \Sigma_1^k \to \Sigma_2^n$ is an (q, δ, ε)-*approximately locally decodable code* (ALDC) if there exists a randomized decoder $M^w(i, r)$, where r is the randomness used, such that

1. M makes $\leq q$ queries
2. for every w which is δ-close to some codeword $C(x)$,

$$\Pr_{i \in [k], r}[M^w(i, r) = x_i] \geq 1 - \varepsilon$$

Again we refer to δ as the radius and ε as the error of an ALDC. First, we observe that a (q, δ, ε)-ALDC is in fact also a (δ, ε)-approximate code. This is because for every w that is δ-close to a codeword, there exists a fixing r_w such that $\Pr_i[M^w(i, r_w)] \geq 1 - \varepsilon$. Thus to (inefficiently) decode any w to error smaller than ε, we can simply do the following. We keep a lookup table of all the randomness strings r_w that have the property above for each possible w. For any input w, we can then lookup r_w and run the local decoding procedure using this randomness for all i's. This fact will be important when we prove bounds on the parameters of approximate codes, as it will in turn prove bounds on ALDCs.

The explicit notion of an ALDC, with deterministic and randomized decoders, has appeared previously in [BET10] and [Sol09] respectively. Some composition results are given, although none for the composition of an ALDC with randomized decoder and an LDC. As discussed in the introduction, [BET10] show that an ALDC with deterministic decoder and the right parameters can be composed with an LDC to increase the decoding radius.

We stress that it is natural to consider randomized decoders as well, and we construct an ALDC in Theorem 1.6 with a randomized decoder with a query to error tradeoff that cannot be achieved with a deterministic decoder.[3] It is simple to extend the composition result of [BET10] for when the decoder is randomized.

Theorem 2.10. *Pick any $\gamma > 0$. If $C_1\colon \Sigma_1^k \to \Sigma_2^n$ is a $(q, (1+\gamma)\delta, \varepsilon)$-LDC and if $C_2\colon \Sigma_2^n \to \Sigma_3^{n'}$ is a (q', δ', δ)-ALDC, then $C_2 \circ C_1$ is a $(qq', \delta', \varepsilon + e^{-\Theta(\gamma^2\delta^2 n)})$-LDC.*

[3] If a (q, δ, ε)-ALDC has a deterministic decoder, then it suffices to corrupt a δ fraction of the codeword in order to fool the decoder on an $\varepsilon \geq \delta/q$ fraction of the message. However, when the decoder is randomized (e.g. Theorem 1.6), we can do better and achieve greater error reduction.

Proof. Consider the behavior of the decoder M_1 for the LDC C_1 when its queries are supplied by the decoder M_2 for the ALDC C_2. Consider running the decoder for C_2 for all its message indices i using independent randomness for every i to obtain a string $s \in \Sigma_2^n$. The behavior of M_1 combined with M_2 is identical to the behavior of M_1 on s, so it suffices to show that s has at most $(1+\gamma)\delta n$ errors. Call the randomness used for index i: r_i. Let Z_i for $i \in [n]$ be the indicator random variable of whether s_i is correct. By definition, $\mathbb{E}_{r_1,\ldots,r_n}[\sum Z_i] \leq \delta n$. Since the Z_i's are independent, Hoeffding's inequality tells us:

$$\Pr\left[\sum Z_i \geq (1+\gamma)\delta n\right] \leq e^{-2\gamma^2\delta^2 n}.$$

So with high probability, s has no more than $(1+\gamma)\delta n$ errors, and hence the local decoder for C_1 sees sufficiently few errors. Finally, note that M_1's queries can be answered in an online manner using M_2, so only the requested symbols of s need be computed. □

To summarize, one can compose an ALDC with an LDC with vanishingly small overhead in the final failure probability of the LDC. In many applications, such a small overhead in error is not necessary, as ε is considered a large fixed constant such as $1/3$. However, we note that how small ε can be relative to δ for a standard LDC has also been the subject of study [GM12]. Thus we present the vanishingly small error overhead as it is relevant for those cases.

As to the existence of approximate locally decodable codes (other than full-strength locally decodable codes which are trivially approximate locally decodable codes), to the best of our knowledge the identity code is the only such code mentioned in the literature ([BET10]), where it is noted that approximate local decoding seems significantly easier than local decoding: there exist approximate locally decodable codes with constant query complexity and polynomial block length (like the identity code), while there are no known constant-query locally decodable codes that achieve polynomial block length.

2.1.2 List Decodable Codes

When a string w is very far from a particular codeword (i.e., has r fraction of errors where r is close to 1), it may be impossible to exactly or even approximately decode, because multiple codewords, corresponding to drastically different messages, may be r-close to w. Even so, if this list of nearby codewords is small for every w, then the code is *list decodable*.

Definition 2.11. A code $C\colon \Sigma_1^k \to \Sigma_2^n$ is an (r,ℓ)-*list decodable code* if for every $w \in \Sigma_2^n$, there are $\leq \ell$ codewords $c \in C$ where $\delta(w,c) \leq r$.

The probabilistic method yields nonexplicit list decodable codes which have nearly-optimal rate:

Proposition 2.12 ([Vad12, Theorem 5.8])**.** *For an arbitrary alphabet Σ, all integers n, ℓ, and $r \in (0, 1-1/|\Sigma|)$, there (nonexplicitly) exists an (r,ℓ)-list decodable code $C\colon \Sigma^k \to \Sigma^n$ with rate $\geq 1-H_{|\Sigma|}(r)-\frac{1}{\ell+1}$. Note that any (r,ℓ)-list decodable code must have rate $\leq 1-H_{|\Sigma|}(r)+\frac{\log_{|\Sigma|}\ell}{n}+o(1)$.*

In the list decoding setting, we can define an analogue to approximate codes: every string w can be decoded to a small list S, such that if $C(m)$ is r-close to w, then m is ε-close to something in S.

Definition 2.13. A code $C\colon \Sigma_1^k \to \Sigma_2^n$ is an (r,ε,ℓ)-*approximate list decodable code* if for every $w \in \Sigma_2^n$, there is a set $S \subseteq \Sigma_1^k$ of cardinality $\leq \ell$ such that for every $m \in \Sigma_1^k$ where $\delta(w, C(m)) \leq r$, there exists $s \in S$ such that $\delta(s,m) \leq \varepsilon$.

Finally, we give a natural definition for a code with a local decoder which outputs a small list of potential symbols for each message index:

Definition 2.14. A code $C\colon \Sigma_1^k \to \Sigma_2^n$ is an $(q, r, \varepsilon, \ell)$-*approximate local weak list decodable code* if there exists a randomized decoder $M^w(i, s)$, where s is the randomness used, which returns a set of symbols $\subseteq \Sigma_1$ such that

1. M makes $\leq q$ queries
2. $\forall w, i, r.\ |M^w(i, r)| \leq \ell$
3. for every $w \in \Sigma_2^n$, if a message $m \in \Sigma_1^k$ has a codeword $C(m)$ which is r-close to w, then

$$\Pr_{i\in[k],s}[m_i \in M^w(i, s)] \geq 1 - \varepsilon$$

This definition makes sense when $|\Sigma_1| > \ell$, or else returning the entire alphabet is a trivial approximate local weak list decoder which does not need to make any queries. We also want to emphasize that this weak list decoding definition significantly differs from approximate local list decodable codes studied in works such as [IJKW10], where the decoder on w must return a small list of efficiently computable functions, such that if $C(m)$ is r-close to w, then m is ε-close to the output of one of the generated functions.

2.2 Samplers

Throughout this work, we will use *samplers* to construct ALDCs. A sampler is a procedure that aims to estimate the average value of a function f by only querying a small subset of possible inputs.

Definition 2.15. A randomized algorithm Samp is a (α, β)-*sampler* with sample complexity D if for every function $f\colon [n] \to [0, 1]$, Samp makes $\leq D$ queries to an oracle for f and satisfies the following condition:

$$\Pr_r\left[\left|\mathrm{Samp}^f(r) - \mathop{\mathbb{E}}_{j'\in[n]}[f(j')]\right| \leq \alpha\right] \geq 1 - \beta$$

We make extensive use of *oblivious samplers*, where the estimate is computed by taking the mean of f on a subset of inputs chosen uniformly randomly from a fixed family; such samplers are equivalently described as bipartite graphs with the set of inputs on the right and the family of potential query sets on the left.

Definition 2.16. A bipartite graph G with bipartition $([k], [n])$ and with left degree D is an *oblivious* (α, β)-*sampler* if for every function $f\colon [n] \to [0, 1]$,

$$\Pr_i\left[\left|\mathop{\mathbb{E}}_{j\in N(i)}[f(j)] - \mathop{\mathbb{E}}_{j'\in[n]}[f(j')]\right| \leq \alpha\right] \geq 1 - \beta$$

where $N(i)$ denotes the right vertices neighboring the ith left vertex in G.

The value of an oblivious sampler is that its graph equivalent provides a blueprint for redistributing message symbols such that if any δ fraction of codeword symbols are corrupted (represented by an indicator function f), then most of the message symbols will still have roughly δ fraction of their relevant codeword symbols affected. Indeed, we use a well-known oblivious sampler based on random walks on expanders in both of our ALDC constructions.

Theorem 2.17 ([Gil98]). *For any $\alpha, \beta > 0$ and for sufficiently large k, there exists an efficiently constructable biregular (α, β)-sampler with k left vertices, left degree $D = O(\frac{1}{\alpha^2} \log \frac{1}{\beta})$, and $n = k/2^{O(D)}$ right vertices.*

Proof sketch. Consider a constant-degree expander graph on n vertices, which is efficiently constructable. By the Expander Chernoff bound, a random walk of length D from a uniformly random starting vertex will visit any given set S with frequency that is α-close to $|S|/n$. Each of the k left vertices represent a unique string of $\log k = \log n + O(D)$ bits which encode the starting vertex and the instructions of the random walk. The right vertices are associated with the vertices of the expander, and every left vertex is connected to every vertex that the random walk visits. □

Note that this sampler has the optimal sample size/left degree, because any sampler must have $D = \Omega(\frac{1}{\alpha^2} \log \frac{1}{\beta})$ [CEG95].

Remark 2.18. It does not hurt to run a random walk for more steps than is needed, as this will only improve the mixing properties. Thus for a given k and any larger desired left degree $D = \Omega\left(\frac{1}{\alpha^2} \log \frac{1}{\beta}\right)$, there is still an explicit (α, β)-sampler with left degree D and $n = k/2^{O(D)}$. This means, for example, that one can run a random walk say, a constant factor longer than is necessary for the desired parameters α, β on a graph that is only a constant factor smaller. We will make use of this fact in our constructions when the required left degree D of the (α, β)-sampler required may be slightly more than the minimum degree needed according to Theorem 2.17.

3 ALDCs with Nearly Optimal Rate

In this section, we will first present an approximate code with high rate. We will then prove that such a rate is (nearly) optimal. Finally we'll show how to use this approximate code to construct an explicit ALDC with efficient encoding and decoding procedure with roughly the same rate.

3.1 An Approximate Code "Surpassing" the Singleton Bound

The rate of a standard code cannot be larger than roughly $1 - d$ for relative distance d by the Singleton bound. As discussed before, the relative distance of an approximate code could be 0, so this bound is meaningless. Since the decoding radius δ is the relevant parameter for an approximate code, one might believe that, analogously to the Singleton bound, the rate of an approximate code is at most roughly $1 - 2\delta$. We first present an approximate code showing that we can in fact do slightly better.

To achieve this slightly better rate, we will make use of nearly-optimal covering codes.

Definition 3.1. $D \subseteq \Sigma^k$ is a *covering code* of radius ε if every string $w \in \Sigma^k$ is ε-close to some $d \in D$.

Proposition 3.2 ([KSV03, Corollary 1.4]). *For all $\varepsilon \geq 3/k$ and all constant size alphabets Σ, there (non-explicitly) exists a covering code $D \subseteq \Sigma^k$ of radius ε such that*

$$|D| \leq O(\varepsilon k \log(\varepsilon k)) \cdot \frac{|\Sigma|^k}{\mathrm{Vol}_{|\Sigma|}(\varepsilon, k)} = |\Sigma|^{(1 - H_{|\Sigma|}(\varepsilon) + f(k))k}$$

where the last equality holds when $\varepsilon < 1 - 1/|\Sigma|$. In particular,

$$f(k) = O\left(\frac{\log_{|\Sigma|}(\varepsilon k)}{k}\right) = o(1)$$

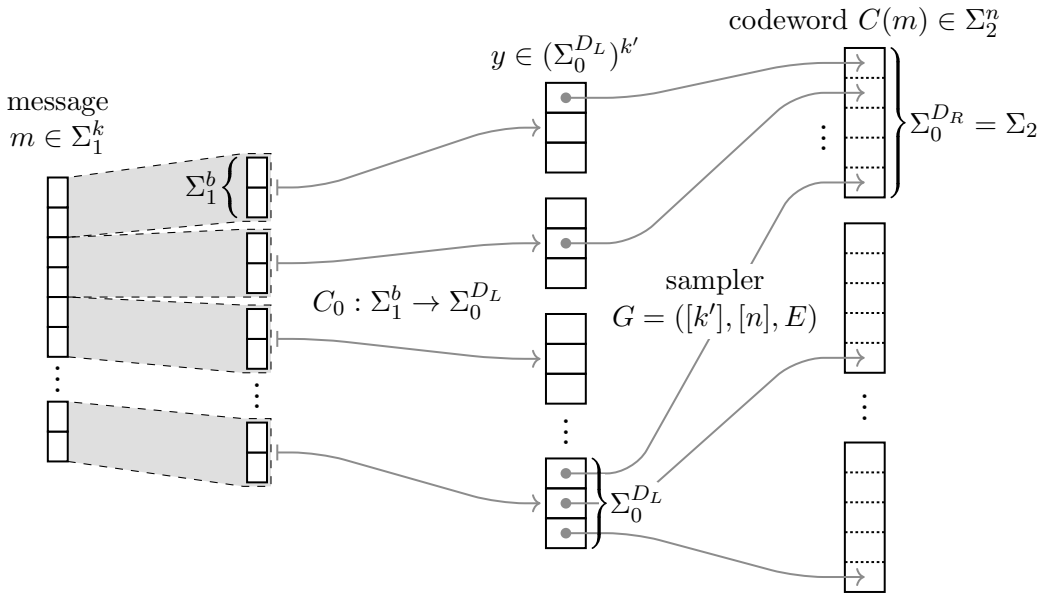

Figure 1: The construction of our ALDC with nearly optimal rate. We first divide the message into blocks of constant length b and encode each block with a near-optimal rate approximate code C_0 found via brute force. On the right hand side, we have a biregular (δ, ε)-sampler between $k' = k/b$ nodes and n nodes with left degree D_L and right degree D_R. Each coordinate of the final codeword consists of the concatenation over all neighbors of the coordinate, of a symbol from the neighbor. The code inherits the rate of C_0.

Remark 3.3. Any covering code $D \subseteq \Sigma^k$ of radius ε must have cardinality

$$|D| \geq \frac{|\Sigma|^k}{\mathrm{Vol}_{|\Sigma|}(\varepsilon, k)} \geq |\Sigma|^{(1-H_{|\Sigma|}(\varepsilon))k}$$

which is achieved when each message is ε-close to a unique covering codeword. Hence, Proposition 3.2 is nearly optimal.

We will need to make use of covering codes for constant length messages in our local decoding construction. Therefore we will need to know how large such constant lengths k must be for the $f(k)$ in Proposition 3.2 to be negligible. Towards this end, we can modify the proposition to get the following remark.

Corollary 3.4. *There exists a universal constant γ such that for any constant $\varepsilon > 0$, and for any $k \geq \gamma/\varepsilon^2$, there (non-explicitly) exists a covering code $D \subseteq \Sigma^k$ of radius ε such that*

$$|\Sigma|^{(1-H_{|\Sigma|}(\varepsilon))k} \leq |D| \leq \frac{|\Sigma|^k}{\mathrm{Vol}_{|\Sigma|}(\varepsilon, k)} = |\Sigma|^{(1-0.99H_{|\Sigma|}(\varepsilon))k}$$

The .99 can be made arbitrarily close to 1 for larger chosen constants γ.

We show that approximate codes exist for any constants $\delta > \varepsilon > 0$ with optimal rate by using a covering code to compress the message space, and then applying Reed-Solomon.

Lemma 3.5. *For any constants $1/2 > \delta > \varepsilon > 0$, any constant size field $\mathbb{F}$, and any $k \geq 3/\varepsilon$, there exists an approximate code $C \colon \mathbb{F}^k \to (\mathbb{F}')^n$ with decoding radius δ, error ε, and rate $\frac{1-2\delta}{1-H_{|\mathbb{F}|}(\varepsilon)+o(1)}$, where $\mathbb{F}'$ is an extension field of $\mathbb{F}$ and $|\mathbb{F}'| \leq |\mathbb{F}|n$.*

Proof. Let $D \subseteq \mathbb{F}^k$ be a covering code of radius ε from Proposition 3.2, so that

$$|D| = |\mathbb{F}|^{(1-H_{|\mathbb{F}|}(\varepsilon)+o(1))k} =: |\mathbb{F}|^{k'}$$

We can (inefficiently) obtain D by brute force. Let $f\colon \mathbb{F}^k \to D$ be a function such that for all $w \in \mathbb{F}^k$, $f(w)$ is some element of D which is ε-close to w (if there are multiple such elements in the covering code, pick one arbitrarily and deterministically). Let $g\colon D \to \mathbb{F}^{k'}$ be an arbitrary bijection. Both f and g can be obtained by brute force.

For any given message $w \in \mathbb{F}^k$, we can then let $x = g(f(w)) \in \mathbb{F}^{k'}$, and then encode this string using Reed–Solomon via Lemma 2.5. To decode a string which is δ-close to a codeword, we can use the distance of $\mathrm{RS}_{\mathbb{F},k',\delta}$ to decode and obtain $s \in \mathbb{F}^{k'}$ which is the message corresponding to the unique nearest Reed–Solomon codeword. Then, $g^{-1}(s) = d \in D$, and d is ε-close to the original message.

It remains to show the rate of this approximate code. Since $k'/k = 1 - H_{|\mathbb{F}|}(\varepsilon) + o(1)$ and the rate of $\mathrm{RS}_{k,\delta}$ is $1 - 2\delta$, the overall rate is

$$\frac{k}{k'} \cdot (1 - 2\delta) = \boxed{\frac{1 - 2\delta}{1 - H_{|\mathbb{F}|}(\varepsilon) + o(1)}}$$

as desired. □

Thus we've shown that when a decoder only needs to return approximate answers, the rate of the code can be slightly improved. Again, in our local decoding construction we will utilize these approximate codes only for constant lengths. Thus we must see how large these constant lengths must be for good. In the same vein as Corollary 3.4 we can show the following:

Corollary 3.6. *There exists a universal constant γ such that for any constants $1/2 > \delta > \varepsilon > 0$, any constant size field $\mathbb{F}$, and any $k \geq \gamma/\varepsilon^2$, there exists an approximate code $C\colon \mathbb{F}^k \to (\mathbb{F}')^n$ with decoding radius δ, error ε, and rate $\frac{1}{1-H_{|\mathbb{F}|}(\varepsilon)} \geq R \geq \frac{1-2\delta}{1-.99H_{|\mathbb{F}|}(\varepsilon)}$, where $\mathbb{F}'$ is an extension field of $\mathbb{F}$ and $|\mathbb{F}'| \leq |\mathbb{F}|n$. Again, the .99 can be made arbitrarily close to 1 by choosing γ accordingly.*

3.2 An "Approximate" Singleton Bound

We now show that the rate of the code above is in fact essentially optimal. To do so, we prove an approximate analogue of the Singleton bound.

Theorem 3.7. *Suppose $C\colon \Sigma_1^k \to \Sigma_2^n$ is a (δ, ε)-approximate code with rate R. Then*

$$R \leq \frac{1 - 2\delta + o(1)}{1 - H_{|\Sigma_1|}(2\varepsilon) - o(1)}$$

Proof. Let $C'\colon \{0,1\}^{k'} \to \Sigma_1^k$ be a standard code of radius ε and rate $R' = 1 - H_{|\Sigma_1|}(2\varepsilon) - o(1)$ guaranteed by the Gilbert–Varshamov bound.

The composition $C \circ C'$ is a code with radius δ. The rate of this code is RR'. Moreover, the rate of this code must obey the Singleton bound. Thus we have:

$$(1 - H_{|\Sigma_1|}(2\varepsilon) - o(1))R \leq RR' \leq 1 - 2\delta + o(1) \implies \boxed{R \leq \frac{1 - 2\delta + o(1)}{1 - H_{|\Sigma_1|}(2\varepsilon) - o(1)}}$$ □

Note that this same approach can be applied to any rate upper bound for $C \circ C'$, and we can pick whichever is strongest for the desired alphabet size.

3.3 An ALDC Approaching the "Approximate" Singleton Bound

Now that we see that our approximate code has nearly optimal rate, we turn to the task of making such codes local. To do so, we closely follow the techniques of [AEL95] and [KMRS17]. For completeness we provide the full construction and its analysis here.

Code construction. Recall that for a given constant sized message alphabet Σ_1 which is a finite field of size a, sufficiently large message length k, and constants $1/2 > \delta > \varepsilon$, we wish to construct a (q, δ, ε)-ALDC $C\colon \Sigma_1^k \to \Sigma_2^n$ with good rate and small q for some Σ_2 and n.

Let $0 < \alpha < \min\{1/4 - \delta/2, \varepsilon\}$ be a parameter. Recall that by Theorem 2.17, for any α and ε there exists a biregular $(\alpha, \varepsilon/2)$-sampler with left degree $D_L^* = \Theta(\frac{1}{\alpha^2}\log\frac{1}{\varepsilon})$. Choose $b > 1$ to be an integer which is $b \geq \gamma/\varepsilon^2$ such that, according to Corollary 3.6, there is a $(\delta + \alpha, \varepsilon/2)$-approximate code $C_0\colon \Sigma_1^b \to \Sigma_0^{D_L}$ with rate at least $\frac{1-2\delta-2\alpha}{1-.99H_{|\Sigma_1|}(\varepsilon/2)}$ and with $D_L \geq D_L^*$. If we choose $b = \Theta(D_L^* \log_{|\Sigma_1|} D_L^*)$ then $D_L^* \leq D_L \leq O\left(\frac{D_L^*}{1-2\delta-2\alpha}\right) \leq O(\frac{1}{\alpha}D_L^*)$ and the conditions for Corollary 3.6 are satisfied.

Given a length k input message x, we divide the message into $k' = k/b$ blocks of length b; if b does not divide k we can add $\leq b$ symbols of padding (which is a constant because D_L is a constant) and only hurt the rate by a $o(1)$ term. We encode each block using C_0. This gives us a string $y \in ((\Sigma_0)^{D_L})^{k'}$.

Incorporating Remark 2.18, there also exists an explicit $(\alpha, \varepsilon/2)$ sampler $G = ([k'], [n], E)$ which has any left degree $D_L \geq D_L^*$ and has right degree $D_R = k'D_L/n$, simply by adding more steps to the random walk. This enhanced sampler is the one we will actually use to encode.

We treat each coordinate of y as a left vertex of G. The final codeword $C(x) \in \Sigma_2^n$ for $\Sigma_2 = (\Sigma_0)^{D_R}$ is defined as follows. Fix any $i \in [n]$. We now define $C(x)_i$. For every neighbor $j \in [k']$ of i, let $e(j) \in [D_L]$ be the number such that the edge entering j from i is the $e(j)$-th edge leaving i in some arbitrary ordering of the edges leaving j. Finally let $\sigma_j \in \Sigma_0$ be the $e(j)$-th symbol in the block corresponding to neighbor j. The i-th symbol of $C(x)$ is the concatenation of all such σ_j-s.

Decoder and analysis. We prove the following theorem:

Theorem 3.8. *For parameter $\min\{1/4 - \delta/2, \varepsilon\} > \alpha > 0$, the code above $C\colon \Sigma_1^k \to \Sigma_2^n$ is a $(D_L, \delta, \varepsilon)$-ALDC with $|\Sigma_2| \leq O(D_L)^{D_R}$. The rate of the code is*

$$\frac{1 - 2\delta - 2\alpha - o(1)}{1 - .99H_{|\Sigma_1|}(\varepsilon/2)}.$$

Proof. We first calculate the rate. Note that in the final output $C(x)$, we copy every symbol in Σ_0 contained in y exactly once, and $\leq b$ additional symbols of padding are added. Thus the rate of the code is the rate between the message and y. This is simply the rate of $(\delta+\alpha, \varepsilon/2)$-approximate code C_0:

$$\frac{1 - 2(\delta + \alpha)}{1 - .99H_{|\Sigma_1|}(\varepsilon/2)} - o(1)$$

We now show that there is a decoder that correctly outputs x_i for at least $1 - \varepsilon$ coordinates i given a word w that is δ-close to $C(x)$. On input i, the decoder does the following.

1. Find the block $i' \in [k']$ containing index i.
2. For each neighbor of i' in G, $j \in \Gamma(i') \subset [n]$, query the j-th coordinate of w. This gives D_L symbols in $\Sigma_2 = \Sigma_0^{D_R}$. Let $\sigma_\ell \in \Sigma_2$ for each $\ell \in [D_L]$ denote the symbol obtained from the ℓ-th neighbor of j.

3. For each σ_ℓ, let $\sigma_{0,\ell} \in \Sigma_0$ denote the symbol within σ_ℓ corresponding to the ℓ-th symbol in block i'. This gives a string $s \in \Sigma_0^{D_L}$.

4. Run the decoding algorithm for C_0 on s. This gives a string in $\{0,1\}^b$ representing the bits in the decoded message for all coordinates in block i'. Return the bit corresponding to coordinate i.

Observe that this decoding procedure is in fact deterministic. We now prove correctness. Let $B \subset [n]$, $|B| \leq \delta n$, be any subset of corrupted coordinates. By the sampler property, all but at most $\varepsilon/2$-fraction of the k' blocks have a neighborhood with at most $(\delta+\alpha)$-fraction of neighbors in the corrupted set. This means that on at least $(1-\varepsilon/2)$-fraction of blocks, the decoding algorithm for C_0 will return a string $s \in \{0,1\}^b$ that is $\varepsilon/2$-close to the true message on that block. Thus the decoder will only err on at most ε-fraction of the message coordinates overall. □

Finally, it remains to express D_L in terms of our parameters. Substituting the value of D_L, observing that C_0 can be found via brute force, and calculating the alphabet size of Σ_2 we get:

Corollary 3.9. *For any constants $\delta, \varepsilon > 0$, any constant parameter $0 < \alpha < \min\{1/4 - \delta/2, \varepsilon\}$, any constant sized alphabet Σ_1 which is a finite field, and any sufficiently large k, there exists an explicit $C\colon \Sigma_1^k \to \Sigma_2^n$ is a (q, δ, ε)-ALDC with $q = O\left(\frac{1}{\alpha^3}\log\frac{1}{\varepsilon}\right)$ with $|\Sigma_2| \leq |\Sigma_1|^{q2^{O(q)}}$. The rate of the code is*

$$\frac{1-2\delta-2\alpha}{1-.99H_{|\Sigma_1|}(\varepsilon/2)} - o(1).$$

Proof. As discussed above, $D_L = \frac{O(D_L^*)}{\alpha} = O\left(\frac{1}{\alpha^3}\log\frac{1}{\varepsilon}\right)$. Since $n = k'/2^{O(D_L)}$, we have $D_R = k'D_L/n = D_L \cdot 2^{O(D_L)}$. Thus Theorem 3.8 shows that this is an ALDC with desired parameters. For explicitness, we simply observe that b is constant, and so we can find the approximate code $C_0\colon \Sigma_1^b \to (\Sigma_0)^{D_L}$ via brute force. □

4 Approximate List Decoding

The technique of message compression via a covering code, which we used in the previous section to build approximate codes, can also be used to yield approximate list decodable codes. We will instantiate this technique using nearly-optimal list decodable codes (see Proposition 2.12).

Lemma 4.1. *If $D \subseteq \Sigma_1^k$ is a covering code of radius ε with $|D| = |\Sigma_1|^{k'}$, and if a code $C\colon \Sigma_1^{k'} \to \Sigma_2^n$ is an (r,ℓ)-list decodable code with rate R, then there exists an (r,ε,ℓ)-approximate list decodable code $C'\colon \Sigma_1^k \to \Sigma_2^n$ with rate kR/k'.*

Proof. In the same manner as Lemma 3.5, let $f\colon \Sigma_1^k \to D$ be a function such that for all $w \in \Sigma_1^k$, $f(w)$ is some element of D which is ε-close to w (if there are multiple such elements in the covering code, pick one arbitrarily and deterministically). Let $g\colon D \to \Sigma_1^{k'}$ be an arbitrary bijection. Both f and g can be obtained by brute force.

For any given message $m \in \Sigma_1^k$, let $C(m) = C(g(f(w))) \in \Sigma_2^n$. Then, consider any string $w \in \Sigma_2^n$. By the list decodability of C, there is a set S of $\leq \ell$ strings $s \in \Sigma_1^{k'}$ such that $\delta(w, C(s)) \leq r$. Hence, $g^{-1}(S)$ is a set of $\leq \ell$ messages such that if $C'(m)$ is r-close to w, then m is ε-close to something in $g^{-1}(S)$. This is because of $C'(m) = C(g(f(m)))$ is r-close to w, then $g(f(m))$ will be in the set S. So, $g^{-1}(S)$ will contain $f(m)$ which by design is ε-close to m. □

Theorem 4.2. *Let Σ be an arbitrary alphabet. Let $0 < \varepsilon < 1/2$, $0 < r < 1 - 1/|\Sigma|$, and $\ell \geq 1$. If $k \geq 3/\varepsilon$, then there is a (nonexplicit) (r, ε, ℓ)-approximate list decodable code $C \colon \Sigma^k \to \Sigma^n$ with rate*

$$\geq \frac{1 - H_{|\Sigma|}(r) - \frac{1}{\ell+1}}{1 - H_{|\Sigma|}(\varepsilon) + o(1)}$$

Proof. Combine the covering code from Proposition 3.2 and the list decodable code from Proposition 2.12 using the previous lemma. □

We will use the Alon–Edmonds–Luby technique once again to lift this approximate *globally* list decodable code to a approximate *locally weak* list decodable code. To do so, we will utilize approximate list decodable codes from Theorem 4.2 with constant length. To that end it is helpful to know how large k must be to upper bound the $o(1)$ terms in the rate.

Corollary 4.3. *Let Σ be an arbitrary alphabet with $|\Sigma| \geq 3$. Let $0 < \varepsilon < 1/2$, $1/2 < r < 1 - 1/a$, and $\ell \geq 1$. There exists a universal constant γ such that if $k \geq \max\{\gamma/\varepsilon^2, \gamma/H_{|\Sigma|}(r)^2\}$, then there is a (nonexplicit) (r, ε, ℓ)-approximate list decodable code $C \colon \Sigma^k \to \Sigma^n$ with rate $R \geq \frac{1 - H_{|\Sigma|}(r) - \frac{1}{\ell+1}}{1 - .99 H_{|\Sigma|}(\varepsilon)}$ and $R \leq \frac{1}{1 - H_{|\Sigma|}(\varepsilon)}$. The .99 can be made arbitrarily close to 1 for larger chosen constants γ.*

Our approximate locally weak list decodable code construction is identical to the construction for high-rate ALDCs (see Corollary 3.9), and we defer to that construction for details. Let $0 < \alpha < \varepsilon$ be a parameter. By Theorem 2.17, for any α and ε there exists a biregular $(\alpha, \varepsilon/2)$-sampler with left degree $D_L^* = \Theta(\frac{1}{\alpha^2} \log \frac{1}{\varepsilon})$, and we need to choose a block size $b \geq \max\{\gamma/\varepsilon^2, \gamma/H_{|\Sigma_1|}(r + \alpha)^2\}$ such that, according to Corollary 4.3, there is a $(r + \alpha, \varepsilon/2, \ell)$-approximate list decodable code $C_0 \colon \Sigma_1^b \to \Sigma_1^{D_L}$ with $D_L \geq D_L^*$, so that we can build a random walk sampler meeting or exceeding the conditions for an $(\alpha, \varepsilon/2)$-sampler, but with left degree matching D_L. It suffices to set $b = \Theta(D_L^*)$ such that $D_L^* \leq D_L \leq O(D_L^*)$ and the conditions for Corollary 4.3 are satisfied. We end up with a code that has rate $R \geq \frac{1 - H_{|\Sigma_1|}(r) - \frac{1}{\ell+1}}{1 - .99 H_{|\Sigma_1|}(\varepsilon)} - o(1)$ because the rate is the same as the constant sized code with a possible $o(1)$ additive loss due to padding. Again, the .99 can be made arbitrarily close to 1.

The decoder for this code will also work almost identically to the high-rate ALDC decoder from Theorem 3.8. Query all of the codeword indices corresponding to the block containing the desired message index i, and then run the approximate list decoding algorithm for the constant-size code on that block in order to recover a set S of $\leq \ell$ strings of length b. Then, return the set of symbols that appear in any string in S at the substring index corresponding to the message index i. For an input w, let $m \in \Sigma_1^k$ such that $C(m)$ is r-close to w. Then, $\geq 1 - \varepsilon/2$ of the blocks will be "good": they contain a $\leq r + \alpha$ fraction of symbols that differ from $C(m)$, and the constant-size approximate list decodable code will return a set of ℓ substrings of length b, one of which is $\varepsilon/2$-close to the corresponding substring of m. Therefore, the decoder acting on a $\geq 1 - \varepsilon/2$ fraction of message indices i in each good block will return a set that contains m_i, and these indices make up a $\geq 1 - \varepsilon$ fraction of the total message. Again, observing that the code from Corollary 4.3 is only used for constant message lengths and thus can be found via brute force, we can construct an explicit approximate local weak list decodable code.

Theorem 4.4. *Let Σ_1 be an arbitrary constant-sized alphabet of size ≥ 3. For parameters $0 < \alpha < \varepsilon < 1/2$, $1/2 < r < 1 - 1/|\Sigma_1|$, and $\ell \geq 1$, there exists an explicit $(q, r, \varepsilon, \ell)$-approximate local weak list decodable code $C \colon \Sigma_1^k \to \Sigma_2^n$ with $|\Sigma_2| \leq |\Sigma_1|^{q 2^{O(q)}}$. The rate of the code is*

$$\geq \frac{1 - H_{|\Sigma_1|}(r + \alpha) - \frac{1}{\ell+1}}{1 - 0.99 H_{|\Sigma_1|}(\varepsilon)} - o(1)$$

and the query complexity is

$$q \leq O\left(\frac{1}{\alpha^2}\log\frac{1}{\varepsilon}\right)$$

5 Achieving Optimal Error Reduction

What is the best possible error reduction that a q-query ALDC can achieve, and what structure would such a decoder have? In fact, we are able to show that the best possible error reduction, even for inefficient ALDCs, is $\varepsilon = \Theta(\delta^{\lceil q/2 \rceil})$ which is achieved by taking the majority of q queries. We construct an efficient q-query ALDC with constant rate and alphabet that achieves this optimal error reduction up to constant factors.

5.1 Majority Lower Bound on Error

Let C be some nonadaptive (q, δ, ε)-ALDC, over any size alphabet and with any rate. Then, we will show that $\varepsilon \geq \Omega(\delta^{\lceil q/2 \rceil})$; in particular, for two queries, this shows that C must have error $\geq \delta$ which is already achieved by the 1-query identity code. Intuitively, if a decoder doesn't trust the majority of the queries that it sees, then it is biased towards returning a certain answer and will do poorly on a different message.

Theorem 5.1. *Let $C\colon \Sigma_1^k \to \Sigma_2^n$ be a nonadaptive (q, δ, ε)-ALDC with $q = O(1)$, $0 < \delta < 1/2$, and sufficiently large n. Then, $\varepsilon \geq \Omega(\delta^{\lceil q/2 \rceil})$.*

Proof. We will construct a string which is δ-close to some codeword, such that C cannot decode more than a $1 - \Omega(\delta^{\lceil q/2 \rceil})$ fraction of message symbols. Let S be a uniformly random subset of $[n]$ of cardinality δn. Then, for sufficiently large n, we will show there exists a noisy codeword w such that $\delta(w, C(m)) \leq \delta$, and

- if q is odd,
$$\Pr_i[M^w(i) \neq m_i] \geq \Pr\left[|[q] \cap S| \geq \frac{q+1}{2}\right] \geq \Omega(\delta^{(q+1)/2}) \tag{1}$$
- if q is even,
$$\Pr_i[M^w(i) \neq m_i] \geq \Pr\left[|[q-1] \cap S| \geq \frac{q}{2}\right] \geq \Omega(\delta^{q/2}) \tag{2}$$

Note that the lower bound for odd q is precisely the probability that a majority of q queries land in S, and the lower bound for even q is identical to the lower bound for $q-1$, so that there is no advantage to making an even number of queries.

Let m be some arbitrary string in the message space of C, and let m' be an arbitrary string such that $m'_i \neq m_i$ for all $i \in [k]$. Let $c = C(m)$ and let $c' = C(m')$. Let $S \subseteq [n]$ be a uniformly chosen subset of cardinality δn. Let w be a string such that indices outside S match c while indices inside S match c', and let w' be defined in the opposite fashion.

Since C has a nonadaptive decoder, for a fixed message index i, the decoder M on a string $\tilde{w}$ is equivalent to returning $D(\tilde{w}|_Q)$, where D is a deterministic function and Q is a set of query locations, each sampled from some distribution depending on i.

$$\Pr_i[M^{\tilde{w}}(i) = \tilde{m}_i] = \mathop{\mathbb{E}}_{i,D,Q}[\Pr[D(\tilde{w}|_Q)] = \tilde{m}_i]$$

We will prove that either w or w' suffices to yield the desired upper bound for the right hand side, by choosing $(\tilde{m}, \tilde{w})$ uniformly from $\{(m, w), (m', w')\}$.

$$\Pr_{S,\tilde{m},\tilde{w},i}[M^{\tilde{w}}(i) = \tilde{m}_i] = \mathop{\mathbb{E}}_{i,D,Q}\left[\mathop{\mathbb{E}}_{S,\tilde{m},\tilde{w}}[\Pr[D(\tilde{w}|_Q)] = \tilde{m}_i\right] = \mathop{\mathbb{E}}_{i,D,Q}\left[\frac{\Pr[D(w|_Q) = m_i] + \Pr[D(w'|_Q) = m'_i]}{2}\right]$$

Consider an arbitrary i, D, and Q. Let $X = |Q \cap S|$ be the random variable representing how many queries fall into the set of indices which are corrupted.

$$p \coloneqq \Pr_S[D(w|_Q) = m_i] = \sum_{j=0}^{q} \Pr[D(w|_Q) = m_i \mid X = j] \cdot \Pr[X = j]$$

Since $m_i \neq m'_i$ for all i, $\Pr[D(w|_Q) = m_i \mid K] + \Pr[D(w|_Q) = m'_i \mid K] \leq 1$ for any event K.

$$\leq 1 - \sum_{j=0}^{q} \Pr[D(w|_Q) = m'_i \mid X = j] \cdot \Pr[X = j]$$

Analogously,

$$p' \coloneqq \Pr_S[D(w'|_Q) = m'_i] = \sum_{j=0}^{q} \Pr[D(w'|_Q) = m'_i \mid X = j] \cdot \Pr[X = j]$$

We can group terms when averaging between p and p', since $\Pr[D(w|_Q) = m_i \mid X = j] = \Pr[D(w'|_Q) = m_i \mid X = q - j]$; on both sides, the decoder sees $q - j$ queries that correspond to c and j that correspond to c'.

$$\begin{aligned}
p + p' &= 2 \Pr_{S,\tilde{m},\tilde{w}}[D(\tilde{w}|_Q) = \tilde{m}_i] \\
&\leq 1 + \sum_{j=0}^{q} (\Pr[D(w|_Q) = m'_i \mid X = q - j] - \Pr[D(w|_Q) = m'_i \mid X = j]) \cdot \Pr[X = j] \\
&\leq 1 + \sum_{j=0}^{\lfloor \frac{q-1}{2} \rfloor} (\Pr[D(w|_Q) = m'_i \mid X = q - j] - \Pr[D(w|_Q) = m'_i \mid X = j]) \cdot \Pr[X = j] \\
&\quad + \sum_{j=0}^{\lfloor \frac{q-1}{2} \rfloor} (\Pr[D(w|_Q) = m'_i \mid X = j] - \Pr[D(w|_Q) = m'_i \mid X = q - j]) \cdot \Pr[X = q - j] \\
&= 1 + \sum_{j=0}^{\lfloor \frac{q-1}{2} \rfloor} (\Pr[D(w|_Q) = m'_i \mid X = q - j] - \Pr[D(w|_Q) = m'_i \mid X = j]) \cdot (\Pr[X = j] - \Pr[X = q - j])
\end{aligned}$$

For sufficiently large n, $\Pr[X = j] \geq \Pr[X = q - j]$ when $j < q/2$ since $\Pr[X = j] = \binom{q}{j}\binom{n-q}{\delta n - j}/\binom{n}{\delta n}$ and $\Pr[X = q - j] = \binom{q}{j}\binom{n-q}{\delta n - (q-j)}/\binom{n}{\delta n}$; then, use the fact that the difference between any two probabilities is ≤ 1. This removes any dependence on i, D, or Q from the bound.

$$\begin{aligned}
&\leq 1 + \sum_{j=0}^{\lfloor \frac{q-1}{2} \rfloor} (\Pr[X = j] - \Pr[X = q - j]) \\
&\leq 1 + \sum_{j=0}^{\lfloor \frac{q-1}{2} \rfloor} \Pr[X = j] - \left(1 - \sum_{j=0}^{\lceil \frac{q-1}{2} \rceil} \Pr[X = j]\right) \\
&\leq \sum_{j=0}^{\lfloor \frac{q-1}{2} \rfloor} \Pr[X = j] + \sum_{j=0}^{\lceil \frac{q-1}{2} \rceil} \Pr[X = j]
\end{aligned}$$

If q is odd, then we have that the probability of success is

$$\begin{aligned}
\frac{p + p'}{2} &\leq \sum_{j=0}^{\frac{q-1}{2}} \Pr[X = j] \\
&= \Pr\left[|[q] \cap S| < \frac{q}{2}\right]
\end{aligned}$$

which is the desired bound (1). If q is even, then

$$\begin{aligned}
\frac{p + p'}{2} &\leq \Pr\left[X < \frac{q}{2}\right] + \frac{1}{2}\Pr\left[X = \frac{q}{2}\right] \\
&= \Pr\left[|[q-1] \cap S| < \frac{q-1}{2}\right]
\end{aligned}$$

using Lemma A.1, which matches the desired bound (2). To complete the proof:

$$\begin{aligned}
\mathop{\mathbb{E}}_{S,\tilde{m},\tilde{w}}[\Pr_i[M^{\tilde{w}}(i) = \tilde{m}_i]] &= \mathop{\mathbb{E}}_{S,\tilde{m},\tilde{w}}\left[\mathop{\mathbb{E}}_{i,D,Q}[\Pr[D(\tilde{w}|_Q)] = \tilde{m}_i]\right] \\
&= \mathop{\mathbb{E}}_{i,D,Q}\left[\mathop{\mathbb{E}}_{S,\tilde{m},\tilde{w}}[\Pr[D(\tilde{w}|_Q)] = \tilde{m}_i]\right] \\
&= \mathop{\mathbb{E}}_{i,D,Q}\left[\frac{\Pr[D(w|_Q) = m_i] + \Pr[D(w'|_Q) = m'_i]}{2}\right] \\
&= \mathop{\mathbb{E}}_{i,D,Q}\left[\frac{p + p'}{2}\right] \\
&= \frac{p + p'}{2}
\end{aligned}$$

By the probabilistic method, there exist fixed S, $\tilde{m}$, and $\tilde{w}$ (i.e., some δ-noisy codeword) such that $\Pr_i[M^{\tilde{w}}(i) = \tilde{m}_i] \leq \frac{p+p'}{2}$. □

Note that this bound is tight when the codeword alphabet is very large: consider the code $C\colon \Sigma^k \to (\Sigma^k)^n$ where each symbol of the codeword is a copy of the entire message; then, the decoder samples q uniformly random codeword indices, and returns the majority of the corresponding portion of the codeword symbols that contains the requested message index. This bound also proves that the identity code (where $C(x) = x$), which is a $(1, \delta, \delta)$-ALDC, is optimal for the ≤ 2 query regime:

Corollary 5.2. *A nonadaptive ALDC with ≤ 2 queries has error $\geq \delta$.*

Proof. A 1-query ALDC is a special case of a 2-query ALDC, so without loss of generality we consider 2-query ALDCs. The upper bound for 2-query ALDCs given by the theorem is the same as the upper bound for 1-query ALDCs, which is

$$\frac{\binom{1}{0}\binom{n-1}{\delta n}}{\binom{n}{\delta n}} = 1 - \delta \qquad \square$$

5.2 Achieving the Majority Lower Bound

Although the majority lower bound applies to all (q, δ, ε)-ALDCs, including ones with inefficient decoders, huge codeword lengths, or huge alphabet size, we will now show that the bound can be achieved up to a constant factor by explicit and efficient $(q, \delta, O(\delta^{\lceil q/2 \rceil}))$-ALDCs with constant rate and alphabet size. To do so, we apply the Alon–Bruck–Naor–Naor–Roth technique [ABN+92] to a sampler graph.

Code construction. Let $0 < \gamma < 1$ be an arbitrarily small constant, q be a positive constant integer, and $0 < \delta < 1/2$. To construct an ALDC $C \colon \Sigma_1^k \to \Sigma_2^n$, begin with a biregular graph G corresponding to an oblivious $(\alpha = \gamma\delta, \beta = \gamma\delta^{\lceil q/2 \rceil})$-sampler with bipartition $([k], [n])$, left degree D_L, and right degree D_R. Let $m \in \Sigma_1^k$ be an arbitrary message. The jth codeword symbol $C(m)_j$ is defined as the string $(m_i)_{i \in N(j)} \in \Sigma_1^{D_R}$ where $N(j)$ is the set of left vertices neighboring the jth right vertex.

Immediately, we can see that the rate of this code will be $1/D_L$ because each message symbol is duplicated D_L times in the codeword, and the alphabet is $\Sigma_2 = \Sigma_1^{D_R}$.

Decoder and analysis. For a message index i and input string w, consider the decoder which independently repeats the following step q times, and takes the majority of the results:

1. Pick a uniformly random right vertex j neighboring left vertex i in the sampler graph G.
2. Query the jth index of w, and then return the symbol of w_j which corresponds to message index i.

If w is δ-close to some codeword $C(m)$, then by the sampler property, a $1 - \beta = 1 - \gamma\delta^{\lceil q/2 \rceil}$ fraction of message coordinates will be "good": these coordinates i have at most a $\leq \delta + \alpha = (1 + \gamma)\delta$ fraction of their neighbors which are corrupted. If a message index i is good, then the probability of a uniformly random neighbor being corrupt is $\leq (1+\gamma)\delta$, and so the probability that the majority of q repetitions is corrupt is

$$\leq \sum_{\ell = \lceil q/2 \rceil}^{q} \binom{q}{\ell} ((1+\gamma)\delta)^{\ell} (1 - (1+\gamma)\delta)^{q-\ell} \leq O(((1+\gamma)\delta)^{\lceil q/2 \rceil})$$

Hence the decoder achieves the following error:

$$\mathop{\mathbb{E}}_{i}[\Pr[M^w(i) \neq m_i]] \leq \beta + (1 - \beta) \cdot O(((1+\gamma)\delta)^{\lceil q/2 \rceil}) \leq O(((1+\gamma)\delta)^{\lceil q/2 \rceil})$$

If $\gamma = 1/100q = O(1)$, then $(1+\gamma)^{\lceil q/2 \rceil} \leq e^{\gamma \lceil q/2 \rceil} \leq e^{1/100}$ and

$$O(((1+\gamma)\delta)^{\lceil q/2 \rceil}) \leq O(\delta^{\lceil q/2 \rceil})$$

Theorem 5.3. *For any $0 < \delta < 1/2$ and constant positive integer q, the code above $C\colon \Sigma_1^k \to \Sigma_2^n$ is a $(q, \delta, O(\delta^{\lceil q/2 \rceil}))$-ALDC with rate $1/D_L$ and alphabet $\Sigma_2 = \Sigma_1^{D_R}$.*

Finally, we can apply the sampler from Theorem 2.17 to complete the construction.

Corollary 5.4. *For any $0 < \delta < 1/2$ and constant positive integer q, there exists an explicit code $C\colon \Sigma_1^k \to \Sigma_2^n$ which is a $(q, \delta, O(\delta^{\lceil q/2 \rceil}))$-ALDC with rate $1/D_L$ and alphabet $\Sigma_2 = \Sigma_1^{D_R}$, where $D_L = O(\frac{q^2}{\delta^2} \log \frac{q}{\delta^{\lceil q/2 \rceil}})$ and $D_R = 2^{D_L 2^{O(D_L)}}$.*

Note that the decoder which chooses q right neighbors without replacement will perform slightly better, but for the purpose of matching the majority error bound asymptotically, it is simpler to analyze the decoder above which chooses neighbors with replacement.

5.3 Binary ALDCs with Nontrivial Error Amplification

The code C described above requires a constant-size alphabet that nevertheless grows quickly in $1/\delta$. We can concatenate this code with the Hadamard code to yield binary ALDCs, at the cost of increased rate and doubled query complexity.

Code construction. Let $C_H\colon \{0,1\}^{D_R} \to \{0,1\}^{2^{D_R}}$ be the Hadamard code for messages of D_R bits. Then, define $C_2\colon \{0,1\}^k \to \{0,1\}^{n'}$ where $n' = 2^{D_R} \cdot n$ to be the code C concatenated with C_H; that is, each symbol of the codeword of C is encoded with C_H, and the entire new codeword is flattened to be a string of bits. The rate of this new code is $D_R/D_L 2^{D_R}$.

Decoder and analysis. The decoder for C_2 is identical to the decoder for C, except that each query to a right vertex is now serviced by the 2-query local Hadamard decoder. That is, given a message index i and input string w, repeat the following q times and return the majority:

1. Pick a uniformly random right vertex j neighboring left vertex i in the sampler graph G.
2. Let $w^{(j)}$ denote the substring of w corresponding to the Hadamard encoding of the jth symbol of the base codeword. In particular, $w^{(j)}$ is the substring of length 2^{D_R} starting at the $((j-1) \cdot 2^{D_R} + 1)$th bit. Then, run the Hadamard 2-query local decoder on $w^{(j)}$ (see Lemma 2.7) to decode the single bit corresponding to message index i. Return the outcome of the Hadamard decoder.

The query complexity is now $2q$ because each query made by the original decoder is serviced by two queries of the Hadamard local decoder. If $w^{(j)}$ has a δ_j fraction of corruption, then the Hadamard decoder will return an incorrect answer with probability $\leq 2\delta_j$. Then, the probability of one iteration of the decoder returning an incorrect answer is

$$p_i \coloneqq \mathop{\mathbb{E}}_{j \in N(i)}[2\delta_j] = 2 \mathop{\mathbb{E}}_{j \in N(i)}[\delta_j]$$

If we define a function on the right vertices $f(j) = \delta_j \in [0,1]$, then by the sampler property, a $1-\beta$ fraction of message coordinates will have $\mathbb{E}_{j \in N(i)}[\delta_j] \leq \delta + \alpha$ and hence $p_i \leq 2(\delta + \alpha)$. So a good coordinate i will be successfully decoded with probability

$$\leq O((2(1+\gamma)\delta)^{\lceil q/2 \rceil}) \leq O((2\delta)^{\lceil q/2 \rceil})$$

using $\gamma = 1/100q$ as before. The total error of the decoder is

$$\mathop{\mathbb{E}}_i[\Pr[M^w(i) \neq m_i]] \leq \beta + (1-\beta) \cdot O((2\delta)^{\lceil q/2 \rceil}) \leq O((2\delta)^{\lceil q/2 \rceil})$$

Theorem 5.5. *For any $0 < \delta < 1/2$ and constant positive integer q, the code above $C_2 \colon \{0,1\}^k \to \{0,1\}^{n'}$ is a $(2q, \delta, O((2\delta)^{\lceil q/2 \rceil}))$-ALDC with rate $D_R/D_L 2^{D_R}$ where $D_L = O(\frac{q^2}{\delta^2} \log \frac{q}{\delta^{\lceil q/2 \rceil}})$ and $D_R = 2^{D_L 2^{O(D_L)}}$.*

6 Open problems

ALDCs with optimal rate and error reduction. We gave two contructions of ALDCs. The first had nearly optimal rate, and a constant number of queries. However, the error reduction is not optimal for the number of queries it uses. The second construction has optimal error reduction the its number of queries. However, the rate is far from optimal (although still constant). It is an open question whether one can construct an ALDC with the best of both worlds or whether it is impossible. That is, do there exist (q, δ, ε)-ALDCs with rate approaching the Singleton bound, and $\varepsilon = O(\delta^{\lceil q/2 \rceil})$?

Binary ALDCs with optimal query complexity. Both of our constructions have a large constant alphabet due to the sampler technique. A natural question to ask is what can be done with a smaller alphabet. In section 5.3 we showed that concatenating our construction for optimal error reduction from section 5.2 with Hadamard gives an ALDC with binary alphabet. However, this doubles the number of queries and so no longer gives optimal error reduction for its query complexity. Thus one can ask what is achievable with binary ALDCs. We do not know whether the same δ to $O(\delta^{\lceil q/2 \rceil})$ error reduction is possible for a binary ALDC.

Strong list decoding with constant number of queries and constant rate. Our list decoding result only works for a weak notion of list decoding. It is natural to ask whether the standard notion of approximate local list decoding can be achieved with constant query complexity and constant rate. The work of [IJKW10] constructs an approximate local list decodable code in the strong sense with a constant number of queries but with a polynomially small rate (see also [BET10, Theorem 2]). The work of [DHK+21] constructs a code with approximate *global* list decoding with constant rate. Both of these codes are derandomized direct product codes.

Acknowledgements

GM is supported by NSF Grant CCF-2200956, an NSF Graduate Research Fellowship (DGE-2137420), and a UT Austin Dean's Prestigious Fellowship Supplement. DM is supported by NSF Grants CCF-1705028, CCF-2200956, and CCF-2312573. JO is supported by NSF Grants CF-2008076 and CCF-2312573. We are grateful to anonymous reviewers for their input.

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-2137420. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[ABN+92] Noga Alon, Jehoshua Bruck, Joseph Naor, Moni Naor, and Ron M. Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs.

IEEE Transactions on Information Theory, 38(2):509–516, March 1992. doi:10.1109/18.119713.

[AEL95] Noga Alon, Jeff Edmonds, and Michael Luby. Linear time erasure codes with nearly optimal recovery. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 512–519, October 1995. ISSN: 0272-5428. doi:10.1109/SFCS.1995.492581.

[AG24] Omar Alrabiah and Venkatesan Guruswami. Near-tight bounds for 3-query locally correctable binary linear codes via rainbow cycles. Technical Report TR24-062, Electronic Colloquium on Computational Complexity (ECCC), April 2024. URL: https://eccc.weizmann.ac.il/report/2024/062/.

[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, page 1438–1448, New York, NY, USA, 2023. Association for Computing Machinery. doi:10.1145/3564246.3585143.

[AL96] Noga Alon and Michael Luby. A linear time erasure-resilient code with nearly optimal recovery. *IEEE Trans. Inf. Theory*, 42(6):1732–1736, 1996. doi:10.1109/18.556669.

[BDG19] Jop Briët, Zeev Dvir, and Sivakanth Gopi. Outlaw distributions and locally decodable codes. *Theory of Computing*, 15(12):1–24, 2019. URL: https://theoryofcomputing.org/articles/v015a012, doi:10.4086/toc.2019.v015a012.

[BET10] Avraham Ben-Aroya, Klim Efremenko, and Amnon Ta-Shma. A note on amplifying the error-tolerance of locally decodable codes. Technical Report TR10-134, Electronic Colloquium on Computational Complexity (ECCC), December 2010. URL: https://eccc.weizmann.ac.il/report/2010/134/.

[BGH+06] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. Robust PCPs of proximity, shorter PCPs, and applications to coding. *SIAM Journal on Computing*, 36(4):889–974, January 2006. Publisher: Society for Industrial and Applied Mathematics. doi:10.1137/S0097539705446810.

[CEG95] Ran Canetti, Guy Even, and Oded Goldreich. Lower bounds for sampling algorithms for estimating the average. *Information Processing Letters*, 53(1):17–25, January 1995. URL: https://www.sciencedirect.com/science/article/pii/002001909400171T, doi:10.1016/0020-0190(94)00171-T.

[CY21] Gil Cohen and Tal Yankovitz. Rate amplification and query-efficient distance amplification for linear LCC and LDC. In Valentine Kabanets, editor, *36th Computational Complexity Conference, CCC 2021, July 20-23, 2021, Toronto, Ontario, Canada (Virtual Conference)*, volume 200 of *LIPIcs*, pages 1:1–1:57. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CCC.2021.1.

[CY22] Gil Cohen and Tal Yankovitz. LCC and LDC: tailor-made distance amplification and a refined separation. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 44:1–44:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ICALP.2022.44.

[DGL21] Marcel Dall'Agnol, Tom Gur, and Oded Lachish. A structural theorem for local algorithms with applications to coding, testing, and privacy. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1651–1665. SIAM, 2021. doi:10.1137/1.9781611976465.100.

[DGY11] Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching vector codes. *SIAM J. Comput.*, 40(4):1154–1178, 2011. doi:10.1137/100804322.

[DH13] Irit Dinur and Prahladh Harsha. Composition of low-error 2-query PCPs using decodable PCPs. *SIAM J. Comput.*, 42(6):2452–2486, 2013. doi:10.1137/100788161.

[DHK+21] Irit Dinur, Prahladh Harsha, Tali Kaufman, Inbal Livni Navon, and Amnon Ta-Shma. List-decoding with double samplers. *SIAM J. Comput.*, 50(2):301–349, 2021. doi:10.1137/19M1276650.

[Efr12] Klim Efremenko. 3-query locally decodable codes of subexponential length. *SIAM J. Comput.*, 41(6):1694–1703, 2012. doi:10.1137/090772721.

[GI02] Venkatesan Guruswami and Piotr Indyk. Near-optimal linear-time codes for unique decoding and new list-decodable codes over smaller alphabets. In *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*, STOC '02, page 812–821, New York, NY, USA, 2002. Association for Computing Machinery. doi:10.1145/509907.510023.

[GI04] Venkatesan Guruswami and Piotr Indyk. Efficiently decodable codes meeting Gilbert-Varshamov bound for low rates. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '04, page 756–757, USA, 2004. Society for Industrial and Applied Mathematics.

[Gil98] David Gillman. A Chernoff bound for random walks on expander graphs. *SIAM Journal on Computing*, 27(4):1203–1220, January 1998. Publisher: Society for Industrial and Applied Mathematics. doi:10.1137/S0097539794268765.

[GKO+16] Sivakanth Gopi, Swastik Kopparty, Rafael Mendes de Oliveira, Noga Ron-Zewi, and Shubhangi Saraf. Locally testable and locally correctable codes approaching the Gilbert-Varshamov bound. Technical Report TR16-122, Electronic Colloquium on Computational Complexity (ECCC), August 2016. URL: https://eccc.weizmann.ac.il/report/2016/122/.

[GL89] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC '89, page 25–32, New York, NY, USA, 1989. Association for Computing Machinery. doi:10.1145/73007.73010.

[GL21] Tom Gur and Oded Lachish. On the power of relaxed local decoding algorithms. *SIAM J. Comput.*, 50(2):788–813, 2021. doi:10.1137/19M1307834.

[GM12] Anna Gal and Andrew Mills. Three-query locally decodable codes with higher correctness require exponential length. *ACM Transactions on Computation Theory*, 3(2):5:1–5:34, January 2012. doi:10.1145/2077336.2077338.

[HRW20] Brett Hemenway, Noga Ron-Zewi, and Mary Wootters. Local list recovery of high-rate tensor codes and applications. *SIAM Journal on Computing*, 49(4):FOCS17–157, January 2020. Num Pages: FOCS17-195 Publisher: Society for Industrial and Applied Mathematics. URL: https://epubs.siam.org/doi/abs/10.1137/17M116149X, doi:10.1137/17M116149X.

[HT18] Pooya Hatami and Madhur Tulsiani. Approximate local decoding of cubic Reed-Muller codes beyond the list decoding radius. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 663–679. SIAM, 2018. doi:10.1137/1.9781611975031.43.

[IJKW10] Russell Impagliazzo, Ragesh Jaiswal, Valentine Kabanets, and Avi Wigderson. Uniform direct product theorems: Simplified, optimized, and derandomized. *SIAM J. Comput.*, 39(4):1637–1665, 2010. doi:10.1137/080734030.

[IW97] Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 220–229. ACM, 1997. doi:10.1145/258533.258590.

[KLT23] Dain Kim, Anqi Li, and Jonathan Tidor. Cubic Goldreich-Levin. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 4846–4892. SIAM, 2023. doi:10.1137/1.9781611977554.ch178.

[KM23] Pravesh K. Kothari and Peter Manohar. An exponential lower bound for linear 3-query locally correctable codes. Technical Report TR23-162, Electronic Colloquium on Computational Complexity (ECCC), November 2023. URL: https://eccc.weizmann.ac.il/report/2023/162/.

[KMRS17] Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally correctable and locally testable codes with sub-polynomial query complexity. *J. ACM*, 64(2):11:1–11:42, 2017. doi:10.1145/3051093.

[KSV03] Michael Krivelevich, Benny Sudakov, and Van H. Vu. Covering codes with improved density. *IEEE Transactions on Information Theory*, 49(7):1812–1815, July 2003. doi:10.1109/TIT.2003.813490.

[KT00] Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, STOC '00, pages 80–86, New York, NY, USA, May 2000. Association for Computing Machinery. doi:10.1145/335305.335315.

[MR10] Dana Moshkovitz and Ran Raz. Two-query PCP with subconstant error. *J. ACM*, 57(5):29:1–29:29, 2010. doi:10.1145/1754399.1754402.

[Sol09] Kiril Solovey. Error reducing locally decodable codes, 2009. URL: http://tau-research-course-2009.wdfiles.com/local--files/error-reducing-locally-decodable-codes/ERLDC.pdf.

[STV01] Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the xor lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001. URL: https://www.sciencedirect.com/science/article/pii/S0022000000917306, doi:https://doi.org/10.1006/jcss.2000.1730.

[Tre03] Luca Trevisan. List-decoding using the XOR lemma. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 126–135. IEEE Computer Society, 2003. doi:10.1109/SFCS.2003.1238187.

[TW14] Madhur Tulsiani and Julia Wolf. Quadratic Goldreich-Levin theorems. *SIAM J. Comput.*, 43(2):730–766, 2014. doi:10.1137/12086827X.

[Vad12] Salil P. Vadhan. Pseudorandomness. *Found. Trends Theor. Comput. Sci.*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.

[Woo07] David P. Woodruff. New lower bounds for general locally decodable codes. Technical Report TR07-006, Electronic Colloquium on Computational Complexity (ECCC), January 2007. URL: https://eccc.weizmann.ac.il/report/2007/006/.

[Yan24] Tal Yankovitz. A stronger bound for linear 3-LCC. Technical Report TR24-036, Electronic Colloquium on Computational Complexity (ECCC), April 2024. URL: https://eccc.weizmann.ac.il/report/2024/036/.

[Yek08] Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *J. ACM*, 55(1):1:1–1:16, 2008. doi:10.1145/1326554.1326555.

A Appendix

Lemma A.1. *When q is an even constant and $\delta \in (0,1)$,*

$$\sum_{j=0}^{q/2-1} \frac{\binom{q}{j}\binom{n-q}{\delta n-j}}{\binom{n}{\delta n}} + \frac{1}{2}\frac{\binom{q}{q/2}\binom{n-q}{\delta n-q/2}}{\binom{n}{\delta n}} = \sum_{j=0}^{q/2-1} \frac{\binom{q-1}{j}\binom{n-q+1}{\delta n-j}}{\binom{n}{\delta n}}$$

Proof. We make use of the Pascal's triangle recurrence ($\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$) by manipulating and reindexing summations. For $k < 0$, let $\binom{n}{k} = 0$ for notation.

$$\begin{aligned}
\binom{q-1}{\frac{q}{2}-2}\binom{n-q}{\delta n-\frac{q}{2}} &= \sum_{j=0}^{\frac{q}{2}-2}\binom{q-1}{j}\binom{n-q}{\delta n-j-2} - \sum_{j=0}^{\frac{q}{2}-3}\binom{q-1}{j}\binom{n-q}{\delta n-j-2} \\
&= \sum_{j=0}^{\frac{q}{2}-2}\binom{q-1}{j}\binom{n-q}{\delta n-j-2} - \sum_{j=1}^{\frac{q}{2}-2}\binom{q-1}{j-1}\binom{n-q}{\delta n-j-1} \\
\binom{q-1}{\frac{q}{2}-2}\binom{n-q}{\delta n-\frac{q}{2}} &= \sum_{j=0}^{\frac{q}{2}-2}\binom{q-1}{j}\binom{n-q}{\delta n-j-2} - \sum_{j=0}^{\frac{q}{2}-2}\binom{q-1}{j-1}\binom{n-q}{\delta n-j-1}
\end{aligned}$$

$$\left(\binom{q}{\frac{q}{2}-1}-\binom{q-1}{\frac{q}{2}-1}\right)\binom{n-q}{\delta n-\frac{q}{2}}=\sum_{j=0}^{\frac{q}{2}-2}\left(\binom{q-1}{j}\left(\binom{n-q+1}{\delta n-j-1}-\binom{n-q}{\delta n-j-1}\right)\right.$$
$$\left.-\binom{q-1}{j-1}\binom{n-q}{\delta n-j-1}\right)$$

$$\binom{q-1}{\frac{q}{2}-1}\binom{n-q}{\delta n-\frac{q}{2}}=\sum_{j=0}^{\frac{q}{2}-2}\left(\binom{q-1}{j}\left(\binom{n-q}{\delta n-j-1}-\binom{n-q+1}{\delta n-j-1}\right)\right.$$
$$\left.+\binom{q-1}{j-1}\binom{n-q}{\delta n-j-1}\right)+\binom{q}{\frac{q}{2}-1}\binom{n-q}{\delta n-\frac{q}{2}}$$
$$=\sum_{j=0}^{\frac{q}{2}-1}\binom{q}{j}\binom{n-q}{\delta n-j-1}-\sum_{j=0}^{\frac{q}{2}-2}\binom{q-1}{j}\binom{n-q+1}{\delta n-j-1}$$
$$=\sum_{j=0}^{\frac{q}{2}-1}\left(\binom{q}{j}\binom{n-q}{\delta n-j-1}-\binom{q-1}{j-1}\binom{n-q+1}{\delta n-j}\right)$$
$$=\sum_{j=0}^{\frac{q}{2}-1}\left(\binom{q}{j}\binom{n-q}{\delta n-j-1}-\binom{q-1}{j-1}\binom{n-q}{\delta n-j}-\binom{q-1}{j-1}\binom{n-q}{\delta n-j-1}\right)$$
$$=\sum_{j=0}^{\frac{q}{2}-1}\left(\binom{q}{j}-\binom{q-1}{j-1}\right)\left(\binom{n-q}{\delta n-j}+\binom{n-q}{\delta n-j-1}\right)$$
$$-\sum_{j=0}^{\frac{q}{2}-1}\binom{q}{j}\binom{n-q}{\delta n-j}$$

$$\binom{q-1}{\frac{q}{2}-1}\binom{n-q}{\delta n-\frac{q}{2}}=\sum_{j=0}^{\frac{q}{2}-1}\binom{q-1}{j}\binom{n-q+1}{\delta n-j}-\sum_{j=0}^{\frac{q}{2}-1}\binom{q}{j}\binom{n-q}{\delta n-j}$$

To complete the proof, note that $\binom{q-1}{q/2-1}=\frac{1}{2}\binom{q}{q/2}$:

$$\frac{1}{2}\binom{q}{\frac{q}{2}}=\frac{1}{2}\cdot\frac{q}{q/2}\cdot\frac{(q-1)(q-2)\cdots(q/2+1)}{(q/2-1)(q/2-2)\cdots 1}=\binom{q-1}{\frac{q}{2}-1}$$

Then, move the second summation to the left hand side and divide both sides by $\binom{n}{\delta n}$. □

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