Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 107 (2014)

Locally Correctable and Testable Codes Approaching the Singleton Bound

Or Meir*

August 10, 2014

Abstract

Locally-correctable codes (LCCs) and locally-testable codes (LTCs) are codes that admit local algorithms for decoding and testing respectively. The local algorithms are randomized algorithms that make only a small number of queries to their input. LCCs and LTCs are both interesting in their own right, and have important applications in complexity theory.

It is a well-known question what are the best rate and distance that such LCCs and LTCs can achieve. When discussing LCCs and LTCs that use a constant number of queries (which is the most common setting), it is known that LCCs can not achieve a constant rate, and it is believed that the same is true for LTCs. However, it has recently been discovered that the situation is radically different when using n^{β} queries ($\beta > 0$): it turns out that there are both LCCs and LTCs that achieve *any* constant rate, while using n^{β} queries.

In this work, we observe that in fact, LCCs and LTCs with n^{β} queries can, for any rate, approach the best possible relative distance. More specifically, recall that, by the Singleton bound, an error-correcting code of rate r can have relative distance of at most $1 - r$. We construct LCCs and LTCs that, for every $r > 0$ and $\varepsilon > 0$, have rate r and relative distance $1-r-\varepsilon$, where the alphabet size is a constant that depends on ε. By applying concatenation to those codes, we obtain binary LCCs and LTCs with n^{β} queries that achieve the Zyablov bound, which constitutes the best known parameters for (explicit) binary codes.

1 Introduction

Locally-correctable codes [BFLS91, STV01, KT00] and locally-testable codes [FS95, RS96, GS06] are codes that admit local algorithms for decoding and testing respectively. More specifically:

- We say that a code C is a locally-correctable code (LCC) if there is an algorithm that, when given a string z that is close to a codeword $c \in C$, and a coordinate i, computes c_i while making only a small number of queries to z.
- We say that a code C is a locally-testable code (LTC) if there is an algorithm that, when given a string z, decides whether w is a codeword of C, or far from C, while making only a small number of queries to z.

The number of queries that are used by the latter algorithms is called the query complexity.

Besides being interesting in their own right, LCCs and LTCs have also played important roles in different areas of complexity theory, such as hardness amplification and derandomization (e.g.

*Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot 76100, Israel. Email: or.meir@weizmann.ac.il

ISSN 1433-8092

[STV01]), and probabilistically checkable proofs [AS98, ALM+98]. It is therefore a natural and well-known question to determine what are the best parameters that LCCs and LTCs can achieve.

LCCs and LTCs were originally studied in the setting where the query complexity was either constant or poly-logarithmic. In those settings, it is believed that LCCs and LTCs must be very redundant, since every bit of the codeword must contain, in some sense, information about every other bit of the codeword. Hence, we do not expect such codes to achieve high rate. In particular, in the setting of constant query complexity, it is known that linear LCCs can not have constant rate [KT00][1], and that LTCs with certain restrictions can not have constant rate [DK11, BSV12]. On the other hand, the best known LCCs have exponential length[2], and the best known LTCs have quasi-linear length [BS08, Din07, Vid13].

It turns out the picture is completely different when considering query complexity of n^β (for any constant $\beta > 0$). In this setting, it has long been known that LTCs can achieve a constant rate [PS94, BS06]. More recently, it has been discovered that both LCCs [KSY11, GKS13, HOW13] and LTCs [Vid11] can achieve rates that are arbitrarily close to 1, where the relative distance is some small constant that depends on the rate sub-optimally. This discovery was quite surprising, considering the state of affairs in the setting of constant query complexity, and the general belief that local correctability and testability require much redundancy.

In this work, we show that LCCs and LTCs with n^β queries can, for any rate, approach the best possible relative distance. This means that, surprisingly, local correctability and local testability with n^β queries do not require "paying" anything in terms of rate and relative distance. In other words, LCCs and LTCs with n^β queries match the best error-correcting codes.

More specifically, recall that, by the Singleton bound [Kom53, Sin64], an error-correcting code of rate r can have relative distance of at most[3] $1 - r$. Our two main results are the following (we refer the reader to Section 2 for the required preliminaries).

Theorem 1.1 (LCCs approaching the Singleton bound)**.** *For every $0 < r < 1$ and $\beta, \varepsilon > 0$ there exists a finite field $\mathbb{H} = \mathbb{F}_{2^p}$ such that the following holds: There exists an infinite family of ($\mathbb{F}_2$-linear) codes $\{C_k\}_k$ such that the code $C_k : \mathbb{F}_2^k \to \mathbb{H}^n$ is an LCC with message length k, rate at least r, relative distance at least $1 - r - \varepsilon$, and query complexity $O(k^\beta)$.*

Theorem 1.2 (Strong LTCs approaching the Singleton bound)**.** *For every $0 < r < 1$ and $\beta, \varepsilon > 0$ there exists a finite field $\mathbb{H} = \mathbb{F}_{2^p}$ such that the following holds: There exists an infinite family of ($\mathbb{F}_2$-linear) codes $\{C_k\}_k$ such that the code $C_k : \mathbb{F}_2^k \to \mathbb{H}^n$ is a strong LTC with message length k, rate at least r, relative distance at least $1 - r - \varepsilon$, and query complexity $O(k^\beta)$.*

By concatenating the codes of Theorems 1.1 and 1.2 with Gilbert-Varshamov codes [Gil52, Var57] of constant length, we immediately obtain binary LCCs and the LTCs that approach the Zyablov bound, which represents the best known explicit construction of binary codes. More specifically, the Zyablov bound [Zya71] provides, for every $\varepsilon > 0$ and any rate $0 < r < 1$, binary codes with rate r, and relative distance

$$Z_\varepsilon(r) \stackrel{\text{def}}{=} \max_{r<R<1} \left\{ (1 - R - \varepsilon) \cdot H^{-1}(1 - \frac{r}{R}) \right\}$$

where H^{-1} is the inverse of the binary entropy function in the domain $(0, \frac{1}{2})$. We have the following results.

[1][KT00] prove a lower bound for the related notion of LDCs (locally decodable codes). Since every linear LCC is also an LDC, their lower bound applies to linear LCCs.

[2]For example, a constant-degree Reed-Muller code is such an LCC.

[3]In fact, the relative distance may be slightly larger, i.e., $1 - r + \frac{1}{n}$ where n is the block length of the code.

Corollary 1.3 (LCCs approaching the Zyablov bound)**.** *For every $\beta, r, \varepsilon > 0$ there exists an infinite family of linear binary codes $\{C_k\}_k$, such that the code C_k is an LCC with message length k, rate at least r, relative distance at least $Z_\varepsilon(r)$, and query complexity $O(k^\beta)$.*

Corollary 1.4 (Strong LTCs approaching the Zyablov bound)**.** *For every $\beta, r, \varepsilon > 0$ there exists an infinite family of linear binary codes $\{C_k\}_k$, such that the code C_k is a strong LTC with message length k, rate at least r, relative distance at least $Z_\varepsilon(r)$, and query complexity $O(k^\beta)$.*

To sum-up, we construct LCCs and LTCs that achieve almost optimal trade-off between rate and relative distance over non-binary alphabets of constant size, and binary LCCs and LTCs that achieve the best known trade-off for binary codes.

Previous work. Our constructions are based on a technique of [AL96], who constructed codes that approach the Singleton bound and can be encoded and decoded from erasures in linear time.

Their construction has two stages: In the first stage, they construct "weak" codes with linear-time encoding and decoding. In the second stage, they use the latter codes to construct codes that approach the Singleton bound. The basic idea that underlies their second stage is the following: the codeword of the "weak" code is broken into blocks of constant size, and each of those blocks is encoded by Reed-Solomon. The decoding procedure works by decoding each of the blocks separately, which can corrects the majority of the erasures. Then, the decoding procedure of the "weak" code is applied to correct the remaining few erasures. Note that both steps can be done in linear time. We mention that the foregoing description is not complete, and that an additional modification needs to be made to make this idea work.

Before proceeding, we mention that the technique of [AL96] was used by [GI05] to construct similar codes that can be decoded from *errors* (rather than *erasures*). [GI05] followed the above scheme of [AL96], and their key step was to replace the codes of the first stage with codes that can be decoded from errors (but otherwise have the same proeprties). After this replacement was done, the second stage was more or less the same. A similar idea was used by [GI02, GR08] to construct capacity-achieving list-decodable codes with constant alphabet, where again the main step was replacing the codes of the first stage with appropriate codes.

Our observation. The key observation of this paper is that the same scheme works as well for LCCs and LTCs. Specifically, we replace the codes of the first stage of [AL96] with the LCCs of [KSY11] (for Theorem 1.1) or with the LTCs of [Vid11] (for Theorem 1.2). Then, we observe that the second stage of [AL96] works just as before. The reason is that decoding the Reed-Solomon blocks can be done locally, since those blocks are of constant size.

More generally, we wish to draw attention to the technique of [AL96]. We believe that it should be viewed as a general scheme for constructing codes that approach the Singleton bound. A "dream version" of such a scheme could be stated as follows.

Claim 1.5 ("Dream version")**.** *For any property $\mathcal{P}$ of codes the following holds. Suppose that for every constant $r_W < 1$, we can construct a "weak" code W such that:*

1. *W has the property $\mathcal{P}$.*
2. *W has rate r_W.*
3. *W has constant relative distance $\delta_W > 0$ which depends on r_W in an arbitrary way.*

Then, we can construct a code C that has the property $\mathcal{P}$ and approaches the Singleton bound (in the sense of Theorems 1.1 and 1.2).

Unfortunately, the technique of [AL96] does not imply such a clean and powerful claim. For example, such a claim obviously does not hold for the propery $\mathcal{P}$ of "not approaching the Singleton bound". Worse yet, we do not have any characterization of the properties $\mathcal{P}$ for which the scheme of [AL96] works. Nevertheless, the scheme of [AL96] does seem to work for many properties $\mathcal{P}$, and we believe that this is a good "take-home message" from this work.

Organization of this paper. We review the required preliminaries in Section 2, construct our LCCs in Section 3, and construct our LTCs in 4.

2 Preliminaries

All logarithms in this paper are in base 2. For any $n \in \mathbb{N}$ we denote $[n] \stackrel{\text{def}}{=} \{1 \ldots, n\}$. For any finite alphabet Σ and any two strings $x, y \in \Sigma^n$, the relative Hamming distance (or, simply, relative distance) between x and y is the fraction of coordinates on which x and y differ, and is denoted by $\operatorname{dist}(x, y) \stackrel{\text{def}}{=} |\{x_i \neq y_i : i \in [n]\}| / n$.

We denote by $\mathbb{F}_{2^p}$ the finite field of 2^p elements. If $\mathbb{F}$ and $\mathbb{H}$ are finite fields such that $\mathbb{H}$ is an extension of $\mathbb{F}$, then we say that a function $f : \mathbb{F}^k \to \mathbb{H}^n$ is $\mathbb{F}$-linear if for every $\alpha, \beta \in \mathbb{F}$ and $x, y \in \mathbb{F}^k$ it holds that

$$f(\alpha \cdot x + \beta \cdot y) = \alpha \cdot f(x) + \beta \cdot f(y).$$

2.1 Error correcting codes

Let $\mathbb{F}$ and $\mathbb{H}$ be finite fields such that $\mathbb{H}$ is an extension field of $\mathbb{F}$. We say that $C : \mathbb{F}^k \to \mathbb{H}^n$ is an $\mathbb{F}$-linear code C with message length k and block length n if it is an injective $\mathbb{F}$-linear function. If $\mathbb{F} = \mathbb{H}$, we say that C is a linear code over $\mathbb{F}$, and if $\mathbb{F} = \mathbb{F}_2$ then we say that C is a binary linear code. The rate r_C of the code C is the ratio $\frac{k \cdot \log|\mathbb{F}|}{n \cdot \log|\mathbb{H}|}$.

We will sometimes identify C with its image $C(\mathbb{F}^k)$. Specifically, we will write $c \in C$ to indicate the fact that there exists $x \in \mathbb{F}^k$ such that $c = C(x)$. In such case, we also say that c is a codeword of C. We say that C has relative distance δ if for every two distinct codewords $c_1, c_2 \in C$ it holds that $\operatorname{dist}(c_1, c_2) \geq \delta$. We will use the notation $\operatorname{dist}(w, C)$ to denote the relative distance of a string $w \in \mathbb{H}^n$ from C, and say that w is ε-close (respectively, ε-far) from C if $\operatorname{dist}(w, C) < \varepsilon$ (respectively, if $\operatorname{dist}(w, C) \geq \varepsilon$).

Reed-Solomon codes. We use the following fact, which states the existence of Reed-Solom codes and their relevant properties.

Fact 2.1 (Reed-Solomon Codes [RS60]). *For every $k, n \in \mathbb{N}$ such that $n \geq k$, and a finite field $\mathbb{F}$ such that $|\mathbb{F}| \geq n$, there exists a linear code $RS_{k,n}$ over $\mathbb{F}$ with message length k, block length n, rate $r = k/n$, and relative distance $1 - \frac{k-1}{n} > 1 - r$.*

We mention that Reed-Solomon codes meet the Singleton bound.

Infinite families of codes. An infinite family of codes $C = \{C_k\}$ is a sequence of codes such that the code C_k has message length k. The block length $n(k)$, rate $R(k)$ and relative distance $\delta(k)$ of such a family are functions of k such that C_k has block length $n(k)$, rate $R(k)$ and relative distance $\delta(k)$. We say that the family has constant rate (resp., constant relative distance) if $R(k)$ (resp., $\delta(k)$) is a constant that is independent of k.

Throughout this paper we will often work with infinite families of codes, and refer to them simply as "codes". For example, we will say that a code C has constant rate, and mean that the family C has constant rate. We will say that a string c is a codeword of C if it is a codeword of one of the codes C_k, and will say that a string $w \in \mathbb{H}^n$ is ε-close to C if it is ε-close to one of the codes C_k.

2.2 Locally-correctable codes

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

Definition 2.2. Let $C = \{C_k\}_k$ be an infinite family of codes with block length $n = n(k)$ and relative distance $\delta = \delta(k)$, and whose codewords belong to $\mathbb{H}^{n(k)}$ for some finite field $\mathbb{H}$. Let $q : \mathbb{N} \to \mathbb{N}$. We say that C is locally correctable with query complexity $q(k)$ if there exists a randomized algorithm A that satisfies the following requirements:

- **Input:** A takes as input a message length $k \in \mathbb{N}$, and a coordinate $i \in [n]$ for $n = n(k)$, and also gets oracle access to a string $z \in \mathbb{H}^n$ that is $\frac{\delta(k)}{2}$-close to a codeword $c \in C_k$.
- **Output:** A outputs c_i with probability at least $\frac{2}{3}$.
- **Query complexity:** A makes at most $q(k)$ queries to the oracle z.

We say that the algorithm A is a local corrector of C.

Remark 2.3. The common definition of LCCs includes an additional parameter $\tau \leq \frac{\delta}{2}$, and only requires that A works when given a string u that is τ-close to C. This definition can be "simulated" by our definition, by pretending that C has smaller relative distance than it actually has.

2.3 Locally-testable codes

Intuitively, a code is said to be locally testable [FS95, RS96, GS00] if, given a string $z \in \mathbb{H}^n$, it is possible to determine whether z is a codeword of C, or rather far from C, by reading only a small part of z. There are two variants of LTCs in the literature, "weak" LTCs and "strong" LTCs. Below, we only give the definition of strong LTCs, since it is simpler and allows us to state a stronger result.

Definition 2.4. Let $C = \{C_k\}_k$ be an infinite family of codes with block length $n = n(k)$, whose codewords belong to $\mathbb{H}^{n(k)}$ for some finite field $\mathbb{H}$. Let $q : \mathbb{N} \to \mathbb{N}$. We say that C is (strongly) locally testable with query complexity $q(k)$ if there exists a randomized algorithm A that satisfies the following requirements:

- **Input:** A takes as input a message length $k \in \mathbb{N}$, and also gets oracle access to a string $z \in \mathbb{H}^{n(k)}$.
- **Completeness:** If z is a codeword of C_k, then A accepts with probability 1.
- **Soundness:** A rejects with probability at least $\mathrm{dist}(z, C_k)/2$.
- **Query complexity:** A makes at most $q(k)$ non-adaptive queries to the oracle z.

We say that the algorithm A is a local tester of C.

Remark 2.5. The common definition of strong LTCs also includes an additional parameter $\rho < 1$, and requires that A rejects with probability $\rho \cdot \mathrm{dist}(u, C_k)$. For simplicity, we chose to fix ρ to $\frac{1}{2}$. If $\rho < \frac{1}{2}$, one can amplify it to $\frac{1}{2}$ by executing A multiple times.

2.4 Expander graphs

Expander graphs are graphs with certain pseudorandom connectivity properties. Below, we state the construction and properties that we need. The reader is referred to [HLW06] for a survey. For a graph G, a vertex s and a set of vertices T, let $E(s,T)$ denote the set of edges that go from s into T.

Definition 2.6. Let $G = (U \cup V, E)$ be a bipartite d-regular graph with $|U| = |V| = n$. We say that G is an (α, γ)-sampler if the following holds for every $T \subseteq V$: for at least $(1 - \alpha)$ fraction of the vertices $s \in U$ it holds that it holds that

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

Theorem 2.7. *For every $\alpha, \gamma > 0$ there exists a constant $d \in \mathbb{N}$ and an infinite family of graphs $\{G_n\}_{n \in \mathbb{N}}$ such that the following holds for each graph $G = G_n$:*

- *$G = (U \cup V, E)$ is a bipartite d-regular graph with $|U| = |V| = n$.*
- *G is an (α, γ)-sampler.*

Furthermore, there exists an algorithm that on input n outputs G_n in time $\mathrm{poly}(n)$.

Proof sketch. We give a brief sketch of the proof, which uses the notions of edge expansion and second eigenvalue. We do not define these notions because they will not be used in the rest of the paper. The interested reader is referred to [HLW06]. In addition, we made no effort to optimize the dependence of d on α and γ.

First, observe that it suffices to prove that there exists a family $\{G'_n\}_{n \in \mathbb{N}}$ of graphs such that G'_n is a *non-bipartite* d-regular graph over n vertices, which has the required sampling property. The reason is that each such graph G'_n can be converted into a bipartite graph G_n with the sampling property, by taking two copies of the vertex set of G'_n and connecting the two copies according to the edges in G'_n.

We start constructing the family $\{G'_n\}_{n \in \mathbb{N}}$ by constructing a family $\{G''_n\}_{n \in \mathbb{N}}$ of graphs which has a constant edge expansion. As noted by [Din07], this can be done as follows: Known constructions of expanders (e.g. [RVW00]) give a graph G''_n with constant edge expansion for every n that is a power of 2. In order to deal with values of n of the form $2^m - k$, we construct an expander over 2^m vertices and merge k pairs of vertices. Then, we maintain the regularity by adding self-loops. The resulting graph has $2^m - k$ vertices, and maintains the edge expansion of the original graph over 2^m vertices.

Next, we note that by the Cheeger inequality for expanders [Dod84, AM85], the family $\{G''_n\}_{n \in \mathbb{N}}$ has second-largest normalized eigenvalue (in absolute value) that is bounded away from 1. This gives the sampling property for some fixed choices of α and γ by the expander mixing lemma [AC88].

Finally, in order to improve α and γ to the required level, we raise the graphs in $\{G''_n\}_{n \in \mathbb{N}}$ to some constant power. We define the family $\{G'_n\}_{n \in \mathbb{N}}$ to be the family of the resulting graphs. ■

3 LCCs approaching the Singleton bound

In this section, we prove Theorem 1.1, restated next.

Theorem 1.1. *For every $0 < r < 1$ and $\beta, \varepsilon > 0$ there exists a finite field $\mathbb{H} = \mathbb{F}_{2^p}$ such that the following holds: There exists an infinite family of ($\mathbb{F}_2$-linear) codes $\{C_k\}_k$ such that the code $C_k : \mathbb{F}_2^k \to \mathbb{H}^n$ is an LCC with message length k, rate at least r, relative distance at least $1 - r - \varepsilon$, and query complexity $O(k^\beta)$.*

To this end, we use the following construction of LCCs with high rate and k^β queries, which follows from Theorem 4 of [KSY11]:

Theorem 3.1 ([KSY11]). *For every $0 < \beta, r_W < 1$ there exists $\delta_W > 0$ such that the following holds. Let $\mathbb{F}$ be a finite field. There exists an infinite family of codes $\{W_k\}_k$ over $\mathbb{F}$ such that W_k has message length k, rate at least r_W, relative distance at least δ_W, and locally correctable with with query complexity $O(k^\beta)$.*

The rest of this section is organized as follows. In Section 3.1, we give an overview of the proof of Theorem 1.1. Then, in Section 3.2, we provide a rigorous construction of the codes of Theorem 1.1 and Corollary 1.3. Finally, in Section 3.3, we prove that those codes are locally correctable.

3.1 Overview

We start with an overview of the construction of the code $C = C_k$. The following construction is due to [AL96]. Fix a rate r and a constant $\varepsilon > 0$. Our goal is to construct a code C with rate r and relative distance $1 - r - \varepsilon$.

Our construction will use two basic ingredients:

- The LCC $W = W_k$ from Theorem 3.1 of rate $\frac{1}{1+\varepsilon/2}$ and relative distance δ_W, where δ_W is some small constant.
- A Reed-Somon code $RS = RS_{b,d}$ from Fact 2.1, with rate $r \cdot (1 + \frac{\varepsilon}{2})$ and relative distance at least $1 - r - \frac{\varepsilon}{2}$.

The idea of the construction is to combine the LCC W and the Reed-Solomon code RS to obtain a code C that enjoys "the best of the all worlds": both the local correctability of W and the good parameters of RS. We do it in two steps: first, we construct a code C' which can be corrected from $\frac{1-r-\varepsilon}{2}$ fraction of *random* errors. Then, we augment C' to obtain a code C that can be corrected from $\frac{1-r-\varepsilon}{2}$ fraction of *adversarial* errors, and hence has relative distance $1 - r - \varepsilon$.

We first describe the construction of C'. The code C' encodes a message x as follows: the code C' first encodes x via W, thus obtaining a codeword $w \in W$. Then, C' partitions w into blocks of constant length b (to be determined later), and encodes each block with the code RS. The resulting string c' is defined to be the encoding of x via C'.

It is easy to see that C' has rate r, as required. We claim that if one applies to a codeword $c' \in C'$ a noise that corrupts each coordinate with probability $\frac{1-r-\varepsilon}{2}$, then the codeword c' can be recovered from its corrupted version with high probability. To see it, first observe that with high probability, almost all the blocks of c' have at most $\frac{1-r-\varepsilon/2}{2}$ fraction of corrupted coordinates. Let us call those blocks "good blocks", and observe that the good blocks can be corrected by decoding them to the nearest codeword of RS. Next, observe that if b is a sufficiently large constant, the fraction of "good blocks" is at least $1 - \delta_W/2$, and hence we can correct the remaining $\delta_W/2$ fraction of errors using the decoding algorithm of W. It follows that C' can be corrected from $\frac{1-r-\varepsilon}{2}$ fraction of random errors.

Next, we show how to augment C' to obtain a code C that is correctable from adversarial errors. This requires two additional ideas. The first idea to apply a pseudorandom permutation to the coordinates of C'. The pseudorandom permutation is determined by the edges of an expander graph (see Section 2.4). This step is motivated by the hope that, after the adversary decided which coordinates to corrupt, applying the permutation to the coordinates will make the errors behave pseudorandomly. This will allow the above analysis for the case of random erros to go through.

Of course, on its own, this idea, since the adversary can take in the permutation into account when it chooses where to place the errors. Here the second idea comes into play: after applying the permutation to the coordinates of C', we will increase the alphabet size of the code, packing each block of symbols into a new big symbol. The motivation for this step is that increasing the alphabet size restricts the freedom of the adversary in choosing the pattern of errors. Indeed, we will show that after the alphabet size is increased, applying permutation to the coordinates of the code makes the errors behave pseudorandomly. This allows us to prove that the code can be decoded from $\frac{1-r-\varepsilon}{2}$ errors, as we wanted.

3.2 Construction

Let $\beta, r, \varepsilon > 0$ be as in Theorem 1.1. Let $r_W \stackrel{\text{def}}{=} \frac{1}{1+\varepsilon/2}$, and let δ_W be the constant relative distance guaranteed by Theorem 3.1. Let $d \in \mathbb{N}$ be a constant that is sufficiently large such that

- There exists a family of d-regular $\left(\frac{1}{2} \cdot \delta_W, \frac{1}{4} \cdot \varepsilon\right)$-samplers $\{G_n\}_n$, as in Theorem 2.7.
- There exists a Reed-Solomon code $RS = RS_{b,d}$ of rate at least $r \cdot (1 + \varepsilon/2)$, relative distance at least $1 - r - \frac{1}{2} \cdot \varepsilon$, and block length d.

We choose the latter code $RS_{b,d}$ to be over a finite field $\mathbb{F}$ that is an extension field of $\mathbb{F}_2$. Let $\mathbb{H}$ be an extension of $\mathbb{F}$ of dimension d, so $|\mathbb{H}| = |\mathbb{F}|^d$. We show how to construct a family of $\mathbb{F}$-linear locally-correctable codes $\left\{C_k : \mathbb{F}^k \to \mathbb{H}^{n(k)}\right\}_k$.

Remark 3.2. Recall that Theorem 1.1 requires us to construct codes C whose messages are taken from $\mathbb{F}_2^k$ rather than $\mathbb{F}^k$. However, every message in $\mathbb{F}^k$ can be interpreted as a message in $\mathbb{F}^{k'}$ (for some $k' = O(k)$), by "unpacking" each symbol in $\mathbb{F}$ into bits. Note that when the codes are reinterpreted in this way, they are $\mathbb{F}_2$-linear.

Fix a message length $k \in \mathbb{N}$. We explain how to construct a code $C = C_k$ in the family. We use the following ingredients:

- The Reed-Solomon code $RS_{b,d}$ from above, over the field $\mathbb{F}$, with message length b and block length d.
- A code $W = W_k$ from Theorem 3.1, where $W : \mathbb{F}^k \to \mathbb{F}^{n_W}$ is a code over $\mathbb{F}$ with rate r_W and relative distance δ_W, and is locally correctable with query complexity $O(k^\beta)$. For simplicity, we assume that b divides n_W, so $n_W = n \cdot b$ for some $n \in \mathbb{N}$.
- A d-regular $\left(\frac{1}{2} \cdot \delta_W, \frac{1}{4} \cdot \varepsilon\right)$-sampler $G = G_n = (U \cup V, E)$ with $|U| = |V| = n$, as in Theorem 2.7 above.

The code C encodes a message $x \in \mathbb{F}^k$ as follows:

- C first encodes x via the code W. This step yields a codeword $w \in W$ of length n_W.
- Next, C partitions the string w into n blocks of length b, and encodes each block via the code $RS_{b,d}$. Let us denote the resulting string by $c' \in \mathbb{F}^{n \cdot d}$ and the resulting codewords of $RS_{b,d}$ by $B_1, \ldots, B_n \in \mathbb{F}^d$.
- Now, C applies a "pseudorandom" permutation to the coordinates of c' as follows: Let $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$ be the left and right vertices of G respectively. For each $i \in [n]$ and $j \in [d]$, we write the j-th symbol of B_i on the j-th edge of u_i. Then, we construct new blocks $S_1, \ldots, S_n \in \mathbb{F}^d$, by setting the j-th symbol of B'_i to be the symbol written on the j-th edge of v_i.

- Finally, we define a string $c \in \mathbb{H}^n$ as follows: the i-th coordinate c_i is the block $S_i \in \mathbb{F}^d$, reinterpreted as a symbol of $\mathbb{H}$. We choose c to be the encoding of x via the code C.

This concludes the definition of the code C. It is not hard to see that C is $\mathbb{F}$-linear. The rate of C is

$$\begin{aligned}\frac{k \cdot \log |\mathbb{F}|}{n \cdot \log |\mathbb{H}|} &= \frac{k \cdot \log |\mathbb{F}|}{n \cdot d \cdot \log |\mathbb{F}|} \\ &= \frac{k}{n \cdot d} \\ &= \frac{k}{n \cdot b} \cdot \frac{b}{d} \\ &= \frac{k}{n_W} \cdot \frac{b}{d} \\ &= r_W \cdot r \cdot \left(1 + \frac{\varepsilon}{3}\right) \\ &= r,\end{aligned}$$

where the last equality follows from the definition of r_W. In the next section, we prove that C is locally correctable, and that the local correction algorithm can correct up to $\frac{1-r-\varepsilon}{2}$ fraction of errors. This will imply that C has the required relative distance $1 - r - \varepsilon$.

3.3 Local correctability

In this section, we complete the proof of Theorem 1.1, by proving that the family $\{C_k\}_k$ is locally correctable from $\frac{1-r-\varepsilon}{2}$ fraction of errors using $O(k^\beta)$ queries. To this end, we describe a local corrector A. The algortihm A is based on the following algorithm A_0, which locally decodes coordinates of W_k from a corrupted codeword of C_k.

Lemma 3.3. *There exists an algorithm A_0 that satisfies the following requirements:*

- ***Input:*** *A_0 takes as input a message length $k \in \mathbb{N}$, a coordinate $i \in [n_W]$, and also gets oracle access to a string $z \in \mathbb{H}^n$ that is $\left(\frac{1-r-\varepsilon}{2}\right)$-close to a codeword $c \in C_k$.*
- ***Output:*** *Let w^c be the codeword of W_k from which c was generated. Then, A_0 outputs w^c_i with probability at least $1 - \frac{1}{3 \cdot b \cdot d}$.*
- ***Query complexity:*** *A makes at most $O(k^\beta)$ queries to the oracle z.*

Before proving Lemma 3.3, we show how to construct the algorithm A given the algorithm A_0. Suppose that the algorithm A is given oracle access to a string z that is $\left(\frac{1-r-\varepsilon}{2}\right)$-close to a codeword $c \in C_k$, and a coordinate $i \in [n]$. The algorithm is required to decode c_i. Let $w^c \in \mathbb{F}^{n_W}$ be the codeword of W from which c was generated, and let $B^c_1, \ldots, B^c_n$ and $S^c_1, \ldots, S^c_n$ be the corresponding blocks.

In order to decode c_i, the algorithm A should decode each of the symbols in the block $S^c_i \in \mathbb{F}^d$. Let $B^c_{j_1}, \ldots, B^c_{j_d}$ be the neighbors of S^c_i in the graph G_n. Each symbol of the block S^c_i belongs to one of the blocks $B^c_{j_1}, \ldots, B^c_{j_d}$, and therefore it suffices to retrieve the latter blocks. Now, each block $B^c_{j_h}$ is the encoding via $RS_{b,d}$ of b symbols of w^c. The algorithm A invokes the algorithm A_0 to decode each of those b symbols of w^c, for each of the blocks $B^c_{j_1}, \ldots, B^c_{j_d}$. By the union bound, the algorithm A_0 decodes all those $b \cdot d$ symbols of w^c correctly with probability at least $1 - \frac{1}{3 \cdot b \cdot d} \cdot b \cdot d = \frac{2}{3}$. Whenever that happens, the algorithm A retrieves the blocks $B^c_{j_1}, \ldots, B^c_{j_d}$, and therefore computes the block S_i^c correctly. This concludes the construction of the algorithm A. Note that the query complexity of A is larger than that of A_0 by a factor of at most $b \cdot d$, and hence it is at most $O(k^\beta)$. It remains to prove Lemma 3.3.

Proof of Lemma 3.3. Let A_W be the local corrector of the code W. By amplificiation, we may assume that A_W errs with probability at most $\frac{1}{3 \cdot b \cdot d}$. Suppose that the algorithm A is invoked on a string $z \in \mathbb{H}^n$ and a coordinate $i \in [n_W]$. The algorithm A_0 invokes the algorithm A_W to retrieve the coordinate i, and emulates A_W in the natural way: Recall that A_W expects to be given access to a corrupted codeword of W, and makes queries to it. Whenever A_W makes a query to a coordinate $q \in [n_W]$, the algorithm A_0 performs the follows steps.

1. A_0 finds the block B_l to which the coordinate q belongs. Formally, $l \stackrel{\text{def}}{=} \lceil q/b \rceil$.
2. A_0 finds the neighbors of the vertex u_l in G. Let us denote those vertices by $v_{j_1}, \dots, v_{j_d}$.
3. A_0 queries the coordinates $j_1, \dots j_d$, thus obtaining the blocks $S_{j_1}, \dots, S_{j_d}$.
4. A_0 reconstructs the block B_l by reversing the permutation of G on $S_{j_1}, \dots, S_{j_d}$.
5. A_0 decodes B_l to the nearest codeword of $RS_{b,d}$.
6. A_0 retrieves the value of the q-th coordinate of w from the latter codeword of $RS_{b,d}$, and feeds it to A_W as an answer to its query.

When the algorithm A_W finishes running, the algorithm A_0 finishes and outputs the output of A_W. It is not hard to see that the query complexity of A_0 is at most d times the query complexity of A_W, and hence it is $O(k^\beta)$. It remains to show that A_0 succeeds in decoding from $\frac{1-r-\varepsilon}{2}$ fraction of errors.

Let $z \in \mathbb{H}^n$ be a string that is $\left(\frac{1-r-\varepsilon}{2}\right)$-close to a codeword $c \in C$. Let $w^c \in \mathbb{F}^{n_W}$ be the codeword of W from which c was generated, and let $B_1^c, \dots, B_n^c$ and $S_1^c, \dots, S_n^c$ be the corresponding blocks. We also use the following definitions:

1. Let $S_1^z, \dots, S_n^z \in \mathbb{F}^d$ be the blocks that correspond to the symbols of z.
2. Let $B_1^z, \dots, B_n^z$ be the blocks that are obtained from $S_1^z, \dots, S_n^z$ by reversing the permutation.
3. Let $w^z \in \mathbb{F}^{n_W}$ be the string that is obtained by decoding each block B_l^z to the nearest codeword of $RS_{b,d}$, and extracting the coordinates of w from the resulting codewords.

It is easy to see that A_0 emulates the action of A_W on w^z. Therefore, if we prove that w^z is $(\delta_W/2)$-close to w^c, we will be done. In order to do so, it suffices to prove that for at least $\left(1 - \frac{\delta_W}{2}\right)$ fraction of the blocks B_l^z, it holds that B_l^c is the codeword of $RS_{b,d}$ that is closest to B_l^z.

To this end, let $j_1, \dots, j_t$ be the coordinates on which z and c differ. In other words, for every $h \in [t]$ it holds that $S_{j_h}^z \neq S_{j_h}^c$. By assumption, $t \leq \left(\frac{1-r-\varepsilon}{2}\right) \cdot n$. Next, observe that since G_n is a $\left(\frac{1}{2} \cdot \delta_W, \frac{1}{4} \cdot \varepsilon\right)$-sampler, it holds that for at least $\left(1 - \frac{\delta_W}{2}\right)$ fraction of the vertices u_l of G, it holds that u_l has at most

$$\left(\frac{1-r-\varepsilon}{2} + \frac{\varepsilon}{4}\right) \cdot d = \frac{(1-r-\varepsilon/2)}{2} \cdot d$$

neighbors among $j_1, \dots, j_t$. Now, for each such vertex u_l, it holds that the block B_l^z is $\left(\frac{1-r-\varepsilon/2}{2}\right)$-close to the block B_l^c. Since the code $RS_{b,d}$ has relative distance $1 - r - \varepsilon/2$, this implies that B_l^c is the codeword of $RS_{b,d}$ that is closest to B_l^z, as required. ■

Obtaining the Zyablov bound. As noted in the introduction, the codes that we constructed in this section can be concatenated with binary Gilbert-Varshamov codes, thus obtaining binary codes that achieve the Zyablov bound. We do not provide the details, since such constructions are standard in coding theory. Nevertheless, there is one subtle point that deserves attention: in order to decode the concatenated codes, we combine the local decoder A described above with the GMD decoder [Jr.66]. However, in order for the GMD decoder to work, the local corrector A needs to deal with both errors and erasures. While the algorithm A described above deals only with errors, it is not hard to modify it to deal with erasures as well. We refer the reader to [GI05] for an example of a similar construction.

4 LTCs approaching the Singleton bound

In this section, we prove Theorem 1.2, restated next.

Theorem 1.2. *For every $0 < r < 1$ and $\beta, \varepsilon > 0$ there exists a finite field $\mathbb{H} = \mathbb{F}_{2^p}$ such that the following holds: There exists an infinite family of ($\mathbb{F}_2$-linear) codes $\{C_k\}_k$ such that the code $C_k : \mathbb{F}_2^k \to \mathbb{H}^n$ is a strong LTC with message length k, rate at least r, relative distance at least $1 - r - \varepsilon$, and query complexity $O(k^\beta)$.*

To this end, we use the following result, which follows from Theorem 3.1 of [Vid11].

Theorem 4.1 ([Vid11])**.** *For every $0 < \beta, r_W < 1$ there exists $\delta_W > 0$ such that the following holds. Let $\mathbb{F}$ be a finite field. There exists an infinite family of codes $\{W_k\}_k$ over $\mathbb{F}$ such that W_k has message length k, rate at least r_W, relative distance at least δ_W, and locally testable with with query complexity $O(k^\beta)$.*

Our construction of the LTCs $\{C_k\}_k$ is the same as the construction of the LCCs of Section 3, with the only difference is that we use the LTCs of Theorem 4.1 instead of the LCCs of Theorem 3.1. Our LTCs have the required rate due to the same considerations as before. In order to show that our LTCs have the required relative distance, we use the same analysis of Section 3.3 to show that the codes can be corrected from $\frac{1-r-\varepsilon}{2}$ fraction of errors, though here the correction is done by an algorithm that is not necessarily local or efficient.

It remains to show that those codes are locally testable. To this end, we describe a local tester A. In what follows, we use the notation of Section 3.2. Let A_W be the local tester of $W = W_k$.

When given oracle access to a purported codeword $z \in \mathbb{H}^n$, the local tester A emulates the action of A_W in the natural way: Recall that A_W expects to be given access to a purported codeword of W, and makes queries to it. Whenever A_W makes a query to a coordinate $q \in [n_W]$, the algorithm A performs the follows steps.

1. A finds the block B_l to which the coordinate q belongs. Formally, $l \stackrel{\text{def}}{=} \lceil q/b \rceil$.
2. A finds the neighbors of the vertex u_l in G. Let us denote those vertices by $v_{j_1}, \ldots, v_{j_d}$.
3. A queries the coordinates $j_1, \ldots j_d$, and retreives the blocks $S_{j_1}, \ldots, S_{j_d}$.
4. A reconstructs the block B_l by reversing the permutation of G on $S_{j_1}, \ldots, S_{j_d}$.
5. If B_l is not a codeword of $RS_{b,d}$, the local tester A rejects.
6. Otherwise, A retrieves the value of the q-th coordinate of w from B_l, and feeds it to A_W as an answer to its query.

If A_W finishes running, then A accepts if and only if A_W accepts. It is easy to see that the query complexity of A is at most d times the query complexity of A_W, and hence it is $O(k^\beta)$. It is also not hard to see that if z is a legal codeword of C, then A accepts with probability 1.

It remains to show that A rejects with probability $\text{dist}(z, C)/2$. To this end, it suffices to prove that A rejects with probability at least $\eta \cdot \text{dist}(z, C)$ for some constant $\eta > 0$, since η can be amplified to $\frac{1}{2}$ by repetition. We use the following definitions:

1. Let $S_1, \ldots, S_n \in \mathbb{F}^d$ be the blocks that correspond to the symbols of z.
2. Let $B_1, \ldots, B_n$ be the blocks that are obtained from $S_1, \ldots, S_n$ by reversing the permutation.,
3. Let $w^z \in (\mathbb{F} \cup \{?\})^{n_W}$ be the string that is obtained by from the blocks $B_1, \ldots, B_n$: for each block B_l that is a legal codeword of $RS_{b,d}$, we extract from B_l the corresponding coordinates of w^z in the natural way. For each block B_l that is not a legal codeword of $RS_{b,d}$, we set the corresponding coordinates of w^z to be "?".

We would like to lower bound the probability that A rejects z in terms of the probability that A_W rejects w^z. However, there is a small technical problem: A_W is defined as acting on strings in $\mathbb{F}^{n_W}$, and not on strings in $(\mathbb{F} \cup \{?\})^{n_W}$. To deal with this technicality, we define an algorithm A'_W that, when given access to a string $y \in (\mathbb{F} \cup \{?\})^{n_W}$, emulates A_W on y, but rejects whenever a query is anwered with "?". We use the following proposition, whose proof we defer to Section 4.1.

Proposition 4.2. *A'_W rejects a string $y \in (\mathbb{F} \cup \{?\})^{n_W}$ with probability at least*

$$\frac{1}{4} \cdot \min \{\text{dist}(y, W), \delta_W\}.$$

Now, it is not had to see that when A is invoked on z, it emulates the action of A'_W on w^z. To finish the proof, note that

$$\text{dist}(w^z, W) \geq \frac{1}{b \cdot d} \cdot \text{dist}(z, C),$$

since every coordinate of C is generated from at most $b \cdot d$ coordinates of W. It thus follows that A rejects z with probability at least

$$\frac{1}{4} \cdot \min \{\text{dist}(w^z, W), \delta_W\} \geq \min \left\{ \frac{1}{4 \cdot b \cdot d} \cdot \text{dist}(z, C), \frac{1}{4} \cdot \delta_W \right\}.$$

We conclude the proof by setting $\eta = \min \left\{ \frac{1}{4 \cdot b \cdot d}, \frac{1}{4} \cdot \delta_W \right\}$.

4.1 Proof of Proposition 4.2

We use the following result.

Claim 4.3. *Let $I \subseteq [n_W]$ be a set of coordinates. The algorithm A_W queries I with probability at least*

$$\frac{1}{2} \cdot \min \left\{ \frac{|I|}{n}, \frac{1}{2} \cdot \delta_W \right\}.$$

Note that this claim only makes sense since we assumed that A_W makes *non-adaptive* queries (we assumed it in Definition 2.4). Without this assumption, the probability that A_W queries I would have depended on the string that A_W gets oracle access to.

Proof. It suffices to prove that for every $I \subseteq [n_W]$ such that $\frac{|I|}{n} \leq \frac{1}{2} \cdot \delta_W$, the algorithm A_W queries I with probability at least $\frac{1}{2} \cdot \frac{|I|}{n}$. Let I be such a set, and let $s \in \mathbb{F}^{n_W}$ be an arbitrary string that contains non-zero values inside I, and contains 0 everywhere outside I. Clearly,

$$\operatorname{dist}(s, W) = \frac{|I|}{n},$$

and therefore A_W rejects s with probability at least $\frac{1}{2} \cdot \frac{|I|}{n}$. On the other hand, A_W can only reject s if it queries I, since otherwise it can not distinguish between s and the all-zeroes codeword. It follows that A_W queries I with probability at least $\frac{1}{2} \cdot \frac{|I|}{n}$, as required. ■

We turn to proving Proposition 4.2. Let

$$E \stackrel{\text{def}}{=} \{i : y_i = ?\}$$

be the set of erasures in y. We consider two cases:

- E **is "large":** Suppose that $\frac{|E|}{n} \geq \frac{1}{2} \cdot \operatorname{dist}(y, W)$. In this case, it holds by Claim 4.3 that A_W queries E with probability at least

$$\frac{1}{4} \cdot \min \{\operatorname{dist}(y, W), \delta_W\}.$$

  Since A'_W rejects y whenever A_W queries E, the proposition follows.

- E **is "small":** Suppose that $\frac{|E|}{n} \leq \frac{1}{2} \cdot \operatorname{dist}(y, W)$. Let $y_0 \in \mathbb{F}^{n_W}$ be an arbitrary string that agrees with y outside E. Clearly,

$$\operatorname{dist}(y, W) \leq \operatorname{dist}(y_0, W) + \frac{|E|}{n_W},$$

  so $\operatorname{dist}(y_0, W) \geq \frac{1}{2} \cdot \operatorname{dist}(y, W)$. Let $\mathcal{E}$ denote the event that A_W queries E. We have that

$$\begin{aligned}
\Pr\left[A'_W \text{ rejects } y\right] &= \Pr[\mathcal{E}] \cdot \Pr\left[A'_W \text{ rejects } y | \mathcal{E}\right] + \Pr\left[\neg\mathcal{E}'\right] \cdot \Pr\left[A'_W \text{ rejects } y | \neg\mathcal{E}\right] \\
&= \Pr[\mathcal{E}] \cdot 1 + \Pr[\neg\mathcal{E}] \cdot \Pr[A_W \text{ rejects } y_0 | \neg\mathcal{E}] \\
&\geq \Pr[\mathcal{E}] \cdot \Pr[A_W \text{ rejects } y_0 | \mathcal{E}] + \Pr[\neg\mathcal{E}] \cdot \Pr[A_W \text{ rejects } y_0 | \neg\mathcal{E}] \\
&= \Pr[A_W \text{ rejects } y_0] \\
&\geq \frac{1}{2} \cdot \operatorname{dist}(y_0, W) \\
&\geq \frac{1}{4} \cdot \operatorname{dist}(y, W),
\end{aligned}$$

  as required.

This concludes the proof.

Acknowledgement. I would like to thank Irit Dinur, Tali Kaufman, Swastik Kopparty and Shubanghi Saraf for useful discussions and ideas. In particular, I would like to thank Swastik and Shubanghi for encouraging me to write this result. I would also like to thank Oded Goldreich and Irit Dinur for helpful comments on an early draft of this work, which improved its presentation.

References

[AC88] Noga Alon and Fan R. K. Chung. Explicit construction of linear sized tolerant networks. *Discrete Mathematics*, 72(1-3):15–19, 1988.

[AL96] Noga Alon and Michael Luby. A linear time erasure-resilient code with nearly optimal recovery. *IEEE Transactions on Information Theory*, 42(6):1732–1736, 1996.

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and intractability of approximation problems. *Journal of ACM*, 45(3):501–555, 1998. Preliminary version in FOCS 1992.

[AM85] N. Alon and V. D. Milman. λ_1, Isoperimetric inequalities for graphs, and superconcentrators. *JOURNAL OF COMBINATORIAL THEORY, Series B*, 38(1):73–88, 1985.

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checkable proofs: A new characterization of NP. *Journal of ACM volume*, 45(1):70–122, 1998. Preliminary version in FOCS 1992.

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *STOC*, pages 21–31, 1991.

[BS06] Eli Ben-Sasson and Madhu Sudan. Robust locally testable codes and products of codes. *Random Struct. Algorithms*, 28(4):387–402, 2006. Preliminary version in APPROX-RANDOM 2004.

[BS08] Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM J. Comput.*, 38(2):551–607, 2008. Preliminary version in STOC 2005.

[BSV12] Eli Ben-Sasson and Michael Viderman. Towards lower bounds on locally testable codes via density arguments. *Computational Complexity*, 21(2):267–309, 2012.

[Din07] Irit Dinur. The PCP theorem by gap amplification. *Journal of ACM*, 54(3):241–250, 2007. Preliminary version in STOC 2006.

[DK11] Irit Dinur and Tali Kaufman. Dense locally testable codes cannot have constant rate and distance. In *APPROX-RANDOM*, pages 507–518, 2011.

[Dod84] Jozef Dodziuk. Difference equations, isoperimetric inequality and transience of certain random walks. 284(2):787–794, 1984.

[FS95] Katalin Friedl and Madhu Sudan. Some improvements to total degree tests. In *ISTCS*, pages 190–198, 1995.

[GI02] Venkatesan Guruswami and Piotr Indyk. Near-optimal linear-time codes for unique decoding and new list-decodable codes over smaller alphabets. In *STOC*, pages 812–821, 2002.

[GI05] Venkatesan Guruswami and Piotr Indyk. Linear-time encodable/decodable codes with near-optimal rate. *IEEE Transactions on Information Theory*, 51(10):3393–3400, 2005.

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

[GKS13] Alan Guo, Swastik Kopparty, and Madhu Sudan. New affine-invariant codes from lifting. In *ITCS*, pages 529–540, 2013.

[GR08] Venkatesan Guruswami and Atri Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008.

[GS00] Oded Goldreich and Shmuel Safra. A combinatorial consistency lemma with application to proving the PCP theorem. *SIAM J. Comput.*, 29(4):1132–1154, 2000.

[GS06] Oded Goldreich and Madhu Sudan. Locally testable codes and PCPs of almost linear length. *Journal of ACM*, 53(4):558–655, 2006. Preliminary version in FOCS 2002, pages 13-22.

[HLW06] Shlomo Hoory, Nati Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of AMS*, 43(4):439–561, 2006.

[HOW13] Brett Hemenway, Rafail Ostrovsky, and Mary Wootters. Local correctability of expander codes. In *ICALP (1)*, pages 540–551, 2013.

[Jr.66] G. David Forney Jr. Generalized minimum distance decoding. *IEEE Transactions on Information Theory*, 12(2):125–131, 1966.

[Kom53] Y Komamiya. Application of logical mathematics to information theory. In *Proc. 3rd Japan. Nat. Cong. Appl. Math*, 1953.

[KSY11] Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. In *STOC*, pages 167–176, 2011.

[KT00] Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *STOC*, pages 80–86, 2000.

[PS94] Alexander Polishchuk and Daniel A. Spielman. Nearly-linear size holographic proofs. In *STOC*, pages 194–203, 1994.

[RS60] Irving S. Reed and Gustave Solomon. Polynomial codes over certain finite fields. *SIAM Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

[RS96] Ronitt Rubinfeld and Madhu Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal of Computing*, 25(2):252–271, 1996.

[RVW00] Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *FOCS*, pages 3–13, 2000. Full version can be retrieved as ECCC TR01-018.

[Sin64] Richard C. Singleton. Maximum distance q -nary codes. *IEEE Transactions on Information Theory*, 10(2):116–118, 1964.

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

[Var57] R. R. Varshamov. Estimate of the number of signals in error correcting codes. *Doklady Akadamii Nauk*, 117:739–741, 1957.

[Vid11] Michael Viderman. A combination of testability and decodability by tensor products. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:87, 2011.

[Vid13] Michael Viderman. Strong ltcs with inverse poly-log rate and constant soundness. *Electronic Colloquium on Computational Complexity (ECCC)*, 20:22, 2013.

[Zya71] Victor V. Zyablov. An estimate on the complexity of constructing binary linear cascade codes. *Problems of Information Transmission*, 7(1):3–10, 1971.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de