Good Locally Testable Codes with Small Alphabet and Small Query Size

Uriya A. First[1]
University of Haifa, Israel

Stav Lazarovici
University of Haifa, Israel

Abstract

Ben-Sasson, Goldreich and Sudan [3] showed that a binary error correcting code admitting a 2-query tester cannot be good, i.e., it cannot have both linear distance and constant rate. They also showed that there are no good codes if the alphabet is a finite field $\mathbb{F}$, the code is $\mathbb{F}$-linear, and the 2-query tester is $\mathbb{F}$-linear. We show that those are essentially the only limitations on the existence of good locally testable codes (LTCs). That is, there are good 2-query LTCs on any alphabet with more than 2 letters, and good 3-query LTCs with a binary alphabet. Similarly, there are good 3-query $\mathbb{F}$-linear LTCs, and for every $\mathbb{F}$-vector space V of dimension greater than 1, there are good 2-query LTCs with alphabet V whose tester is $\mathbb{F}$-linear. This completely solves, for every $q \geq 2$ and alphabet (resp. $\mathbb{F}$-vector space) Σ, the question of whether there is a good q-query LTC (resp. $\mathbb{F}$-LTC) with alphabet Σ. Our proof builds on the recent good 2-query $\mathbb{F}$-LTCs of the first author and Kaufman [11], by establishing a general method for reducing the alphabet size of a good low-query LTC.

2012 ACM Subject Classification Mathematics of computing → Coding theory; Theory of computation → Computational complexity and cryptography

Keywords and phrases error correcting code, locally testable code, property testing

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.70

Category RANDOM

Related Version *Extended Version*: `https://arxiv.org/abs/2512.16082` [10]

Funding This research was supported by ISF grant no. 721/24.

Acknowledgements We are grateful to the anonymous referees for many useful suggestions, and to Esty Kelman for finding a mistake in an earlier version of Section 4.

1 Introduction

Throughout, Σ is a (finite) alphabet and $\mathbb{F}$ is a finite field. We let $C \subseteq \Sigma^n$ be an error correcting code, which we think of as ranging in an infinite family of codes with block length tending to infinity. The normalized Hamming distance in Σ^n is denoted $\delta(\cdot,\cdot)$. We recall other relevant coding theory terminology in Section 2.

Informally, the code $C \subseteq \Sigma^n$ is said to be a locally testable code (LTC) if it admits a randomized algorithm – called a *tester* – that can estimate with high probability whether a word $w \in \Sigma^n$ is far from C or not by reading only a constant number of letters from w. Formally, we require the tester for C to read at most q letters from w, accept all words in C, and reject every $w \in \Sigma^n - C$ with probability at least $\mu \cdot \delta(w, C)$ for some $\mu > 0$. Here, q and μ are independent of n and w. When we wish to specify the implicit constants q and μ, we will say that $C \subseteq \Sigma^n$ is a q-query LTC with soundness μ. Also, if not indicated otherwise, we assume that any tester is *non-adaptive*, i.e., it decides which positions to read from the input word before reading them.

[1] Corresponding author

The notion of an LTC arose in the 1990s from the many works on the developing theories of *property testing* and *probabilistically checkable proofs* (PCPs). It first appeared in print in [13, Dfn. 9] in a more lax version, in which the requirement that T rejects every $w \in \Sigma^n$ with probability at least $\mu \cdot \delta(w, C)$ was imposed only for words that are $\Omega(1)$-far from C. Such a codes are known as *weak LTCs*. The definition of LTCs which we use here, known as *strong LTCs*, originates from [15, Dfn. 2.2], which intitiated the systematic study of LTCs as objects of independent interest. See [14] for a more detailed history of how LTCs emerged.

Since their introduction, it was not clear whether there are LTCs that are also *good codes*, i.e., LTCs whose relative distance and rate is bounded away from 0; see [14, Open Problem 2], for example. Indeed, Ben-Sasson, Goldreich and Sudan [3] showed that there are no good 2-query LTCs on a binary alphabet, and no $\mathbb{F}$-linear 2-query LTCs. (Here, an LTC is said to be $\mathbb{F}$-linear if its alphabet is $\mathbb{F}$ and its tester checks $\mathbb{F}$-linear constraints.) Further limitations on 2-query LTCs and 3-query LTCs appeared in [6, 18]. See also [4]. Roughly at the same time, a series of works including [23, 8, 5, 25, 19, 16] established the existence of "almost" good LTCs (both weak and strong), e.g., LTCs with rate $\mathrm{poly}(\log n)^{-1}$ and constant relative distance, or good codes admitting a tester reading $(\log n)^{O(\log \log n)}$ letters. The existence of good LTCs was finally settled by Dinur–Evra–Livne–Lubotzky–Mozes [9] and Panteleev–Kalachev [22] (independently), who showed that good $\mathbb{F}$-linear LTCs (with large query size) do exist. Shortly after, the first author and Kaufman [11] (see also [12]) showed that there are also good 2-query LTCs (with a large alphabet). The latter codes are $\mathbb{F}$-LTCs, meaning that their alphabet is an $\mathbb{F}$-vector space, and their tester checks $\mathbb{F}$-linear constraints.

Main Results

In this work, we completely resolve the question of what are the integers $q \geq 2$ and alphabets Σ for which there exists a good q-query LTC (resp. $\mathbb{F}$-LTC) with alphabet Σ.

▶ **Theorem 1.** *Let $q \geq 2$ be an integer and let Σ be an alphabet.*

(i) *There exists a good q-query LTC with alphabet Σ if and only if $(q, |\Sigma|) \neq (2, 2)$.*

(ii) *Suppose Σ is an $\mathbb{F}$-vector space. Then there exists a good q-query $\mathbb{F}$-LTC with alphabet Σ if and only if $(q, \dim \Sigma) \neq (2, 1)$.*

Note that the non-existence statements of Theorem 1 were established by Ben-Sasson, Goldreich and Sudan [3]. Theorem 1 therefore says that the restrictions on the existence of good LTCs (resp. $\mathbb{F}$-LTCs) from [3] are actually the only restrictions.

We prove the existence part of Theorem 1 by introducing an alphabet reduction method for LTCs and applying it to the good 2-query $\mathbb{F}$-LTCs of [12, Thms. 9.7, 9.8] (see also [11, Cor. 7.2] where the construction is given for $\mathbb{F} = \mathbb{F}_2$). This is the content of the following two theorems, which together with [12, Thms. 9.7, 9.8] and [3] imply Theorem 1.

▶ **Theorem 2.** *If there exists a good 2-query LTC with a possibly adaptive tester on some alphabet Σ, then there exist*

(i) *a good 2-query LTC with alphabet Δ for every Δ having more than 2 letters, and*

(ii) *a good 3-query LTC with a binary alphabet.*

▶ **Theorem 3.** *If there exists a good 2-query $\mathbb{F}$-LTC, then there exist*

(i) *a good 2-query $\mathbb{F}$-LTC with alphabet Δ for every $\mathbb{F}$-vector space Δ with $\dim \Delta \geq 2$, and*

(ii) *a good 3-query $\mathbb{F}$-linear LTC.*

In both Theorem 2 and Theorem 3, the relative distance, rate and soundness of the LTCs promised by the theorem are at least proportional to the corresponding parameters of the given LTC. In Theorem 3, all the parameters are diminished by a factor that is at most polynomial in $|\Sigma|$, while in Theorem 2, that factor is at most exponential in $|\Sigma|$; see Theorems 28 and 29. However, if the LTC given in Theorem 2 is an $\mathbb{F}_2$-LTC, then the relative distance, rate and soundness are diminished only by a factor polynomial in $|\Sigma|$; see Theorem 30. If the randomness complexity of the given LTC's tester is $r(n)$ (a function of the block length n), then the randomness complexity of the promised LTC's tester is $\max\{r(n), \log_2 n\} + O(1)$; see Remark 32.

Analogous statements hold for q-query LTCs. Again, see Theorems 28 and 29.

We hope that our alphabet reduction results will find other uses.

▶ Remark 4. Some methods for reducing the number of queries of a good LTC are also known. Specifically, by [12, Thm. 3.11, Prop. 3.6], if there exists a good q-query LTC satisfying some assumptions (which are satisfied by the good LTCs of [9]) on alphabet Σ, then there exists a good 2-query LTC with alphabet $\Sigma' \subseteq \Sigma^q$. A different construction of this kind, which also needs assumptions on the LTC, appears in [25, Cor. 3.4]. It is further known that the existence of a good *weak* $\mathbb{F}$-linear LTC implies the existence of a good weak 3-query $\mathbb{F}$-linear LTC [6, Thm. A.1]. We do not know if there is a query complexity reduction method that works for a general good LTC.

Proof Idea

Our alphabet reduction method for LTCs (Theorems 2 and 3) is simple in nature and consists of concatenating[2] the given 2-query LTC $C \subseteq \Sigma^n$ with a special code $D \subseteq \Delta^k$ (which remains fixed as n grows) and showing that the concatenated code $C \circ D \subseteq \Delta^{nk}$ is also a good 2-query LTC. While concatenation is a well-known method for reducing a good code's alphabet size, it usually fails to preserve the property of being a 2-query LTC. The novelty of our approach is in showing that the inner code $D \subseteq \Delta^k$ (and the identification of Σ with D) can be chosen so that the property of being a 2-query LTC will persist.

Let us now explain in more detail how our concatenation approach works in the context of Theorem 2(i). To begin, let us first see why $C \circ D \subseteq \Delta^{nk}$ is in general not a 2-query LTC. Let T denote the 2-query tester of C, and choose some encoding function $f : \Sigma \to D \subseteq \Delta^k$. Then each codeword of $C \circ D$ has the form

$$f(w_1)f(w_2)\cdots f(w_n) \in \Delta^{nk}$$

for some $w \in C$. Let $u \in \Delta^{nk}$ and write $u = u_1 u_2 \cdots u_n$ with $u_1, \ldots, u_n \in \Delta^k$. A natural way to test whether u is in $C \circ D$ would be to try one of the following:

(1) Check that $u_i \in D$ for some $i \in [n]$;

(2) Emulate the tester T of C: run T to get two coordinates $i, j \in [n]$ to read, find the letters $a, b \in \Sigma$ satisfying $f(a) = u_i$, $f(b) = u_j$ (or reject if there are no such a, b), and return what T would have returned upon reading a and b in positions i and j.

A naive implementation of (1) and (2) requires making k or $2k$ queries to u, so the 2-query testability is seemingly lost.

[2] Concatenation of codes is recalled in Section 3.

We solve the large number of queries in (1) by requiring that $D \subseteq \Delta^k$ is defined by 2-letter constraints. Then, instead of checking whether $u_i \in D$, we check whether u_i satisfies a random 2-letter constraint. (Note that since $D \subseteq \Delta^k$ remains fixed as n grows, we do not need the stronger assumption that $D \subseteq \Delta^k$ is included in a family of codes that is a 2-query LTC.)

Reducing the number of queries in (2) is possible under special assumptions on T and f. Given a random seed s for T, denote by $i_s, j_s \in [n]$ the coordinates queried by T, and by $T^s(w)$ the output of T on $w \in \Sigma^n$ when given the seed s. What we need from T and f is that for every s, there are coordinates $i', j' \in [k]$ (depending on s) such that one can determine $T^s(w)$ – which depends only on w_{i_s} and w_{j_s} – from the i'-th letter of $f(w_{i_s})$ and the j'-th letter of $f(w_{j_s})$. In other words, writing $f_{i'} : \Sigma \to \Delta$ for the i'-th coordinate of f (so that $f(a) = (f_1(a), \ldots, f_k(a))$), we want to be able to determine $T^s(w)$ from $f_{i'}(w_{i_s})$ and $f_{j'}(w_{j_s})$ for some i', j' depending on s. When this holds for every seed s, we say that T is *f-compatible*; see Section 4 for details. Provided that T is f-compatible, we can emulate the action of T^s on $u = u_1 \cdots u_n$ by reading just two letters from u, namely, $(u_{i_s})_{i'}$ and $(u_{j_s})_{j'}$, and thus perform (2) using only 2 queries.

We show in Theorem 8 that if $D \subseteq \Delta^k$ is defined by 2-letter constraints and T is f-compatible, then the concatenated code $C \circ D$ is indeed a 2-query LTC. With this at hand, we need to find a code $D \subseteq \Delta^k$ and an encoding function $f : \Sigma \to D$ such that (i) T is f-compatible, and (ii) $D \subseteq \Sigma^k$ is defined by 2-letter constraints.

The code $D \subseteq \Delta^k$ that we choose for the task is a generalization of the *long code* of [2, §3]. Formally, letting $f_1, \ldots, f_k$ denote *all* the functions from Σ to Δ (so that $k = |\Delta|^{|\Sigma|}$), we choose D to be the code $L(\Sigma, \Delta) := \{(f_1(a), \ldots, f_k(a)) \mid a \in \Sigma\} \subseteq \Delta^k$ and take $f : \Sigma \to D$ to be the obvious encoding $f(a) = (f_1(a), \ldots, f_k(a))$. We call $L(\Sigma, \Delta)$ a *generalized long code*; the usual long code is the special case where $\Sigma = \{0,1\}^t$ and $\Delta = \{0,1\}$. The rationale behind this choice of D is that having any function $g : \Sigma \to \Delta$ in the collection $\{f_i\}_{i=1,\ldots,k}$ helps in securing that T is f-compatible. (This is still not enough to guarantee f-compatibility, and we solve this below.) The drawback of choosing D to be $L(\Sigma, \Delta)$ is that it *a priori* it does not fulfill the requirement of being defined by 2-letter constraints. We prove this nontrivial statement in Theorem 17 under the assumption $|\Delta| \geq 3$ (the hardest case is $|\Delta| = 3$). By contrast, the "usual" long code $L(\{0,1\}^t, \{0,1\})$ is *not* defined by 2-letter constraints; see Remark 20.

In order to finish, it remains to check that the tester T is f-compatible. We show in Proposition 23 that the f-compatibility of T is equivalent to another condition called *Δ-separability*. While this condition fails in general, we show in Theorem 26 that T can be replaced with a Δ-separable 2-query tester with soundness that is proportional to that of T. Conveniently, this replacement is also the usual trick to turn an adaptive tester into a nonadaptive tester, so we can also treat non-adaptive 2-query testers. After that replacement, we can finally conclude that the concatenated code $C \circ D \subseteq \Delta^{nk}$ is a 2-query LTC.

The proof of Theorem 3(i) follows a similar path except that we define $D \subseteq \Delta^k$ by choosing $f_1, \ldots, f_k$ to be all the $\mathbb{F}$-linear functions from Σ to Δ. This gives rise to what we call a *generalized Hadamard code*; see § 5.2. Similarly to the generalized long code, it can be defined by 2-letter constraints if and only if $\dim \Delta \geq 2$ (Theorem 16).

▶ Remark 5. Passing from $C \subseteq \Sigma^n$ to $C \circ D \subseteq \Delta^{nk}$ as described above scales down the soundness by a constant factor depending on Σ and D. There are three sources for this scale-down.

The first is the soundness of the 2-query tester for D (cf. (1) above) – we use the naive lower bound $\frac{1}{R}$ where R is the number of 2-letter constraints defining D; for the generalized long code this factor is $|\Delta|^{-O(|\Sigma|)}$. This can probably be improved significantly. That is,

when $|\Delta| \geq 3$, we expect that the generalized long code $L(\Sigma, \Delta) \subseteq \Delta^{|\Delta|^{|\Sigma|}}$ should have a 2-query tester with soundness that is *independent of* $|\Sigma|$. Similarly, in the linear case, we expect the generalized Hadamard code (§ 5.2) to have the same property when $\dim \Delta \geq 2$. We pose this as a problem.

The second source for loosing on the soundness is the block length k of $D \subseteq \Delta^k$. This $\frac{1}{k}$ factor comes up in the analysis of the tester of $C \circ D$, and is forced in situations in which the coordinates $i', j' \in [k]$ from the definition of f-compatibility (which are determined by the random seed of T) are very unevenly distributed. See Remark 9 for details and possible ways one might overcome it (which were not pursued in this work).

The final cause for loosing on the soundness is the replacement of T by a Δ-separable tester. In more detail, in the non-linear case, the trick is to first guess what are the two letters that will be read by T, and apply T only if these letters are actually encountered (otherwise accept the word). This multiplies the soundness by a factor of $|\Sigma|^{-2}$. We do not know how to avoid this factor in general. However, in the linear case, T is morally close to being Δ-separable (it is always Δ^m-separable for $m = \lceil \frac{2 \dim \Sigma}{\dim \Delta} \rceil$), and one can use a different trick which reduces the soundness by a factor of $O(\log |\Sigma|)$.

Relation to Other Works on Alphabet Reduction

Concatenation with the "ordinary" long code and Hadamard code was used in the literature for alphabet reduction in the construction of PCPs and 2-query LTCs. For example, in [8, §§7–8], resp. [20, §6.4], this technique is used to construct non-linear, resp. linear, weak 2-query LTCs having linear distance and inverse-polylogarithmic rate. The same method was pushed further in [25] to construct (strong) $\mathbb{F}_2$-linear 3-query LTCs (with a binary alphabet) and $\mathbb{F}_2$-LTCs with an 8-letter alphabet having similar distance and rate.

In a nutshell, the idea of these alphabet reductions is to encode both the letters and the constraints of the code; this is different from our approach which encodes only the letters. To explain this, let us describe an oversimplified version of the alphabet reduction of [8]: We start with a code $C \subseteq \Sigma^n$ having a 2-query tester T. Choose some embedding of Σ in $\{0,1\}^t$ and use it to view words in Σ^n as strings of tn bits. Think of T as checking at random one of (say) R 2-letter constraints which define $C \subseteq \Sigma^n$. Since $\Sigma \subseteq \{0,1\}^t$, every such constraint is a circuit on $2t$ bits. Now construct a new code $C' \subseteq \{0,1\}^N$ by modifying the codewords $w \in C$ as follows. First, view w as a tn-bit string using that $\Sigma \subseteq \{0,1\}^t$. Then, for each of the R constraints defining C, attach to w the long-code encoding of the $2t$ bits participating in that constraint.[3] This results in a string of $N = tn + 2^{2^{2t}} R$ bits, and we take C' to be the collection of all such strings. Since the long-code encoding of $2t$ bits includes the evaluation of any circuit on those bits, if $w' \in \{0,1\}^N$ is obtained from some $w \in \Sigma^n$ by the above procedure, then we can emulate the action of T on w by reading just 1 bit from w'. The long code[4] has two additional relevant properties. First, it has a 3-query tester. Second, it has a 2-local decodability property – if $y \in \{0,1\}^{2^{2^{2t}}}$ is close to the encoding of some $x \in \{0,1\}^{2t}$, then there is a randomized algorithm which can determine x_i with high probability by reading just 2 letters from y.[5] Using these properties and other mild assumptions, it can be shown that $C' \subseteq \{0,1\}^N$ has a 3-query tester with soundness

[3] This is a little different from [8, §7], where a certain puncturing of the long code is used.
[4] More precisely, its puncturing that is used in [8, §7].
[5] The algorithm: choose uniformly at random two functions $f, g : \{0,1\}^{2t} \to \{0,1\}$ such that $f + g$ is the i-th coordinate function $e_i : \{0,1\}^{2t} \to \{0,1\}$, and return $y_f + y_g$. See [8, Thm. 7.1] for a precise statement. See [20, Lem. 6.21] for a corresponding statement for the Hadamard code.

that is proportional to that of T. Briefly, the idea is that given $u \in \{0,1\}^N$, we can detect if u is very corrupted in its last $2^{2^{2t}}R$ bits (the long-code encoding of the constraints) using the 3-query tester of the long code. If that is not the case, then we can use the 2-local decodability property of the long code to check if the last $2^{2^{2t}}R$ bits of u are consistent with the first tn bits (this requires querying $2+1=3$ bits from u). When they are almost consistent, meaning that u is very close to a word coming from some $w \in \Sigma^n$, we can emulate T on w.

There are two reasons why we cannot use this alphabet reduction approach to prove our main results. First, the above construction increases the block length by $O(R)$, where $\log_2 R$ is the randomness complexity of T, while leaving the message length the same. Therefore, applying it to a good LTC would result in a good LTC only if $R = O(n)$, equiv. the randomness complexity of T is $\log n + O(1)$. By contrast, our alphabet reduction method (Theorems 2 and 3) makes no assumptions on the randomness complexity. Second, the alphabet reduction just described (loosely) relies on the fact that the long code (resp. Hadamard code) is 2-locally decodable, which is one reason why the resulting tester needs to query $2+1=3$ letters (the other reason is that the long code's tester queries 3 letters). If we were to use this approach to get a 2-query LTC, we would need to replace the long code with a 1-locally decodable code, and there are no such useful codes. That said, it seems likely that applying the alphabet reduction of [8], [20], [25] to a good 2-query LTC (or even any good LTC) whose tester has randomness complexity $\log_2(n) + O(1)$, e.g., those of [11] (or [9], [22]), would result in a good 3-query LTC on a binary alphabet.

Finally, in terms of soundness, the alphabet reduction of [8], etc. is more economical than ours as it decreases the soundness by a constant factor (independent of $|\Sigma|$); see Remark 5 for what causes the soundness loss in our alphabet reduction. As for rate and distance, our alphabet reduction reduces the rate by a smaller factor than that of *op. cit.*, and both alphabet reductions reduce the distance by a constant factor independent of $|\Sigma|$.

Organization

The paper is organized as follows: Section 2 is preliminary and recalls necessary definitions and facts about codes and testers. In Section 3, we recall concatenation of codes. In Section 4, we prove a theorem giving sufficient conditions for the concatentation of two codes to be a q-query LTC. The inner codes to which this result will be applied – the generalized long code and the generalized Hadamard code – are presented in Section 5. In that section, it is also shown that these codes are defined by 2-letter constraints. Section 6 is concerned with showing that one can replace the tester of any LTC by another tester to which our result about concatenation of LTCs can be applied. Finally, in Section 7, all previous results are combined to prove Theorems 2 and 3.

2 Preliminaries

2.1 General Conventions

Throughout this paper, $\mathbb{F}$ is a finite field, and $[n]$ denotes the set $\{1,\dots,n\}$. Vector spaces are over $\mathbb{F}$ and are assumed to be finite-dimensional. (Unlike some other texts, the letter q will be reserved for the number of queries performed by a tester and will *not* denote the size of $\mathbb{F}$.) An alphabet is a finite set with at least two elements.

Recall that a distribution on a countable set X is a collection of non-negative real numbers $p = (p_x)_{x \in X}$ adding up to 1; the number p_x is the probability of drawing x when sampling an element from X according to p. We write $x \sim p$ to indicate that $x \in X$ is chosen at random accroding to the distribution p. For a finite set X, we write $x \sim X$ to denote that x is chosen from X uniformly at random.

2.2 Error Correcting Codes

Let Σ be an alphabet and $n \in \mathbb{N}$. We write Σ^n for the set of n-letter words in the alphabet Σ, and unless indicated otherwise, write w_i for the i-th letter of a word $w \in \Sigma^n$. As usual, the normalized Hamming distance on Σ^n, denoted $\delta(\cdot,\cdot)$, is given by $\delta(u,v) = \frac{1}{n} \cdot \#\{i \in \{1,\dots,n\} : u_i \neq v_i\}$ for all $u, v \in \Sigma^n$.

In this work, an *error correcting code*, or a *code* for short, with alphabet Σ and block length n is a nonempty subset $C \subseteq \Sigma^n$. Recall that the *relative distance* of $C \subseteq \Sigma^n$ is

$$\delta(C) := \min\{\delta(u,v) \,|\, u, v \in C,\, u \neq v\}$$

and its *rate* is

$$r(C) := \frac{\log_{|\Sigma|} |C|}{n}.$$

We say that C has relative distance δ (resp. rate r) when $\delta(C) \geq \delta$ (resp. $r(C) \geq r$) and add the word "exactly" to indicate that equality holds.

It is common to think of a code $C \subseteq \Sigma^n$ as ranging in a *family of codes*, i.e., a sequence of codes $\{C_i \subseteq \Sigma^{n_i}\}_{i \in \mathbb{N}}$ with n_i tending to ∞. Recall that the family $\{C_i \subseteq \Sigma^{n_i}\}_{i \in \mathbb{N}}$ is said to be a *good code* if there are $r, \delta > 0$ such that $r(C_i) \geq r$ and $\delta(C_i) \geq \delta$ for all i. In this case, we also say that the family $\{C_i \subseteq \Sigma^{n_i}\}_{i \in \mathbb{N}}$ has relative distance δ and rate r.

2.3 Testers

Let $q \in \mathbb{N}$. Recall that a *q-query tester*, or a *q-tester* for short, for a code $C \subseteq \Sigma^n$ is a randomized algorithm T which, given oracle access to some $w \in \Sigma^n$, reads at most q letters from w and returns 1 – meaning "accept" – or 0 – meaning "reject" – subject to the requirement that any $w \in C$ is accepted. The tester T is said to have soundness μ ($\mu \geq 0$) if

$$\Pr(T(w) = 0) \geq \mu\, \delta(w, C) \qquad \forall w \in \Sigma^n.$$

The tester T is called *non-adaptive* if it does not use information from previous queries to w to determine which position to read next.

Unless explicitly indicated, we always assume that testers are non-adaptive.

Observe that in order to specify a (non-adaptive) q-tester for $C \subseteq \Sigma^n$, it is enough to give the following data: a finite set I, a probability distribution $p = (p_i)_{i \in I}$ on I, a q-tuple $(a_1^{(i)}, \dots, a_q^{(i)}) \in [n]^q$ for each $i \in I$, and a function $T^{(i)} : \Sigma^q \to \{0,1\}$ for every $i \in I$. The corresponding tester T then works as follows: Given $w \in \Sigma^n$, choose $i \in I$ according to the distribution p, read the letters in positions $a_1^{(i)}, \dots, a_q^{(i)}$ from w and return $T^{(i)}(w_{a_1^{(i)}}, \dots, w_{a_q^{(i)}})$. We write this simply as

$$T = (T^{(i)}, (a_1^{(i)}, \dots, a_q^{(i)}), p_i)_{i \in I}.$$

A code $C \subseteq \Sigma^n$ is said to be *defined by q-letter constraints* if it has a q-query tester with positive soundness. This is equivalent to saying that there is a list of constraints on words in Σ^n, each involving q or less letters, such that C is the set of words in Σ^n satisfying all those constraints.

A family of codes $\{C_i \subseteq \Sigma^{n_i}\}_{i\in\mathbb{N}}$ is called a *q-query locally testable code* (q-query LTC) if there is $\mu > 0$ such that each code in the family admits a q-tester with soundness μ. In this case, we also say that the q-query LTC $\{C_i \subseteq \Sigma^{n_i}\}_{i\in\mathbb{N}}$ has soundness μ. A *locally testable code* (LTC) is a family of codes that is a q-query LTC for some $q \in \mathbb{N}$.

2.4 Linear Codes

Let $\mathbb{F}$ be a finite field, and suppose that the alphabet Σ is also an $\mathbb{F}$-vector space (hence $\Sigma \neq 0$). A code $C \subseteq \Sigma^n$ is said to be an *$\mathbb{F}$-code* if C is a subspace of Σ^n. An *$\mathbb{F}$-linear code* is an $\mathbb{F}$-code with alphabet $\mathbb{F}$.

Let C be an $\mathbb{F}$-code. We say that a q-tester $T = (T^{(i)}, (a_1^{(i)}, \ldots, a_q^{(i)}), p_i)_{i\in I}$ is *$\mathbb{F}$-linear* if for each $i \in I$, there is a linear subspace $V_i \subseteq \Sigma^q$ such that

$$T^{(i)}(u) = \begin{cases} 1 & u \in V_i \\ 0 & u \notin V_i \end{cases}$$

for all $u \in \Sigma^q$. For example, when $\Sigma = \mathbb{F}$ and V_i has codimension 1 in Σ^q, the test $T^{(i)}$ merely checks that the input $w \in \mathbb{F}^n$ satisfies a single homogeneous linear equation involving at most q of the coordinates.

Any q-tester for an $\mathbb{F}$-code can be replaced by an $\mathbb{F}$-linear q-query tester having soundness greater than or equal to the soundness of the original tester.

A *q-query $\mathbb{F}$-LTC* (resp. *$\mathbb{F}$-linear LTC*) is a family of $\mathbb{F}$-codes (resp. $\mathbb{F}$-linear codes) for which there is $\mu > 0$ such that each code in the family admits an $\mathbb{F}$-linear q-tester with soundness μ.

3 Concatenation of Codes

We proceed by recalling concatenation of codes, setting notation along the way. Let $C \subseteq \Sigma^n$ be a code with alphabet Σ, let Δ be another alphabet, and let $f : \Sigma \to \Delta^k$ be an injective function with image D. We think of D as a code inside Δ^k and of $f : \Sigma \to D \subseteq \Delta^k$ as its encoding function. Recall that the *concatenated code* $C \circ D \subseteq \Delta^{kn}$ is obtained by encoding the letters of each codeword in C using the code D. Formally,

$$C \circ D := \{f(w_1) \cdots f(w_n) \,|\, w \in C\} \subseteq \Delta^{kn}.$$

The codes C and D are often called the *outer code* and *inner code*, respectively. When the encoding map f is not clear from the context, we shall write $C \circ_f D$ for $C \circ D$. For more details, see [24, Chp. 12] or [17, §10.1], for instance.

Note that if C is an $\mathbb{F}$-code, Δ is an $\mathbb{F}$-vector space and $f : \Sigma \to \Delta^k$ is $\mathbb{F}$-linear, then $C \circ D$ is an $\mathbb{F}$-code. When $\Delta = \mathbb{F}$, the code $C \circ D$ is moreover $\mathbb{F}$-linear.

The behavior of the relative distance and rate of $C \circ D$ is well-known and summarized in the following proposition.

▶ **Proposition 6.** *With notation as above, we have*

$$\delta(C \circ D) \geq \delta(C)\delta(D) \qquad \text{and} \qquad r(C \circ D) = r(C)r(D).$$

In applications of concatenation in this paper, the outer code C will range over a family of codes $\{C_i \subseteq \Sigma^{n_i}\}_{i\in\mathbb{N}}$, while the inner code $D \subseteq \Delta^k$ will remain fixed. Then, by Proposition 6, the family $\{C_i \circ_f D \subseteq \Sigma^{kn_i}\}_{i\in\mathbb{N}}$ will be a good code as long as $\{C_i \subseteq \Sigma^{n_i}\}_{i\in\mathbb{N}}$ is a good code; this holds no matter how poor the relative distance and rate of D are.

4 Concatenation of Codes with Testers

In general, the concatenation of codes admitting q-testers with positive soundness does not share the same property. However, in this section, we will show that under certain assumptions, this is indeed the case.

▶ **Definition 7.** *Let $C \subseteq \Sigma^n$ be a code, let $T = (T^{(i)}, (a_1^{(i)}, \dots, a_q^{(i)}), p_i)_{i \in I}$ be a q-tester for C (see § 2.3), and let $f : \Sigma \to \Delta^k$ be an injective function. We denote the j-coordinate of f as $f_j : \Sigma \to \Delta$. We say that T is* f-compatible *if for every $i \in I$, there are $b_1, \dots, b_q \in [k]$ and a function $g : \Delta^q \to \{0,1\}$ such that for every $w_1, \dots, w_q \in \Sigma$,*

$$T^{(i)}(w_1, \dots, w_q) = g(f_{b_1}(w_1), \dots, f_{b_q}(w_q)).$$

In the other words, we can compute $T^{(i)}(w_1, \dots, w_q)$ by reading q letters from the word $f(w_1) \cdots f(w_q)$ in Δ^{qk}.

▶ **Theorem 8.** *Let $C \subseteq \Sigma^n$ be a code admitting a q_C-tester T_C with soundness $\mu_C > 0$. Let $f : \Sigma \to \Delta^k$ be an injective function such that the code $D := \mathrm{im}(f) \subseteq \Delta^k$ has a q_D-tester T_D with soundness $\mu_D > 0$. Suppose further that T_C is f-compatible. Then the concatenated code $E := C \circ_f D \subseteq \Delta^{nk}$ admits a $\max\{q_C, q_D\}$-tester T_E with soundness*

$$\frac{\mu_C \mu_D}{qk + \mu_C + \mu_D}.$$

Proof. We abbreviate q_C to q and write $T = T_C = (T^{(i)}, (a_1^{(i)}, \dots, a_q^{(i)}), p_i)_{i \in I}$. Given $i \in I$, denote the $b_1, \dots, b_q \in [k]$ and the function $g : \Delta^q \to \{0,1\}$ promised by Definition 7 by $b_1^{(i)}, \dots, b_q^{(i)} \in [k]$ and $g^{(i)} : \Delta^q \to \{0,1\}$.

Fix some distribution $\rho = (\rho_1, \rho_2, \rho_3)$ on the set $\{1,2,3\}$; we will specify ρ at the end. Using ρ, we define a tester T_E for $E \subseteq \Delta^{nk}$ as follows: Given $u \in \Delta^{nk}$, write $u = u_1 \cdots u_n$ with $u_1, \dots, u_n \in \Delta^k$. Then choose $x \in \{1,2,3\}$ according to the distribution ρ and perform routine (x) from the following list.

(1) Choose $\ell \in [n]$ uniformly at random and return $T_D(u_\ell)$ (q_D letters are read from u).

(2) Choose $i \in I$ according to $p = (p_i)_{i \in I}$ and return $g^{(i)}\left((u_{a_1^{(i)}})_{b_1^{(i)}}, \dots, (u_{a_q^{(i)}})_{b_q^{(i)}}\right)$ (q_C letters are read from u).

(3) Choose $i \in I$ according to $p = (p_i)_{i \in I}$, choose $\ell \in [q_C]$ uniformly at random and return $T_D(u_{a_\ell^{(i)}})$ (q_D letters are read from u).

The rationale behind (1) and (2) was explained in the introduction. Routine (3) is included in order to deal with situations where $a_1^{(i)}, \dots, a_q^{(i)}$ are very unevenly distributed; otherwise it is subsumed by routine (1).

It is straightforward to see that T_E accepts every word in E. It remains to prove that it has the claimed soundness.

Let $u \in \Delta^{nk}$ and write $\varepsilon := \delta(u, E)$. Then

$$\Pr(T(u) = 0) = \sum_{x=1}^{3} \rho_x \cdot \Pr(T(u) = 0 \,|\, (x) \text{ is executed}).$$

We will bound the three summands on the right hand side from below. To that end, we think of D^n as a subset of Δ^{nk} and define the following words in Δ^{nk}:

- u' is an element of D^n with minimal distance from u.
- u'' is an element of E with minimal distance from u'.

We further let $w' \in \Sigma^n$ denote the word satisfying $f^n(w') = u'$. Obseve that

$$\varepsilon = \delta(u, E) \le \delta(u, u'') \le \delta(u, u') + \delta(u', u''). \tag{4.1}$$

Now, since T_D has soundness μ_D, we have

$$\begin{aligned}\Pr(T_E(u) = 0 \,|\, (1) \text{ is exec.}) &= \Pr_{\ell\sim[n]}\left(T_D(u_\ell) = 0\right) \ge \frac{1}{n}\sum_{\ell=1}^{n} \mu_D\, \delta_{\Delta^k}(u_\ell, D) \\ &= \mu_D\, \delta_{\Delta^{nk}}(u, D^n) = \mu_D\, \delta(u, u').\end{aligned}$$

Next, denote by R the probability that $u_{a_t^{(i)}} \in D$ for all $t = 1, \dots, q$ (equivalently, $u_{a_t^{(i)}} = u'_{a_t^{(i)}}$ for $t = 1, \dots, q$) when $i \in I$ is chosen according to the distribution p. Then

$$\begin{aligned}\Pr(T_E(u) = 0 \,|\, (2) \text{ is exec.}) &= \Pr_{i\sim p}\left[g^{(i)}\left((u_{a_1^{(i)}})_{b_1^{(i)}}, \dots, (u_{a_q^{(i)}})_{b_q^{(i)}}\right) = 0\right] \\ &\ge \Pr_{i\sim p}\left[g^{(i)}\left((u_{a_1^{(i)}})_{b_1^{(i)}}, \dots, (u_{a_q^{(i)}})_{b_q^{(i)}}\right) = 0 \text{ and } u_{a_t^{(i)}} = u'_{a_t^{(i)}}\ \forall t \in [q]\right] \\ &= \Pr_{i\sim p}\left[g^{(i)}\left((u'_{a_1^{(i)}})_{b_1^{(i)}}, \dots, (u'_{a_q^{(i)}})_{b_q^{(i)}}\right) = 0 \text{ and } u_{a_t^{(i)}} = u'_{a_t^{(i)}}\ \forall t \in [q]\right] \\ &\ge \Pr_{i\sim p}\left[T^{(i)}(w'_{a_1^{(i)}}, \dots, w'_{a_q^{(i)}}) = 0\right] - \Pr_{i\sim p}\left[\exists t \in [q] \,:\, u_{a_t^{(i)}} \ne u'_{a_t^{(i)}}\right] \\ &\ge \mu_C\, \delta(w', C) - (1 - R) \\ &\ge \mu_C\, \delta(u', E) - (1 - R) = \mu_C\, \delta(u', u'') - (1 - R),\end{aligned}$$

while

$$\begin{aligned}\Pr(T_E(u) = 0 \,|\, (3) \text{ is exec.}) &= \Pr_{i\sim p, \ell\sim[q]}(T_D(u_{a_\ell^{(i)}}) = 0) \\ &\ge (1 - R) \cdot \Pr_{i\sim p, \ell\sim[q]}\left[T_D(u_{a_\ell^{(i)}}) = 0 \,\middle|\, u_{a_\ell^{(i)}} \notin D \text{ for some } \ell\right] \\ &\ge (1 - R) \cdot \frac{1}{q} \cdot \frac{\mu_D}{k}\end{aligned}$$

Here, the second-to-last inequality holds since $u_{a_\ell^{(i)}} \notin D$ implies $\delta_{\Delta^k}(u_{a_\ell^{(i)}}, D) \ge \frac{1}{k}$.

Putting everything together gives

$$\begin{aligned}\Pr(T_E(u) = 0) &\ge \rho_1 \cdot \delta(u, u')\mu_D + \rho_2 \cdot (\mu_C\, \delta(u', u'') - (1 - R)) + \rho_3 \cdot (1 - R)\frac{\mu_D}{qk} \\ &= \rho_1\mu_D\delta(u, u') + \rho_2\mu_C\, \delta(u', u'') + (1 - R)\left(-\rho_2 + \rho_3 \cdot \frac{\mu_D}{qk}\right) =: (\star)\end{aligned}$$

Suppose that ρ is chosen so that $\rho_2 \le \frac{\mu_D}{qk}\rho_3$. Then by (4.1), we have

$$(\star) \ge \rho_1\mu_D\delta(u, u') + \rho_2\mu_C\, \delta(u', u'') \ge \varepsilon \cdot \min\{\rho_1\mu_D, \rho_2\mu_C\}.$$

Subject to the requirements $\rho_1 + \rho_2 + \rho_3 = 1$ and $\rho_2 \le \frac{\mu_D}{qk}\rho_3$, the right hand side is maximized for

$$\rho_1 = \frac{\mu_C}{\mu_C + \mu_D + qk}, \quad \rho_2 = \frac{\mu_D}{\mu_C + \mu_D + qk}, \quad \rho_3 = \frac{qk}{\mu_C + \mu_D + qk},$$

in which case it evaluates to $\frac{\mu_C\mu_D}{\mu_C + \mu_D + qk} \cdot \varepsilon$. This is what we want. ◀

▶ Remark 9. Without additional assumptions in Theorem 8, the tester T_E described in its proof cannot have soundness larger than $\Omega(\frac{1}{k})$. The reason is that we make no assumption

on the distribution of the coordinates $b_1^{(i)}, \ldots, b_q^{(i)} \in [k]$ from Definition 7 as $i \in I$ distributes according to p. For example, consider an extreme situation in which $b_1^{(i)} = \cdots = b_q^{(i)} = 1$ for all $i \in I$ and $\delta(D) > \frac{1}{k}$. Start with a word $w' \in \Sigma^n$ that is δ-far from C, let $w \in C$ be the closest word to w', and let $u = f(w_i) \cdots f(w_n)$ and $u'_i = f(w'_i) \cdots f(w'_n)$. Then, for every $i \in [n]$ such that $w_i \neq w'_i$, replace the $(ki+1)$-letter of u' with the corresponding letter in u; denote the resulting word by u''. Now, the word u'' satisfies $\delta(u'', C \circ D) \geq \delta(u', C \circ D) - \delta(u'', u') \geq \delta \cdot (\delta(D) - \frac{1}{k}) = \Omega(\delta)$. However, when we attempt to emulate T_C on u'' (routine (2) above), we cannot distinguish between u and u''. Thus, the only way we could detect that u'' is not in $C \circ D$ is by applying the tester of D to one of the k-letter blocks of u'' (routines (1) or (3)). Regardless of how this block is chosen, it differs by only one letter from a word in D, and so we are only guaranteed to succeed with probability $\mu_D \cdot \frac{1}{k}$ or less. It follows that T_E rejects u'' with probability $O(\frac{1}{k})$, while u'' is $\Omega(\delta)$-far from $C \circ D$.

One can overcome the $\Omega(\frac{1}{k})$ limitation by requiring that for every $t \in [q]$, the position $k(a_t^{(i)} - 1) + b_t^{(i)}$ distributes nearly uniformly in $[nk]$ when $i \sim p$. This requires imposing additional assumptions on T, f, D. We omit the details because these assumptions are not guaranteed in the situations in which we apply Theorem 8.

Another approach to bypass this limitation is to take into advantage the fact that in routine (3) above, we only need to check that the block $u_{a_t^{(i)}}$ was not corrupted *in position* $b_t^{(i)}$. Calling $T_D(u_{a_t^{(i)}})$ is wasteful since it does not use this information. To use it, however, we need D to be equipped with a "local tester" able to detect with high probability whether a given word is far from D *or corrupted in a given position*. Such local testers were not considered in the literature, and are out of the scope of this work.

Theorem 8 implies in particular that if $C \subseteq \Sigma^n$ admits a q-tester with soundness $\mu > 0$ and if Δ is an alphabet containing Σ, then the code $C \subseteq \Delta^n$ (i.e., C thought of as a code in Δ^n) has a q-tester with soundness $\frac{\mu}{q+1+\mu}$ (take $f : \Sigma \to \Delta^1$ to be the embedding of Σ in Δ). In this case, however, there is a tester with larger soundness.

▶ **Proposition 10.** *Let $C \subseteq \Sigma^n$ be a code admitting a q-tester T with soundness $\mu > 0$, and let Δ be an alphabet containing Σ. Then $C \subseteq \Delta^n$ admits a tester T' with soundness $\frac{\mu}{\mu+1}$.*

Proof. Write $T = (T^{(i)}, (a_1^{(i)}, \ldots, a_q^{(i)}), p_i)_{i \in I}$. Fix $\rho_1, \rho_2 > 0$ with $\rho_1 + \rho_2 = 1$ and consider the following tester T' for $C \subseteq \Delta^n$: Given $u \in \Delta^n$, choose $x \in \{1, 2\}$ according to the distribution (ρ_1, ρ_2) and perform routine (x) from the following:

(1) Choose $\ell \in [n]$ uniformly at random and accept u if and only if $u_\ell \in \Sigma$.

(2) Choose $i \in I$ according to p. Read $u_{a_1^{(i)}}, \ldots, u_{a_q^{(i)}}$. If one of these letters is not in Σ, then reject u. Otherwise, return $T^{(i)}(u_{a_1^{(i)}}, \ldots, u_{a_q^{(i)}})$.

As in the proof of Theorem 8, to show that T' has the claimed soundness, let $u' \in \Sigma^n$ be a word with minimal distance from u, and let $u'' \in C$ be a word with minimal distance from u'. Then

$$\delta(u, u') + \delta(u', u'') \geq \delta(u, C) =: \varepsilon.$$

Moreover, $\Pr(T'(u) = 0 \,|\, (1) \text{ exec.}) = \delta(u, u')$. Next, as in the proof of Theorem 8, writing R for the probability that $u_{a_1^{(i)}}, \ldots, u_{a_q^{(i)}} \in \Sigma$ when $i \in I$ distributes according to p, we have

$$\begin{aligned}\Pr(T'(u) = 0 \,|\, (2) \text{ is exec.}) &= (1-R) \\ &\quad + \Pr_{i\sim p}\left[T^{(i)}(u_{a_1^{(i)}}, \ldots, u_{a_q^{(i)}}) = 0 \text{ and } u_{a_1^{(i)}}, \ldots, u_{a_q^{(i)}} \in \Sigma\right] \\ &\geq (1-R) + [\mu\,\delta(u', C) - (1-R)] \geq \mu\,\delta(u', u'').\end{aligned}$$

It follows that

$$\Pr(T'(u) = 0) \geq \rho_1\,\delta(u, u') + \rho_2\mu\,\delta(u', u'') \geq \min\{\rho_1, \rho_2\mu\}\varepsilon.$$

Taking $\rho_1 = \frac{\mu}{\mu+1}$ and $\rho_2 = \frac{1}{\mu+1}$ completes the proof of proposition. ◀

▶ Remark 11. In Theorem 8, if Σ and Δ are $\mathbb{F}$-vector spaces, $f : \Sigma \to \Delta$ is $\mathbb{F}$-linear, and the code C is an $\mathbb{F}$-code, then $E = C \circ D \subseteq \Delta^{nk}$ is also an $\mathbb{F}$-code. Moreover, it is clear from the proof that if the testers T_C and T_D are $\mathbb{F}$-linear, then so is the tester T_E of E.

Similarly, in Proposition 10, if Σ and Δ are $\mathbb{F}$-vector spaces such that Σ is a subspace of Δ and T is $\mathbb{F}$-linear, then so is the tester T' for $C \subseteq \Delta^n$.

5 The Generalized Hadamard Code and the Generalized Long Code

In this section, we introduce two types of codes generalizing the (linear) Hadamard code and the (non-linear) long code, respectively. They will ultimately serve as the inner code when we apply Theorem 8, and to that end, we will show that they can be defined using 2-letter constraints. This stands in contrast to the original Hadamard code and long code which, as we also show, cannot be defined by 2-letter constraints.

5.1 A General Construction

We begin with a general construction which is natural in the context of Theorem 8. Let S be a finite set, let Δ be an alphabet and let $f_1, \ldots, f_n : S \to \Delta$ be some functions. We associate with S, Δ and the collection $\{f_i\}_{i=1}^n$ the code

$$D(\{f_i\}_{i=1}^n) := \{(f_1(s), \ldots, f_n(s)) \,|\, s \in S\} \subseteq \Delta^n.$$

If we further define $f : S \to \Delta^n$ by $f(s) = (f_1(s), \ldots, f_n(s))$ and assume that f is injective, then this recovers the setting of Theorem 8. In fact, every code $D \subseteq \Delta^n$ equipped with an encoding function $f : S \to D$ is of the form $D(\{f_i\}_{i=1}^n)$, but the point of our construction is to build a code in Δ^n from functions $f_1, \ldots, f_n : S \to \Delta$ rather than doing the opposite.

The construction recovers two well-known codes. When $S = \{0,1\}^k$, $\Delta = \{0,1\}$ and $f_1, \ldots, f_n$ consist of all the functions from S to $\{0,1\}$ (so that $n = 2^{2^k}$), the code C is known as the *long code*, e.g., see [2, §3]. Next, when $S = V$ with V an $\mathbb{F}$-vector space, $\Delta = \mathbb{F}$ and $f_1, \ldots, f_n$ are all the $\mathbb{F}$-linear functions from V to $\mathbb{F}$ (so that $n = |V|$), the code C is (up to equivalence) the $|\mathbb{F}|$-ary *Hadamard code* associated to the vector space V, e.g., see [17, Exercise 2.22] (we also recall the definition in § 5.2).

We now introduce a q-tester for $D(\{f_i\}_{i=1}^n)$. To that end, we say that a collection of functions $f_1, \ldots, f_q : S \to \Delta$ is *dependent* if

$$\mathrm{im}(f_1 \times \cdots \times f_q) = \{(f_1(s), \ldots, f_q(s)) \,|\, s \in S\} \subsetneq \Delta^q.$$

Informally, this means that knowing (say) the values of $f_1, \ldots, f_{q-1}$ on some input $s \in S$ gives some information on the value of f_q on that input, even without knowing what s is. (Note that it is possible for a single function f_1 to be dependent, namely, if its image is strictly smaller than Δ.) With this at hand, we introduce:

▶ **Definition 12** (q-dependence tester)**.** *With notation as a above, given a word $w \in \Delta^n$, the* q-dependence tester *for $D(\{f_i\}_{i=1}^n)$ chooses uniformly at random a tuple $(i_1, \ldots, i_q) \in [n]^q$ such that $f_{i_1}, \ldots, f_{i_q}$ are dependent, and accepts w if and only if $(w_{i_1}, \ldots, w_{i_q}) \in \mathrm{im}(f_{i_1} \times \cdots \times f_{i_q})$.*

If the q-dependence tester has positive soundness, then this soundness is at least n^{-q}.

▶ **Example 13.** To demonstrate how the q-dependence tester works, suppose for simplicity that $\Delta = \mathbb{F}$, and let $w \in \Delta^k$ be a word that is always accepted by the 3-dependence tester. We claim that if f_i, f_j, f_k satisfy $f_i + f_j = f_k$, then we must have $w_i + w_j = w_k$. Indeed, this is forced by the fact that $\mathrm{im}(f_i \times f_j \times f_k) \subseteq \{(\alpha, \beta, \alpha+\beta) \,|\, \alpha, \beta \in \mathbb{F}\}$. Similarly, if $f_i = \alpha f_j$ for some $\alpha \in \mathbb{F}$, then $w_i = \alpha w_j$, because for any $k \in [n]$, $\mathrm{im}(f_i, f_j, f_k) \subseteq \{(\beta, \alpha\beta, \gamma) \,|\, \beta, \gamma \in \mathbb{F}\}$ (f_k was added artificially to f_i and f_j to make a dependent triple of functions). More generally, for a general Δ, if there is a function $g : \Delta \times \Delta \to \Delta$ such that $g \circ (f_i, f_j) = f_k$ (i.e. $g(f_i(s), f_j(s)) = f_k(s)$ for all $s \in S$), then $g(w_i, w_j) = w_k$.

The following useful property of the q-dependence tester will be used freely in the sequel.

▶ **Proposition 14.** *With notation as above, the code $D(\{f_i\}_{i=1}^n) \subseteq \Delta^n$ is defined by q-letter constraints (i.e., it has a q-tester with positive soundness) if and only if its q-dependence tester has positive soundness.*

Proof. The "if" direction is clear, so we treat the "only if" direction. Write $D := D(\{f_i\}_{i=1}^n)$ and let $T = (T^{(i)}, (a_1^{(i)}, \ldots, a_q^{(i)}), p_i)_{i \in I}$ be a q-tester for D having positive soundness. Let $w \in \Delta^n - D$. It is enough to prove that w fails the q-dependence test with some positive probability. Since T has positive soundness, there is $i \in I$ such that $T^{(i)}(w_{a_1^{(i)}}, \ldots, w_{a_q^{(i)}}) = 0$. Since T accepts all words in D, we must also have $T^{(i)}(f_{a_1^{(i)}}(s), \ldots, f_{a_q^{(i)}}(s)) = 1$ for all $s \in S$. Consequently, $(w_{a_1^{(i)}}, \ldots, w_{a_q^{(i)}}) \notin \mathrm{im}(f_{a_1^{(i)}} \times \cdots \times f_{a_q^{(i)}})$. This means that $f_{a_1^{(i)}}, \ldots, f_{a_q^{(i)}}$ are dependent and that w will be rejected if the q-dependence test chooses these functions. ◀

5.2 The Generalized Hadamard Code

Keeping the notation above, suppose that both S and Δ are $\mathbb{F}$-vector spaces. We write V in place of S to denote that. Let $f_1, \ldots, f_n$ enumerate all the $\mathbb{F}$-linear functions from V to Δ (so that $n = |\mathbb{F}|^{\dim V \cdot \dim \Delta} = |\Delta|^{\dim V} = |V|^{\dim \Delta}$). The *generalized Hadamard code* associated to V and Δ is defined to be

$$H(V, \Delta) := D(\{f_i\}_{i=1}^n) = \{(f_1(v), \ldots, f_n(v)) \,|\, v \in V\} \subseteq \Delta^n.$$

It easy to see that $H(V, \Delta) \subseteq \Delta^n$ is an $\mathbb{F}$-code of relative distance $1 - \frac{1}{|\Delta|}$ and rate $\frac{\log_{|\Delta|} |V|}{|\Delta|^{\dim V}} = \frac{\dim V}{\dim \Delta \cdot |\Delta|^{\dim V}} = \frac{\log_{|\Delta|} n}{n \dim \Delta}$.

▶ Remark 15. Recall that the Hadamard code associated to an $\mathbb{F}$-vector space V is $\mathrm{Hom}_{\mathbb{F}}(V, \mathbb{F})$ – the set of linear functions from V to $\mathbb{F}$ – viewed as a subset of $\mathbb{F}^V$, i.e., all functions from V to $\mathbb{F}$. More concretely, writing $v_1, \ldots, v_n$ for the vectors in V, the Hadamard code of V is

$$H' = \{(f(v_1), \ldots, f(v_n)) \,|\, f : V \to \mathbb{F} \text{ is linear}\} \subseteq \mathbb{F}^n.$$

This code is in fact equivalent to $H(V, \mathbb{F})$, which is the reason why we call $H(V, \Delta)$ a generalized Hadamard code. Indeed, let V^* denote the dual vector space of V. Then $f_1, \ldots, f_n$ are the vectors of V^*. Since every linear map g from V^* to $\mathbb{F}$ admits a unique $v \in V$ such that $g(f) = f(v)$ for all $f \in V^*$, the code H' is just $H(V^*, \mathbb{F})$. Choosing an isomorphism $V \to V^*$ then gives an equivalence between H' and H.

It is well-known that the code $H(V,\mathbb{F})$ is 3-testable. Indeed, it is equivalent to the ordinary Hadamard code $\mathrm{Hom}_{\mathbb{F}}(V,\mathbb{F}) \subseteq \mathbb{F}^V$, which is defined by 3-letter constraints, namely,

$$\mathrm{Hom}_{\mathbb{F}}(V,\mathbb{F}) = \{f : V \to \mathbb{F} \,:\, f(x+y) = f(x) + f(y) \text{ and } f(\alpha x) = \alpha f(x) \text{ for all } x, y \in V,\ \alpha \in \mathbb{F}\}.$$

Tracing this back to $H(V,\mathbb{F}) \subseteq \mathbb{F}^n$ ($n = |V|$) shows that a word $w \in \mathbb{F}^n$ is in $H(V,\mathbb{F})$ if and only if

(1) $w_i = w_j + w_k$ for all $i,j,k \in [n]$ such that $f_i = f_j + f_k$ and

(2) $w_i = \alpha w_j$ for all $i,j \in [n]$ and $\alpha \in \mathbb{F}$ such that $f_i = \alpha f_j$.

Consequently, the 3-dependence tester of $H(V,\mathbb{F})$ has positive soundness (cf. Example 13). (In fact, for some fields $\mathbb{F}$, e.g. $\mathbb{F}_2$, the Hadamard code is known to have 3-testers with soundness that is independent of V; see [7], [1] and the many followup works.) On the other hand, it is not difficult to see that the 2-dependence tester for $H(V,\mathbb{F})$ has soundness 0 if $\dim V > 1$; the reason is that a pair of $\mathbb{F}$-linear functions $f, g : V \to \mathbb{F}$ is dependent if and only if they are proportional. As a result, $H(V,\mathbb{F})$ is not defined by 2-letter constraints (Proposition 14). By contrast, we now show that $H(V,\Delta)$ is defined by 2-letter constraints when $\dim \Delta \geq 2$.

▶ **Theorem 16.** *Let Δ be a vector space of dimension > 1 and let V be any vector space. Then the 2-dependence tester (Definition 12) for the code $H(V,\Delta) \subseteq \Delta^n$ has positive soundness (depending on n).*

Proof. We may assume without loss of generality that $\Delta = \mathbb{F}^m$ for some $m > 1$. Given functions $g_1, \ldots, g_r \in \mathrm{Hom}_{\mathbb{F}}(V,\mathbb{F})$ ($r \leq m$), we denote by $(g_1, \ldots, g_r, \vec{0})$ the function from V to $\Delta = \mathbb{F}^m$ sending $v \in V$ to $(g_1(v), \ldots, g_r(v), 0, \ldots, 0) \in \mathbb{F}^m = \Delta$. Let $g_1, \ldots, g_t$ enumerate all the linear functions from V to $\mathbb{F}$ ($t = |V|$). We may assume that $f_1, \ldots, f_n$ are numbered so that $f_i = (g_i, \vec{0})$ whenever $1 \leq i \leq t$.

Let $w \in \Delta^n$ be a word that is always accepted by the 2-dependence tester of $H(V,\Delta)$. We need to show that $w \in H(V,\Delta)$.

Since $\mathrm{im}(f_1), \ldots, \mathrm{im}(f_t) \subseteq \mathbb{F} \times \{0\}^{m-1}$ and w passes the 2-dependence test, we must have $w_1, \ldots, w_t \in \mathbb{F} \times \{0\}^{m-1}$. Thus, for each $i \in [t]$, we can write $w_i = (u_i, 0, \ldots, 0)$ with $u_i \in \mathbb{F}$.

Let $u = u_1 \cdots u_t$. We claim that $u \in D(\{g_i\}_{i=1}^t) = H(V,\mathbb{F})$. As we noted earlier, in order to show this, it is enough to check that for all $i,j,k \in [t]$ with $g_i + g_j = g_k$ (equiv. $f_i + f_j = f_k$), we have $u_i + u_j = u_k$, and for all $i,j \in [t]$ and $\alpha \in \mathbb{F}$ with $g_i = \alpha g_j$ (equiv. $f_i = \alpha f_j$), we have $u_i = \alpha u_j$. The second statement holds because w passes the 2-dependence test, so we only need to establish the first. There is $\ell \in [n]$ such that $f_\ell = (g_i, g_j, \vec{0})$. Then $\mathrm{im}(f_i \times f_\ell) \subseteq \{((\alpha,0,\ldots,0),(\alpha,\beta,0,\ldots,0)) \,|\, \alpha,\beta \in \mathbb{F}\}$, and so our assumption on w implies that w_ℓ has the form $(u_i, *, 0, \ldots, 0)$. Repeating this argument with f_j in place of f_i gives $w_\ell = (u_i, u_j, 0, \ldots, 0)$. Now, since $\mathrm{im}(f_k \times f_\ell) \subseteq \{((\alpha+\beta,0,\ldots,0),(\alpha,\beta,0,\ldots,0)) \,|\, \alpha,\beta \in \mathbb{F}\}$, we must have $u_k = u_i + u_j$, which is what we want. We conclude that $u \in H(V,\mathbb{F})$.

Let $v \in V$ be a vector such that $u_i = g_i(v)$ for all $i \in [t]$. We finish the proof by showing that $w_i = f_i(v)$ for all $i \in [n]$. Let $i \in [n]$. Then there are $i_1, \ldots, i_m \in [t]$ such that $f_i = (g_{i_1}, \ldots, g_{i_m})$. Now, for every $\ell \in [m]$, the dependence between f_i and f_{i_ℓ} implies that the ℓ-th component of $w_i \in \mathbb{F}^m$ is $u_{i_\ell} = g_{i_\ell}(v)$. As this holds for every ℓ, it follows that $w_i = (g_{i_1}(v), \ldots, g_{i_m}(v)) = f_i(v)$, as required. ◀

5.3 The Generalized Long Code

Returning to the general setting of § 5.1, let $f_1, \ldots, f_n$ denote all functions from S to Δ (so that $n = |\Delta|^{|S|}$). We call

$$L(S, \Delta) := D(\{f_i\}_{i=1}^n) = \{(f_1(s), \ldots, f_n(s)) \,|\, s \in S\} \subseteq \Delta^n$$

the *generalized long code* associated to S and Δ. This code has relative distance $1 - \frac{1}{|\Delta|}$ and rate $\frac{\log_{|\Delta|} |S|}{|\Delta|^{|S|}} = \frac{\log_{|\Delta|} \log_{|\Delta|} n}{n}$. As we noted earlier, this is a generalization of the *long code*, which arises as the special case where $S = \{0,1\}^k$ and $\Delta = \{0,1\}$.

We will show in Remark 20 that $L(S, \Delta)$ cannot be defined by 2-letter constraints when $|\Delta| = 2$. However, it turns out that it can be defined by 2-letter constraints when $|\Delta| > 2$.

▶ **Theorem 17.** *Let S be a set and let Δ be an alphabet with more than 2 letters. Then the 2-dependence tester of the generalized long code $L(S, \Delta) \subseteq \Delta^n$ has positive soundness (depending on n).*

In order to prove the theorem, we first note that $L(S, \{0,1\})$ has a 3-tester with positive soundness.

▶ **Proposition 18.** *With notation as above, suppose that $\Delta = \mathbb{F}_2$ and the functions $f_1, \ldots, f_n$ from S to Δ are numbered so that f_1 is the function sending every element of S to 1. Then a word $w \in \mathbb{F}_2^n$ is in $L(S, \Delta)$ if and only if*
(1) *$w_i + w_j = w_k$ (in $\mathbb{F}_2$) for every $i, j, k \in [n]$ such that $f_i + f_j = f_k$,*
(2) *$w_i \cdot w_j = w_k$ for every $i, j, k \in [n]$ such that $f_i \cdot f_j = f_k$, and*
(3) *$w_1 = 1$.*
In particular, $L(S, \mathbb{F}_2)$ is defined by 3-letter constraints.

Proof. This is well-known for $S = \{0,1\}^k$ [2, Prp. 3.2]. For the general case, see the extended version of this paper [10, Prp. 5.7]. ◀

We shall also need the following key lemma.

▶ **Lemma 19.** *Let $g_1, \ldots, g_t : S \to \{0,1\}$ be distinct functions. For every $i, j \in [t]$, let $g_{i,j}$ and $g'_{i,j}$ denote the functions from S to $\{0,1,2\}$ defined as follows:*

$$g_{i,j}(s) = \begin{cases} 0 & f_i(s) = 0 \\ 1 & f_i(s) = 1 \wedge f_j(s) = 0 \\ 2 & f_i(s) = 1 \wedge f_j(s) = 1 \end{cases} \qquad g'_{i,j}(s) = \begin{cases} 1 & f_i(s) = 1 \\ 0 & f_i(s) = 0 \wedge f_j(s) = 1 \\ 2 & f_i(s) = 0 \wedge f_j(s) = 0 \end{cases}$$

Let $f_1, \ldots, f_n : S \to \{0,1,2\}$ enumerate the functions in $\{g_i, g_{i,j}, g'_{i,j} \,|\, i, j \in [t]\}$. If the code $D' := D(\{g_i\}_{i=1}^t) \subseteq \{0,1\}^t$ is defined by 3-letter constraints, then $D := D(\{f_j\}_{j=1}^n) \subseteq \{0,1,2\}^n$ is defined by 2-letter constraints.

Proof. We may assume that $f_1, \ldots, f_n$ are numbered so that $f_i = g_i$ when $i \in [t]$.

Suppose $w \in \{0,1,2\}^n$ always passes the 2-dependence test of D. We need to show that $w \in D$. We begin by noting that since $\mathrm{im}(f_i) \subseteq \{0,1\}$ for all $i \in [t]$, we must have $w_i \in \{0,1\}$ for $i \in [t]$.

Next, fix $i, j \in [t]$ and let $k, \ell \in [n]$ be the numbers for which $f_k = g_{i,j}$ and $f_\ell = g'_{i,j}$. Observe that $f_i = g_i$ and $f_k = g_{i,j}$ are dependent, and $f_j = g_j$ and $f_k = g_{i,j}$ are dependent. These dependencies and our assumption on w imply that (i) $w_k = 0$ if and only if $w_i = 0$, (ii) if $w_k = 1$, then $w_j = 0$, and (iii) if $w_k = 2$, then $w_j = 1$. This means that

$$w_k = \begin{cases} 0 & w_i = 0 \\ 1 & w_i = 1 \wedge w_j = 0 \\ 2 & w_i = 1 \wedge w_j = 1. \end{cases} \tag{5.1}$$

Arguing similarly with ℓ in place of k gives

$$w_\ell = \begin{cases} 1 & w_i = 1 \\ 0 & w_i = 0 \wedge w_j = 1 \\ 2 & w_i = 0 \wedge w_j = 0. \end{cases} \tag{5.2}$$

At this point, if we could show that $w' := w_1 \cdots w_t$ is in D', then we could conclude that $w \in D$. Indeed, if it were the case that $w' \in D'$, then there would be an $s \in S$ such that $w_i = g_i(s) = f_i(s)$ for all $i \in [t]$. Since for every $p \in [n] - [t]$, the function f_p is either $g_{i,j}$ or $g'_{i,j}$ for some $i, j \in [t]$, by the previous paragraph, we would have that w_p is $g_{i,j}(s)$ or $g'_{i,j}(s)$, respectively, and consequently $w_p = f_p(s)$.

Suppose for the sake of contradiction that $w' \notin D'$. By assumption, D' admits a 3-tester $T = (T^{(u)}, (a_u, b_u, c_u), p_u)_{u \in U}$ with positive soundness. Then there is some $u \in U$ such that $T^{(u)}(w_{a_u}, w_{b_u}, w_{c_u}) = 0$. Write $(i, j, p) = (a_u, b_u, c_u)$ and $(x_1, x_2, x_3) = (w_{a_u}, w_{b_u}, w_{c_u})$. By the pigeonhole principle, at least two of x_1, x_2, x_3 are equal. We reorder (a_u, b_u, c_u) to have $x_1 = x_2$. Suppose that $x_1 = x_2 = 1$. The fact that $T^{(u)}(v_i, v_j, v_p) = 1$ for all $v \in D'$ while $T^{(u)}(x_1, x_2, x_3) = 0$ means that

$$\mathrm{im}(f_i \times f_j \times f_p) \subseteq \{0,1\}^3 - \{(x_1, x_2, x_3)\} = \{0,1\}^3 - \{(1,1,x_3)\}.$$

As a result, $\mathrm{im}(g_{i,j} \times f_p) \subseteq \{0,1,2\} \times \{0,1\} - \{(2, x_3)\}$. Writing $g_{i,j} = f_k$ with $k \in [n]$, this means that f_k and f_p are dependent (as functions from S to $\{0,1,2\}$), and thus $(w_k, w_p) \in \{0,1,2\} \times \{0,1\} - \{(2, x_3)\}$. However, since $w_i = x_1 = 1$ and $w_j = x_2 = 1$, we have $w_k = 2$ by (5.1), and since $w_p = x_3$, it follows that $(w_k, w_p) = (2, x_3)$, a contradiction to our earlier conclusion. Similarly, when $x_1 = x_2 = 0$, we reach a contradiction using (5.2). As all possibilities lead to contradiction, our assumption $w' \notin D'$ must have been false. This completes the proof. ◀

We are now ready to prove Theorem 17.

Proof of Theorem 17. We may assume without loss of generality that $\{0,1,2\} \subseteq \Delta$ and that the functions $f_1, \ldots, f_n : S \to \Delta$ are numbered so that $f_1, \ldots, f_t$ ($t = 2^{|S|}$) are all the functions from S to $\{0,1\}$.

Suppose that $w \in \Delta^n$ always passes the 2-dependence test. We need to show that $w \in L(S, \Delta)$.

We begin by showing that the exists $s \in S$ such that $w_i = f_i(s)$ for all $i \in [t]$. To that end, let $g_i = f_i$ for $i \in [t]$, and for all $i, j \in [t]$, define $g_{i,j}, g'_{i,j} : S \to \{0,1,2\}$ as in Lemma 19. We may assume that there is $m \in \{t, \ldots, n\}$ such that $f_1, \ldots, f_m$ enumerate all the functions in the set $\{g_i, g_{i,j}, g'_{i,j} \mid i, j \in [t]\}$. Now, since w passes the 2-dependence, and since $\mathrm{im}(f_i) \subseteq \{0,1,2\}$ for all $i \in [m]$, we have $w_i \in \{0,1,2\}$ for all $i \in [m]$. We may therefore think of $w' := w_1 \cdots w_m$ as a word in $\{0,1,2\}^m$. Our assumption on w implies that w' also passes the 2-dependence test for the code $D(\{f_i\}_{i=1}^m)$, so by Lemma 19 and Proposition 18 (applied to $D(\{f_i\}_{i=1}^t) = L(S, \{0,1\})$), we have $w' \in D(\{f_i\}_{i=1}^m)$. This means in particular that there is $s \in S$ such that $w_i = f_i(s)$ for all $i \in [t]$ (and even for all $i \in [m]$).

We finish by showing that $w_i = f_i(s)$ for all $i \in [n]$. Fix some $i \in [n]$. For every $a \in \Delta$, let $g_a : \Delta \to \{0,1\}$ be the function mapping a to 1 and all other elements of Δ to 0. There is some $j \in [t]$ such that $f_j = g_a \circ f_i$. The dependency between f_i and f_j and our assumption on w now imply that $w_j = g_a(w_i)$ (cf. Example 13). Since $w_j = f_j(s)$, we get that $g_a(w_i) = f_j(s) = g_a(f_i(s))$. By the definition of g_a, this means that $w_i = a$ if and only if $f_i(s) = a$. As this holds for every $a \in \Delta$, we conclude that $w_i = f_i(s)$. ◀

▶ Remark 20. By contrast to Theorem 17, the code $L(S, \{0,1\}) \subseteq \{0,1\}^n$ ($n = 2^{|S|}$) is not defined by 2-letter constaints when $|S| > 2$. To show this, it is enough to exhibit a word $w \in \{0,1\}^n$ that always passes the 2-dependence test and is not in $L(S, \{0,1\})$. Fix distinct $x_1, x_2, x_3 \in S$ and define the word $w \in \{0,1\}^n$ by $w_i = \mathrm{maj}(f_i(x_1), f_i(x_2), f_i(x_3))$ for all $i \in [n]$, where $\mathrm{maj}(-)$ is the majority function. The word w is not in $L(S, \{0,1\})$ because for every $s \in S$, there is $i \in [n]$ such that $f_i(s) \neq \mathrm{maj}(f_i(x_1), f_i(x_2), f_i(x_3)) = w_i$.

Let us now show that w always passes the 2-dependence test. Suppose that $f_i, f_j : S \to \{0,1\}$ are dependent. We need to show that $(w_i, w_j) \neq (a, b)$ for every $(a, b) \in \{0,1\}^2 - \mathrm{im}(f_i \times f_j)$. Fix such (a, b). If $w_i \neq a$, then it is clear that $(w_i, w_j) \neq (a, b)$. On the other hand, if $w_i = a$, then $\mathrm{maj}(f_i(x_1), f_i(x_2), f_i(x_3)) = a$, and so at least 2 of $f_i(x_1), f_i(x_2), f_i(x_3)$ equal a. Since $(a, b) \notin \mathrm{im}(f_i \times f_j)$, at least 2 of $f_j(x_1), f_j(x_2), f_j(x_3)$ are different from b. This means that $w_j \neq b$ and again we get $(w_i, w_j) \neq (a, b)$, as required.

When $|S| = 2$, the long code $L(S, \{0,1\})$ is $\{0101, 0011\} \subseteq \{0,1\}^4$ (up to equivalence), and it is easy to see that it is defined by 2-letter constaints.

6 Separable Testers

In order to apply Theorem 8 and change the alphabet of an LTC from Σ to Δ, we need to have a suitable function $f : \Sigma \to \Delta^k$ such that the tester T of our code is f-compatible (Definition 7). In this section, we give a necessary and sufficient condition on T to be compatible with *some* f, and moreover show that every tester can be replaced with one that satisfies our condition and has proportional soundness.

▶ **Definition 21.** *Let $C \subseteq \Sigma^n$ be a code and let $T = (T^{(i)}, (a_1^{(i)}, \ldots, a_q^{(i)}), p_i)_{i \in I}$ be a q-tester for C. Let Δ be another alphabet. We say that the tester T is* Δ-separable *if for every $i \in I$, there are functions $g_1, \ldots, g_q : \Sigma \to \Delta$ such that $T^{(i)} : \Sigma^q \to \{0,1\}$ factors via the function $g_1 \times \cdots \times g_q : \Sigma^q \to \Delta^q$, i.e., there is $g : \Delta^q \to \{0,1\}$ such that $T^{(i)} = g \circ (g_1 \times \cdots \times g_q)$.*

When C is an $\mathbb{F}$-code, T is $\mathbb{F}$-linear, and Δ is an $\mathbb{F}$-vector space, we say that T is linearly Δ-separable *if $g_1, \ldots, g_q$ can be taken to be $\mathbb{F}$-linear maps.*

▶ **Example 22.** Suppose that $C \subseteq \Sigma^n$ is an $\mathbb{F}$-code and $T = (T^{(i)}, (a_1^{(i)}, \ldots, a_q^{(i)}), p_i)_{i \in I}$ is q-tester such that each $T^{(i)}$ checks a single linear constraint involving at most q letters. Then T is linearly $\mathbb{F}$-separable. Indeed, by assumption, for every $i \in I$, there is a linear map $\varphi_i : \Sigma^q \to \mathbb{F}$ such that for every $w \in \Sigma^q$, we have $T^{(i)}(w) = 1$ if and only if $\varphi_i(w) = 0$. For such φ_i, there are linear functionals $g_1, \ldots, g_q : \Sigma \to \mathbb{F}$ such that $\varphi_i(w) = g_1(w_1) + \cdots + g_q(w_q)$. In particular, we can determine from $g_1(w_1), \ldots, g_q(w_q)$ the value of $T^{(i)}(w)$, meaning that $T^{(i)}$ factors via $g_1 \times \cdots \times g_q : \Sigma^q \to \mathbb{F}^q$.

We now show that Δ-separability is a necessary and sufficient condition for a tester T to be f-compatible with some $f : \Sigma \to \Delta^k$.

▶ **Proposition 23.** *Let $C \subseteq \Sigma^n$ be a code, let T be a q-tester for C, and let Δ be another alphabet.*

- **(i)** *If there is a function $f : \Sigma \to \Delta^k$ such that T is f-compatible, then T is Δ-separable.*
- **(ii)** *If T is Δ-separable, then there is $k \in \mathbb{N}$ and an injective function $f : \Sigma \to \Delta^k$ such that T is f-compatible. In fact, f can be constructed as follows: Let $k = |\Delta^\Sigma|$, let $f_1, \ldots, f_k$ be all the functions from Σ to Δ, and take $f(w) = (f_1(w), \ldots, f_k(w))$.*

Proof. Write $T = (T^{(i)}, (a_1^{(i)}, \dots, a_q^{(i)}), p_i)_{i \in I}$.

(i) For $j \in [k]$, let $f_j : \Sigma \to \Delta$ denote the j-th component of f. Let $i \in I$. Our assumption that T is f-compatible means that there exist a function $g : \Delta^q \to \{0,1\}$ and $b_1, \dots, b_q \in [k]$ such that for every $w_1, \dots, w_q \in \Sigma$, we have $T^{(i)}(w_1, \dots, w_q) = g(f_{b_1}(w_1), \dots, f_{b_q}(w_q))$. This is equivalent to saying that $T^{(i)} = g \circ (f_{b_1} \times \cdots \times f_{b_q})$, so $T^{(i)}$ factors via $f_{b_1} \times \cdots \times f_{b_q} : \Sigma^q \to \Delta^q$, and we have shown that T is Δ-separable.

(ii) Define f as in the statement. We need to show that T is f-compatible. Let $i \in I$. Then there are functions $g_1, \dots, g_q : \Sigma \to \Delta$ and a function $g : \Delta^q \to \{0,1\}$ such that $g \circ (g_1 \times \cdots \times g_q) = T^{(i)}$. By the construction of f, there are $b_1, \dots, b_q \in [k]$ such that $g_\ell = f_{b_\ell}$ for all $\ell \in [q]$. Then $g \circ (f_{b_1} \times \cdots \times f_{b_q}) = T^{(i)}$, or rather, $T^{(i)}(w_1, \dots, w_q) = g(f_{b_1}(w_1), \dots, f_{b_q}(w_q))$ for all $w \in \Sigma^q$. This is exactly what we need to show. ◀

▶ **Proposition 24.** *Let $C \subseteq \Sigma^n$ be an $\mathbb{F}$-code, let T be an $\mathbb{F}$-linear q-tester for T, and let Δ be a nonzero $\mathbb{F}$-vector space.*

(i) *If there is an $\mathbb{F}$-linear function $f : \Sigma \to \Delta^k$ such that T is f-compatible, then T is linearly Δ-separable.*

(ii) *If T is linearly Δ-separable, then there is $k \in \mathbb{N}$ and an injective function $f : \Sigma \to \Delta^k$ such that T is f-compatible. In fact, f can be constructed as follows: Let $k = |\operatorname{Hom}_{\mathbb{F}}(\Sigma, \Delta)|$, let $f_1, \dots, f_k$ be all the linear functions from Σ to Δ, and take $f(w) = (f_1(w), \dots, f_k(w))$.*

Proof. This is completely analogous to the proof of Proposition 23. ◀

▶ Remark 25. In Proposition 23(ii), the image of $f : \Sigma \to \Delta^k$ is the generalized long code $L(\Sigma, \Delta)$ of § 5.3. In Proposition 24(ii), the image of $f : \Sigma \to \Delta^k$ is the generalized Hadamard code $H(\Sigma, \Delta)$ of § 5.2.

Next, we show that every code admitting a q-tester also admits a Δ-separable q-tester with soundness that is proportional to that of the original tester. In particular, when given an LTC, we can always assume that its tester is Δ-separable.

▶ **Theorem 26.** *Let $C \subseteq \Sigma^n$ be a code, let T be a possibly adaptive q-tester for C with soundness $\mu > 0$, and let Δ be another alphabet. Then C admits a Δ-separable tester T' with soundness $\frac{\mu}{|\Sigma|^q}$.*

This proof is a mild modification of a standard trick to turn T into a non-adaptive q-tester. The idea is to guess what will be the q letters that the adaptive tester T would see. Based on this guess, we can determine which coordinates T would read, and then read letters in those positions. If the letters did match our guess, we return what T would return, and if our guess was wrong, we accept the word.

Proof. Let I be the set of random seeds for T (say, $I = \{0,1\}^r$ if T flips r coins). We define a non-adaptive tester T' as follows: Given $w \in \Sigma^n$, choose $u = (u_1, \dots, u_q) \in \Sigma^q$ and $i \in I$ uniformly at random. Let $a_1^{(i,u)}, \dots, a_q^{(i,u)}$ be the coordinates that T would have read on the seed i if the letters it would query in these positions were $u_1, u_2, \dots$, and let c be the output of T in that situation. Now query $w_{a_1^{(i,u)}}, \dots, w_{a_q^{(i,u)}}$. If this sequence is different from $u_1, \dots, u_q$, then w is accepted. Otherwise, T' returns c.

It is straightforward to see that $\Pr(T'(w)=0) \geq \frac{1}{|\Sigma^q|}\Pr(T(w)=0)$, so T' has soundness $\frac{\mu}{|\Sigma|^q}$. It remains to show that T' is Δ-separable. To that end, note that

$$T' = \left(T^{(i,u)}, (a_1^{(i,u)}, \ldots, a_q^{(i,u)}), \tfrac{1}{|I|\cdot|\Sigma|^q}\right)_{(i,u)\in I\times\Sigma^q},$$

where $T^{(i,u)} : \Sigma^q \to \{0,1\}$ a function that returns 1 for every $u' \in \Sigma^q - \{u\}$ and for u returns what T would have returned on the random seed i if it would read the letters $u_1, u_2, \ldots$ in positions $a_1^{(i,u)}, a_2^{(i,u)}, \ldots$.

Fix $(i,u) \in I \times \Sigma^q$. We need to show that $T^{(i,u)} = g \circ (g_1 \times \cdots \times g_q)$ for some $g_1, \ldots, g_q : \Sigma \to \Delta$ and $g : \Delta^q \to \{0,1\}$. Without loss of generality, we may assume that $0, 1 \in \Delta$. If $T^{(i,u)}(u) = 1$, then $T^{(i,u)}$ is identically 1 and we can take $g_1, \ldots, g_q, g$ to be the functions which map everything to 1. Otherwise, $T^{(i,u)}(u) = 0$ and $T^{(i,u)}(w) = 1$ for all $w \in \Sigma^q - \{u\}$. In this case, define $g_\ell : \Sigma \to \Delta$ by

$$g_\ell(w) = \begin{cases} 0 & w_\ell = u_\ell \\ 1 & w_\ell \neq u_\ell \end{cases}$$

and let $g : \Delta^q \to \{0,1\}$ be the function which maps $(0,\ldots,0)$ to 0 and all other elements to 1. It is straightforward to check that $T^{(i,u)} = g \circ (g_1 \times \cdots \times g_q)$, so T' is indeed Δ-separable. ◀

▶ **Theorem 27.** *Let $C \subseteq \Sigma^n$ be an $\mathbb{F}$-code, let T be an $\mathbb{F}$-linear q-tester for C with soundness $\mu > 0$, and let Δ be a nonzero $\mathbb{F}$-vector space. Then C admits a linearly Δ-separable tester T' with soundness $\left\lceil \frac{q \dim \Sigma}{\dim \Delta} \right\rceil^{-1} \mu$.*

Proof. Write $T = (T^{(i)}, (a_1^{(i)}, \ldots, a_q^{(i)}), p_i)_{i\in I}$ and put $m = \lceil \frac{q\dim\Sigma}{\dim\Delta} \rceil$. Then for every $i \in I$, there is a subspace $V_i \subseteq \Sigma^q$ such that $T^{(i)}(u) = 1$ if and only if $u \in V_i$ ($u \in \Sigma^q$). By the definition of m, we have $\dim \Delta^m = m \dim \Delta \geq \dim \Sigma^q$. Thus, there is a linear map $h_i : \Sigma^q \to \Delta^m$ with kernel V_i. For $j \in [m]$, denote by $h_{ij} : \Sigma^q \to \Delta$ the j-th component of h_i.

We now define an $\mathbb{F}$-linear q-tester T' as follows: Given $w \in \Sigma^n$, choose $i \in I$ according to p and $j \in [m]$ uniformly at random, and accept w if and only if $h_{ij}(w_{a_1^{(i)}}, \ldots, w_{a_q^{(i)}}) = 0$. It is clear that T' accepts every $w \in C$. Moreover, since $V_i = \ker h_i = \bigcap_{j=1}^m \ker h_{ij}$, whenever $T^{(i)}(w_{a_1^{(i)}}, \ldots, w_{a_q^{(i)}}) = 0$, there is at least one $j \in [m]$ such that $h_{ij}(w_{a_1^{(i)}}, \ldots, w_{a_q^{(i)}}) \neq 0$. This means that $\Pr(T'(w)=0) \geq \frac{1}{m}\Pr(T(w)=0)$, so T' has soundness $\frac{\mu}{m}$.

It remains to show that T' is linearly Δ-separable. To that end, observe that

$$T' = (T^{(i,j)}, (a_1^{(i,j)}, \ldots, a_q^{(i,j)}), p_{(i,j)})_{(i,j)\in I\times[m]},$$

where $p_{(i,j)} := \frac{p_i}{m}$, $a_\ell^{(i,j)} := a_\ell^{(i)}$ ($i \in I$, $j \in [m]$, $\ell \in [q]$), and $T^{(i,j)}(u)$ ($u \in \Sigma^q$) is defined to be 1 if $u \in \ker h_{ij}$ and 0 otherwise. Fix some $i \in I$ and $j \in [m]$. Then there are linear maps $g_1, \ldots, g_q : \Sigma \to \Delta$ such that $h_{ij}(u) = g_1(u_1) + \cdots + g_q(u_q)$ for all $u \in \Sigma^q$. This means that we can recover $T^{(i,j)}(u)$ from $g_1(u_1), \ldots, g_q(u_q)$, so $T^{(i,j)}$ factors via $g_1 \times \cdots \times g_q : \Sigma^q \to \Delta^q$. ◀

7 Alphabet Reduction for Good LTCs

We finally put the results of the previous sections together to prove our alphabet reduction results for good LTCs.

We begin with results about $\mathbb{F}$-codes. Theorem 3 is an immediate consequence of the following theorem applied with $(q,c) = (2,2)$ and $(q,c) = (3,1)$.

▶ **Theorem 28.** *Let $q \geq 2$ be an integer and let Δ be an $\mathbb{F}$-vector space of dimension $d > 0$. We further fix an auxiliary parameter $c \in \{1, \dots, d\}$ and require that $q \geq 3$ if $c = 1$. Suppose that $C \subseteq \Sigma^n$ is an $\mathbb{F}$-code with relative distance δ and rate r admitting an $\mathbb{F}$-linear q-query tester T with soundness $\mu > 0$. Then there exists an $\mathbb{F}$-code $C' \subseteq \Delta^{|\Sigma|^c n}$ with relative distance $(1 - |\mathbb{F}|^{-c})\delta$ and rate $\frac{1}{d} \cdot \frac{\dim \Sigma}{|\Sigma|^c} \cdot r$ admitting an $\mathbb{F}$-linear q-query tester T' with soundness*

$$\frac{c\mu\nu}{(q \dim \Sigma + c)(q|\Sigma|^c + \nu) + c\mu(1+\nu)},$$

where ν is the largest possible soundness of a q-query tester for the generalized Hadamard code $H(\Sigma, \mathbb{F}^c)$ (see § 5.2). Moreover, $\nu \geq |\Sigma|^{-2c}$ if $c > 1$ and $\nu \geq |\Sigma|^{-3}$ if $c = 1$.

Our requirement about q implies that the theorem cannot be applied with 1-dimensional Δ and $q = 2$.

Note that increasing the auxiliary parameter c decreases the rate of C', and also the ratio between the block-lengths of C and C'. On the other hand, we expect that increasing c would increase ν, and thus the soundness of T'. In particular, we believe that ν is much larger than the naive lower bound provided in the theorem. For example, when $q = 3$ and $c = 1$, we have $\nu \geq \frac{1}{6}$ by [20, Lem. B.2]; see also [21, Thm. 4.8] which shows that ν is close to 1 when $c = 1$.

Proof. Fix a c-dimensional subspace $\Delta' \subseteq \Delta$. By Theorem 27, we may assume that T is Δ'-separable at the cost of reducing the soundness from μ to $\mu' := \frac{c}{q \dim \Sigma + c}\mu$ (note that $\lceil \frac{q \dim \Sigma}{c} \rceil^{-1} \geq (\frac{q \dim \Sigma}{c} + 1)^{-1} = \frac{c}{q \dim \Sigma + c}$). Now, by Proposition 24(ii), there is an $\mathbb{F}$-linear injective function $f : \Sigma \to (\Delta')^k$ such that T is f-compatible. Moreover, by Remark 25, $D := \mathrm{im}(f)$ is the generalized Hadamard code $H(\Sigma, \Delta')$, and so $k = |\Sigma|^c$. By the definition of ν, the code $D \subseteq (\Delta')^k$ admits an $\mathbb{F}$-linear q-tester with soundness ν. Moreover, since $q \geq 2$ when $c > 1$ (resp. $q \geq 3$ with $c = 1$), ν is greater than or equal to the soundness of the 2-dependence (resp. 3-dependence) tester for $H(\Sigma, \Delta')$. The latter is positive by Theorem 16 (resp. the discussion in § 5.2), so $\nu \geq k^{-2} = |\Sigma|^{-2c}$ (resp. $\nu \geq k^{-3} = |\Sigma|^{-3}$).

Consider the code $C'' := C \circ_f D \subseteq (\Delta')^{kn}$. By Proposition 6, it has relative distance $\delta(H(\Sigma, \Delta'))\delta(C) = (1 - |\Delta'|^{-1})\delta$ and rate $r(H(\Sigma, \Delta'))r(C) = \frac{r \dim \Sigma}{c|\Sigma|^c}$, and by Theorem 8, it has an $\mathbb{F}$-linear q-tester with soundness

$$\mu'' := \frac{\mu'\nu}{qk + \mu' + \nu} = \frac{c\mu\nu}{(q \dim \Sigma + c)(q|\Sigma|^c + \nu) + c\mu}.$$

Finally, we take C' to be C'' viewed as a code with alphabet Δ (instead of Δ'). This multiplies the rate by $\frac{\dim \Delta'}{\dim \Delta} = \frac{c}{d}$, has no effect on the relative distance, and by Proposition 10, C' still admits an $\mathbb{F}$-linear q-tester with soundness $\frac{\mu''}{\mu'' + 1} = \frac{c\mu\nu}{(q \dim \Sigma + c)(q|\Sigma|^c + \nu) + c\mu(1+\nu)}$. ◀

Theorem 2 follows by applying following theorem with $(q, c) = (2, 3)$ and $(q, c) = (3, 2)$.

▶ **Theorem 29.** *Let $q \geq 2$ be an integer and let Δ be an alphabet with d elements. We further fix an auxiliary parameter $c \in \{2, \dots, d\}$ and require that $q \geq 3$ if $c = 2$. Suppose that $C \subseteq \Sigma^n$ is a code with relative distance δ and rate r admitting a possibly adaptive q-query tester T with soundness $\mu > 0$. Then there exists a code $C' \subseteq \Delta^{c^{|\Sigma|} n}$ with relative distance $(1 - c^{-1})\delta$ and rate $\frac{\log_d |\Sigma|}{c^{|\Sigma|}} \cdot r$ admitting a q-query tester T' with soundness*

$$\frac{\mu\nu}{|\Sigma|^q(qc^{|\Sigma|} + \nu) + \mu(\nu + 1)},$$

where ν is the largest possible soundness of a q-query tester for the generalized long code $L(\Sigma, \{1, \dots, c\})$ (see § 5.3). Moreover, $\nu \geq c^{-2|\Sigma|}$ if $c > 2$ and $\nu \geq 2^{-3|\Sigma|}$ if $c = 2$.

Our requirement about q implies that the theorem cannot be applied when $q = 2$ and $|\Delta| = 2$. As in Theorem 28, the parameter c allows a trade-off between code-theoretic properties of C' and the soundness of the tester T', where again, we believe that ν is much larger than the naive bounds given in the theorem.

Proof. The proof strategy is similar to that of proving Theorem 28.

Let Δ' be a subset of Δ having c elements. We use Theorem 26 to replace T with a Δ'-separable tester with soundness $\mu' := \frac{\mu}{|\Sigma|^q}$. By Proposition 23(ii), there is an injective function $f : \Sigma \to (\Delta')^k$ such that T is f-compatible, and by Remark 25, it can be chosen so that $D := \mathrm{im}(f)$ is the generalized long code $L(\Sigma, \Delta')$. In particular, $k = c^{|\Sigma|}$. By Theorem 17 (resp. Proposition 18), when $c > 2$ (resp. $c = 2$), the 2-dependence (resp. 3-dependence) for D has positive soundness. As a result, when $c > 2$ (resp. $c = 2$), D has a q-tester with soundness $\nu \geq k^{-2}$ (resp. $\nu \geq k^{-3}$).

The proof is finished as in Theorem 28 by taking C' to be the code $C \circ_f D \subseteq (\Delta')^{kn}$ and viewing it as a code inside Δ^{kn}. ◀

Next, we note that the parameters of the code C' and the tester T' promised by Theorem 29 can be improved if C is assumed to be an $\mathbb{F}_2$-code and T is $\mathbb{F}_2$-linear. This is the content of the following theorem, which we state only in the case $|\Delta| = 3$ for simplicity.

▶ **Theorem 30.** *Suppose that $C \subseteq \Sigma^n$ is an $\mathbb{F}_2$-code with relative distance δ and rate r admitting an $\mathbb{F}_2$-linear q-query tester T with soundness $\mu > 0$. Then there exists a code $C' \subseteq \{0,1,2\}^{(2|\Sigma|^2+|\Sigma|)n}$ with relative distance $\frac{\delta}{2|\Sigma|^2+|\Sigma|}$ and rate $\frac{\log_3 |\Sigma|}{2|\Sigma|^2+|\Sigma|} \cdot r$ admitting a q-query tester T' with soundness*

$$\frac{\mu\nu}{q \dim \Sigma \cdot (2q|\Sigma|^2 + q|\Sigma| + \nu) + \mu},$$

where $\nu \geq (2|\Sigma|^2 + |\Sigma|)^{-2}$ is the maximal possible soundness of a 2-query tester for a certain code $D \subseteq \{0,1,2\}^{2|\Sigma|^2+|\Sigma|}$ specified in the proof.

We will need the following lemma, whose straightforward proof may be found in the extended version of this paper [10, Lem. 7.4].

▶ **Lemma 31.** *Let $C \subseteq \Sigma^n$ be a code with a q-tester T. Let Δ be an alphabet and let Δ' be another alphabet containing Δ. Let $f_1, \ldots, f_k : \Sigma \to \Delta$ and let $f'_1, \ldots, f'_{k'} : \Sigma \to \Delta'$. Define $f : \Sigma \to \Delta^k$ by $f(a) = (f_1(a), \ldots, f_k(a))$ and define $f' : \Sigma \to (\Delta')^{k'}$ similarly. Suppose that for every $i \in [k]$, there is $i' \in [k']$ such that $f'_{i'} = f_i$ as functions from Σ to Δ'. If T is f-compatible, then it is f'-compatible.*

Proof of Theorem 30. By Theorem 27, we may replace T with a linearly $\mathbb{F}_2$-separable tester at the cost of reducing the soundness to $\mu' := \frac{\mu}{q \dim \Sigma}$.

Let $g_1, \ldots, g_t$ be all the linear $\mathbb{F}_2$-functions from Σ to $\mathbb{F}_2$ (so $t = |\Sigma|$), and let $g : \Sigma \to \{0,1\}^t$ be defined by $g(a) = (g_1(a), \ldots, g_t(a))$. By Proposition 24(ii), T is g-compatible. Now define $f_1, \ldots, f_\ell : \Sigma \to \{0,1,2\}$ as in Lemma 19 (with ℓ in place of n). By construction, $\ell \leq t + 2t^2 =: k$, and if $\ell < t + 2t^2$ we define $f_{\ell+1}, \ldots, f_k : \Sigma \to \{0,1,2\}$ to be identically 0. Now define $f : \Sigma \to \{0,1,2\}^k$ similarly to g. By Lemma 31, T is also f-compatible.

Put $D = D(\{f_i\}_{i=1}^k) \subseteq \{0,1,2\}^k$ and let $C' = C \circ_f D \subseteq \{0,1,2\}^{kn}$. By Lemma 19, $D \subseteq \{0,1,2\}^k$ has a 2-query tester with soundness $k^{-2} = (t + 2t^2)^{-2}$, so $\nu \geq (t + 2t^2)^{-2}$. Furthermore, by Theorem 8, $C' \subseteq \{0,1,2\}^{kn}$ has a 2-query tester with soundness

$$\frac{\mu'\nu}{qk + \mu' + \nu} = \frac{\mu\nu}{q \dim \Sigma \cdot (2q|\Sigma|^2 + q|\Sigma| + \nu) + \mu}.$$

To finish, note that by Proposition 6, $\delta(C') \geq \delta(D)\delta(C) \geq \frac{\delta}{k}$ and $r(C') = r(D)r(C) = \frac{\log_3 |\Sigma|}{k} \cdot r$. ◀

▶ Remark 32. By tracing the proofs of Theorems 28, 29, 30, one sees that if the tester T of C has randomness complexity R, then the promised tester T' for C' has randomness complexity $\max\{R, \log_2 n\} + O(1)$, where the $O(1)$-factor depends on μ, $|\Sigma|$, $|\Delta|$.

References

1 M. Bellare, D. Coppersmith, J. Hastad, M. Kiwi, and M. Sudan. Linearity testing in characteristic two. *IEEE Transactions on Information Theory*, 42(6):1781–1795, 1996. doi:10.1109/18.556674.

2 Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, PCPs, and nonapproximability—towards tight results. *SIAM Journal on Computing*, 27(3):804–915, 1998. doi:10.1137/S0097539796302531.

3 Eli Ben-Sasson, Oded Goldreich, and Madhu Sudan. Bounds on 2-query codeword testing. In *Approximation, randomization, and combinatorial optimization*, volume 2764 of *Lecture Notes in Comput. Sci.*, pages 216–227. Springer, Berlin, 2003. doi:10.1007/978-3-540-45198-3_19.

4 Eli Ben-Sasson, Venkatesan Guruswami, Tali Kaufman, Madhu Sudan, and Michael Viderman. Locally testable codes require redundant testers. *SIAM J. Comput.*, 39(7):3230–3247, July 2010. doi:10.1137/090779875.

5 Eli Ben-Sasson and Madhu Sudan. Robust locally testable codes and products of codes. *Random Structures Algorithms*, 28(4):387–402, 2006. doi:10.1002/rsa.20120.

6 Eli Ben-Sasson and Michael Viderman. Towards lower bounds on locally testable codes via density arguments. *Comput. Complexity*, 21(2):267–309, 2012. doi:10.1007/s00037-012-0042-8.

7 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549–595, 1993. doi:10.1016/0022-0000(93)90044-W.

8 Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):Art. 12, 44, 2007. doi:10.1145/1236457.1236459.

9 Irit Dinur, Shai Evra, Ron Livne, Alexander Lubotzky, and Shahar Mozes. Locally testable codes with constant rate, distance, and locality. In *STOC '22—Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 357–374. ACM, New York, 2022.

10 Uriya First and Stav Lazarovici. Good locally testable codes with small alphabet and small query size. *arXiv preprint*, 2025. doi:10.48550/arXiv.2512.16082.

11 Uriya A. First and Tali Kaufman. Cosystolic expansion of sheaves on posets with applications to good 2-query locally testable codes and lifted codes. In *STOC'24—Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1446–1457. ACM, New York, 2024. doi:10.1145/3618260.3649625.

12 Uriya A. First and Tali Kaufman. Cosystolic expansion of sheaves on posets with applications to good 2-query locally testable codes and lifted codes. *arXiv preprint*, 2024. doi:10.48550/arXiv.2403.19388.

13 Katalin Friedl and Madhu Sudan. Some improvements to total degree tests. In *Third Israel Symposium on the Theory of Computing and Systems (Tel Aviv, 1995)*, pages 190–198. IEEE Comput. Soc. Press, Los Alamitos, CA, 1995. doi:10.1109/ISTCS.1995.377032.

14 Oded Goldreich. Short locally testable codes and proofs. In *Studies in complexity and cryptography*, volume 6650 of *Lecture Notes in Comput. Sci.*, pages 333–372. Springer, Heidelberg, 2011. doi:10.1007/978-3-642-22670-0_25.

15 Oded Goldreich and Madhu Sudan. Locally testable codes and PCPs of almost-linear length. *J. ACM*, 53(4):558–655, 2006. doi:10.1145/1162349.1162351.

16 Sivakanth Gopi, Swastik Kopparty, Rafael Oliveira, Noga Ron-Zewi, and Shubhangi Saraf. Locally testable and locally correctable codes approaching the Gilbert-Varshamov bound. *IEEE Trans. Inform. Theory*, 64(8):5813–5831, 2018. doi:10.1109/TIT.2018.2809788.

17 Venkatesan Guruswami, Atri Rudra, and Madhu Sudan. Essential coding theory, 2025. Preprint. Aug 26, 2025 version. Available at `https://cse.buffalo.edu/faculty/atri/courses/coding-theory/book/`.

18 Gillat Kol and Ran Raz. Bounds on 2-query locally testable codes with affine tests. *Inform. Process. Lett.*, 116(8):521–525, 2016. `doi:10.1016/j.ipl.2016.03.008`.

19 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):Art. 11, 42, 2017. `doi:10.1145/3051093`.

20 Or Meir. Combinatorial construction of locally testable codes. *SIAM J. Comput.*, 39(2):491–544, 2009. `doi:10.1137/080729967`.

21 Tushant Mittal and Sourya Roy. A general framework for low soundness homomorphism testing. In *17th Innovations in Theoretical Computer Science Conference*, volume 362 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 103, 18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. `doi:10.4230/lipics.itcs.2026.103`.

22 Pavel Panteleev and Gleb Kalachev. Asymptotically good quantum and locally testable classical LDPC codes. In *STOC '22 – Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 375–388. ACM, New York, 2022.

23 Alexander Polishchuk and Daniel A. Spielman. Nearly-linear size holographic proofs. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '94, pages 194–203, New York, NY, USA, 1994. Association for Computing Machinery. `doi:10.1145/195058.195132`.

24 Ron Roth. *Introduction to Coding Theory*. Cambridge University Press, 2006.

25 Michael Viderman. Strong LTCs with inverse polylogarithmic rate and soundness. In *2013 IEEE Conference on Computational Complexity – CCC 2013*, pages 255–265. IEEE Computer Soc., Los Alamitos, CA, 2013. `doi:10.1109/CCC.2013.34`.