Robust Locally Testable Codes and Products of Codes

Eli Ben-Sasson[1] and Madhu Sudan[2]

[1] Radcliffe Institute for Advanced Study, 34 Concord Avenue, Cambridge, MA 02138, USA. eli@eecs.harvard.edu.

[2] MIT and Radcliffe IAS, The Stata Center Rm. G640, 32 Vassar Street, Cambridge, MA 02139, USA. madhu@mit.edu.

Abstract. We continue the investigation of locally testable codes, i.e., error-correcting codes for whom membership of a given word in the code can be tested probabilistically by examining it in very few locations. We give two general results on local testability: First, motivated by the recently proposed notion of *robust* probabilistically checkable proofs, we introduce the notion of *robust* local testability of codes. We relate this notion to a product of codes introduced by Tanner, and show a very simple composition lemma for this notion. Next, we show that codes built by tensor products can be tested robustly and somewhat locally, by applying a variant of a test and proof technique introduced by Raz and Safra in the context of testing low-degree multivariate polynomials (which are a special case of tensor codes).

Combining these two results gives us a generic construction of codes of inverse polynomial rate, that are testable with poly-logarithmically many queries. We note these locally testable tensor codes can be obtained from *any* linear error correcting code with good distance. Previous results on local testability, albeit much stronger quantitatively, rely heavily on algebraic properties of the underlying codes.

1 Introduction

Locally testable codes (LTCs) are error-correcting codes that admit highly efficient probabilistic tests of membership. Specifically, an LTC has a tester that makes a small number of oracle accesses into an oracle representing a given word w, accepts if w is a codeword, and rejects with constant probability if w is far from every codeword. LTCs are combinatorial counterparts of probabilistically checkable proofs (PCPs), and were defined in [18, 24, 2], and their study was revived in [20].

Constructions of locally testable codes typically come in two stages. The first stage is algebraic and gives local tests for algebraic codes, usually based on multivariate polynomials. This is based on a rich collection of results on “linearity testing” or “low-degree testing” [1, 3–9, 13, 14, 16–18, 20, 22, 24]. This first stage either yielded codes of poor rate (mapping k information symbols to codewords of length $\exp(k)$) as in [14], or yielded codes over large alphabets as in [24].

To reduce the alphabet size, a second stage of "composition" is then applied. In particular, this is done in [20, 13, 11] to get code mapping k information bits to codewords of length $k^{1+o(1)}$, over the *binary alphabet*. This composition follows the lines of PCP composition introduced in [4], but turns out to be fairly complicated, and in most cases, even more intricate than PCP composition. The one exception is in [20, Section 3], where the composition is simple, but based on very specific properties of the codes used. Thus while the resulting constructions are surprisingly strong, the proof techniques are somewhat complex.

In this paper, we search for simple and general results related to local testing. A generic (non-algebraic) analysis of low-degree tests appears in [19], and a similar approach to PCPs appears in [15]. Specifically, we search for generic (non-algebraic) ways of getting codes, possibly over large alphabets, that can be tested by relatively local tests, as a substitute for algebraic ways. And we look for simpler composition lemmas. We make some progress in both directions. We show that the "tensor product" operation, a classical operation that takes two codes and produces a new one, when applied to linear codes gives codes that are somewhat locally testable (See Theorem 1). To simplify the second stage, we strengthen the notion of local testability to a "robust" one. This step is motivated by an analogous step taken for PCPs in [11], but is naturally formulated in our case using the "Tanner Product" for codes [26]. Roughly speaking, a "big" Tanner Product code of block-length n is defined by a "small" code of block-length $n' = o(n)$ and a collection of subsets $S_1, \ldots, S_m \subset [n]$, each of size n'. A word is in the big code if and only if its projection to every subset S_i is a word of the small code. Tanner Product codes have a natural local test associated with them: to test if a word w is a codeword of the big code, pick a random subset S_j and verify that w restricted to S_j is a codeword of the small code. The normal soundness condition would expect that if w is far from every codeword, then for a constant fraction of such restrictions, w restricted to S_j is not a codeword of the small code. Now the notion of robust soundness strengthens this condition further by expecting that if w is far from every codeword, then many (or most) projections actually lead to words that are *far* from codewords of the small code. In other words, a code is robust if global distance (from the large code) translates into (average) local distance (from the small code). A simple, yet crucial observation is that robust codes compose naturally. Namely, if the small code is itself locally testable by a robust test (with respect to a tiny code, of block-length $o(n')$), then distance from the large code (of block-length n) translates to distance from the tiny code, thus reducing query complexity while maintaining soundness. By viewing a tensor product as a robust Tanner product code, we show that a $(\log N/\log\log N)$-wise tensor product of *any* linear code of length $n = \operatorname{poly}\log N$ and relative distance $1 - \frac{1}{\log N} = 1 - \frac{1}{n^{\epsilon}}$, which yields a code of length N and polynomial rate, is testable with $\operatorname{poly}(\log N)$ queries (Theorem 2). Once again, while stronger theorems than the above have been known since [6], the generic nature of the result above might shed further light on the notion of local testability.

Organization. We give formal definitions and mention our main theorems in Section 2. In Section 3 we analyze the basic tester for tensor product codes. Finally in Section 4 we describe our composition and analyze some tests based on our composition lemma.

2 Definitions and Main Results

Throughout this paper Σ will denote a finite alphabet, and in fact a finite field. For positive integer n, let $[n]$ denote the set $\{1, \ldots, n\}$. For a sequence $x \in \Sigma^n$ and $i \in [n]$, we will let x_i denote the ith element of the sequence. The Hamming distance between strings $x, y \in \Sigma^n$, denoted $\Delta(x, y)$, is the number of $i \in [n]$ such that $x_i \neq y_i$. The relative distance between $x, y \in \Sigma^n$, denoted $\delta(x, y)$, is the ratio $\Delta(x, y)/n$.

A code C of length n over Σ is a subset of Σ^n. Elements of C are referred to as codewords. When Σ is a field, one may think of Σ^n as a vector space. If C is a linear subspace of the vector space Σ^n, then C is called a linear code. The crucial parameters of a code, in addition to its length and the alphabet, are its dimension (or information length) and its distance, given by $\Delta(C) = \min_{x \neq y \in C}\{\Delta(x, y)\}$. A linear code of dimension k, length n, distance d over the alphabet Σ is denoted an $[n, k, d]_\Sigma$ code. For a word $r \in \Sigma^n$ and a code C, we let $\delta_C(r) = \min_{x \in C}\{\delta(r, x)\}$. We say r is δ'-proximate to C (δ'-far from C, respectively) if $\delta_C(r) \geq \delta'$ ($\delta_C(r) \geq \delta'$, respectively).

Throughout this paper, we will be working with infinite families of codes, where their performance will be measured as a function of their length.

Definition 1 (Tester). *A tester T with query complexity $q(\cdot)$ is a probabilistic oracle machine that when given oracle access to a string $r \in \Sigma^n$, makes $q(n)$ queries to the oracle for r and returns an accept/reject verdict. We say that T tests a code C if whenever $r \in C$, T accepts with probability one; and when $r \notin C$, the tester rejects with probability at least $\delta_C(r)/2$. A code C is said to be locally testable with $q(n)$ queries if there is a tester for C with query complexity $q(n)$.*

When referring to oracles representing vectors in Σ^n, we emphasize the queries by denoting the response of the ith query by $r[i]$, as opposed to r_i. Through this paper we consider only non-adaptive testers, i.e., testers that use their internal randomness R to generate q queries $i_1, \ldots, i_q \in [n]$ and a predicate $P : \Sigma^q \rightarrow \{0, 1\}$ and accept iff $P(r[i_1], \ldots, r[i_q]) = 1$.

Our next definition is based on the notion of Robust PCP verifiers introduced by [11]. We need some terminology first.

Note that a tester T has two inputs: an oracle for a received vector r, and a random string s. On input the string s the tester generates queries $i_1, \ldots, i_q \in [n]$ and fixes circuit $C = C_s$ and accepts if $C(r[i_1], \ldots, r[i_q]) = 1$. For oracle r and random string s, define the robustness of the tester T on r, s, denoted $\rho^T(r, s)$, to be the minimum, over strings x satisfying $C(x) = 1$, of relative distance of $\langle r[i_1], \ldots, r[i_q]\rangle$ from x. We refer to the quantity $\rho^T(r) \stackrel{\text{def}}{=} \mathbf{E}_s[\rho^T(r, s)]$ as the expected robustness of T on r. When T is clear from context, we skip the superscript.

Definition 2 (Robust Tester). *A tester T is said to be c-robust for a code C if for every $r \in C$, the tester accepts w.p. one, and for every $r \in \Sigma^n$, $\delta_C(r) \leq c \cdot \rho^T(r)$.*

Having a robust tester for a code C δ implies the existence of a tester for C, as illustrated by the following proposition. It's proof appears in the full version of the paper [12].

Proposition 1. *If a code C has a c-robust tester T for C making q queries, then it is locally testable with $O(c \cdot q)$ queries.*

The main results of this paper focus on robust local testability of certain codes. For the first result, we need to describe the tensor product of codes.

Tensor Products and Local Tests Recall that an $[n, k, d]_\Sigma$ linear code C may be represented by a $k \times n$ matrix M over Σ (so that $C = \{xM | x \in \Sigma^k\}$). Such a matrix M is called a generator of C. Given an $[n_1, k_1, d_1]_\Sigma$ code C_1 with generator M_1 and an $[n_2, k_2, d_2]_\Sigma$ code C_2 with generator M_2, their tensor product (cf. [21], [25, Lecture 6, Section 2.4]), denoted $C_1 \otimes C_2 \subseteq \Sigma^{n_2 \times n_1}$, is the code whose codewords may be viewed as $n_2 \times n_1$ matrices given explicitly by the set $\{M_2^T X M_1 | X \in \Sigma^{k_2 \times k_1}\}$. It is well-known that $C_1 \otimes C_2$ is an $[n_1 n_2, k_1 k_2, d_1 d_2]_\Sigma$ code.

Tensor product codes are interesting to us in that they are a generic construction of codes with "non-trivially" local redundancy. To elaborate, every linear code of dimension k does have redundancies of size $O(k)$, i.e., there exist subsets of $t = O(k)$ coordinates where the code does not take all possible Σ^t possible values. But such redundancies are not useful for constructing local tests; and unfortunately generic codes of length n and dimension k may not have any redundancies of length $o(k)$. However, tensor product codes are different in that the tensor product of an $[n, k, d]_\Sigma$ code C with itself leads to a code of dimension k^2 which is much larger than the size of redundancies which are $O(k)$-long, as asserted by the following proposition.

Proposition 2. *A matrix $r \in \Sigma^{n_2 \times n_1}$ is a codeword of $C_1 \otimes C_2$ if and only if every row is a codeword of C_1 and every column is a codeword of C_2.*

In addition to being non-trivially local, the constraints enumerated above are also redundant, in that it suffices to insist that all columns are codewords of C_2 and only k_2 (prespecified) rows are codewords of C_1. Thus the insistence that other rows ought to be codewords of C_1 is redundant, and leads to the hope that the tests may be somewhat robust. Indeed we may hope that the following might be a robust test for $C_1 \otimes C_2$.

> **Product Tester:** Pick $b \in \{1, 2\}$ at random and $i \in [n_b]$ at random. Verify that r with bth coordinate restricted to i is a codeword of C_{3-b}.

While it is possible to show that the above is a reasonable tester for $C_1 \otimes C_2$, it remains open if the above is a robust tester for $C_1 \otimes C_2$. (Note that the query complexity of the test is $\max\{n_1, n_2\}$, which is quite high. However if the test were robust, there would be ways of reducing this query complexity in many cases, as we will see later.)

Instead, we consider higher products of codes, and give a tester based on an idea from the work of Raz and Safra [23]. Specifically, we let C^m denote the code $\underbrace{C \otimes \cdots \otimes C}_{m}$. We consider the following test for this code:

> m**-Product Tester:** Pick $b \in [m]$ and $i \in [n]$ independently and uniformly at random. Verify that r with bth coordinate restricted to i is a codeword of C^{m-1}.

Note that this tester makes $N^{1-\frac{1}{m}}$ queries to test a code of length $N = n^m$. So its query complexity gets worse as m increases. However, we are only interested in the performance of the test for small m (specifically $m = 3, 4$). We show that the test is a robust tester for C^m for every $m \geq 3$. Specifically, we show

Theorem 1. *For a positive integer m and $[n, k, d]_\Sigma$-code C, such that $\left(\frac{d-1}{n}\right)^m \geq \frac{7}{8}$, m-Product Tester is 2^{16}-robust for C^m.*

This theorem is proven in Section 3. Note that the robustness is a constant, and the theorem only needs the fractional distance of C to be sufficiently large as a function of m. In particular a fractional distance of $1 - \frac{1}{O(m)}$ suffices. Note that such a restriction is needed even to get the fractional distance of C^m to be constant.

The tester however makes a lot of queries, and this might seem to make this result uninteresting (and indeed one doesn't have to work so hard to get a non-robust tester with such query complexity). However, as we note next, the query complexity of robust testers can be reduced significantly under some circumstances. To describe this we need to revisit a construction of codes introduced by Tanner [26].

Tanner Products and Robust Testing For integers (n, m, t) an (n, m, t)-ordered bipartite graph is given by n left vertices $[n]$, and m right vertices, where each right vertex has degree t and the neighborhood of a right vertex $j \in [m]$ is ordered and given by a sequence $\ell_j = \langle \ell_{j,1}, \ldots, \ell_{j,t} \rangle$ with $\ell_{j,i} \in [n]$.

A Tanner Product Code (TPC), is specified by an $[n, m, t]$ ordered bipartite graph G and a code $C_{\text{small}} \subseteq \Sigma^t$. The product code, denoted $\text{TPC}(G = \{\ell_1, \ldots, \ell_m\}, C_{\text{small}}) \subseteq \Sigma^n$, is the set

$$\{r \in \Sigma^n \mid r|_{\ell_j} \overset{\text{def}}{=} \langle r_{\ell_{j,1}}, \ldots, r_{\ell_{j,t}} \rangle \in C_{\text{small}},\ \forall j \in [m]\}.$$

Notice that the Tanner Product naturally suggests a test for a code. "Pick a random right vertex $j \in [m]$ and verify that $r|_{\ell_j} \in C_{\text{small}}$." Associating this test with such a pair (G, C_{small}), we say that the pair is c-robust if the associated test is a c-robust tester for $\text{TPC}(G, C_{\text{small}})$.

The importance of this representation of tests comes from the composability of robust tests coming from Tanner Product Codes. Suppose (G, C_{small}) is c-robust and C_{small} is itself a Tanner Product Code, $\text{TPC}(G', C_{\text{small}}{}')$ where G' is an (d, m', t')-ordered bipartite graph and $(G', C_{\text{small}}{}')$ is c'-robust. Then $\text{TPC}(G, C_{\text{small}})$ has an $c \cdot c'$-robust tester that makes only t' queries. (This fact is completely straightforward and proven in Lemma 3.)

This composition is especially useful in the context of tensor product codes. For instance, the tester for C^4 is of the form (G, C^3), while C^3 has a robust tester of the form (G', C^2). Putting them together gives a tester for C^4, where the tests verify appropriate projections are codewords of C^2. The test itself is not surprising, however the ease with which the analysis follows is nice. (See Lemma 4.) Now the generality of the tensor product tester comes in handy as we let C itself be C'^2 to see that we are now testing C'^8 where tests verify some projections are codewords of C'^4. Again composition allows us to reduce this to a C'^2-test. Carrying on this way we see that we can test any code of the form C^{2^t} by verifying certain projections are codewords of C^2. This leads to a simple proof of the following theorem about the testability of tensor product codes.

Theorem 2. *Let $\{C_i\}_i$ be any infinite family of codes with C_i being an $[n_i, k_i, d_i]_{\Sigma_i}$ code, with $n_i = p(k_i)$ for some polynomial $p(\cdot)$. Further, let t_i be a sequence of integers such that $m_i = 2^{t_i}$ satisfies $d_i/n_i \geq 1 - \frac{1}{7m_i}$. Then the sequence of codes $\{C_i' = C_i^{m_i}\}_i$ is a sequence of codes of inverse polynomial rate and constant relative distance that is locally testable with polylogarithmic number of queries.*

This theorem is proven in Section 4. We remark that it is possible to get code families C_i such as above using Reed-Solomon codes, as well as algebraic-geometric codes.

3 Testing Tensor Product Codes

Recall that in this section we wish to prove Theorem 1. We first reformulate this theorem in the language of Tanner products.

Let G_m^n denote the graph that corresponds to the tests of C^m by the m-Product Tester, where $C \subseteq \Sigma^n$. Namely G_m^n has n^m left vertices labelled by elements of $[n]^m$. It has $m \cdot n$ right vertices labelled (b, i) with $b \in [m]$ and $i \in [n]$. Vertex (b, i) is adjacent to all vertices $(i_1, \ldots, i_m)$ such that $i_b = i$. The statement of Theorem 1 is equivalent to the statement that (G_m^n, C^{m-1}) is 2^{16}-robust, provided $\left(\frac{d-1}{n}\right)^m \geq \frac{7}{8}$. The completeness of the theorem follows from Proposition 2, which implies $C^m = \text{TPC}(G_m^n, C^{m-1})$. For the soundness, we first introduce some notation.

Consider the code $C_1 \otimes \cdots \otimes C_m$, where $C_i = [n_i, k_i, d_i]_\Sigma$ code. Notice that codewords of this code lie in $\Sigma^{n_1 \times \cdots \times n_m}$. The coordinates of strings in $\Sigma^{n_1 \times \cdots \times n_m}$ are themselves m-dimensional vectors over the integers (from $[n_1] \times \cdots \times [n_m]$). For $r \in \Sigma^{n_1 \times \cdots \times n_m}$ and $i_1, \ldots, i_m$ with $i_j \in [n_j]$, let $r[i_1, \ldots, i_m]$

denote the $\langle i_1, \ldots, i_m \rangle$-th coordinate of r. For $b \in [m]$, and $i \in [n_b]$, let $r_{b,i} \in \Sigma^{n_1 \times \cdots \times n_{b-1} \times n_{b+1} \times \cdots \times n_m}$ be the vector obtained by projecting r to coordinates whose bth coordinate is i, i.e., $r_{b,i}[i_1, \ldots, i_{m-1}] = r[i_1, \ldots, i_{b-1}, i, i_b, \ldots, i_{m-1}]$.

The following simple property about tensor product codes will be needed in our proof. It's proof appears in the full version of the paper [12].

Proposition 3. *For $b \in \{1, \ldots, m\}$ let C_b be an $[n_b, k_b, d_b]_\Sigma$ code, and let I_b be a set of cardinality at least $n_b - d_b + 1$. Let C'_b be the code obtained by the projection of C_b to I_b. Then every codeword c' of $C'_1 \otimes \cdots \otimes C'_m$ can be extended to a codeword c of $C_1 \otimes \cdots \otimes C_m$.*

Recall that the m-Product tester picks a random $b \in [m]$ and $i \in [n]$ and verifies that $r_{b,i} \in C^{m-1}$. The robustness of this tester for oracle r on random string $s = (b, i)$ is given by $\rho(r, (b, i)) = \delta_{C^{m-1}}(r_{b,i})$, and its expected robustness is given by $\rho(r) = \mathbf{E}_{b,i}[\delta_{C^{m-1}}(r_{b,i})]$. We wish to show for every r that $\delta_{C^m}(r) \leq 2^{16} \cdot \rho(r)$.

We start by first getting a crude upper bound on the proximity of r to C^m and then we use the crude bound to get a tighter relationship. To get the crude bound, we first partition the random strings into two classes: those for which the robustness $\rho(r, (b, i))$ is large, and those for which it is small. More precisely, for $r \in \Sigma^{n^m}$ and a threshold $\tau \in [0, 1]$, define the τ-soundness-error of r to be the probability that $\delta_{C^{m-1}}(r_{b,i}) > \tau$, when $b \in [m]$ and $i \in [n]$ are chosen uniformly and independently. Note that the $\sqrt{\rho}$-soundness error of r is at most $\sqrt{\rho}$ for $\rho = \rho(r)$. We start by showing that r is $O(\tau + \epsilon)$-close (and thus also $O(\sqrt{\rho})$-close) to some codeword of C^m.

Lemma 1. *If the τ-soundness-error of r is ϵ for $\tau + 2\epsilon \leq \frac{1}{12} \cdot \left(\frac{d-1}{n}\right)^m$, then $\delta_{C^m}(r) \leq 16 \cdot \left(\frac{n}{d}\right)^{m-1} \cdot (\tau + \epsilon)$.*

Proof. For every $i \in [n]$ and $b \in [m]$, fix $c_{b,i}$ to be a closest codeword from C^{m-1} to $r_{b,i}$. We follow the proof outline of Raz & Safra [23] which when adapted to our context goes as follows: (1) Given a vector r and an assignment of codewords $c_{b,i} \in C^{m-1}$, we define an "inconsistency" graph G. (Note that this graph is *not* the same as the graph G^n_m that defines the test being analysed. In particular G is related to the word r being tested.) (2) We show that the existence of a large independent set in this graph G implies the proximity of r to a codeword of C^m (i.e., $\delta_{C^m}(r)$ is small). (3) We show that this inconsistency graph is sparse if the τ-soundness-error is small. (4) We show that the distance of C forces the graph to be special in that every edge is incident to at least one vertex whose degree is large.

Definition of G. The vertices of G are indexed by pairs (b, i) with $b \in [m]$ and $i \in [n]$. Vertex (b_1, i_1) is adjacent to (b_2, i_2) if *at least* one of the following conditions hold:

1. $\delta_{C^{m-1}}(r_{b_1,i_1}) > \tau$.
2. $\delta_{C^{m-1}}(r_{b_2,i_2}) > \tau$.

3. $b_1 \neq b_2$ and c_{b_1,i_1} and c_{b_2,i_2} are inconsistent, i.e., there exist $j = \langle j_1, \ldots, j_m \rangle \in [n]^m$, with $j_{b_1} = i_1$ and $j_{b_2} = i_2$ such that $c_{b_1,i_1}[j^{(1)}] \neq c_{b_2,i_2}[j^{(2)}]$, where $j^{(c)} \in [n]^{m-1}$ is the vector j with its b_cth coordinate deleted.

Independent sets of G and proximity of r. It is clear that G has mn vertices. We claim next that if G has an independent set I of size at least $m(n-d)+d+1$ then r has distance at most $1 - (|I|/(mn))(1-\tau)$ to C^m.

Consider an independent set $I = I_1 \cup \cdots \cup I_m$ in G with I_b of size n_b being the set of vertices of the form $(b,i), i \in [n]$. W.l.o.g. assume $n_1 \geq \cdots \geq n_m$. Then, we have $n_1, n_2 > n - d$ (or else even if $n_1 = n$ and $n_2 = n - d$ we'd only have $\sum_b n_b \leq n + (m-1)(n-d)$). We consider the partial vector $r' \in \Sigma^{I_1 \times n \times \cdots \times n}$ defined as $r'[i, j_2, \ldots, j_m] = c_{1,i}[j_2, \ldots, j_m]$ for $i \in I_1$, and $j_2, \ldots, j_m \in [n]$. We show that r' can be extended into a codeword of C^m and that the extended word is close to r and this will give the claim.

First, we show that any extension of r' is close to r: This is straightforward since on each coordinate $i \in I_1$, we have r agrees with r' on $1-\tau$ fraction of the points. Furthermore I_i/n is at least $|I|/(mn)$ (since n_1 is the largest). So we have that r' is at most $1 - (|I|/(mn))(1-\tau)$ far from r.

Now we prove that r' can be extended into a codeword of C^m. Let $C_b = C|_{I_b}$ be the projection (puncturing) of C to the coordinates in I_b. Let r'' be the projection of r' to the coordinates in $I_1 \times I_2 \times [n] \times \cdots \times [n]$. We will argue below that r'' is a codeword of $C_1 \otimes C_2 \otimes C^{m-2}$, by considering its projection to axis-parallel lines and claiming all such projections yield codewords of the appropriate code. Note first that the restriction of r' to any line parallel to the b-th axis is a codeword of C, for every $b \in \{2, \ldots, m\}$, since $r'_{1,i}$ is a codeword of C^{m-1} for every $i \in I_1$. Thus this continues to hold for r'' (except that now the projection to a line parallel to the 2nd coordinate axis is a codeword of C_2). Finally, consider a line parallel to the first axis, given by restricting the other coordinates to $\langle i_2, \ldots, i_m \rangle$, with $i_2 \in I_2$. We claim that for every $i_1 \in I_1$, $r''[i_1, \ldots, i_m] = c_{2,i_2}[i_1, \ldots, i_m]$. This follows from the fact that the vertices $(1, i_1)$ and $(2, i_2)$ are not adjacent to each other and thus implying that c_{1,i_1} and c_{2,i_2} are consistent with each other. We conclude that the restriction of r'' to every axis parallel line is a codeword of the appropriate code, and thus (by Proposition 2), r'' is a codeword of $C_1 \otimes C_2 \otimes C^{m-2}$. Now applying Proposition 3 to the code $C_1 \otimes C^{m-1}$ and its projection $C_1 \otimes C_2 \otimes C^{m-2}$ we get that there exists a unique extension of r'' into a codeword c' of the former. We claim this extension is exactly r' since for every $i \in I_1$, $c'_{1,i}[j,k] = r'[i,j,k]$. Finally applying Proposition 3 one more time, this time to the code C^m and its projection $C_1 \otimes C^{m-1}$, we find that $r' = c'$ can be extended into a codeword of the former. This concludes the proof of this claim.

Density of G. We now see that the small τ-soundness-error of the test translates into a small density γ of edges in G. Below, we refer to pairs (b,i) with $b \in [m]$ and $i \in [n]$ as "planes" (since they refer to $(m-1)$-dimensional planes in $[n]^m$) and refer to elements of $[n]^m$ as "points". We say a point $p = \langle p_1, \ldots, p_m \rangle$ lies on a plane (b,i) if $p_b = i$. Now consider the following test: Pick two random planes

(b_1, i_1) and (b_2, i_2) subject to the constraint $b_1 \neq b_2$ and pick a random point p in the intersection of the two planes and verify that c_{b_1,i_1} is consistent with $r[p]$. Let κ denote the rejection probability of this test. We bound κ from both sides.

On the one hand we have that the rejection probability is at least the probability that we pick two planes that are τ-robust and incident to each other in G (which is at least $\frac{m\gamma}{m-1} - 2\epsilon$) and the probability that we pick a point on the intersection at which the two plane codewords disagree (at least $(d/n)^{m-2}$), times the probability that the codeword that disagrees with the point function is the first one (which is at least 1/2). Thus we get $\kappa \geq \frac{d^{m-2}}{2(n)^{m-2}} \left(\frac{m\gamma}{m-1} - 2\epsilon \right)$.

On the other hand we have that in order to reject it must be the case that either $\delta_{C^{m-1}}(r_{b_1,i_1}) > \tau$ (which happens with probability at most ϵ) or $\delta_{C^{m-1}}(r_{b_1,i_1}) \leq \tau$ and p is such that r_{b_1,i_1} and c_{b_1,i_1} disagree at p (which happens with probability at most τ). Thus we have $\kappa \leq \tau + \epsilon$. Putting the two together we have $\gamma \leq \frac{m-1}{m} \left(2\epsilon + \frac{2n^{m-2}}{d^{m-2}}(\tau + \epsilon) \right)$.

Structure of G. Next we note that every edge of G is incident to at least one high-degree vertex. Consider a pair of planes that are adjacent to each other in G. If either of the vertices is not τ-robust, then it is adjacent to every vertex of G. So assume both are τ-robust.

W.l.o.g., let these be the vertices $(1, i)$ and $(2, j)$. Thus the codewords $c_{1,i}$ and $c_{2,j}$ disagree on the $(m-2)$-dimensional surface with the first two coordinates restricted to i and j respectively. Now let $S = \{\langle k_3, \ldots, k_m \rangle \mid c_{1,i}[j, k_3, \ldots, k_m] \neq c_{2,j}[i, k_3, \ldots, k_m]\}$ be the set of disagreeing tuples on this line. By the distance of C^{m-2} we know $|S| \geq d^{m-2}$. But now if we consider the vertex (b, k_b) in G for $b \in \{3, \ldots, m\}$ and k_b such that there exists $k_1, \ldots, k_{m-2}$ satisfying $k = (k_1, \ldots, k_{m-2}) \in S$, it must be adjacent at least one of $(1, i)$ or $(2, j)$ (it can't agree with both at the point (i, j, k). Furthermore, there exists d such k_b's for every $b \in \{3, \ldots, m\}$. Thus the sum of the degrees of $(1, i)$ and $(2, j)$ is at least $(m-2)d$, and so at least one has degree at least $(m-2)d/2$.

Putting it together. From the last paragraph above, we have that the set of vertices of degree less than $(m-2)d/2$ form an independent set in the graph G. The fraction of vertices of degree at least $(m-2)d/2$ is at most $2(\gamma mn)/((m-2)d)$. Thus we get that if $mn \cdot (1 - 2(\gamma mn)/((m-2)d)) \geq m(n-d) + d + 1$, then r is δ-proximate to C^m for $\delta \leq \tau + (1 - \tau) \cdot 2(\gamma mn)/((m-2)d)$. The lemma now follows by simplifying the expressions above, using the upper bound on γ derived earlier. The calculations needed to derive the simple expressions can be found in the full version of this paper [12].

Next we improve the bound achieved on the proximity of r by looking at the structure of the graph G_m^n (the graph underlying the m-Product tester) and its "expansion". Such improvements are a part of the standard toolkit in the analysis of low-degree tests based on axis parallel lines (see e.g., [7, 6, 16, 17] etc.) We omit the proof of this part from this version — it may be found in the full version of this paper [12]. We state the resulting lemma and show how Theorem 1 follows from it.

Lemma 2. *Let m be a positive integer and C be an $[n,k,d]_\Sigma$ code with the property $d^{m-1}/n^{m-1} \geq \frac{7}{8}$. If $r \in \Sigma^{n^m}$ and $c \in C^m$ satisfy $\delta(r,c) \leq \frac{1}{4}$ then $\delta(r,c) \leq 8\rho(r)$.*

Proof (Theorem 1). Let $c = 2^{14} \cdot \left(\frac{n}{d-1}\right)^{2m}$. We will prove that the m-Product Tester is c-robust for C^m. Note that $c \leq 2^{16}$ as required for the theorem, and $\sqrt{\frac{1}{c}} \leq \min\{\frac{1}{36} \cdot \left(\frac{d-1}{n}\right)^m, \frac{1}{128} \cdot \left(\frac{d}{n}\right)^{m-1}\}$ (as will be required below).

The completeness (that codewords of C^m have expected robustness zero) follows from Proposition 2. For the soundness, consider any vector $r \in \Sigma^{n^m}$ and let $\rho = \rho(r)$. If $\rho \geq 1/c$, then there there is nothing to prove since $\delta_{C^m}(r) \leq 1 \leq c \cdot \rho$. So assume $\rho \leq 1/c$.

Note that r has $\sqrt{\rho}$-soundness-error at most $\sqrt{\rho}$. Furthermore, by the assumption on ρ, we have $3\sqrt{\rho} \leq 3\sqrt{\frac{1}{c}} \leq \frac{1}{12} \cdot \left(\frac{d-1}{n}\right)^m$ and so, by Lemma 1, we have $\delta_{C^m}(r) \leq 16 \cdot \left(\frac{n}{d}\right)^{m-1} \cdot 2 \cdot \sqrt{\rho}$. Now using $\sqrt{\rho} \leq \sqrt{\frac{1}{c}} \leq \frac{1}{128} \cdot \left(\frac{d}{n}\right)^{m-1}$, we get $\delta_{C^m}(r) \leq \frac{1}{4}$. Let v be a codeword of C^m closest to r. We now have $\delta(r,v) \leq \frac{1}{4}$ and $\left(\frac{d}{n}\right)^{m-1} \geq \frac{7}{8}$, and so, by Lemma 2, we get $\delta_{C^m}(r) = \delta(r,v) \leq 8\rho$. This concludes the proof.

4 Tanner Product Codes and Composition

In this section we define the composition of two Tanner Product Codes, and show how they preserve robustness. We then use this composition to show how to test C^m using projections to C^2. All proofs of Lemmas in this Section appear in the full version of the paper [12].

Recall that a Tanner Product Code is given by a pair (G, C_{small}). We start by defining a composition of graphs that corresponds to the composition of codes.

Given an (N,M,D)-ordered graph $G = \{\ell_1, \ldots, \ell_M\}$ and an additional (D,m,d)-ordered graph $G' = \{\ell'_1, \ldots, \ell'_m\}$, their Tanner Composition, denoted $G \textcircled{c} G'$, is an $(N, M \cdot m, d)$-ordered graph with adjacency lists $\{\ell''_{j,j'} | j \in [M], j' \in [m]\}$, where $\ell''_{(j,j'),i} = \ell_{j,\ell'_{j',i}}$.

Lemma 3 (Composition). *Let G_1 be an (N,M,D)-ordered graph, and $C_1 \subseteq \Sigma^D$ be a linear code with $C = \mathrm{TPC}(G_1, C_1)$. Further, let G_2 be an (D,m,d)-ordered graph and $C_2 \subseteq \Sigma^d$ be a linear code such that $C_1 = \mathrm{TPC}(G_2, C_2)$. Then $C = \mathrm{TPC}(G_1 \textcircled{c} G_2, C_2)$ (giving a d-query local test for C). Furthermore if (G_1, C_1) is c_1-robust and (G_2, C_2) is c_2-robust, then $(G_1 \textcircled{c} G_2, C_2)$ is $c_1 \cdot c_2$-robust.*

We continue by recasting the results of Section 3 in terms of robustness of associated Tanner Products. Recall that G^n_m denotes the graph that corresponds to the tests of C^m by the m-Product Tester, where $C \subseteq \Sigma^n$.

Note that G^n_m can be composed with G^n_{m-1} and so on. For $m' < m$, define $G^n_{m,m'} = G^n_m$ if $m' = m-1$ and define $G^n_{m,m'} = G^n_m \textcircled{c} G^n_{m-1,m'}$ otherwise. Thus we have that $C^m = \mathrm{TPC}(G^n_{m,m'}, C^{m'})$. The following lemma (which follows easily from Theorem 1 and Lemma 3 gives the robustness of $(G^n_{4,2}, C^2)$.

Lemma 4. *Let C be an $[n,k,d]_\Sigma$ code with $(d-1/n)^4 \leq \frac{7}{8}$. Then $(G^n_{4,2}, C^2)$ is 2^{32}-robust.*

Finally we define graphs H^n_t so that $C^{2^t} = \mathrm{TPC}(H^n_t, C^2)$. This is easily done recursively by letting $H^n_2 = G^n_{4,2}$ and letting $H^n_t = G^{n^{2^{t-2}}}_{4,2} \copyright H^n_{t-1}$ for $t > 2$. We now analyze the robustness of (H^n_t, C^2).

Lemma 5. *There exists a constant c such that the following holds: Let t be an integer and C be an $[n,k,d]_\Sigma$ code such that $d-1 \geq (1-\frac{1}{10m})\cdot n$. Then (H^n_t, C^2) is c^t-robust.*

We are ready to prove Theorem 2.

Proof (Theorem 2). Let c be the constant given by Lemma 5. Fix i and let $C = C_i$, $n = n_i$ etc. (i.e., we suppress the subscript i below). Then C^m is an $[N,K,D]_q$ code, for $N = n^m$, $K = k^m$ and $D = d^m$. Since $d/n \geq 1 - \frac{1}{2m}$, we have C^m has relative distance $d^m/n^m \geq \frac{1}{2}$. Furthermore, the rate of the code is inverse polynomial, i.e., $N = n^m = (p(k))^m \leq \mathrm{poly}(k^m) = \mathrm{poly}(K)$. Finally, we have $C^m = \mathrm{TPC}(H^n_{\log_2 m}, C^2)$, where $(H^n_{\log_2 m}, C^2)$ is a $c^{\log_2 m}$-robust tester for C^m and this tester has query complexity $O(n^2)$. From Proposition 1 we get that there is a tester for C that makes $O(n^2 c^{O(\log_2 m)}) = \mathrm{poly}\log N$ queries.

Acknowledgments

We wish to thank Irit Dinur, Oded Goldreich and Prahladh Harsha for valuable discussions.

References

1. Noga Alon, Tali Kaufman, Michael Krivelevich, Simon Litsyn, and Dana Ron. Testing Reed-Muller codes. In *Proc. RANDOM 2003*, pages 188–199, 2003.
2. Sanjeev Arora. *Probabilistic checking of proofs and the hardness of approximation problems.* PhD thesis, University of California at Berkeley, 1994.
3. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, May 1998.
4. Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45(1):70–122, January 1998.
5. Sanjeev Arora and Madhu Sudan. Improved low-degree testing and its applications. In *Proc. STOC97*, pages 485–495, El Paso, Texas, 4-6 May 1997.
6. László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proc. STOC91*, pages 21–32. ACM, New York, 1991.
7. László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1(1):3–40, 1991.

8. Mihir Bellare, Don Coppersmith, Johan Håstad, Marcos Kiwi, and Madhu Sudan. Linearity testing over characteristic two. *IEEE Transactions on Information Theory*, 42(6):1781–1795, November 1996.
9. Mihir Bellare, Shafi Goldwasser, Carsten Lund, and Alex Russell. Efficient probabilistically checkable proofs and applications to approximation. In *Proc. STOC93*, pages 294–304. ACM, New York, 1993.
10. Mihir Bellare and Madhu Sudan. Improved non-approximability results. In *Proc. STOC94*, pages 184–193, Montreal, Quebec, Canada, 23-25 May 1994.
11. Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. Robust PCPs of proximity, shorter PCPs and applications to coding. In *Proc. STOC04*, (to appear), 2004.
12. Eli Ben-Sasson and Madhu Sudan. Robust Locally Testable Codes and Products of Codes. Available at Electronic Colloquium on Computational Complexity `http://eccc.uni-trier.de/eccc-reports/2004/TR04-046/index.html`
13. Eli Ben-Sasson, Madhu Sudan, Salil Vadhan, and Avi Wigderson. Randomness efficient low-degree tests and short PCPs via ϵ-biased sets. In *Proc. STOC03*, pages 612–621, 2003.
14. 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.
15. Irit Dinur and Omer Reingold. Assignment-Testers: Towards a Combinatorial Proof of the PCP-Theorem. *Manuscript*, 2004.
16. Uriel Feige, Shafi Goldwasser, Laszlo Lovasz, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, 1996.
17. Katalin Friedl, Zsolt Hatsagi, and Alexander Shen. Low-degree tests. In *Proc. SODA94*, pages 57–64, 1994.
18. Katalin Friedl and Madhu Sudan. Some improvements to total degree tests. In *Proc. Israel STCS*, pages 190–198, Tel Aviv, Israel, 4-6 January 1995.
19. Oded Goldreich and Muli Safra. A Combinatorial Consistency Lemma with application to the PCP Theorem. In *SIAM Jour. on Comp.*, Volume 29, Number 4, pages 1132-1154, 1999.
20. Oded Goldreich and Madhu Sudan. Locally testable codes and PCPs of almost-linear length. In *Proc. FOCS02*, Vancouver, Canada, 16-19 November 2002.
21. F. J. MacWilliams and Neil J. A. Sloane. *The Theory of Error-Correcting Codes*. Elsevier/North-Holland, Amsterdam, 1981.
22. Alexander Polishchuk and Daniel A. Spielman. Nearly linear-size holographic proofs. In *Proc. STOC94*, pages 194–203, Montreal, Canada, 1994.
23. Ran Raz and Shmuel Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proc. STOC97*, pages 475–484. ACM Press, 1997.
24. Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM J. Comp.*, 25(2):252–271, 1996.
25. Madhu Sudan. Algorithmic introduction to coding theory. Lecture notes, Available from http://theory.csail.mit.edu/˜madhu/FT01/, 2001.
26. R. Michael Tanner. A recursive approach to low complexity codes. *IEEE Transactions of Information Theory*, 27(5):533–547, September 1981.