Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 175 (2021)

On the Locally Testable Code of Dinur *et al.* (2021)

Oded Goldreich

December 7, 2021

Abstract

This text provides a high-level description of the locally testable code constructed by Dinur, Evra, Livne, Lubotzky, and Mozes (*ECCC*, TR21-151). In particular, the group theoretic aspects are abstracted as much as possible.

1 The result

Ever since Dinur's seminal proof of the PCP Theorem [4], which provided as a "by product" a locally testable code (LTC) of 1/polylog rate, the question resolved by Dinur *et al.* [5] was on the table. I would not say that this question was on the table before [4], because I think that even the 1/polylog rate was not seen on the horizon. Prior to [4], we were still making slow progress at much lower rates (i.e., even rate $n^{-o(1)}$, for block-length n, was not known).

In any case, inspired by prior studies of High-Dimensional Expanders, but actually stepping away from them, the current work provides a LTC of constant rate, where here and above I refer to the regime of constant number of queries (as opposed to prior work that achieved constant rate with a quasi-polylogarithmic number of queries [6, Sec. 13.4.3]) and take constant relative distance for granted.

Needless to say, the current work challenges the two-regimes perspective (i.e., the constant query regime vs the constant rate regime) as well as the possibility that there is a trade-off between the level of locality (i.e., number of queries) and the rate of the code.

The result of Dinur *et al.* [5] refers to the strongest used notion of locally testable codes (cf. [6, Sec. 13.2]). Specifically, it is required that the tester always accepts any codeword, and that any non-codeword is rejected with probability that is proportional to its distance from the code. Needless to say, things are stated in asymptotic terms, where n is viewed as a varying parameter, but all other parameters (i.e., rate, relative distance, and the number of queries made by the tester) are all constants.

Definition 1.1 (LTC, for this text, loosely stated): *The code $C \subset \{0,1\}^n$ has* rate $\frac{\log_2 |C|}{n}$ *and* (relative) distance $\min_{x \neq y \in C}\{\Delta(x,y)\}$, *where* $\Delta(x,y) = |\{i \in [n] : x_i \neq y_i\}|/n$. *We say that C is* locally testable *if there exists an oracle machine, T, that makes a constant number of queries and satisfies the following two conditions:*

1. *For every $x \in C$, it holds that* $\Pr[T^x(n)=1] = 1$.
2. *For every $x \in \{0,1\}^n \setminus C$, it holds that* $\Pr[T^x(n) \neq 1] = \Omega(\Delta_C(x))$, *where* $\Delta_C(x) = \min_{y \in C}\{\Delta(x,y)\}$.

ISSN 1433-8092

In this case, we say that C is a locally testable code.

In terms of property testing, a tester as in Definition 1.1 constitutes a proximity oblivious tester with linear detection probability for the property C [6, Def. 1.7]. The main result of Dinur *et al.* [5] is thus stated as follow.

Theorem 1.2 (LTCs exist and can be explicitly constructed): *For any n, there exists a locally testable code $C \subset \{0,1\}^n$ of constant rate and constant relative distance. Furthermore, C is a linear subspace, and a basis for it can be found in* $\mathrm{poly}(n)$*-time.*

It follows that C has an efficient encoding algorithm (a bijection mapping $\Omega(n)$-bit strings to codewords of C). It also has an efficient decoding (with errors) algorithm; but this (only) follows from the proof provided in [5]. The presentation in [5] only supports n's in a "linearly dense" set (i.e., $n_{i+1} - n_i = O(n_i)$, where n_j is the j^{th} smallest integer in the set), but this can be fixed by padding.

2 The construction

The construction "lifts" the expander codes of [10], where the lifting is highly non-trivial because of an extra feature required from the ingredients (cf., the 4-cycles). This feature (and its utilization) is the key to the success of the new construction.

2.1 The ingredients

For a sufficiently large constant d, we use two d-regular (expander) graphs, G' and G'', on the same vertex set V. These graphs are represented by their incidence functions $g'_i, g''_i : V \to V$ (for $i \in [d]$) such that $g'_i(v)$ (resp., $g''_i(v)$) denotes the i^{th} neighbor of v in the first (resp., second) graph.[1] Furthermore, we assume that these functions are actually bijections. Indeed, each of these graphs is an expander in the sense that its second eigenvalue (i.e., random-walk convergence rate) is sufficiently small (as a function of other parameters). Moreover, we require:

1. *The neighborhoods of a vertex in the two graphs are disjoint*; that is, for every $v \in V$ and $i, j \in [d]$, it holds that $g'_i(v) \neq g''_j(v)$.

2. *Symmetry of the incidence functions*; that is, for every $i \in [d]$ there exists $j \in [d]$ such that $g'_j(g'_i(v)) = v$ holds for all $v \in V$. Without loss of generality, we may assume that g'_{2i-1} is the inverse of g'_{2i}; that is, $g'_{2i-1}(g'_{2i}(v)) = v$. Ditto for g''_i.

3. *Two interleaving steps form a 4-cycle in $G' \cup G''$*: For every $v \in V$ and $i, j \in [d]$, it holds that $g''_j(g'_i(v)) = g'_i(g''_j(v))$. Hence, $c_{v,i,j} \stackrel{\text{def}}{=} (v, g'_i(v), g''_j(g'_i(v)), g''_j(v), v)$ forms a 4-cycle in the graph $G = (V, E)$ that is formed by superimposing G' and G'' (i.e., $E = (V, E' \cup E'')$, where $G' = (V, E')$ and $G'' = (V, E'')$). We denote this set of (ordered) 4-cycles by Q; that is,

$$Q \stackrel{\text{def}}{=} \{(v, g'_i(v), g''_j(g'_i(v)), g''_j(v), v) : v \in V \ \& \ i, j \in [d]\}. \qquad (1)$$

   Note: Although there may be other 4-cycles in the graph G, in the sequel, whenever we refer to 4-cycles, we mean the 4-cycles in Q only.

[1] For simplicity, we use the same degree in both graphs and the same bound on the second eigenvalue.

Indeed, the last requirement appears hardest to meet. Dinur *et al.* [5] achieve it by using left and right multiplication (in a non-Abelian group). Specifically, they use Cayley graphs over the vertex-set (group) V, with adequate generator-sets $A = \{a_i : i \in [d]\}$ and $B = \{b_i : i \in [d]\}$, and let $g'_i(v) = a_i \cdot v$ and $g''_i(v) = v \cdot b_i$.

Base codes: We also use constant-size codes $C', C'' \subset \{0,1\}^d$ of rate $r_0 > 3/4$ and relative distance $\delta_0 > \lambda$, where $\lambda > 0$ is an upper bound on the (normalized) second eigenvalue of each of the graphs. Furthermore, we pick these codes so that their tensoring yields a relatively "robust" tensor code (see [5, Def. 2.8 & Lem. 2.9]).[2]

2.2 The constructed code and its tester

For a function $f : Q \to \{0,1\}$, we denote by f_v its restriction to the set of 4-cycles that are "rooted" at the vertex $v \in V$; that is, 4-cycles that have the form $(v, g'_i(v), g''_j(g'_i(v)), g''_j(v), v) = c_{v,i,j}$ for some $i, j \in [d]$. Indeed, letting $Q_v = \{c_{v,i,j} : i, j \in [d]\}$, the function $f_v : Q_v \to \{0,1\}^{d \times d}$ is viewed as a d-by-d Boolean matrix in which the $(i,j)^{\text{th}}$ entry equals $f(v, g'_i(v), g''_j(g'_i(v)), g''_j(v), v)$. The new code, denoted C, consists of all Boolean functions $f : Q \to \{0,1\}$ whose f_v-restrictions are codewords of the tensor code $C' \otimes C''$, where $C' \otimes C''$ is the set of all d-by-d matrices whose rows are codewords of C' and columns are codewords of C''. That is,

$$C \stackrel{\text{def}}{=} \{f : Q \to \{0,1\} \mid (\forall v \in V)\ f_v \in C' \otimes C''\}. \tag{2}$$

The tester is the natural one; that is, it selects one condition at random and checks it. Specifically, given oracle access to $f : Q \to \{0,1\}$, the tester select uniformly $v \in V$, retrieves the d-by-d matrix $f_v = (f(v, g'_i(v), g''_j(g'_i(v)), g''_j(v), v))_{i,j \in [d]}$ by querying f on all 4-cycles in Q_v, and accepts if and only if f_v is a codeword of $C' \otimes C''$.

Comment: In the foregoing presentation each 4-cycle is represented four times (since each of its vertices can be used as the "start vertex" (or "root")).[3] In contrast, in [5], the four representations are identified so that the value on each of them is obtained from the value on a canonical representation of the relevant 4-cycle.[4]

3 The analysis (flavor only)

The analysis of the rate and distance of the code C follows the analysis of the expander codes of [10], but the real issue is analyzing the foregoing tester. (Recall that generic expander codes are not locally testable.)

[2] In Dinur *et al.* [5], the base codes are denoted C_A and C_B, and they are shown to exists in [5, Lem. 5.1].

[3] The other three representation of $c_{v,i,j} = (v, g'_i(v), g''_j(g'_i(v)), g''_j(v), v)$ are $c_{g'_i(v),i',j}$, $c_{g''_j(g'_i(v)),i',j'}$ and $c_{g''_j(v),i,j'}$, where $g'_{i'}$ is the inverse of g'_i and $g''_{j'}$ is the inverse of g''_j.

[4] This operation is called *folding* [3]; it replaces a potential auxiliary test (which queries the four representations) that enforces all four representation to hold the same value.

Rate. Recalling that the code is a linear subspace, we lower-bound its dimension by $\frac{1}{4} \cdot |V| \cdot d^2 - |V| \cdot 2d \cdot (d - r_0 \cdot d)$, where $\frac{1}{4}$ compensates for the four representations of each 4-cycle and $2d \cdot (d - r_0 \cdot d)$ is an upper bound on the number of linear constraints imposed on each f_v (i.e., $2d$ is the number of rows and columns in each matrix Q_v, and $d - r_0 \cdot d$ is the co-dimension of the base codes). Hence, we obtain a rate of at least $\frac{1}{4} - 2 \cdot (1 - r_0)$, which is a positive constant provided that $r_0 > 7/8$.

Distance. Since the code is linear, we lower-bound the weight of its non-zero codewords. For any $f \in C$ and each $i \in [n]$, let $f^{(i)}$ be a function on the edges of G'' such that $f^{(i)}(\{v, g''_j(v)\}) = f(c_{v,i,j})$, which is well-defined by the folding (see Footnote 4). Now, assuming that $f(c_{v^*,i^*,j^*}) = 1$ for some v^*, i^*, j^*, it follows (by the distance of C') that, for at least for a δ_0 fraction of the $i \in [d]$, it holds that the i^{th} row of f_{v^*} is not an all-zero codeword (of C'). Hence, for at least a δ_0 fraction of the $i \in [d]$, the function $f^{(i)}$ is non-zero. Considering only the graph G'' (and the based code C''), we apply the analysis of expander codes to $f^{(i)}$ (see [5, Lem. 4.4], which reduces to [5, Lem. 2.1]). It follows that a non-zero $f^{(i)}$ must have relative weight at least a $\delta_0 \cdot (\delta_0 - \lambda)$, where λ upper-bounds the second (normalized) eigenvalue of G''. Recalling that at least a δ_0 fraction of the $f^{(i)}$'s are non-zero, we conclude that the relative weight of non-codewords of C is at least a $\delta_0^2 \cdot (\delta_0 - \lambda)$.

Local testability – take 1. *How come the new code is locally testable whereas expander codes are not*? As observed by numerous experts, generic expander codes (as generic LDPC codes) are defined in terms of a *low-density parity-check* matrix, which (generically) may be of full rank. In that case, removing a single parity-check from the matrix yield a larger code that may still have large distance. But then the resulting code contains codewords that are far from the original code, although they violate a single linear constraint of the original code. Hence, the natural tester that checks a single linear constraint (in the original matrix) fails poorly.

In contrast, the tester associated with the new code C selects at random a set of highly dependent linear constraints, which are associated with a (random) vertex, such that the sets associated with different choices (i.e., vertices) have significant pairwise intersections. Specifically, for every two neighboring vertices, u and v, the inspected d-by-d matrices (i.e., f_u and f_v) share d-entries that correspond to the edge $\{u, v\}$. Hence, violating a single constraint (of C) leads to violating many other (different) constraints. In particular, dropping few constraints from the low-density parity-check matrix that corresponds to C leaves the code invariant.

Needless to say, the foregoing is extremely far from establishing the local testability of C. It merely asserts that C passes a sanity check that the expander codes fail.

Local testability – take 2. As is often the case in property testing (cf. [9, Chap. 3]), the analysis of the foregoing tester uses a self-correction process (in order to establish the contrapositive). Specifically, Dinur *et al.* [5] present a decoding algorithm and prove that if the *natural tester* (which selects a random vertex $v \in V$ and accepts if and only if $f_v \in C' \otimes C''$) rejects f with probability η, then the decoding algorithm finds a codeword (of C) that is $O(\eta)$-close to f.[5] It follows (by the contrapositive) that each $f : Q \to \{0, 1\}$ is rejected by the natural test with probability that is lower-bounded by a constant fraction of f's distance from C.

[5]The actual constant in the O-notation is $4(2d+1)$, and the claim holds provided that $\lambda \leq \alpha \cdot \delta_0$, where α depends on the "robustness" parameter of the tensor code $C' \otimes C''$.

The key issue, of course, is to design and analyze a decoding algorithm that satisfies the foregoing condition. That is, given any $f : Q \to \{0,1\}$, the decoder must find a codeword of C that is $O(\eta(f))$-close to f, where $\eta(f)$ is the probability that the natural tester rejects f. A natural idea is to iteratively modify f such that in each iteration we select an arbitrary 4-cycle c and reset $f(c)$ such that it satisfies a majority of the checks that look at it (i.e., $f(c) = \sigma$ if c is assigned σ in a majority of the d-by-d matrices f_v that contain c).[6] The decoding process terminates when no addition modification is possible (i.e., where for each $c \in Q$ the value of $f(c)$ equals the majority value assigned to c by the relevant f_v's).

The foregoing decoder is analogous to the one used for decoding expander codes. It seems that this candidate decoder works well (i.e., correctly decodes f) in the case that f is close to C, but the intended application of this decoder is showing that every f is $O(\eta(f))$-close to C (by showing that, on any input f, the decoder finds a codeword that is $O(\eta(f))$-close to f).[7] We stress that it may be that the foregoing decoder works well on any input f (i.e., it always finds a codeword that is $O(\eta(f))$-close to f), but this is currently unknown.

Local testability – take 3. In light of the foregoing, a different approach to decoding is taken. The following decoding algorithm is based on the agreement testing paradigm, which arose with the proof composition paradigm of PCPs [2, 1]. The foregoing paradigm links the agreement probability of partial assignments to suitable intersecting subsets of the domain (i.e., "local agreement") to the existence of a global function that approximately fits these partial assignments (i.e., "global agreement"). This paradigm will be applied here to the set of d-by-d matrices that correspond to the codewords of $C' \otimes C''$ that are closest to the matrices f_v (for all $v \in V$). Note that d-by-d matrices that correspond to neighboring vertices have a common row (or column)[8], and the agreement test will be applied (as a mental experiment) to these pairs of matrices. Also, the disagreement probability (between the foregoing pairs of codewords) is at most twice $\eta(f)$; see [5, Eq. (4.5)].

In general, for every $\overline{w} = (w_v)_{v\in V} \in (\{0,1\}^{d\times d})^{|V|}$, we define the local disagreement of $\overline{w}$ as the probability that the pair of matrices that correspond to a random edge agree on the row (or column) that corresponds to the 4-cycles that contain this edge. That is, we consider

$$\mathtt{D}(\overline{w}) \stackrel{\rm def}{=} \Pr_{e=\{u,v\}\in E}[w_u|_e \neq w_v|_e] \tag{3}$$

where $w_u|_e$ (resp., $w_v|_e$) denotes the restriction of w_u (resp., w_v) to the row (or column) that corresponds to the 4-cycles that contain the edge e (i.e., the 4-cycles in $Q_u \cap Q_v$, where $\{u,v\} = e$). (Recall that E is the edge-set of the graph defined in Section 2.1.)

Decoding is done in iterations such that in each iteration we pick an arbitrary vertex v and modify the current w_v so to minimize $\mathtt{D}(\overline{w})$ subject to the new w_v being in $C' \otimes C''$; that is, w_v is replaced by $w' \in C' \otimes C''$ if w' minimizes $\Pr_{e=\{u,v\}\in E}[w_u|_e \neq w'|_e]$ (over all $C' \otimes C''$). Initially, on input $f : Q \to \{0,1\}$, for every $v \in V$, we let w_v be a codeword of $C' \otimes C''$ that is closest to f_v, and the decoder halts when no modification is possible (i.e., no modification decreases the value of

[6]That is, we consider all f_v's such that $c = c_{v,i,j}$ for some $i, j \in [d]$.

[7]Hence, the foregoing is insufficient for two reasons. First, we need the decoder to work on any input f, and not only on inputs that are close to C; that is, the closeness to C is the desired conclusion, and can not be the hypothesis. Second, even in case f is $o(1)$-close to C, which implies that $\eta(f) = o(d^2) = o(1)$, we need to upper-bound f's distance to C in terms of $\eta(f)$; that is, we seek a quantitative result (i.e., $O(\eta(f)$-closeness) not merely a qualitative result (e.g., if $\eta(f) = o(1)$, then f is $o(1)$-close to C).

[8]In contrast, in the case of expander codes, neighboring vertices have only a single edge in common.

$\mathtt{D}$).[9] At termination, either $\mathtt{D}(\overline{w}) > 0$, which is considered a failure, or $\mathtt{D}(\overline{w}) = 0$, which implies that $\overline{w}$ corresponds to a codeword of C (i.e., there exists $f' \in C$ such that $w_v = f'_v$ for every $v \in V$).

Indeed, the main result of [5, Sec. 4] is that this decoder works well, which yields the desired LTC, once a suitable graph is constructed (in [5, Sec. 5]). Specifically, Dinur *et al.* [5] proved

Theorem 3.1 (the foregoing decoder works well [5, Prop. 4.7&4.8]): *Let* $f : Q \to \{0,1\}$ *and* $\eta(f) \stackrel{\rm def}{=} \Pr_{v \in V}[f_v \notin C' \otimes C'']$. *For some universal constant* $\eta_0 > 0$ (i.e., $\eta_0 = (\Omega(\delta_0) - \lambda)/2d$), *if* $\eta(f) < \eta_0$, *then the foregoing decoder never fails but rather outputs a codeword of* C *that is at distance at most* $O(d) \cdot \eta(f)$ *from* f.

In particular, [5, Prop. 4.8] asserts that if $\eta(f) < \eta_0$, then the decoder does not fail, whereas [5, Prop. 4.7] asserts that in this case the output (codeword of C) is $O(d) \cdot \eta(f)$-closet to f.[10] Needless to say, if $\eta(f) \geq \eta_0$, then the claim holds triviality (since every f is $O(\eta_0)$-close to C).

Theorem 3.2 (construction of suitable graphs, follows from [5, Lem. 5.2]): *For every* $\lambda > 0$, *there exists a constant* d *such that, for every* $n \in \mathbb{N}$, *a pair of* $\Theta(n)$*-vertex graphs as in Section 2.1 can be constructed in* $\mathrm{poly}(n)$*-time. In particular, each graph is* d*-regular and its second* (normalized) *eigenvalue is at most* λ. *Furthermore, incidence queries regarding each of the graphs can be answered in* $\mathrm{poly}(\log n)$*-time.*

(The foregoing is simplified: Dinur *et al.* [5, Lem. 5.2] obtain such graphs for any d that is a multiple of some $d_0 \in \mathbb{N}$, and use this fact in order to present suitable base codes (see [5, Lem. 5.1]).)[11]

On the proof of Theorem 3.1. The easy part (proved in [5, Prop. 4.7]) is showing that if the decoder does not fail, then the codeword f' that it outputs is $O(\eta(f))$-close to f. Letting $\overline{w}^{\mathtt{init}}$ (resp., $\overline{w}^{\mathtt{fin}}$) denote the initial (resp., final) value of $\overline{w}$, observe that $\Delta(f, f') \leq \frac{|V^{\mathtt{init}}| + |V^{\mathtt{fin}}|}{|V|}$, where $V^{\mathtt{init}} = \{v \in V : w_v^{\mathtt{init}} \neq f_v\}$ and $V^{\mathtt{fin}} = \{v \in V : w_v^{\mathtt{fin}} \neq w_v^{\mathtt{init}}\}$. Next, note that $|V^{\mathtt{init}}| \leq \eta(f) \cdot |V|$ (since $f_v \in C' \otimes C''$ implies $w_v^{\mathtt{init}} = f_v$) and $|V^{\mathtt{fin}}| \leq \mathtt{D}(\overline{w}^{\mathtt{init}}) \cdot |E|$ (since each modification step decreases $\mathtt{D}$ by at least $1/|E|$), whereas $\mathtt{D}(\overline{w}^{\mathtt{init}}) \leq 2\eta(f)$ (since $\{u, v\}$ contributes to $\mathtt{D}(\overline{w}^{\mathtt{init}})$ only if either $f_u \notin C' \otimes C''$ or $f_v \notin C' \otimes C''$)[12] and $|E| = d \cdot |V|$.

The more difficulty part (proved in [5, Prop. 4.8]) is showing that the decoder may fail only when $\eta(f) \geq \eta_0$. It is actually shown that if the algorithm fails (i.e., $\mathtt{D}(\overline{w}^{\mathtt{fin}}) > 0$), then $\mathtt{D}(\overline{w}^{\mathtt{fin}}) \geq 2\eta_0$ must hold, which implies $\eta(f) \geq \eta_0$. (Recall that $\overline{w}^{\mathtt{fin}}$ is stable in the sense that $\mathtt{D}$ cannot be decreased by any modification to $\overline{w}^{\mathtt{fin}}$.)

At a very high level, the foregoing claim is proved as follows. First, it is proved (in [5, Clm. 4.10]) that if some edge e contributes to $\mathtt{D}(\overline{w}^{\mathtt{fin}})$ (per the r.h.s of Eq. (3)), then a constant fraction of the edges that participate in 4-cycles that contain e also contribute to this count (i.e, to $\mathtt{D}(\overline{w}^{\mathtt{fin}})$). This means that disagreements are propagated locally; that is, disagreement propagates from a single

[9] Note that if $\mathtt{D}$ is decreased by the modification, then $\mathtt{D}$ decreases by at least $1/|E|$ units.

[10] In [5, Prop. 4.7] the constant factor is $4 \cdot (2d + 1)$, but our presentation is a bit different (i.e., we use all four representations of each 4-cycle) and this may affect the constant.

[11] The point is that they used a result that requires d to be a multiple of some given d_0. We believe that this is not really necessary. Alternatively, obtaining d that is a multiple of d_0 is quite trivial if one does not aim at optimal expansion (i.e., Ramanujan graphs), which is immaterial for the current application.

[12] Otherwise, $w_u^{\mathtt{init}} = f_u$ and $w_v^{\mathtt{init}} = f_v$, which contradicts the hypothesis regarding $\{u, v\}$. Note, however, that the same vertex may contribute to $2d$ edges. Hence, we have $\mathtt{D}(\overline{w}^{\mathtt{init}}) \cdot |E| \leq 2d \cdot \eta(f) \cdot |V|$.

edge to many edges in the various 4-cycles that contain this edge. Next, using the robustness of the tensor code $C' \otimes C''$ (and the stability of $\overline{w}^{\mathtt{fin}}$), it is proved (in [5, Clm. 4.11]) that disagreements on edges that are incident at a vertex v translate to a proportional number of disagreements on the edges that are in 4-cycles that contain vertex v but are not incident to it. Finally, the expansion properties of the graphs are used in order to prove (in [5, Clm. 4.12 & Lem. 4.13]) that these local disagreements translate to global ones; that is, if there are many disagreements in the 4-cycles that touch a vertex, then there are many disagreements globally (i.e., in the entire graph). This means that $\mathtt{D}(\overline{w}^{\mathtt{fin}}) > 0$ implies $\mathtt{D}(\overline{w}^{\mathtt{fin}}) = \Omega(1)$.

On the proof of Theorem 3.2. One may indeed wonder whether there exist pairs of graphs satisfying the conditions stated in Section 2.1. The cue is using left and right multiplication (in a non-Abelian group); specificaly, Dinur *et al.* [5, Lem. 5.2] use Cayley graphs over the vertex-set (group) V, with generator-sets $A = \{a_i : i \in [d]\}$ and $B = \{b_i : i \in [d]\}$, and let $g'_i(v) = a_i \cdot v$ and $g''_i(v) = v \cdot b_i$. Hence, $g'_i \circ g''_j = g''_j \circ g'_i$, whereas guaranteeing the $g'_i(v) \neq g''_j(v)$ holds (for all $v \in V$ and $i, j \in [d]$) does not seem problematic (yet, it is far from trivial, since we need these graphs to be expanders (see [5, Sec. 6])).

4 Concluding comments

An interesting feature of the locally testable code of Dinur *et al.* [5] is that it is the first known LTC of subquadratic block-length that comes in a single-step construction, which (in particular) does not utilize any PCP machinery.[13] (We mention that the LTCs of Meir [7] and Viderman [11, 12] also avoids PCP machinery, but these constructions proceed in several steps, which mimic various ideas of PCP constructions.)

I was told that Panteleev and Kalachev [8] have, independently but later, also proved Theorem 1.2.[14] Their construction seems (essentially) identical to the one of Dinur *et al.* [5], but their analysis seems somewhat different.

References

[1] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof Verification and Intractability of Approximation Problems. *JACM*, Vol. 45, pages 501–555, 1998. Extended abstract in *33rd FOCS*, 1992.

[2] Sanjeev Arora and Shmuel Safra. Probabilistic Checkable Proofs: A New Characterization of NP. *JACM*, Vol. 45, pages 70–122, 1998. Extended abstract in *33rd FOCS*, 1992.

[3] Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free Bits, PCPs and Non-Approximability – Towards Tight Results. *SICOMP*, Vol. 27, No. 3, pages 804–915, 1998. Extended abstract in *36th FOCS*, 1995.

[4] Irit Dinur. The PCP Theorem by Gap Amplification. *JACM*, Vol. 54 (3), Art. 12, 2007. Extended absract in *38th STOC*, 2006.

[13] An explicit LTC of almost quadratic block-length that does not utilize any PCP machinery follows by starting from a suitable low-degree test and using alphabet reduction; see [6, Sec. 13.3.2.1].

[14] The result of Dinur etal [5] was publicly announced in September 2021.

[5] Irit Dinur, Shai Evra, Ron Livne, Alex Lubotzky, and Shahar Mozes. Locally Testable Codes with Constant Rate, Distance, and Locality. *ECCC*, TR21-151, 2021.

[6] Oded Goldreich. *Introduction to Property Testing.* Cambridge University Press, 2017.

[7] Or Meir. Combinatorial Construction of Locally Testable Codes. *SICOMP*, Vol. 39 (2), pages 491–544, 2009. Extended abstract in *40th STOC*, 2008.

[8] Pavel Panteleev and Gleb Kalachev. Asymptotically Good Quantum and Locally Testable Classical LDPC Codes. `arXiv:2111.03654[cs.IT]`, November 2021.

[9] Dana Ron. Algorithmic and analysis techniques in property testing. *Foundations and Trends in Theoretical Computer Science*, Vol. 5, pages 73–205, 2010.

[10] Michael Sipser and Daniel Spielman. Expander codes. *IEEE Trans. Inf. Theory*, Vol. 42 (6), pages 1710–1722, 1996. Preliminary version in *35th FOCS*, 1994.

[11] Michael Viderman. Strong LTCs with Inverse Poly-Log Rate and Constant Soundness. In *54th FOCS*, pages 330–339, 2013.

[12] Michael Viderman. Explicit Strong LTCs with Inverse Poly-Log Rate and Constant Soundness. In *22nd RANDOM*, LIPIcs (Vol. 116), pages 58:1–58:14, 2018.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il