Lecture 10: LTC (cont.) and quantum LDPC codes

Irit Dinur

January 11, 2023

In this lecture we will complete the proof that the LTCs that we constructed last week are in fact locally testable. We will then describe the so-called quantum LDPCs show how the same 2-chain gives rise to a quantum LDPC with constant relative rate and distance.

1 LTCs

We have defined the left right Cayley complex given a group G and two sets of generators $A, B \subset G$. Given two codes $C_A \subseteq \mathbb{F}_2^A$ and $C_B \subseteq \mathbb{F}_2^B$ we have defined the code

$$C(X, C_A, C_B) = \left\{ f \in \mathbb{F}_2^{X(2)} \;\middle|\; \forall a, g, b \; f([a, g, \cdot]) \in C_B \;\text{ and }\; f([\cdot, g, b]) \in C_A \right\}$$

We assume that $|A| = |B| = d$ for simplicity, and let $k = \dim(C_A) = \dim(C_B)$. We started proving that this code is locally testable, and cosidered the following local correction algorithm,

Algorithm: given a word $f \in \mathbb{F}_2^{X(2)}$.

1. Every $g \in G$ chooses $w_g \in C_0 \otimes C_0$ that is closest to $f(\cdot, g, \cdot)$.
2. Let $E' = \{\{g, g'\} \in X(1) \mid \; w_g \nsim w_{g'}\}$, where $w_g \nsim w_g'$ means that the local views disagree on some common square.

   For each g, if there is another choice of w_g that minimizes the number of sets in E' touching g, then switch to that local view.

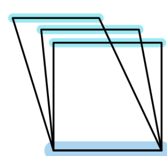
   Repeat until no more available switches.
3. If $E' = \phi$ output $\tilde{f}$ the codeword obtained from the combined local views. Else output fail.

We saw that the algorithm doesn't get stuck in an infinite loop, and that if it doesn't fail then $\mathrm{dist}(\tilde{f}, f) = O(\mathrm{wt}(Hf))$. It remains to prove,

Lemma 1.1. *If $E' \neq 0$ at the end of the algorithm, then $|E'| = \Omega(|E|)$.*

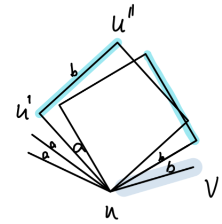
Proof. We prove this by propagation. We will devise a random walk from edge to edge and show that it expands and that starting from E' there is some decent probability to reach E' after one step. This will imply, via the Alon-Chung lemma, that E' is large. The random walk starts from an edge e and then with probability $\frac{1}{2}$ goes to a *parallel* edge, and with probability $\frac{1}{2}$ goes to an *opposite* edge; where

- Parallel edge: Given an edge $[a, g]$ a parallel edge is an edge $[a, gb]$ for any b. Given an edge $[g, b]$ a parallel edge is an edge $[ag, b]$ for any a.
- Opposite edge: Given an edge $\{u, v\}$ an opposite edge is obtained by first choosing one of the endpoints, say u, and then moving to a neighbor u' of u in the graph $(X(0), X(1))$, and then taking a random edge $\{u', u''\}$ containing u'.



parallel walk

$$\{g, ag\} \underset{\text{b-step}}{\longrightarrow} \{gb, agb\}$$



opposite walk

$$\{u, v\} \underset{\text{down}}{\longrightarrow} u \underset{\text{nbr}}{\longrightarrow} u' \underset{\text{up}}{\longrightarrow} \{u', u''\}$$

It is not too hard to see that this random walk is an arbitrarily good expander.

Claim 1.2. Let λ upper bound the second largest eigenvalue of $Cay(G, A)$ and $Cay(G, B)$.

- Let M_{opp} be the random walk moving from e to an opposite edge e' according to the above process. Then $\lambda(M_{opp}) \leqslant \lambda$.
- Let $M_{||}$ be the random walk moving from e to a parallel edge e' according to the above process. The edges split to at most $|A| + |B|$ connected components, and on each one, $\lambda(M_{||}) \leqslant \lambda$.

How is this walk useful for us? we now show that every edge $e \in E'$ implies that many of its neighbors are also in E'.

1. If $uv \in E'$ then there are a constant fraction of squares s touching uv for which $w_u(s) \neq w_v(s)$.
2. For each such square, suppose it is $s = \{u, v, w_1, w_2\}$. Either $vw_1 \in E'$ or $w_1w_2 \in E'$ or $w_2u \in E'$ because they cannot all agree on s.
3. Either u or v is "heavy", namely many of its adjacent edges are in E'; or the edge uv has many parallel edges (like w_1w_2) that are in E'.
4. If u (or v) is heavy, then many of the "opposite" edges are in E'.

For the last item, we rely on the coboundary expansion of the tensor code, also known as *agreement testability*. Fix a heavy g. Let M_A be the $d \times d$ matrix whose rows are collected from the A neighbors of g. Let M_B be the $d \times d$ matrix whose columns are collected from the B neighbors of g. We chose g as a heavy vertex, which means that an ε fraction of its neighbors disagree with it. Since we are at the end of the run of the algorithm, the local view w_g is the tensor codeword that agrees with a maximal number of rows in M_A and columns in M_B.

Now we recall the definition of agreement-testability:

Definition 1.3 (agreement testability)**.** Let $\beta > 0$. Let $C_i \subset \{f : [n_i] \to \mathbb{F}_2\}$ for $i = 1, 2$. We say that $C_1 \otimes C_2$ is β-agreement testable if for every $w_1 \in C_1 \otimes \mathbb{F}_2^{n_2}$, $w_2 \in \mathbb{F}_2^{n_1} \otimes C_2$, there exists $w \in C_1 \otimes C_2$ such that

$$\mathop{\mathbb{P}}_{i\in[n_1],j\in[n_2]}[w_1(i,j) \neq w_2(i,j)] \geqslant \frac{\beta}{2} \cdot (\mathop{\mathbb{P}}_i[w_1(i,\cdot) \neq w(i,\cdot)] + \mathop{\mathbb{P}}_j[w_2(\cdot,j) \neq w(\cdot,j)]).$$

In our case, if $C_A \otimes C_B$ is β-agreement testable then for $w_1 = M_A$ and $w_2 = M_B$, there is some M that satisfies the definition, so

$$wt(M_A + M_B) \geqslant \beta \cdot \frac{1}{2}(\mathop{\mathbb{P}}_i[M(i,\cdot) \neq M_A(i,\cdot)] + \mathop{\mathbb{P}}_j[M(\cdot,j) \neq M_B(\cdot,j)]).$$

We get, for w_g that is at least as good as M, $wt(M_A + M_B) \geqslant \beta \cdot \frac{1}{2}(\mathrm{dist}_{rows}(M_A, w_g) + \mathrm{dist}_{cols}(M_B, w_g)) = \beta\varepsilon$.

What is $\mathrm{wt}(M_A + M_B)$? it is the fraction of entries (a, b) on which $M_A(a, b) \neq M_B(a, b)$. This corresponds to the fraction of neighbor pairs ag, gb of g that disagree on $[a, g, b]$, namely:

$$w_{ag}([a, g, b]) \neq w_{gb}([a, g, b])$$

Whenever this happens, either the edge $[ag, b] \in E'$ or $[gb, a] \in E'$. Therefore, we get a large fraction of opposite edges that are in E'. □

All in all we have seen that if $Cay(G, A)$ and $Cay(G, B)$ are good expanders, and if the tensor code $C_A \otimes C_B$ is a coboundary expander, then the global code is locally testable.

2 Tensor codes: agreement testability and coboundary expansion

Let $C \subset \mathbb{F}_2^n$ and denote $k = \dim(C)$, and $m = n - k$. Let $G \in \mathbb{F}_2^{n\times k}$ be a generator check matrix for C.

$$\begin{tikzcd}
 & \mathbb{F}_2^{k\times n} \arrow[dr, "G\otimes I"] & \\
\mathbb{F}_2^{k^2} \arrow[ur, "I\otimes G"] \arrow[dr, "G\otimes I"'] & & \mathbb{F}_2^{n^2} \\
 & \mathbb{F}_2^{n\times k} \arrow[ur, "I\otimes G"'] &
\end{tikzcd}$$

Figure 1: Chain complex of a tensor code.

In more compressed form we have $\mathbb{F}_2^{k^2} \xrightarrow{\delta_0} (\mathbb{F}_2^k)^{2n} \xrightarrow{\delta_1} \mathbb{F}_2^{n^2}$, where $\delta_0 = I \otimes G + G \otimes I$ and $\delta_1 = G \otimes I + I \otimes G$.

Claim 2.1. This is an exact sequence, namely $Ker(\delta_1) = Im(\delta_0)$ which is the space corresponding to tensor codewords.

Given $f \in (\mathbb{F}_2^k)^{2n}$, if $\mathrm{wt}(\delta_1 f)$ is small, does it mean that $\mathrm{dist}(f, Im(\delta_0))$ is small? This depends whether the chain in Figure 2 has coboundary expansion. This turns out to be equivalent to the notion of agreement testability. Recall the definition

Lemma 2.2. *The chain in Figure 2 has β coboundary expansion iff the code $C \otimes C$ is β-agreement-testable.*

3 A chain complex

Let us give a cohomological interpretation to the code and proof we have just seen. The collection of w_g can be packaged as $w \in (C_A \otimes C_B)^{X(0)}$, $w(g) = w_g$. We can define a map δ_0 from the such chains on thevertices to chains on the edges, where the edge uv sums the appropriate row of w_u and of w_v. So for $w \in (C \otimes C)^{X(0)}$,

$$\delta_0 w([a, g]) = w(g)(a, \cdot) + w(ag)(a^{-1}, \cdot)$$

and similarly for an edge $[g, b]$. We get $\delta_0 w \in C^{X(1)}$.

Next, we can define a map from the edges to the squares by having each square sum over the four appropriate bits on its four edges. So, for $f \in C^{X(1)}$,

$$\delta_1 f([a, g, b]) := f([a, g])(b) + f([a, gb])(b^{-1}) + f([g, b])(a) + f([ag, b])(a^{-1}).$$

We get a chain complex

$$(C_A \otimes C_B)^{X(0)} \xrightarrow{\delta_0} {C_A}^{X_B(1)} \times C_B^{X_A(1)} \xrightarrow{\delta_1} \mathbb{F}_2^{X(2)} \tag{3.1}$$

Claim 3.1. This is a 2-chain, namely $\delta_1 \circ \delta_0 = 0$.

Proof. For a fixed square and each of its four vertices, the vertex sends the value to the square twice, for two edges, and this gets cancelled. □

Even more intrestingly, the lemma stated above can be cast in these terms. Letting w be the collection of local views at the end of the algorithm, and letting $f = \delta w$, we see that $E' = supp(f)$. We say that f is locally minimal with respect to vertex moves if the weight of f is minimal with respect to changes of the form $f \leftarrow f - w_g$ for any $w_g \in C_A \otimes C_B$. The proof we have seen for Lemma 1.1 above shows that

Lemma 3.2. *For any $0 \neq f \in \operatorname{Ker} \delta_1$, if f is locally minimal with respect to vertex moves, then $\mathrm{wt}(f) \geqslant \Omega(1)$.*

Indeed, in the proof, the structural property of $E' = supp(f)$ that was used is the following. If $e \in supp(f)$ then it affects a δ fraction of the squares s touching it. In order for $\delta_1 f(s) = 0$, there must be at least one other edge in the square that is non-zero, so $e' \in supp(f)$. This also gives

Corollary 3.3 (cosystolic distance)**.** . *If $f \in \mathrm{Ker}(\delta_1) \backslash \mathrm{Im}(\delta_0)$, then $\mathrm{dist}(f, \mathrm{Im}(\delta_0)) \geqslant \Omega(1)$.*

4 Quantum CSS codes

A quantum (CSS stabilizer) code is a subspace of quantum states on n qubits. For more on this see [2, 1]. Errors are modeled as single bit flips in either the X or Z basis. A code is designed so that even if some bounded number of errors (= bitflips in X or Z basis) occur, the original state can be recovered.

The code subspace is specified by a local Hamiltonian, and is a simultaneous eigenspace of a bunch of "parity check" operators. Unlike the classical case, the parity checks can apriori be applied in a continuum of bases. By linearity it suffices to restrict to two bases for each bit (which span all others): the X and the Z basis. So designing this codespace amounts to designing two parity check matrices H_X, H_Z specifying parity checks in X basis, and in Z basis.

The dimension of the code is the number of qubits minus the dimensions of the parity checks, namely $n - \dim(H_X) - \dim(H_Z)$.

The codes C_X, C_Z are not arbitrary, rather, H_X, H_Z must have mutually orthogonal rows (mod 2). Namely,

$$rows(H_Z) \perp rows(H_X)$$

or $C_Z^{\perp} \subseteq C_X$, or $H_X \subseteq H_Z^{\perp}$.

The reason for this is that each row in H_X stands for an operator of the form $I \otimes I \otimes X \otimes X \otimes X \otimes I$, with the X's placed where the 1's would be. Similarly for the matrix H_Z. The code is the simultaneous eigenspace of the operators in all of the rows. In order for it to be non-empty, all operators (i.e. all rows) must *commute*. Clearly every pair of operators from H_X commute with each other, and similarly for a pair of operators from H_Z. In order for an H_X operator to commute with an H_Z operator, we must have the corresponding rows have inner product 0 mod 2, because X and Z operators anti-commute, $(XZ = -ZX)$.

Every element in C_Z is viewed as an 'error' as it moves a codeword to another codeword. However, elements in $C_X^{\perp}$, as parity checks, by definition stabilize the codewords so these are *not errors*. Therefore, the Z errors are only $C_Z \setminus C_X^{\perp}$, and the X errors, symmetrically are $C_X \setminus C_Z^{\perp}$. The minimum distance is the minimum waight of any of these words. We define the quantum minimum distance to be

$$d_Q = \min(d_Z, d_X)$$

where

- $d_X = \min\left\{|x| \,\middle|\, x \in C_X \setminus C_Z^{\perp}\right\}$
- $d_Z = \min\left\{|z| \,\middle|\, z \in C_Z \setminus C_X^{\perp}\right\}$

Homological point of view

Let $H_Z^T = \delta_Z$ and let $H_X = \delta_X$ and we get

$$\mathbb{F}_2^{r_Z} \xrightarrow{\delta_Z} \mathbb{F}_2^n \xrightarrow{\delta_X} \mathbb{F}_2^{r_X} \tag{4.1}$$

In which case the quantum code is the cohomology $Ker\delta_X / Im\delta_Z$.

Quantum LDPC codes on the left-right complex

The chain defined in (3.1) essentially gives rise to a quantum LDPC code.

$$
\begin{tikzcd}
 & (\mathbb{F}_2^k)^{X_A(1)} \arrow[dr, "\delta_{1,B}"] & \\
(\mathbb{F}_2^{k^2})^{X(0)} \arrow[ur, "\delta_{0,A}"] \arrow[dr, "\delta_{0,B}"'] & & \mathbb{F}_2^{X(2)} \\
 & (\mathbb{F}_2^k)^{X_B(1)} \arrow[ur, "\delta_{1,A}"'] &
\end{tikzcd}
$$

Here $\delta_{0,A}$ applies $G_A \otimes I$ to $f(g)$ for every vertex g, and then distributes the k-bit rows of the result among the A neighboring edges. Every edge adds the twi k-bit vectors that it gets from its two vertex endpoints. We get

$$\delta_{0,A} f([a,g]) = (G_A \otimes I) f(g)[a,\cdot] + (G_A \otimes I) f(ag)[a^{-1},\cdot].$$

and similarly

$$\delta_{0,B} f([g,b]) = (I \otimes G_B) f(g)[\cdot,b] + (I \otimes G_B) f(gb)[\cdot,b^{-1}].$$

Next, the map $\delta_{1,B}$, for each vertex g, collects the $d \times k$ bits from the A neighbors of g, and applies $(I \otimes G_B)$ to this matrix. It then gets an $d \times d$ matrix which it distribute to the neighboring squares. Similarly $\delta_{1,A}$ does the same, and each square adds the four bits it receives from each of its neighbors.

One can see that both maps δ_0, δ_1 are LDPC: each output bit depends on a constant number of input bits.

Rate. If $\dim(C_A) = (1-\varepsilon)d$ and $\dim(C_B) = \varepsilon d$ we get positive rate.

Distance. For one direction this essentially follows from Lemma 3.3. Let $f \in \mathbb{F}_2^{X(1)\cdot k}$. Assume $f \in \mathrm{Ker}(\delta_1)$. To show distance, we assume $f \neq 0$ has small weight, and deduce it must belong to $\mathrm{Im}\,\delta_0$. We use an algorithm.

Algorithm: given a word $f \in \mathbb{F}_2^{X(1)\cdot k}$.

1. For every $g \in G$, if there is some $w_g \in \mathbb{F}_2^{k^2}$ so that $f - \delta_0(w_g)$ is zero on more edges touching g than before, let $f \leftarrow f - \delta_0(w_g)$.[1]
2. Repeat until no more available moves.
3. Output f.

Clearly, every step of the algorithm decreases the number of edges on which f is non-zero, so if $\mathrm{wt}(f) = \varepsilon$ initially, it can only be less than ε after the algorithm terminates. Let

$$E' = \{e \in E \mid \; f(e) \neq 0\}.$$

We will use the same walk as above to show that E' must be large.

For each edge e, let w_e be the result of stretching $f(e)$ to a codeword using C_A or C_B. We view w_e as an assignment for the squares touching e. Since $\delta_1 f = 0$, every square receives contributions from its four edges that sum to zero. Therefore, if $e \in E'$, by the distance of C_A, C_B, at least δ fraction of its squares are nonzero. Each of these squares must receive contribution from an additional edge, therefore one of its three other edges must also be in E'.

[1]We are thinking of $w_g \in \mathbb{F}_2^{X(0)\times k^2}$ by putting zero everywhere other than on g.

1. If $e = uv \in E'$ then there are a constant fraction of squares touching e for which $w_e(s) \neq 0$.

2. For each such square, suppose it is $s = \{u, v, w, x\}$. either $vw \in E'$ or $wx \in E'$ or $xu \in E'$ because the total sum must be zero.

3. Either u or v is "heavy", namely many of e's adjacent edges are in E'; or e has many parallel edges that are in E'.

4. If u (or v) is heavy, then many of the "opposite" edges are in E'.

The only step that needs checking is the last step. Suppose v is "heavy". By coboundary expansion, many of the neighbor-pairs of v must disagree, otherwise v would have "made a move" to reduce the number of non-zero edges touching it.

References

[1] Anthony Leverrier and Gilles Zémor. Quantum tanner codes. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 872–883. IEEE, 2022. 5

[2] Pavel Panteleev and Gleb Kalachev. Asymptotically good quantum and locally testable classical LDPC codes. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 375–388. ACM, 2022. 5