M1 Internship:
Homological product code and weight of random tensors

Joël Felderhoff
ENS de Lyon
Internship realized under the supervision of Péter Vrana
at QMATH, University of Copenhagen

1 Introduction

1.1 Context, motivation and problematics

Quantum computer desing is a current field of research. To build such computers would lead to the ability to solve problems in less time than with classical computers. For example, the discrete logarithm problem is solved in linear time with quantum computation while the best classical algorithm solving it is exponential.

A big difference between classical and quantum computers is that in the classical case, no error is supposed to append during computations, which is not the case in quantum computation. The problem of the resiliant storage and transmission is also a problem with quantum computers, as in the classical case. In order to build quantum computers, good procedures to detect and correct errors are then needed. The problem of finding equivalent of error correcting codes in the quantum case is then a current and active field of research.

The problem is that if the case of classical error correcting code is well studied and known, the case of quantum code still has open problems. It was shown by Calderbank, A Robert and Shor [4] that good Quantum Error Correcting Codes (QECC) exists (i.e. codes which encode n logical qubits into $O(n)$ physical qubits and are able to correct $O(n)$ errors), but it is not known by now whether good Low Density Parity Check (LDPC) QECC exists. As a matter of fact, the best codes known by now have either distance $O(n)$ but weight $O(\sqrt{n})$ [3] or weight $O(1)$ but distance $O(\sqrt{n})$ [10]. Researchers are looking for LDPC code because the weight of a code is linked with the number of quantum gates used to implement it, and then with the price and the complexity of implementing it in an actual quantum computer.

In 2013, Bravyi and Hastings published "Homological Product Code" [3]. In that paper, they give a proof of the existence of $[[n, O(n), O(n), O(\sqrt{n})]]$ codes, by making the *homological product* of two random codes. The goal of my internship was to study whether this construction could be generalized to the product of 3 (or more) random codes, which is expected to give raise to $[[n, O(n), O(n), O(\sqrt[3]{n})]]$ (or $[[n, O(n), O(n), O(\sqrt[k]{n})]]$) codes. This construction was suggested in the paper of Bravyi and Hastings, but they noted that it seems more difficult than in the two factor case.

As the proof of Bravyi and Hastings is related to the distribution of the weight of tensor in certain orbit of 2-tensors under the action of GL_n^2, I studied the action of GL_n^3 over the space of 3-tensors, examined some invariants and studied how the weight of random tensors behave in function of their orbit under the action of this group.

1.2 My internship

I did my internship under the supervision of Péter Vrana, in the QMATH group at the University of Copenhagen. The internship lasted 12 weeks.

I spent the first time of my internship acquiring the knownledge needed to understand the subject. I worked on the general theory of QECC and stabilizer codes, and I studied a part of the graph homology theory, in order to link the article's theoretical construction to objects I understand better.

Then, my second work was to understand the paper of Bravyi and Hastings, and to link it with the knowledge acquired before.

The third part of the internship was the longest. During that part I tried to have a deeper understanding of the proof of Bravyi and Hastings in order to see which part of it could be generalized and what part were more difficult to adapt to the multiple factor case. Over that period, I had the opportunity to attend the Simons Program: QMath Masterclass on Tensors: Geometry and Quantum Information at the department, where my tutor and several expert of tensor theory had lectures. It provided me some insights about quantum information theory and (besides giving me culture on a mathematical object I did not know) it gave me more ease to work in my subject.

1.3 Organisation of this report

In the first part of this report (Section 2) I will make some definitions and statements useful to understand the objective behind the objects we are going to study. I will introduce what a QECC is and some standard constructions on them. Once this will be defined, I will, in Section 3, get into the main subject of the internship, the proof of the paper of Bravyi and Hastings. I will define what the homological product code is, and make a sketch of the proof of [3]. Then I will present the different directions I took to try to generalize this proof in Section 4.

My generalisations attemps are presented in two ways. First I will explain my study of different tensors invariants (4.1) and after that I will emphase a part of my work about the link between the weight of a matrix and its rank (4.2).

1.4 Special thanks

I would like to thank Péter Vrana for accepting of being my tutor during the 3 months of this internship, and for his help and support during it. Thanks to Danaé, Solène, Louis and my parents for their huge help and support during the everyday life abroad.

Thanks to Marc and Nicolas for the relecture of my report.

2 Quantum error correcting codes

2.1 Basics about quantum mechanics

In quantum mechanics, states of physical systems are modeled by norm 1 elements of a Hilbert space (that is to say a complex vector space endowed with an Hermitian inner product). The dimension of this space is the number of "degrees of freedom" of the system. For example, the *spin* of an electron can be **up** or **down**, then this quantity will be modelised as a normalized element of $\mathbb{C}^2$.

Definition-Proposition 2.1.1. *[2, A-II-§3.1] Take A and B two finite dimensional F vector spaces (with F a field), with basis $(a_i)_{i=1...n}$ and $(b_i)_{i=1...m}$.*

*The **tensor product** $A \otimes B$ is the vector space spanned by the rank 1 tensors $(a \otimes b)_{\substack{a\in A,\\ b\in B}}$, subject to the following relations, for all $a, a' \in A, b, b' \in B, \lambda \in F$:*

- $\lambda(a \otimes b) = (\lambda a) \otimes b = a \otimes (\lambda b)$.
- $a \otimes b + a' \otimes b = (a + a') \otimes b$
- $a \otimes b + a \otimes b' = a \otimes (b + b')$

It is a finite dimensional vector space of dimension nm whose basis is $(a_i \otimes b_j)_{\substack{i=1...n\\ j=1...m}}$.

Remark 2.1.2. *[2, A-II-§3.8] This construction is (up to isomorphism) associative: $(A \otimes B) \otimes C = A \otimes (B \otimes C) = A \otimes B \otimes C$.*

If we have two systems, one in the Hilbert space $\mathcal{H}_1$ the other in $\mathcal{H}_2$, and we want to model their interaction, we are going to consider the overall system to be an element of the product space $\mathcal{H}_1 \otimes \mathcal{H}_2$. One could see that intuitively as if the first system has n degrees of freedom and the second m degrees of freedom, then the product system has nm degrees of freedom.

As our goal is to build a quantum computer, we are interested in how to create "bits" from physical systems. A **qubit** is a system with 2 degrees of freedom. The basis of its Hilbert space will be denoted by $|0\rangle, |1\rangle$.

As we will be interested in systems with a certain number of qubits, we will consider the tensor product of n 2 dimensional Hilbert spaces. In the following, I will denote $E = (\mathbb{C}^2)^{\otimes n}$. Its basis is $(|a_1 a_2 \dots a_n\rangle)_{(a_i)\in\mathbb{Z}_2^n}$.

The computations on qubits are done by unitary operators (linear operators that preserve the norm). For example, the controled-NOT (CNOT) gate acts on a system of 2 qubits like this:

$$F|00\rangle = |00\rangle, F|01\rangle = |01\rangle$$
$$F|10\rangle = |11\rangle, F|11\rangle = |10\rangle$$

The CNOT gate makes the qubits **interact** with each other. We will use a lot the **local actions** on qubits.

Definition 2.1.3. *Take A and B two finite dimensional F vector spaces (with F a field), with basis $(a_i)_{i=1\dots n}$ and $(b_i)_{i=1\dots m}$. Let $M, N \in Hom(A) \times Hom(B)$ two operators on A and B. The **local operator** $M \otimes N$ is defined to be (for all $a \in A, b \in B$):*

$$(M \otimes N)(a \otimes b) = Ma \otimes Nb$$

These actions can be understood as operations performed on only subparts of the system. For example, if one studies the interaction between two electrons, then a local action would be an operation performed in each electrons without interactions. Those operations are important because they are cheaper to implement than the global one. More generally, transformations acting on only a few qubits are experimentally more accessible than global ones, and are sufficient to build any transformations on an arbitrary large number of qubits.

2.2 Quantum error correcting codes

The idea behind the quantum error correction is to take k qubits, called **logical qubits** (that is to say the space $(\mathbb{C}^2)^{\otimes k}$), and to encode them into a subspace of $(\mathbb{C}^2)^{\otimes n}$.

Definition 2.2.1. *A **Quantum Error Correcting Code (QECC)** is a subspace of E.*

Before defining our first examples of QECC, I have to state some definitions. First, we want to know what the errors that can occurs are.

Definition 2.2.2. *The **Pauli matrices** are:*

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \text{ and } Y = XZ = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

*The **nth Pauli Group** is the (multiplicative) group:*

$$\mathscr{P}_n = \left\{ \pm X^{a_1} Z^{b_1} \otimes \dots \otimes X^{a_n} Z^{b_n}, (a_i)_{i=1\dots n}, (b_i)_{i=1\dots n} \in \mathbb{Z}_2 \right\}$$

This group acts on E by left multiplication.

The action of Pauli matrices have easy interpretation in term of qubit action:

- The action of X is a bit flip: $X|0\rangle = |1\rangle, X|1\rangle = |0\rangle$.
- The action of Z is a phase flip: $Z|0\rangle = |0\rangle, Z|1\rangle = -|1\rangle$.

The Pauli group encodes the local actions on qubits. An important property of this group is that it spans the space of operators on E. One can prove that if two errors can be corrected by a QECC, then any linear combination of them can be corrected, which implies that restricting the study of errors to Pauli errors does not lose the generality of error correction.

2.3 Stabilizer codes

As the Pauli group encodes some actions that would modify the result of our computations (bit flip for example), it is legitimate to try find subspaces of E that are invariant by the action of some of the elements of the Pauli group. When such kind of space has been found, one is able to encode qubits on them in order to protect them from errors.

An important class of QECC are the stabilizer codes, which are the quantum analogs of classic linear codes.

Definition 2.3.1. *Let S be a subgroup of $\mathscr{P}_n$. Then the **stabilizer code** of S is the space of the elements of E invariant by the action of S : $C_S \equiv E^S$.*

In the following, I will denote by $I \in \mathscr{P}_n$ the operator $I \otimes \cdots \otimes I$.

Proposition 2.3.2. *Take S a subgroup of $\mathscr{P}_n$. Then if $-I \in S$, then C_S is trivial, and furthermore if $-I \notin S$, then*

$$dim(C_S) = \frac{2^n}{|S|} \tag{1}$$

Proof. First, we can note that $\mathrm{tr}(A \otimes B) = \mathrm{tr}(A)\,\mathrm{tr}(B)$ and $\mathrm{tr}(X) = \mathrm{tr}(Z) = \mathrm{tr}(Y) = 0$.

Hence if $P \in \mathscr{P}_n$, if $P = I$, $\mathrm{tr}(P) = 2^n$, if $P = -I$, $\mathrm{tr}(P) = -2^n$. Else, $\mathrm{tr}(P) = 0$.

The theory of representations of finite groups (see [9] for example), gives us that

$$\dim(C_S) = \dim(E^S) = \frac{1}{|S|}\sum_{s\in S}\mathrm{tr}(s)$$

Of course, $I \in S$. Hence if $-I \in S$, the sum is equal to $2^n - 2^n + 0 = 0$, and then C_S is trivial. Else, this sum is equal to 2^n which gives the result. □

Direct computations show the following properties:

Proposition 2.3.3.

- $X^2 = Z^2 = I$, $Y^2 = -I$.
- $\left(\bigotimes_{i=1}^n X^{a_i} Z^{b_i}\right)\left(\bigotimes_{i=1}^n X^{a'_i} Z^{b'_i}\right) = (-1)^{\boldsymbol{a'}\cdot\boldsymbol{b}} \bigotimes_{i=1}^n X^{a_i+a'_i} Z^{b_i+b'_i}$ *where* $\boldsymbol{a'}\cdot\boldsymbol{b} = \sum_{i=1}^n a'_i b_i$ *and every addition is performed in* $\mathbb{Z}_2$.
- $\forall P \in \mathscr{P}_n, P^{-1} = \pm P$

Corollary 2.3.4. $[\mathscr{P}_n, \mathscr{P}_n] = \{I, -I\}$

Proof. Take $P = \bigotimes_{i=1}^n X^{a_i} Z^{b_i}, Q = \bigotimes_{i=1}^n X^{a'_i} Z^{b'_i} \in \mathscr{P}_n^2$.

$$[P,Q] = PQP^{-1}Q^{-1} = \pm PQPQ = \pm\left(\pm\bigotimes_{i=1}^n X^{a_i+a'_i} Z^{b_i+b'_i}\right)^2$$
$$= \pm\bigotimes_{i=1}^n X^{2(a_i+a'_i)} Z^{2(b_i+b'_i)} = \pm\bigotimes_{i=1}^n I = \pm I$$

□

Hence, the condition for a stabilizer code to be non-trivial can be rephrased in the following way.

Proposition 2.3.5. *If a subgroup S of $\mathscr{P}_n$ gives rises to a nontrivial code, then it is comutative.*

Proof. With Proposition 2.3.2, we have that C_S is nontrivial iff $-I \notin S$. So if C_S nontrivial, as $[S,S] \subset [\mathscr{P}_n, \mathscr{P}_n]$, $[S,S] = \{I\}$, hence S is commutative. □

This property enables us to think about stabilizer codes in terms of subgroups of $(\mathscr{P}_n/\{\pm I\},\cdot) \simeq (\mathbb{F}_2^{2n},+)$, that is to say in terms of subspaces of $\mathbb{F}_2^{2n}$.

More precisely, set

$$\begin{array}{rccl} \langle\cdot,\cdot\rangle: & \mathscr{P}_n\times\mathscr{P}_n & \longrightarrow & \mathbb{F}_2 \\ & x,y & \longmapsto & \begin{cases} 0, & \text{if } xy=yx \\ 1, & \text{if } xy=-yx \end{cases} \end{array}$$

This map is a homomorphism in both arguments, invariant by multiplication of one of its arguments by $-I$, so it factors through $((\mathscr{P}_n/\{\pm I\})^2,\cdot) \simeq (\mathbb{F}_2^{2n},+)$.

The isomorphism between $(\mathscr{P}_n/\{\pm I\},\cdot)$ and $(\mathbb{F}_2^{2n},+)$ is done in the following way:

$$\bigotimes_{i=1}^{n} X^{a_i}Z^{b_i} \mapsto (a_1,\ldots,a_n,b_1,\ldots,b_n)$$

With this isomorphism, seen as a bilinear form in $\mathbb{F}_2^{2n}$, the matrix of $\langle\cdot,\cdot\rangle$ is given by $\Sigma = \begin{pmatrix} 0 & I_n \\ I_n & 0 \end{pmatrix}$.

Definition-Proposition 2.3.6. *A bilinear form $f: E^2 \longrightarrow \mathbb{F}$ is said to be **symplectic** if*

1. *It is non-degenerate: $\forall u\in E, (\forall v\in E, f(u,v)=0 \implies u=0)$*
2. *It is alternating: $\forall u\in E, f(u,u)=0$*

Then $\langle\cdot,\cdot\rangle: (\mathbb{F}_2^{2n})^2 \longrightarrow \mathbb{F}_2$ is symplectic.

If $f: E^2\longrightarrow\mathbb{F}$ is a symplectic form and F is a subspace of E, one can define

$$\hat{F} \equiv \{y\in E, \forall x\in F\ f(x,y)=0\}$$

*A subspace F of E is said to be **totally isotropic** for f if $F\subset\hat{F}$.*

It turns out that totally isotropic subspaces of $\mathbb{F}_2^{2n}$ for the bilinear map $\langle\cdot,\cdot\rangle$ are very important in the study of stabilizer codes. In the following, when I will talk about a totally isotropic subspace, it will mean totally isotropic for $\langle\cdot,\cdot\rangle$.

Proposition 2.3.7. *Every commutative subgroup S of $\mathscr{P}_n$ induces a totally isotropic subspace of $\mathbb{F}_2^{2n}$. Conversely, every totally isotropic subspace of $\mathbb{F}_2^{2n}$ is the (non unique) image of a subgroup S of $\mathscr{P}_n$ such that $-I\notin S$.*

Proof. If H is a commutative subgroup of $\mathscr{P}_n$, then $\forall g,h\in H, \langle u,v\rangle = 0$ so the embedding of H in $\mathbb{F}_2^{2n}$ is totally isotropic.

Conversely, take V a totally isotropic subspace of $\mathbb{F}_2^{2n}$. Let $e_1,\ldots,e_s$ be a basis of V in $\mathscr{P}_n/\{\pm I\}$, let $f_1,\ldots,f_s$ be arbitrary preimages in $\mathscr{P}_n$ of $e_1,\ldots,e_s$. Then $\forall i,j \langle f_i,f_j\rangle = \langle e_i,e_j\rangle = 0$ so $f_if_j=f_jf_i$ hence the group generated by $f_1,\ldots,f_s$ is commutative. Let us call it S.

If $-I\in S$, then there exist $a_1,\ldots a_s\in\mathbb{F}_2^s$ not all equal to zero such that $-I=\prod_i f_i^{a_i}$, then when we embbed this equality in $\mathbb{F}_2^{2n}$, we get $\sum_i a_ie_i=0$, with the a_i not all equal to zero which is impossible since (e_i) is a basis of V, and hence is linearly independant.

The image of this group in $\mathbb{F}_2^{2n}$ is clearly V, which concludes the proof. □

Proposition 2.3.8. *If S_1 and S_2 are two subgroup of $\mathscr{P}_n$ obtained this way, then there exist $g\in\mathscr{P}_n$ such that $S_1 = gS_2g^{-1}$*

Proof. Let H be a totally isotropic subspace of E. Let $H\subset L$ be a maximal subspace of E such that $\langle\cdot,\cdot\rangle$ is zero on it. There is a basis $(e_1,\ldots e_s,e_{s+1},\ldots e_l,f_1,\ldots f_k)$ of E such that $e_1,\ldots,e_s$ is a basis of H, $e_1,\ldots,e_l$ a basis of L with $\langle e_i,f_j\rangle=\delta_{ij}$, $\langle e_i,e_j\rangle=\langle f_i,f_j\rangle=0$ (this is proven by a slightly modified version of the Gram–Schmidt process).

Take $g_1,\ldots,g_s$ be the generating set of a Pauli subgroup S. The other possible preimage of g_i inside $\mathscr{P}_n$ is $-g_i$. Let's say that I change the sign of one of the g_i (say the sign of g_1), and take S' this new subgroup. Take h a preimage of f_1, then I claim that $S'=hSh^{-1}$.

Take $i\geq 2$, then $\langle h,g_i\rangle = \langle f_1,g_i\rangle = 0$ then $hg_ih^{-1}=g_i$.

$\langle h,g_1\rangle = \langle f_1,g_1\rangle = 1$, so $hg_1h^{-1}=-g_1$. □

Those properties show that to study stabilizer subgroup of $\mathscr{P}_n$ is equivalent to study isotropic subspaces of $\mathbb{F}_2^{2n}$.

Definition 2.3.9. *A stabilizer code C_S is said to be an $[[n,k]]$-code if it encodes k (logical) qubits into n (physical) qubits, that is to say if it is a subspace of $(\mathbb{C}^2)^{\otimes n}$ of dimension 2^k.*

Note that this definition is valid even if one is not talking about stabilizer codes. As in the classical case, it is simple to give a definition of the distance of a stabilizer code.

If we take a code defined by a subspace V of $\mathbb{F}_2^{2n}$, it can be shown ([7, Theorem 10.8]) that every error not in $\hat{V} \setminus V$ can be properly detected and corrected. The interesting errors are then the elements of the Pauli group defined by the elements of $\hat{V} \setminus V$.

Definition 2.3.10. *Let C be an $[[n,k]]$ stabilizer code given by a totally isotropic subspace V of $\mathbb{F}_2^{2n}$. V is called the set of recoverable errors.*

C is said to have distance d (then one will say C is $[[n,k,d]]$ code) if one have $d \leq \min_{v \in \hat{V} \setminus V} |v|$.

$d(C)$ is defined as the largest d such that C is a $[[n,k,d]]$ code.

A simple way to create stabilizer QECC is to use the fact that $\mathbb{F}_2^{2n} \equiv \mathbb{F}_2^n \times \mathbb{F}_2^n$, and to take two $(n,k)_2$ classical linear code C_X, C_Z, the first in order to correct bit flip errors, the second to correct phase flip errors.

Definition-Proposition 2.3.11. *Take C^X, C^Z two linear classical code encoding respectively k_1 and k_2 bits into n bits (that is to say C^X -resp C^Z- is a subspace of $\mathbb{F}_2^n$ with dimension k_1 -resp k_2-), and such as $C^X \subset (C^Z)^\perp$ (or equivalently $C^Z \subset (C^X)^\perp$), then the space $C \equiv C^X \oplus C^Z$ is a totally isotropic subspace of $\mathbb{F}_2^{2n}$ of dimension k_1+k_2, and then the stabilizer code defined by C is a $[[n, n-k_1-k_2]]$-code.*

*This kind of code is called a **CSS code**.*

Proof. Let (u_X, u_Z) and (v_X, v_Z) be two vectors of $C^X \oplus C^Z$, then $\langle (u_X,u_Z),(v_X,v_Z)\rangle = u_X^T v_Z + u_Z^T v_X = 0 + 0 = 0$. Hence, $C^X \oplus C^Z$ is isotropic.

The fact that it is an $[[n, n-k_1-k_2]]$-code is just an application of Equation (1). □

This codes have been invented by Robert Calderbank, Peter Shor and Andrew Steane [4], and are called CSS codes from the name of their inventors.

Definition 2.3.12. *Take (C^X, C^Z) an $[[n,k,d]]$ CSS code, given by the parity check matrices H^X, H^Z. This code is said to have **weight** w if every columns and rows of H^X and H^Z has weight at most w (that is to say at most w non-zero coefficients).*

In that case one says that the code is an $[[n,k,d,w]]$ code.

Note that this definition of the weight depends on the parity check matrices, and not only on the QECC itself (since the same CSS code can be given by differents parity check matrices). This implies that the weight of a CSS code is a question of matrices and not of QECC anymore.

3 Homological product of random codes

3.1 Homological codes

A special kind of CSS code can be created by looking at a certain class of operator.

In everything that follows, I will denote $E = \mathbb{F}_2^n$.

Definition 3.1.1. *Let $\partial : E \to E$ be a linear map such that $\partial^2 = 0$. This kind of operator is called a **boundary operator**.*

With ∂ a boundary operator, one can define a CSS code : let $C^Z \equiv \operatorname{Im} \partial$ and $C^X \equiv \operatorname{Im} \partial^T$.

$x^T \partial y = (\partial^T x)^T y$ so $(C^Z)^\perp = \ker \partial^T$ and similarly $(C^X)^\perp = \ker \partial$. As $\partial^2 = 0$, $\operatorname{Im} \partial \subset \ker \partial$, so (C^X, C^Z) defines a $[[n, n - 2\operatorname{rank} \partial]]$ CSS code given by the matrices of ∂ and ∂^T. So this code has low weight if the matrix of ∂ is low weighted.

This kind of operator is studied in homological algebra, which has its own way of naming objects.

The elements of $\ker \partial$ are called the **cycles** of ∂. An element of $\operatorname{Im} \partial$ is said to be a **trivial cycle** or a **boundary**.

The space $H(\partial) = \ker \partial / \operatorname{Im} \partial$ is called the homology space of ∂. ∂ is said to have **homological dimension** H if $\dim(H(\partial)) = H$. In that case, if $\dim \operatorname{Im} \partial = L$, $\dim E = H + 2L$. One can then see that if $\dim E$ and $\dim(H(\partial))$ are fixed, then $\dim \operatorname{Im} \partial$ is fixed too. With those notations, (C^X, C^Z) is then a $[[\dim(E), H]]$ QECC.

If C is the code defined by ∂, the set of recoverable errors of C are the elements of $\operatorname{Im} \partial \bigoplus \operatorname{Im} \partial^T$, that is to say the trivial cycles. The non-recoverable errors of C are indeed exactly the nontrivial cycles of ∂, which is shown by this property:

Proposition 3.1.2 ([1]). *Let ∂ be a boundary operator, let $V = \operatorname{Im} \partial \bigoplus \operatorname{Im} \partial^T$ the code defined by this operator. Then the set of non-recoverable errors $\hat{V} = \ker \partial \bigoplus \ker \partial^T$*

The maximal distance of C is then given by

$$\min_{\varphi \in (\ker \partial \setminus \operatorname{Im} \partial) \cup (\ker \partial^T \setminus \operatorname{Im} \partial^T)} |\varphi|.$$

In [10], Tillich and Zémor defined an operation made to combine two existing homological codes in order to create a new. They called it **hypergraph product**, but Bravyi and Hastings, in [3] used the name **homological product**.

If one takes two boundary operators $\delta_1 : E_1 \to E_1$ and $\delta_2 : E_2 \to E_2$, one can define their product as

$$\partial = \delta_1 \otimes I + I \otimes \delta_2 : E_1 \otimes E_2 \to E_1 \otimes E_2$$

Proposition 3.1.3 ([3]). *If the δ_i are boundary operators defining $[[n_i, k_i, d_i]]$ codes with weights w_i, then ∂ is a boundary operator, which defines an $[[n_1 n_2, k_1 k_2, \min(d_1, d_2), w_1 + w_2]]$ code.*

One can see that when one takes the product of two homological codes, the weight of the product grows in an additive way when the size of the code grows in a multiplicative way. In their article [3], Bravyi and Hastings used that fact in order to construct families of homological QECC whose size and distance grows linearly (with parameter n) and its weight grows as the square root of n.

3.2 Product of random homological codes

Theorem 3.2.1 ([3]). *If δ_1 and δ_2 are two random boundary operators, with space of size n and homological dimension $H = \rho_{enc} n$. Let ∂ be the homological product of δ_1 and δ_2, let C be the code defined by ∂. For sufficiently large n, and for sufficiently small c and ρ_{enc} (independent of n),*

$$\mathbb{P}\left(d(C) \leq cn^2\right) = o_n(1)$$

This theorem implies the existence of a family of $[[n^2, (\rho_{enc} n^2), cn^2, O(n)]]$ codes, with n going to infinity, that is to say a family of $[[n, O(n), O(n), O(n^{1/2})]]$ codes.

In the following, I give a sketch of the proof of this theorem.

3.3 Proof sketch

I take the same notation as in the statement of the theorem.

The idea of the proof is the following: if ψ is a non-trivial cycle of ∂ with total weight $\leq cn^2$ (for a certain $c \in]0, 1[$, then $\psi \in \mathbb{F}_2^n \otimes \mathbb{F}_2^n$ can be seen as an $n \times n$ matrix. Then one can choose $r \in]c, 1[$, and set $n' = rn$, and $c' \in]0, 1[$ (arbitrarly small as c tends to 0) such that a submatrix of ψ of size $n' \times n'$ has weight of each of its columns and rows $\leq c'n'$ (they call this condition Uniform Low Weight, or ULW). They call this submatrix a **reduced cycle** of ∂.

Then, the probability that a nontrivial cycle with low weight exist is bounded by $\binom{n}{n'}^2 \mathbb{P}_{red} + o(1)$, where $\mathbb{P}_{red}$ is the probability that there exist a non-zero reduced cycle of ∂ whose every column and row has weight $\leq cn'$ and the $o(1)$ stands for the probability that the reduced cycles are zero. The goal of the rest of the proof is to prove that $\mathbb{P}_{red}$ is exponentially small.

After that, Bravyi and Hastings introduce the notion of good operators. δ_1 and δ_2 are good if they do not have any element of their kernel with support in the last $n - n'$ coordinates. They prove that the δ_i are good with high probability (increasing to 1 as n is growing), so for the remaining of the proof, they assume that δ_1 and δ_2 are good.

It appears that as the δ_i are good, this is indeed easy to count the number of reduced cycles (seen as matrices) of a given rank.

Proposition 3.3.1. *If we denote by $\Gamma(R)$ the number of reduced cycles of rank R of ∂, then*

$$\Gamma(R) \leq O(1) \cdot 2^{(n+H)R-R^2} \quad \text{if } R \leq H \tag{2}$$

$$\Gamma(R) \leq O(1) \cdot 2^{(n+H/2)R-R^2/2} \quad \text{if } R \geq H \tag{3}$$

In order to prove this result, they restrict the image of δ_i to their first n' coordinates (no information about the kernel of δ_i is lost here, because δ_i is supposed to be good), and quotient by the image of the $n - n'$ last coordinates. This construction gives two new boundary operators δ'_1 and δ'_2, then define ∂' their homological product, and count the number of possible extensions of each vector in the kernel of δ' to a vector in the kernel of δ. They prove the following formula:

$$\Gamma(R) = \sum_{r=0}^{\min(K,R)} \#\left\{h \in \ker \partial', \mathrm{rank}(h) = r\right\} \cdot E_{K,r}^{n',R}$$

With K the size of the quotient space, and $E_{K,r}^{n',R}$ is the number of extensions of a $K \times K$ matrix of rank r to a $n' \times n'$ matrix of rank R.

They prove that

$$E_{a,r}^{A,R} \leq O(1) \cdot 2^{(2A-a)R-ar-R^2+(r+R)^2/4}$$

$$\#\left\{h \in \ker \partial', \mathrm{rank}(h) = r\right\} \leq O(1) \cdot 2^{2(H+L)r-r^2} \cdot \sum_{f=0}^{\infty} 2^{-2f^2+2f(r-H)}$$

By using those two formulas, they have the wanted bound on $\Gamma(R)$.

After that, they prove that there exists an enumeration of the reduced cycles of rank R such that if one fix an index j, the jth element of the enumeration is uniform in the set of matrices of rank R (when δ_1 and δ_2 are uniform in the set of boundary operators).

This enumeration has this property because it is invariant by the action of GL_n^2, and because the rank is a characteristic invariant of the orbits of M_n by GL_n^2.

They now only have to bound the probability that a random matrix of a given rank has weight of its columns and rows of weight $c'n'$, and to do an union bound.

They prove that the probability that an uniform $n \times n$ matrix of rank R has all its rows and columns is upper bounded by $O(1)2^{R^2-(1-\epsilon)nR}$.

The final bound they obtain is the folowing (for a small η):

$$\mathbb{P}_{red}^{good} \leq O(1) \cdot 2^{-n/2+n\eta/2}$$

This concludes the proof, since $\binom{n}{n'} \cdot 2^{-n/2+n\eta/2}$ is $o(1)$ for sufficiently small c and ρ_{enc}.

4 Generalisation of the result

As the weight of the product of two homological code is linear in the weight of the two factors and (with high probability) the distance of the product if the product of the distance of the two factors, we get a $[[n^2, O(n^2), O(n^2), O(n)]]$ QECC when we take the homological product of two random codes. The natural question to ask is whether the same thing is true if we take a product of k factors. If the distance is still multiplicative, then we would have a $[[n^k, O(n^k), O(n^k), O(n)]]$ QECC, that is to say a $[[n, O(n), O(n), O(n^{1/k})]]$ QECC, which would decrease a lot the previous best weight for good QECC that we saw in introduction.

Proposition 4.0.1. *The homological product is associative.*

Proof. Direct application of the definitions. □

Some results generalize well, like the **Künneth formula**:

Proposition 4.0.2. *If the δ_i are boundaries operators, and ∂ their product, then*

$$\ker \partial = \bigotimes_{i=1}^{n} \ker \delta_i + \operatorname{Im} \partial$$

Proof. The case $n = 2$ is treated in [3], let's do the other cases by induction.

Assume the formula is true for $n-1$ ($n \geq 3$).

Let δ_i be n boundaries operators, and ∂ their product. Let ∂' be the product of $\delta_1, \dots \delta_{n-1}$. By the induction hypothesis, $\ker \partial' = \bigotimes_{i=1}^{n-1} \ker \delta_i + \operatorname{Im} \partial$.

As the homological product is associative, ∂ is the homological product of ∂' and δ_n. By the $n = 2$ case of the **Künneth formula**,

$$\ker \partial = \bigotimes_{i=1}^{n} \ker \delta_i + \operatorname{Im} \partial' \otimes \ker \delta_n + \operatorname{Im} \partial$$

Let us show that the second term $\operatorname{Im} \partial' \otimes \ker \delta_n$ is included in $\operatorname{Im} \partial$.

Take $y \in \operatorname{Im} \partial' \otimes \ker \delta_n$. $y = \sum_{i=1}^{r} \partial'(h_i) \otimes g_i$ with $h_i \in E$ and $g_i \in \ker \delta_n$. Then I claim that $y = \partial \sum_{i=1}^{r} h_i \otimes g_i$.

$$\begin{aligned} \partial \sum_{i=1}^{r} h_i \otimes g_i &= \sum_{i=1}^{r} \partial h_i \otimes g_i = \sum_{i=1}^{r} (\partial' \otimes I) h_i \otimes g_i + (I \otimes \delta_n) h_i \otimes g_i \\ &= \sum_{i=1}^{r} \partial' h_i \otimes g_i + h_i \otimes g_i = \sum_{i=1}^{r} \partial' h_i \otimes g_i = y \end{aligned}$$

□

In order to simplify the study, I only made precise computations about the 3 factor case.

Let ∂ be the homological product of δ_1, δ_2 and δ_3. The dimension of the space is n.

Take ϕ a non trivial cycle for ∂ with weight less than cn^3, then one can reduce it to a reduced cycle with all it 2d slices of weight less than cn'^2. I call this property, as in the 2-tensor case, the Uniform Low Weight (ULW) condition.

First, I will introduce some invariants I studied in order to answer this questions, then I will explain another question I was ask, which is the question of the distribution of the weight of the matrices given their rank.

4.1 Invariants

The reason why the rank appears in the formula of Γ (3) is because this is a quantity invariant under the action of $GL_n(\mathbb{F}_2)^2$. More precisely, the rank is the characteristic invariant of the orbits of $(F_2^n) \otimes (F_2^n)$ under the action of $GL_n(\mathbb{F}_2)^2$.

Proposition 4.1.1. *Let K be an arbitrary field, $r \leq n$ two integers.*

$$\forall M \in M_n(K), rk(M) = r \iff \exists P, Q \in GL_n(K), PMQ = \begin{pmatrix} I_r & 0 \\ 0 & 0 \end{pmatrix}$$

In the 3-tensor case, the set $\left(F_2^n \otimes F_2^n \otimes F_2^n\right) / GL_n(\mathbb{F}_2)^3$ is not well understood, we then cannot use the same argument: we do not know characteritic invariant of the orbit of $F_2^n \otimes F_2^n \otimes F_2^n$ under the action of $GL_n(\mathbb{F}_2)^3$.

I then studied some invariants under the action of $GL_n(\mathbb{F}_2)^3$ in order to find one which has the following characteristics:

- We want to be able to compute the probability that a uniform tensor with this invariant has the LWC.
- We want to compute the number of cycles with a given value of this invariant, or at least a good upper bound on it.

Those characteristics are here to allow me to link probabilities to algebraic properties that I can study.

None of those quantities is a characteristic invariant of the orbits, but as the orbit are not well understood, it seemed reasonable to look at only weak invariants.

In what follows, n will be the dimension of the space. It is a power of 2.

4.1.1 Rank

Definition 4.1.2. *Let $\psi \in A \otimes B \otimes C$ be a tensor. Its rank is defined as:*

$$rk(\psi) = \min \left\{ r \in \mathbb{Z}_{\geq 0} \;\middle|\; \psi = \sum_{i=1}^{r} a_i \otimes b_i \otimes c_i, a_i, b_i, c_i \in A, B, C \right\}$$

The tensor rank is an invariant under the action of $GL_n(\mathbb{F}_2)^3$, but it is not characteristic in general, in contrary to the matrix case (see for example [8]).

The first problem was that computing the rank of a tensor is an NP-complete problem ([5]), which firstly tends to indicate that question linked to the tensor rank should be complicated and secondly increase the complexity of doing numerical computations and samples.

In order to understand if the rank of the tensor could be a good candidate to be an invariant for our study, we looked at the case of rank 1 tensors which are simpler to consider since they form a single orbit.

In the case of the study of the rank, since this was the first invariant I studied, I tried to study it in the case of the product of k factors.

Proposition 4.1.3. *Let ∂ be the homological product of the $(\delta_i)_{i=1\dots k}$, then let $\Gamma(1)$ be the number of reduced cycles of ∂ of rank 1. Then*

$$\Gamma(1) \leq O(1) \cdot 2^{krn + \frac{1+\rho_{enc}}{2}^{k/2} n^{k/2}}$$

With $n' = (1-r)n$.

Proof. The number of extensions of a given rank 1 tensor of size $n' \times \cdots \times n'$ to a rank 1 tensor of size $n \times \cdots \times n$ in $\mathbb{F}_2$ is $2^{n-n'^k} = 2^{krn}$.

I bound the number of rank 1 tensor in the kernel of ∂' by using the same kind of bound proven in [3] by considering ∂ as the homological product product of two boundary operators (on the space $E^{\otimes k/2}$). The product of those two quantities leads to the bound. □

The probability for a rank 1 tensor to be of uniform low weight is easy to compute

Proposition 4.1.4. *Let $\psi = v_1 \otimes \ldots \otimes v_k$ be a uniform rank 1 tensor. Then*

$$\mathbb{P}(\psi \ ULW) = \mathbb{P}(\forall i, w(v_i) \leq c'n') = \mathbb{P}(w(v_1) \leq c'n')^k = O(1)2^{-nk(1-h(c'))}$$

With $h(\cdot)$ the binary entropy function.

The overall bound is then, with ψ uniform rank 1 tensor.

$$\Gamma(1)\mathbb{P}(\psi \text{ ULW}) \leq O(1)2^{-nk(1-h(c'))+krn+\frac{1+\rho_{enc}}{2}^{k/2} n^{k/2}}$$

Which is clearly not $o(1)$ when $k \geq 3$.

I have to say that I think that this bound is far from close. I wrote a program to approximate the number of rank 1 vector in the kernel of an homological operator, by Bayesian inference. The amount of memory needed to do my computation (when we do the computation for a 4 factor product, we need to store $O(n^8)$ values for the matrix of ∂) is too large to raise n too large, but the logarithm of the number of rank 1 vector in the kernel seems to grows linearly in n, and not in n^2, as implied by the bound. More precise computation seems to be needed in order to refine the bound.

The rank did not seems to be a nice invariant to work with, seen the fact that it computation is very difficult and the bound I managed to prove were not good.

4.1.2 Analytic rank

Another invariant I studied during the end of the internship was the analytic rank.

Definition 4.1.5. *Let $T \in A \otimes B \otimes C$ be a tensor, seen as a multilinear form $T : A \times B \times C \to \mathbb{F}_2$. Then the bias of T is defined to be*

$$bias(T) = \mathbb{E}_{x,y,z}((-1)^{T(x,y,z)}) = \mathbb{P}_{x,y}(T(\cdot, x, y) = 0)$$

And from this quantity, the analytic rank is defined: $arank(T) = -\log(bias(T))$

This quantity is clearly invariant by the action of GL_n^3, and seems more related to the weight of the slices of the tensor.

Proposition 4.1.6.

$$\mathbb{P}_{x,y}(T(\cdot, x, y) = 0) = \mathbb{P}_{x,y}(T(e_i, x, y) = 0 \ \forall i)$$

This way of considering a tensor as a collection of some linear form, even if it has not be as usefull as expected, lead me to consider another problem.

Let take $T \in A \otimes A \otimes A$ a tensor, with $A = \mathbb{F}_2^m$. Let U_n, V_n, W_n be a sequence of random variables i.i.d. in $M_{n,m}(\mathbb{F}_2)^3$ for all n. Now let $T_n = (U_n \otimes V_n \otimes W_n)T \in (F_2^n)^{\otimes 3}$.

The question is then to compute the asymptotic probability of T_n to be ULW.

The first fact to emphase is that evaluating a coeficient $(i, j, k)$of T_n is the same as evaluating T_n at (e_i, e_j, e_k), and that this is the same as evaluating T at the point u_i, v_j, w_k, if u_i (resp v_j, w_k) is the ith (resp the jth or kth) column vector of U_n (resp V_n, W_n), which is an uniform random vector of A.

Another fact is that for a sufficiently large n (say $n \gg 2^m$), we can expect with high probability that every vector of A will be drawn in the columns vectors of U_n, and V_n.

This justifies the introduction of the family of random variables $(N_a^U)_{a\in A}$, $(N_a^V)_{a\in A}$ by:

$$N_a^U = \#\{i \in [1, m],\ u_i = a\}$$
$$N_a^V = \#\{i \in [1, m],\ v_i = a\}$$

Where $U_n = [u_1, \ldots, u_n]$, $V_n = [[v_1, \ldots, v_n]$.

The random variables N_a^U (resp N_a^V) follow the multinomial law, of parameters $\left(n, \frac{1}{|A|}\right)$. The interesting value is the weight of a slice of T_n, that is to say, for $k \in [1, n]$ (Where $W = [w_1, \ldots, w_n]$):

$$S = \sum_{a,b\in A^2 \ T(a,b,w_k)\neq 0} N_a^U N_b^V$$

We then want to bound the probability for S to be less or equal to cn^2. We can do that using a Gaussian approximation. For that, we need the expectation of those random variables.

First, let $\alpha_k = \frac{\#\{a,b\in A^{\otimes 2},\ T(a,b,w_k)=1\}}{|A|^2}$ and $p = 1/|A|$. The collection of the α_k is invariant In the following, I will omit the k.

Proposition 4.1.7.

$$\mathbb{E}(S) = \alpha n^2$$

Proof. First, let us recall that the mean of a multinomial is: $\mathbb{E}(N_a^U) = np$. Then the formula follows directly from the fact the N_a^U and N_b^U are independent from each other. □

The variance is also needed to compute any gaussian approximation of S, but I made several attempt to compute it and leaded to complicated, unusable (and possibly false) formulas. In order to have easier computations, I used a process called "Poissonization".

Proposition 4.1.8. *Let n be a Poisson random variable with parameter λ, and (N_a^U), (N_a^V) independent multinomial of parameters (n, p). Then the marginal distribution of (N_a^U), (N_a^V) are independent poisson variable of parameters $p\lambda$.*

That independence gives us an easier way to compute the mean and variance of S.

Proposition 4.1.9.

$$\mathbb{E}(S) = \alpha\lambda^2$$

$$\mathbb{V}(S) = \alpha(\lambda^2 + 2p\lambda^3)$$

Proof. The proof for the mean is the same as before.

$$\mathbb{V}(n_a n_b) = \mathbb{E}(n_a)^2\mathbb{V}(n_b) + \mathbb{E}(n_b)^2\mathbb{V}(n_a) + \mathbb{V}(n_a)\mathbb{V}(n_b) = 2(p\lambda)^3 + (p\lambda)^2.$$

As the N_a^U, N_a^U are iid, (I take $T = T(\cdot, \cdot, w_k)$)

$$\mathbb{V}(S) = \sum_{a,b \in A\times A} T(a,b)^2\mathbb{V}(N_a^U N_b^V) = \#A^2\alpha(2(p\lambda)^3 + (p\lambda)^2) = \alpha(\lambda^2 + 2p\lambda^3)$$

□

A remaining thing to do would be to bound the probability that T_n has ULW by gaussian approximation, which I started at the very end of my internship and therefore had not the time to finish.

4.1.3 Slice collection

Definition 4.1.10. *Let $A, B, C = F_2^n$. Let $\psi \in A \otimes B \otimes C$. ψ can be seen as a linear map $\psi : A^* \to B \otimes C$, that is to say a map $\psi : F_2^n \to M_n(F_2)$.*

*We call the **slice collection** of ψ the collection $(s_1, \dots, s_n)$, with*

$$\forall r,\ s_r = \#\{M \in \mathrm{Im}(\psi),\ \mathrm{rank}(M) = r\}$$

Remark 4.1.11. *The choice of considering $\psi : A^* \to B \otimes C$ and not $\psi : B^* \to A \otimes C$ or $\psi : C^* \to A \otimes B$ is arbitrary (this is the "slice direction"), but the results presented here are independent of it.*

Proposition 4.1.12. *The slice collection of a tensor is invariant by the action of Gl_n^3*

Proof. Let $\psi : A^* \to B \otimes C$ a tensor (same notation as in the definition).

Let $g_1, g_2, g_3 \in GL_n^3$. It acts as follows: $\forall x \in A^*$, $(g_1, g_2, g_3)\psi(x) = g_2\psi(g_1 x)g_3$. Hence, the action of g_1 does not change the image of ψ, and for all $a \in A^*$, we have that $\mathrm{rank}(g_2\psi(a)g_3) = \mathrm{rank}(\psi(a))$. □

When a slice collection is fixed, if we take the slice decomposition (see Fig 1) a random tensor with this slice collection, then this is easy to compute the probability that the ith slice of this tensor has rank r_i. With this known, we can start to try to compute the weight of the slices of such a tensor.

The problem is then to bound the probability of a matrix of a given rank to have total weight less than cn^2, this is the question of the next subsection.

First, we have the following result:

Proposition 4.1.13. *[6, Theorem 2.1] If M is sampled uniformly in the set of rank r matrices, then for all i, j, $\mathbb{P}(M_{ij} = 1) = \mathbb{P}(M_{11} = 1)$, this quantity is called the average weight per entry.*

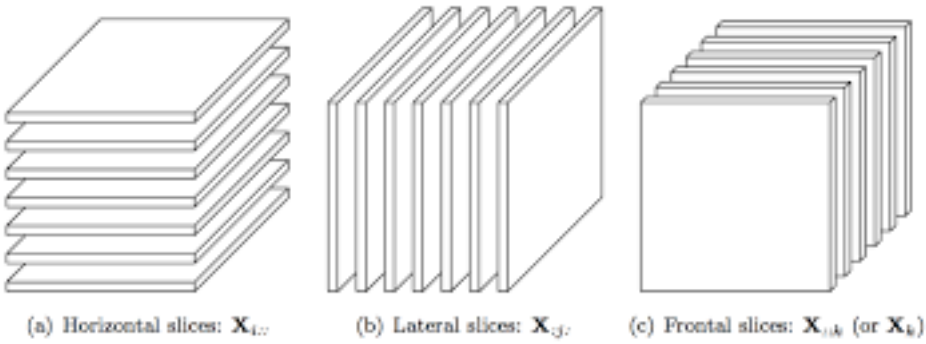

Figure 1: Slices of a 3-tensor

And it appears that we have an exact formula for this quantity

Proposition 4.1.14. *[6, Theorem 2.3] If M is sampled uniformly in the set of $n \times n$ matrices of rank r, then*

$$\mathbb{P}(M_{ij} = 1) = \frac{1}{2} \frac{1 - 1/2^r}{(1 - 1/2^n)^2}$$

Corollary 4.1.15. *If M is sampled uniformly in the set of $n \times n$ matrices of rank r, then*

$$\mathbb{E}(W(M)) = \frac{n^2}{2} \frac{1 - 1/2^r}{(1 - 1/2^n)^2}$$

The problem is then to find more precise informations about the weight of random matrices of a given rank in order to understand them better.

In order to have experimental results, I made several Python program in order to see how the weight of random matrices behave in function of the rank. The codes and method used to have those curves are presented in the Appendix A.2.

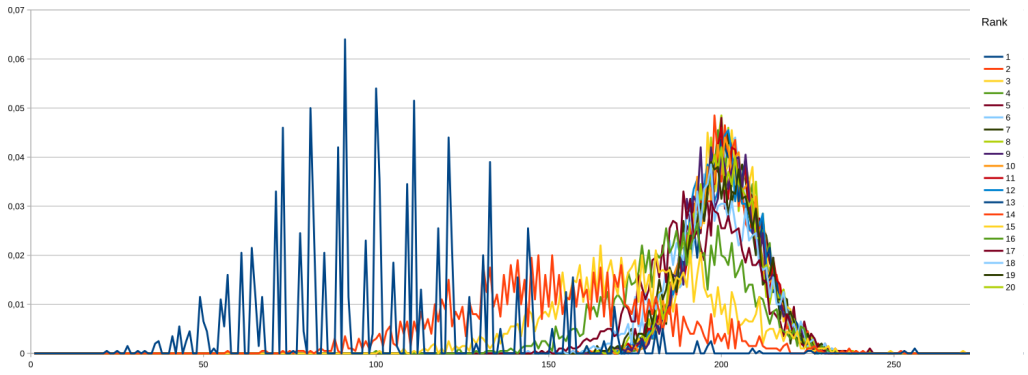

Figure 2: Distribution of the weight of randomly choosen 20×20 matrices in $\mathbb{F}_2$ in function of their rank

As we can see in Figure 2, the higher the rank is, the more the distribution of the weight seems to behave like a Gaussian, centered in its expected value.

As the exact distribution of the weight of fixed rank matrices should be very hard to compute, I wanted to approximate this by a Gaussian. This asks to know the second moment of the weight of a random matrix of fixed rank.

4.2 Weight of a matrix of a given rank

I explored several ways of constructing matrices, with control of their rank.

The idea behind it is that if one can contruct a random variable M_r for $r = 1 \dots n$ such that M_r is of rank at most r, such that the distribution of M_r given its rank is uniform in the set of matrix of a given rank and such that ones knows some information about the moments of the weight of M_r, we will be able to find the kth moment of a uniform rank r matrix.

More formally:

Proposition 4.2.1. *If (M_r) is a collection of random variables with values in $M_n(\mathbb{F}_2)$, $f : M_n(\mathbb{F}_2) \to \mathbb{Z}_{\geq 0}$ a function such that for all $r = 1 \dots n$:*

- $\text{rank}(M_r) \leq r$
- *For all $k = 1 \dots r$, the random variable $(M_r / \text{rank} = k)$ is uniform in the set of rank k matrices*
- *We know $\mathbb{E}(f(M_r))$ and $\mathbb{P}(rk(M_r) = k)$ for all k*

Then we can, by a matrix inversion, deduce $\mathbb{E}(f(M))$ with M uniform of rank $r = 1 \dots n$, by the collection of equations (for all r):

$$\mathbb{E}(f(M_r)) = \sum_{k=0}^{r} \mathbb{E}_{\text{rank}(M)=k}(f(M))\mathbb{P}(rk(M_r) = k)$$

The interesting functions f are the powers of the weight of the matrices, because with them we can hope to compute some Gaussian approximation of the weight of a random matrix of fixed rank by computing the moments of the weight.

4.3 Sum of rank 1 matrices

Proposition 4.3.1.

$$\text{rank}(M) = \min \left(r \in \mathbb{Z}_{\geq 0}, \quad M = \sum_{i=0}^{r} {v_i}^t w_j \right)$$

Proof. This is a direct application of the Gaussian elimination algorithm. □

Then a way of constructing matrices of a certain maximal rank is to add up random products of vectors.

More precisely, I take n an integer (the size of the space), r an integer less than or equal to n, and $X_1 \dots X_r$ and $Y_1 \dots Y_r$ some independent random uniform vectors in $\mathbb{F}_2^n$. I then set

$$M_r = \sum_{i=1}^{r} X_i {}^t Y_i$$

Proposition 4.3.2. *This construction satisfies the hypothesis of Prop 4.2.1.*

Proposition 4.3.3.

- $\mathbb{E}(M_r) = \frac{n^2}{2}\left(1 - \frac{1}{2^r}\right)$
- $\mathbb{E}\left(M_r^2\right) = n^2\left(\frac{3}{4} - \frac{3}{2}\frac{1}{2^r} - \frac{3}{4}\frac{1}{2^{2r}} + n\left(\frac{1}{2^r} - \frac{1}{2^{2r}}\right) + n^2\left(\frac{1}{4} - \frac{1}{2^{r+1}} + \frac{1}{2^{2r+2}}\right)\right)$

Proof. See Appendix A.1 □

As we add random matrices at each "step", the rank of M_r is a Markov chain. We can then try to study it in order to find the quantity $\mathbb{P}(rk(M_r) = k)$.

For simplicity, by now we will have a fixed collection of random variables $X_1 \dots X_n$ and $Y_1 \dots Y_n$ (n is fixed) iid uniform in $\mathbb{F}_2^n$, and we let $M_l = \sum_{i=1}^{l} X_i {}^t Y_i$.

At each step, the rank of M_t can only increase of 1, decrease of 1 or stay the same. We denote by respectively those probabilites u_r, d_r and s_r (they depend only on the rank r of M_t)

Proposition 4.3.4.

- $u_r = (1 - 2^{n-r})^2$
- $d_r = \frac{(2^r-1)2^{r-1}}{2^{2n}}$

Proof. $M_{t+1} = M_t + v_{t+1}{}^t w_{t+1}$. If w_{t+1} or v_{t+1} are 0, the rank does not change, so let's assume they are different from 0.

If M_t is of rank r, there are $P, Q \in GL_n$ such that $PM_tQ = J_r$ with $J_r = \begin{pmatrix} I_r & 0 \\ 0 & 0 \end{pmatrix}$. As $\operatorname{rank} M_t + v_{t+1}{}^t w_{t+1} = \operatorname{rank} P(M_t + v_{t+1}{}^t w_{t+1})Q$ and as the application of invertible operators leaves the distribution of v_t and w_t unchanged, we can assume that at step t, $M_t = J_r$.

Let $I_t =< v_1, \dots, v_t >, K_t =< w_1, \dots, w_t >$. The rank will increase if and only if $v_{t+1} \notin I_t$ and $W_{t+1} \notin K_t$, so with probability $u_r = (1 - 2^{n-r})^2$.

If v_{t+1} is in I_t (with probability $\frac{2^r-1}{2^n}$) (and different from 0), then there exists $A \in GL(I_t)$ that sends v_{t+1} to e_1. Then we can conjugate by $\begin{pmatrix} A & 0 \\ I & 0 \end{pmatrix}$ without changing the rank and changing the distribution of w_{t+1}.

Then we have, with $(w_1, \dots, w_n)$ uniform, $\operatorname{rank}(M_{t+1}) = \operatorname{rank}(S)$ with S the following matrix:

$$\begin{bmatrix} 1+w_1 & w_2 & w_3 & \cdots & w_r & \cdots & w_n \\ 0 & 1 & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & 0 & \ddots & & 0 & \cdots & 0 \\ 0 & 0 & \cdots & \cdots & 1 & \cdots & 0 \\ 0 & 0 & \cdots & \cdots & 0 & \cdots & 0 \\ & & & (0) & & & \end{bmatrix}$$

And the rank of this matrix is $r - 1$ if and only if $w_1 = (1, w_2, \dots, w_r, 0, \dots, 0)$, which occurs with probability $\frac{2^{r-1}}{2^n}$. The same holds is $w_{t+1} \in W_t$, then the overall probability is then the expected one. □

The transition matrix of this Markov chain is then:

$$\begin{bmatrix} 1-(1-2^n)^2 & (1-2^n)^2 & 0 & \cdots & 0 & 0 & 0 \\ 2^{-2n} & 1-u_1-d_1 & \left(1-2^{n-1}\right)^2 & \cdots & 0 & 0 & 0 \\ \vdots & & & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & d_{n-1} & s_{n-1} & u_{n-1} \\ 0 & 0 & 0 & \cdots & 0 & \frac{(2^n-1)2^{n-1}}{2^{2n}} & 1-\frac{(2^n-1)2^{n-1}}{2^{2n}} \end{bmatrix} \tag{4}$$

Conjecture 4.3.5. *The eigenvalues of this matrix are* $1, 2^{-1}, 2^{-2}, \dots, 2^{-n}$

I ended up with that conjecture after entering this matrix in Python and computed its eigenvalues for $n = 1 \dots 20$.

I started to compute the eigenvector and the eigenvalues of this transition matrix, but it appeared that it was very complicated and lead to unusable formulas, I then looked for another Markov chain.

4.4 Adding up columns

Another approach to the problem is to build a matrix by simply adding columns one after the other.

If we want to formalize this construction: take $X_1, \dots X_n$ n iid uniform random column vectors in $\mathbb{F}_2^n$, and set $M_t = [X_1, \dots X_t]$.

Then the rank of M_t is a Markov chain. At each step the rank can only stay the same or increase.

Proposition 4.4.1.

- $\mathbb{P}(\operatorname{rank}(M_{t+1}) = r + 1 / \operatorname{rank}(M_t) = r) = 1 - 2^{r-n}$
- $\mathbb{P}(\operatorname{rank}(M_{t+1}) = r / \operatorname{rank}(M_t) = r) = 2^{r-n}$

Proof. The rank of M_{t+1} is the same as the rank of M_t if and only if the vector added is inside the space spanned by the first t vectors, which is of dimension $\text{rank}(M_t)$. □

The transition matrix of this Markov chain is then:

$$\Sigma = \begin{bmatrix} 2^{-n} & 1-2^{-n} & 0 & \cdots & 0 & 0 \\ 0 & 2^{1-n} & 1-2^{1-n} & \cdots & 0 & 0 \\ 0 & 0 & \ddots & & \vdots & \vdots \\ \vdots & \vdots & (0) & & 1/2 & 1/2 \\ 0 & 0 & & & 0 & 1 \end{bmatrix}$$

This matrix is simpler than the one of the Subsection 4.3 because it is triangular. We then know its eigenvalues, which are $(2^{-i})_{i=0\dots n}$ and its eigenvectors can be computed.

Proposition 4.4.2. *Let $i \in \{0 \dots n\}$, and $x^{(i)} \in \mathbb{R}^{n+1}$ be the following vector:*

$$\forall r \in \{0 \dots n\}\,,\ x_r^{(i)} = 2^{-ri} \prod_{k=0}^{r-1} \frac{1-2^{k-n+i}}{1-2^{k-n}} \tag{5}$$

$$= \prod_{k=0}^{r-1} \frac{2^{n-k-i}-1}{2^{n-k}-1} \tag{6}$$

Then for all $i \in \{0 \dots n\}$, $x^{(i)}$ is the eigenvector of Σ associated to the eigenvalue 2^{-i}.

Proof. The equation $\Sigma x = 2^i x$ can be rephrased:

$$\text{For } r = 0 \dots n-1, \quad 2^{r-n}x_r + (1-2^{r-n})x_{r+1} = 2^{-i}x_r \tag{7}$$

$$\iff \quad x_{r+1} = \frac{2^{-i}-2^{r-n}}{1-2^{r-n}} x_r \tag{8}$$

$$\iff \quad x_{r+1} = 2^{-i}\frac{1-2^{r-n+i}}{1-2^{r-n}} x_r \tag{9}$$

$$\text{For } r = n, \quad x_n = 2^{-i}x_n \tag{10}$$

These equations are satisfied by the $x^{(i)}$. Eq 8 follows directly from Eq 5, and Eq 10 follows from the fact that if $i \neq 0, x_n^i = 0$. □

The problem is that the computation of the nth power is hard and I didn't manage to make it in the time I had.

5 Conclusion

5.1 Work done

After understanding the theory of QECC, I acquired a good understanding of the proof of [3]. proceeded to identify and determine whether it could be extended in terms of application.

The main effort and focus consisted on understanding the different invariants of the 3-tensors under the action of $GL_n(\mathbb{F}_2)^3$. For each of those invariants, I did attempt to understand the distribution of the weight of the tensors to create bounds on the probability that a reduced cycle has the uniform low weight condition. I specifically worked on the ranking, the analytic rank and the slice collection. I then worked on the related question of the distribution of the weight of matrices in function of their rank.

To do so, I studied several invariants and several matrix constructions algorithms which led me to study further to Markov's chains.

5.2 Perspectives

The work performed here was to dig in several directions in order to see the whether the result could be improved. This question remains open, and so is the question of the existence of $[[n, O(n), O(n), O(\sqrt[k]{n})]]$ QECC for $k \geq 3$.

In the case of the 3-factors homological product of random codes, the answer should be related to the invariants of the 3-tensors and hence those invariants still need to be studied. Another potetial axis of research is the case of the product of more than 3 homological codes. As the Künneth formula still works with more than 3 factors, the case of 4 or more factors could be studied.

Some additional work remains to be completed regarding the constructions I have developed. I did not end up with the second moments of the weight of a uniform matrix of rank r. If this result is known for the rank 1, I did not find any reference about the general case, or about full rank matrices. I think the Markov chains developed during this internship are worth studying since they seems to have interesting properties. A start point could be to prove that the Markov Chain 4, for whose I strongly conjecture that the eigenvalues are the inverse of the power of two. Understanding and diagonalizing this Markov Chain could lead to a formula for the moments of the weight of random rank r matrices.

In order to have more experimental results, one could also improve a lot the Python code I produced, in order to obtain more data, better algorithms than Gaussian elimination could be used for example (for example algorithms based on the Strassen algorithm, in $O(n^{2.81})$).

5.3 Context of the internship

This internship was realized in QMATH, University of Copenhagen under the supervision of Peter Vrana.

It was the first time that I went abroad alone, and this was a difficult exercise. I didn't know the culture nor the language of Denmark, and I didn't know anyone there. I learned to join and cooperate with an existing team and to speak English daily, which was quite challenging at the beginning.

Even if I experienced challenging times, especially during the first weekends when alone, I learned how to overcome quickly this loneliness by actively working on numerous activities such as reading and planning weekends in France. I think that from now on and considering my future work and experiences abroad I will now be able and prepared to manage a speedy integration with other cultures and people. Apart from this initial challenge, I have found the team to be very friendly and I enjoyed working with them. There were several nationalities represented and I was very well welcomed by Peter and by the rest of the team.

References

[1] H. Bombin and M. A. Martin-Delgado. Homological error correction: Classical and quantum codes. 48, 06 2006.

[2] N. Bourbaki. *Commutative Algebra: Chapters 1-7*. Springer-Verlag New York, 1989.

[3] S. Bravyi and M. B. Hastings. Homological product codes. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 273–282. ACM, 2014.

[4] A. R. Calderbank and P. W. Shor. Good quantum error-correcting codes exist. *Physical Review A*, 54(2):1098, 1996.

[5] J. Håstad. Tensor rank is np-complete. *Journal of Algorithms*, 11(4):644 – 654, 1990.

[6] T. Migler, K. E. Morrison, and M. Ogle. Weight and rank of matrices over finite fields. *arXiv preprint math/0403314*, 2004.

[7] M. A. Nielsen and I. Chuang. *Quantum computation and quantum information*. AAPT, 2002.

[8] M. R. Bremner and J. Hu. Canonical forms of small tensors over f_2. 06 2012.

[9] J.-P. Serre. *Linear representations of finite groups*, volume 42. Springer Science & Business Media, 2012.

[10] J. Tillich and G. Zémor. Quantum LDPC codes with positive rate and minimum distance proportional to n^{1/2}. *CoRR*, abs/0903.0566, 2009.

A Appendix

A.1 Misc proofs

Proof of Proposition 4.3.3.

First, let's take $X_1, \dots, X_r$ be vectors sampled in F_2^s iid according to some distribution such that $\mathbb{P}((X_1)_k = 1) = p$ and does not depends on k.

Let $S = \sum_{i=1}^r X_i$ and let $W = |S|$ the Hamming weight of S. We are going to compute the first two moments of W.

Let $Y_i = \sum_{k=1}^r (X_k)_i \mod 2$.

$$\begin{aligned}\mathbb{E}(Y_i) &= \sum_{\substack{k=0 \\ k \text{ odd}}}^{r} \binom{r}{k} p^k (1-p)^{n-k} \\ &= \sum_{k=0}^{r} \binom{r}{k} \frac{1-(-1)^k}{2} p^k (1-p)^{n-k} \\ &= \frac{1}{2}\left(1-(1-2p)^r\right)\end{aligned}$$

Then as $W = \sum_{i=1}^n Y_i$, $\mathbb{E}(W) = \frac{s}{2}\left(1-(1-2p)^r\right)$.

We have $Y_i^2 = Y_i$ because $Y_i \in \{0,1\}$. Then in order to compute $\mathbb{E}(W^2)$, we have to compute $\mathbb{E}(Y_i Y_j)$ with $i \neq j$.

Let $i \neq j$. For $a, b \in 0, 1$, let $p_{ab} = \mathbb{P}\left((X_1)_i = a, (X_1)_j = b\right)$. Note that it can depends on i and j (this dependence will not be written for the sake of simplicity).

$$\begin{aligned}\mathbb{E}(Y_i Y_j) &= \sum_{\substack{k_{00},k_{01},k_{10},k_{11} \\ \sum k_{ij} = r \\ k_{01}+k_{11} \text{ odd} \\ k_{10}+k_{11} \text{odd}}} \binom{r}{k_{00}, k_{01}, k_{10}, k_{11}} p_{00}^{k_{00}} p_{10}^{k_{10}} p_{01}^{k_{01}} p_{11}^{k_{11}} \\ &= \sum_{\substack{k_{00},k_{01},k_{10},k_{11} \\ \sum k_{ij} = r}} \binom{r}{k_{00}, k_{01}, k_{10}, k_{11}} \frac{1-(-1)^{k_{01}+k_{11}}}{2} \frac{1-(-1)^{k_{10}+k_{11}}}{2} p_{00}^{k_{00}} p_{10}^{k_{10}} p_{01}^{k_{01}} p_{11}^{k_{11}} \\ &= \frac{1}{4}\left(1-(p_{00}-p_{01}+p_{11}+p_{10})^r - (p_{00}-p_{01}-p_{11}-p_{10})^r + (p_{00}-p_{01}+p_{11}-p_{10})^r\right)\end{aligned}$$

When $s = n^2$, and the X_i are distributed as the product of 2 vectors. Let $U_1, \dots, U_r$ and $V_1, \dots, V_r$ uniform in $\mathbb{F}_2^n$ and $X_i = U_i {}^t V_i$.

Then, in that case we have, for $k \in [1, n^2]$.

$$\mathbb{P}((X_i)_k = 1) = \mathbb{P}((U_i)_k (V_i)_k = 1) = \mathbb{P}((U_i)_k = 1, (V_i)_k = 1) = \frac{1}{4}$$

And then in that case $\mathbb{E}(W) = \frac{n^2}{2}(1 - \frac{1}{2^r})$.

Now let us write the $k \in [1, n^2]$ as the coordinate of a matrix. Take a, b, c, d with $a \neq c$ and $b \neq d$, then

$$p_{11} = \frac{1}{16}, p_{01} = p_{10} = \frac{3}{16}, p_{00} = \frac{9}{16}$$

Now if $a = c$ or (exclusive or) $b = d$, computations shows that

$$p_{11} = p_{10} = p_{01} = \frac{1}{8}, p_{00} = \frac{5}{8}$$

Now if we apply those values to the preceding formulas (with $a \neq c, b \neq d$):

$$\mathbb{E}(Y_{ab}Y_{cd}) = \frac{1}{4}(1 - 2\frac{1}{2^r} + \frac{1}{2^{2r}}) \text{ and } \mathbb{E}(Y_{ab}Y_{ad}) = \frac{1}{4}(1 - \frac{1}{2^{2r}})$$

Then we have:

$$\begin{aligned}
\mathbb{E}(W^2) &= \sum_{a,b,c,d=1}^{r} \mathbb{E}(Y_{ab}Y_{cd}) \\
&= \sum_{a,b=1}^{r} \mathbb{E}(Y_{ab}^2) + \sum_{\substack{a,b,c=1 \\ b\neq c}}^{r} \mathbb{E}(Y_{ab}Y_{ac}) + \sum_{\substack{a,b,c=1 \\ a\neq c}}^{r} \mathbb{E}(Y_{ab}Y_{cb}) + \sum_{\substack{a,b,c,d=1 \\ a\neq c \\ b\neq d}}^{r} \mathbb{E}(Y_{ab}Y_{cd}) \\
&= n^2\mathbb{E}(Y_{11}^2) + 2n\binom{n}{2}\mathbb{E}(Y_{12}Y_{13}) + \binom{n}{2}^2\mathbb{E}(Y_{12}Y_{34})
\end{aligned}$$

And some more computations leads to the wanted formula.

A.2 Sampling random matrices

Proposition A.2.1. *Let $r \leq n$. If P, Q are two uniform elements of $Gl_n(\mathbb{F}_2)$, then the random variable PJ_rQ is uniform in the set of $\mathbb{F}_2$ $n \times n$ matrices of rank r. Where $J_r = \begin{pmatrix} I_r & 0 \\ 0 & 0 \end{pmatrix}$.*

Proof. Take A_1 and A_2 two rank r matrices. There exit P_1, Q_1 and P_2, Q_2 such that $P_iA_iQ_i = J_r$. Then, let P, Q be two random invertible matrices. The application $M \mapsto P_iMQ_i$ is bijective, therefore

$$\begin{aligned}
\mathbb{P}\left(PJ_rQ = A_1\right) &= \mathbb{P}\left(P_1^{-1}PJ_rQQ_1^{-1} = J_r\right) \\
&= \mathbb{P}\left(P_2P_1^{-1}PJ_rQQ_1^{-1}Q_2 = A_2\right)
\end{aligned}$$

And as $P_2P_1^{-1}$ and $Q_1^{-1}Q_2$ are invertible, the distribution of $P_2P_1^{-1}P$ and $QQ_1^{-1}Q_2$ is uniform in the set of invertible matrices, so

$$\mathbb{P}\left(P_2P_1^{-1}PJ_rQQ_1^{-1}Q_2 = A_2\right) = \mathbb{P}\left(PJ_rQ = A_2\right).$$

□

I used that property in order to sample uniform rank r matrices. The only remaining problem was to sample uniform elements of Gl_n. This is done by the technique of **sample and reject**.

Proposition A.2.2. *Let $E \subset F$ be two finite sets. Let X be uniform in F, then X conditioned on event "belongs to E" is uniformly distributed in E.*

The idea is then to sample uniformly choosen matrices in M_n, to test whether it is invertible or not, and if this is not the case, repeat this procedure until we find an invertible matrix.

Since the python library numpy does not handle the finite field case, I had to program a version of the gaussian pivot in order to test those things. The overall complexity is $O(1/p \cdot n^3)$. With p the probability of a matrix to be invertible.

Proposition A.2.3. *Let $\mathbb{F}_q$ be the field with q elements. Let p be the probability that a uniformly random $n \times n$ matrix is invertible. Then*

$$p \geq e^{-\frac{1}{q-1}}$$

Remark A.2.4. *Note that this bound does not depend on n*

Proof. Let n be an integer. Let us enumerate the number of invertible $n \times n$ matrices.

First, I choose the first row vector, it just has to be nonzero: $q^n - 1$ choices. Then, I choose the second row vector, it just has to not belong to the vector space spanned by the first row: $q^n - q$ choices. This procedure generalizes: at step k, I have to choose a vector not belonging to the vector space spanned by the $k-1$ first rows vectors: $q^n - q^{k-1}$ choices.

The total number of invertible $n \times n$ matrices in $\mathbb{F}_q$ is then

$$(q^n - 1)(q^n - q) \cdots (q^n - q^{n-1})$$

The probability for a matrix to be invertible is then

$$\frac{(q^n - 1)(q^n - q) \cdots (q^n - q^{n-1})}{q^{n^2}} = (1 - q^{-n})(1 - q^{-(n-1)}) \cdots (1 - q^{-1}) \geq \prod_{k=1}^{\infty} (1 - q^{-k})$$

Then, I apply ln:

$$\sum_{k \geq 1} \ln\left(1 - q^{-k}\right) \geq \sum_{k \geq 1} q^{-k} = \frac{1}{1 - q}$$

And then by application of exp, the result follows.

□

A.3 Code

You will find next some of the code I produced in order to generate random matrices and random tensors of a given rank.

```
def gaussian_pivot_F2(M, invert_only=False):
    m, n = M.shape
    M = np.mat(np.bmat([M, np.identity(m)]), dtype=np.int_) # m lines, n+m cols
    P = np.identity(n)
    j = 0
    l = 0

    nulCol = -1

    while j<=n+nulCol:
        piv = None
        for k in range(l, m):
            if M[k, j] != 0:
                piv = k
                break

        if piv is None:

            for f in range(m):
                M[f, j], M[f, n+nulCol] = M[f, n+nulCol], M[f, j]
            for f in range(n):
                P[f, j], P[f, n+nulCol] = P[f, n+nulCol], P[f, j]

            nulCol-=1
            continue
        else:
            for f in range(m+n):
                M[piv, f], M[l, f] = M[l, f], M[piv, f]
```

```
        for k in range(m):
            if k!= l and M[k, j] != 0:
                for f in range(m+n):
                    M[k, f] ^= M[l, f]
        j += 1
        l += 1

    if not invert_only:
        M1 = M.copy()
        return M[0:,:n], M[0:, n:], M1[0:l,:n], M1[0:l, n:], P, l
    else:
        if m !=n:
            return False
        if l != n:
            return False
        else:
            return M[0:, n:]

  def sample_invertible_matrix(n):
    mat = np.matrix([ [0]*n for _ in range(n)], dtype=np.int_)
    t = 1
    while True:
        mat = random_matrix_F2(n, n)
        t+=1
        inv = linalg.gaussian_pivot_F2(mat, True)
        if inv is False:
            continue
        return mat, inv

def sample_rank_r_matrix(n, r):
    matRank = np.bmat([[np.identity(r), np.zeros((r, n-r))],
                       [np.zeros((n-r,  r)), np.zeros((n-r, n-r))]])
    (U, _), (V, _) = sample_invertible_matrix(n), sample_invertible_matrix(n)
    return (U*matRank*V)%2
```