Exploring Abelian and Non-Abelian Quantum Codes

Simon David Burton

Doctor of Philosophy
University of Sydney
2016

Declaration

I declare that this thesis was composed by myself and that the work contained therein is my own, except where explicitly stated otherwise in the text.

(*Simon David Burton*)

To my friends

Abstract

Stabilizer codes... from the point of view of category theory joining stabilizer codes together comes for free once we have a suitable choice of what is meant by a homomorphism of such a code. The category of length chain complexes over Z_2 provides a natural homomorphism, namely the chain-maps. Category theory then automatically gives a definition of joining and this corresponds (partly?) to the previously discovered notion of welding of stabilizer codes. Turning to non-Abelian Hamiltonians as a possible way to construct quantum memories, we examine the eigenstructure of Hamiltonians with Pauli operator terms, from the perspective of the representation theory of the group formed by these terms. Finally, we show how to simulate an error correction scenario involving non-Abelian excitations in a two dimensional topologically ordered system.

Contents

Chapter 1

Fibered Sum of Stabilizer Codes

1.1 Introduction

1.2 Symplectic structure of stabilizer codes

We work with vector spaces over the field $\mathbb{Z}_2$.

A quantum CSS code is given by two parity check matrices S_z and S_x.

Such a code will be called *regular* when the parity checks have full rank.

Given a regular code, we can a symplectic structure is any solution to the following (block) matrix equation:

$$\begin{pmatrix} L_z \\ S_z \\ T_z \end{pmatrix} \begin{pmatrix} L_x \\ T_x \\ S_x \end{pmatrix}^{\top} = I,$$

where I denotes the appropriate identity matrix, and the small T is matrix transposition.

In general this is a non-linear equation because of the presense of the quadratic term: $T_z T_x^{\top} = 0$.

To construct solutions given S_z and S_x we proceed as follows:

(1) Find L_z. The rows of L_z lie in the kernel of S_x, chosen to be (arbitrary) elements of the cosets of $S_z^{\top}$. Ie. the rows of L_z span $\ker(S_x)/Im(S_z^t)$.

(2) Find L_x. We first repeat step *(1)* on the dual code (S_x and S_z swapped) to find

L'_x. We look for L_x such that $L_z L_x^\top = I$ knowing that the rows of L_x lie in the span of L'_x. Ie. $L_x = AL'_x$ for some A. Now solve $L_z L_x^\top A^\top = I$ for A.

(3) Find T_z. This will be a solution of the linear system:

$$\begin{aligned} (*) \quad S_x T_z^\top &= I \\ L_x T_z^\top &= 0. \end{aligned}$$

The solution space has kernel spanned by the rows of S_z.

(4) T_z is now a solution of the linear system:

$$\begin{aligned} S_z T_x^\top &= I \\ L_z T_x^\top &= 0 \\ T_z T_x^\top &= 0. \end{aligned}$$

From $(*)$ above, we know that T_z has full rank, and so this system has a unique solution T_x.

1.3 The boundary of the boundary is empty

1.3.1 Chain complexes

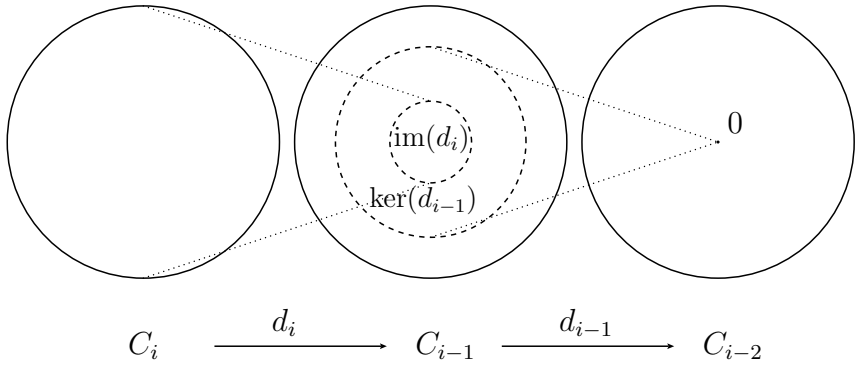
We introduce the category of chain complexes, $\widetilde{\text{Chain}}$.

A *chain complex* C is given by a sequence of vector spaces C_i and linear maps $d_i : C_i \to C_{i-1}$ such that $d_{i-1}d_i = 0$ for all i.

Here is a diagram:

$$\ldots \xrightarrow{d_{i+2}} C_{i+1} \xrightarrow{d_{i+1}} C_i \xrightarrow{d_i} C_{i-1} \xrightarrow{d_{i-1}} \ldots$$

The condition $d_{i-1}d_i = 0$ is equivelant to requiring the image of d_i to be contained within the kernel of d_{i-1}:



Elements of the space $B_i := \mathrm{im}(d_{i+1})$ are known as *boundaries*, and elements of $Z_i := \ker(d_i)$ are also known as *cycles*.

We form the quotient vector space $H_i(C) := B_i(C)/Z_i(C)$, called the i'th homology group (the group operation is given by the vector space addition.)

The sequence of spaces H_i will also be denoted as simply H. It can be taken to be a chain complex with the zero boundary map.

We can always consider finite length chain complexes by appending/prepending zero vector spaces and maps, for example $A \to B \to C$ can be extended as

$$... \to 0 \to A \to B \to C \to 0 \to ...$$

1.3.2 Chain maps

A chain map $f : C \to C'$ is a sequence of linear maps $f_i : C_i \to C'_i$ that commute (intertwine) with the boundary map: $f_{i-1}d_i = d'_i f_i$. Or in diagram form:

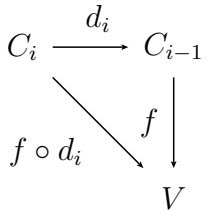
$$\begin{array}{ccccccccc} & \longrightarrow & C_2 & \xrightarrow{d_2} & C_1 & \xrightarrow{d_1} & C_0 & \longrightarrow & \\ ... & & \downarrow{\scriptstyle f_2} & & \downarrow{\scriptstyle f_1} & & \downarrow{\scriptstyle f_0} & & ... \\ & \longrightarrow & C'_2 & \xrightarrow{d'_2} & C'_1 & \xrightarrow{d'_1} & C'_0 & \longrightarrow & \end{array}$$

The main point about a chain map is that it induces a (linear) map of homology groups:

$$\tilde{f}_i : H_i \to H'_i.$$

1.3.3 The Hom functor

We consider the boundary operator acting by pre-composition:



Given a chain complex C and an arbitrary vector space V we see that the boundary map d_i acts on maps $f : C_{i-1} \to V$ to give a map $C_i \to V$. We will fix V to be the underlying field $\mathbb{Z}_2$ then this action is "multiplying on the right", ie. the transpose operation. In this way we construct the dual cochain.

$$\ldots \longleftarrow C^{i+1} \xleftarrow{d^i} C^i \xleftarrow{d^{i+1}} C^{i-1} \longleftarrow \ldots$$

This is the familiar covector construction. In general, the so-called hom functor reverses the direction of all the arrows (of some diagram.) In our case this means simply that the transpose of a product reverses the product: $(AB)^\top = B^\top A^\top$.

1.3.4 Classical linear codes

A classical linear code may be specified as the kernel of a parity check matrix $S : \mathbb{Z}_2^n \to \mathbb{Z}_2^m$. As a chain complex, this is the homology group at $\mathbb{Z}_2^m$.

As a notational convenience we will sometime denote a vector space by its (integer) dimension, eg. $S : n \to m$.

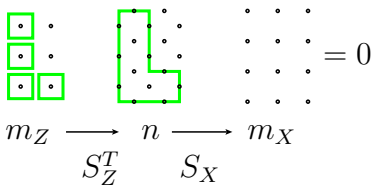
Chain	Cochain
$0 \longrightarrow n \xrightarrow{S} m \longrightarrow 0$	$0 \longrightarrow m \xrightarrow{S^T} n \longrightarrow 0$
$H_1 = \ker(S) =: L$	$H^0 = \ker(S^\top) =: L^\top$
$H_0 = m/\mathrm{im}(S) = \mathrm{coker}(S)$	$H^1 = n/\mathrm{im}(S^\top) = \mathrm{coker}(S^\top)$

1.3.5 Quantum stabilizer codes

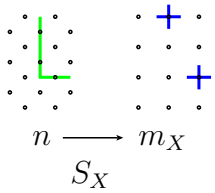
A quantum (CSS) code is given by two parity check matrices $S_X : n \to m_X$ and $S_Z : n \to m_Z$, such that the chain condition $S_X S_Z^\top = 0$ is satisfied.

Chain	Cochain
$0 \longrightarrow m_Z \xrightarrow{S_Z^T} n \xrightarrow{S_X} m_X \longrightarrow 0$	$0 \longrightarrow m_X \xrightarrow{S_X^T} n \xrightarrow{S_Z} m_Z \longrightarrow 0$
$0 \longrightarrow C_2 \longrightarrow C_1 \longrightarrow C_0 \longrightarrow 0$	$0 \longrightarrow C^0 \longrightarrow C^1 \longrightarrow C^2 \longrightarrow 0$
$H_2 = \ker(S_Z^\top)$	$H^0 = \ker(S_X^\top)$
$H_1 = \ker(S_X)/\mathrm{im}(S_Z^\top) =: L_X$	$H^1 = \ker(S_Z)/\mathrm{im}(S_X^\top) =: L_Z$
$H_0 = m_X/\mathrm{im}(S_X) = \mathrm{coker}(S_X)$	$H^2 = m_Z/\mathrm{im}(S_Z) = \mathrm{coker}(S_Z)$

In the chain we are thinking of the space m_Z as the space of "2-dimensional" objects. In the toric code this is the space of plaquettes, but more generally we can think of these as "generator labels". The space n is associated with the physical qubits, this is where the pauli operators reside.



In the toric code n is the space of "1-dimensional" error operators. The space m_X is then the space of (X-type) syndrome measurements. In the toric code these are the "zero-dimensional" end-points of (Z-type) error operators.



1.4 Tensor product

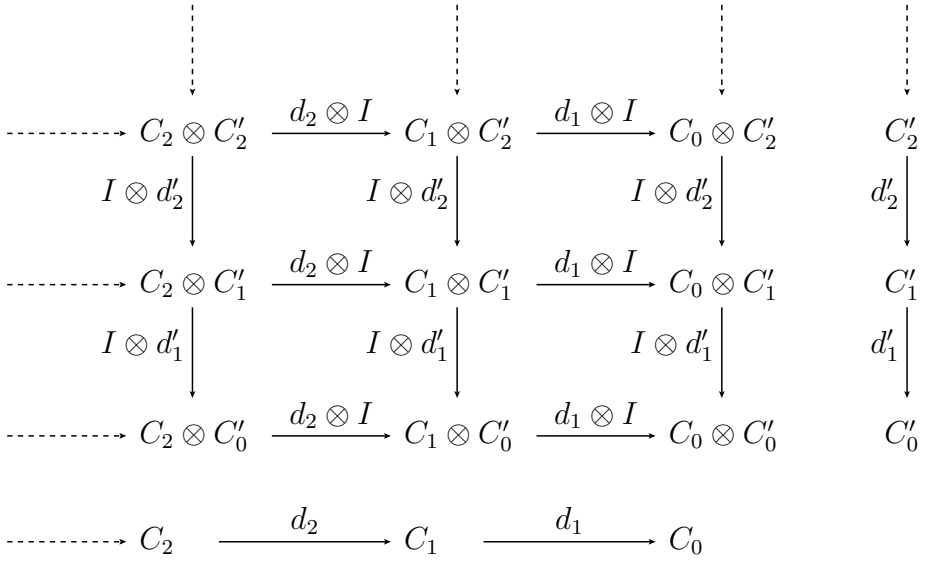
The tensor product $C \otimes C'$ of two chain complexes (C, d) and (C', d') is given by

$$(C \otimes C')_i = \sum_{j+k=i} C_j \otimes C'_k$$

with boundary map

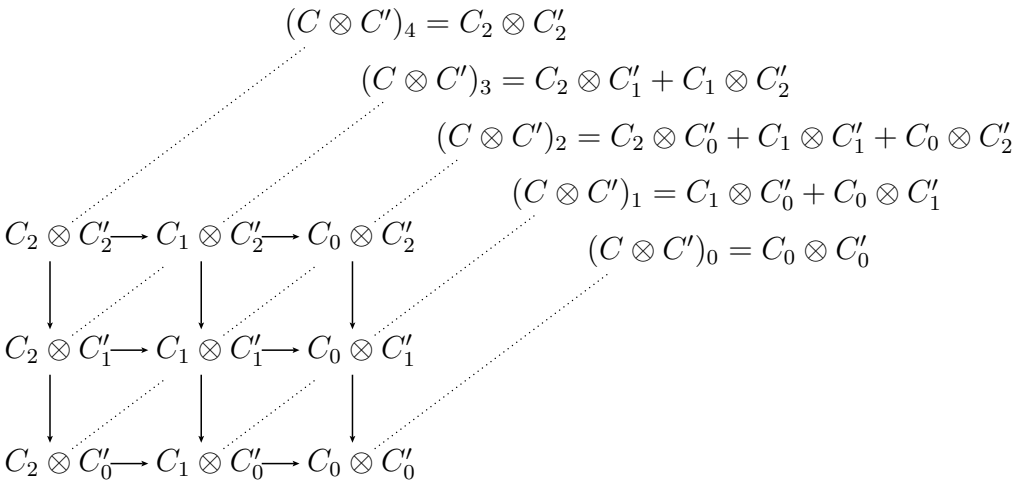
$$d(c \otimes c') = d(c) \otimes c' + (-1)^{deg(c)} c \otimes d'(c').$$

To motivate these formulae, consider the following two dimensional complex:



where I indicates the appropriate identity map on each vector space.

To reduce this to a one dimensional structure we (direct) sum along the diagonals, for example:



Now we add the arrows in an alternating fashion to get a boundary map. The composition of two arrows in the same direction is evidently zero, and we use an alternating

weight to force the two paths around each square to cancel:

$$\begin{array}{ccccc}
C_2 \otimes C_2' & \longrightarrow & C_1 \otimes C_2' & \longrightarrow & C_0 \otimes C_2' \\
+\downarrow & & -\downarrow & & +\downarrow \\
C_2 \otimes C_1' & \longrightarrow & C_1 \otimes C_1' & \longrightarrow & C_0 \otimes C_1' \\
+\downarrow & & -\downarrow & & +\downarrow \\
C_2 \otimes C_0' & \longrightarrow & C_1 \otimes C_0' & \longrightarrow & C_0 \otimes C_0'
\end{array}$$

With this definition of tensor product the category of chain complexes becomes a monoidal category (See [4], section 2.3 for a helpful discussion of monoidal categories.)

1.4.1 The Kunneth formula

The homology group of the product inherits the same structure as the underlying chain complex. This is the import of the Kunneth formula:

$$H_i(C \otimes C') = \sum_{j+k=i} H_j(C) \otimes H_k(C')$$

We now define a homomorphism $f : H(C) \otimes H(C') \to H(C \otimes C')$ by its action on the subspaces:

$$f : H_j(C) \otimes H_k(C') \to H_{j+k}(C \otimes C').$$

defined by choosing $u_j \in \ker(d_j), u_k' \in \ker(d_k')$ and then noting that

$$(d_j \otimes I)(u_j \otimes u_k') = 0, \quad (I \otimes d_k')(u_j \otimes u_k') = 0$$

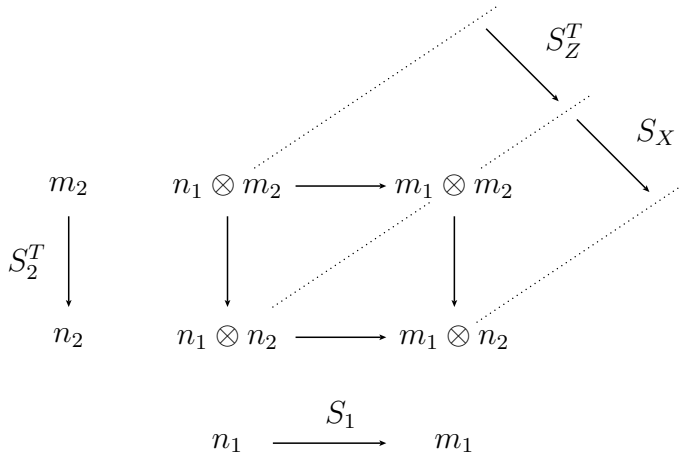
which means $u_j \otimes u_k'$ is in the kernel of the tensor product boundary map, and so represents an element of $H_{j+k}(C \otimes C')$. Next check that the choice of u_i, u_j' did not matter...

Weight of stabilizers... Weight of logops...

1.4.2 Product of two classical codes

The hypergraph product of Tillich and Zemor [34] is the product of a classical code and the dual of a classical code. We didn't define such a product above, but evidently

if we follow the arrows in the same way (or alternativy, relabel the cochain) it should all work out.



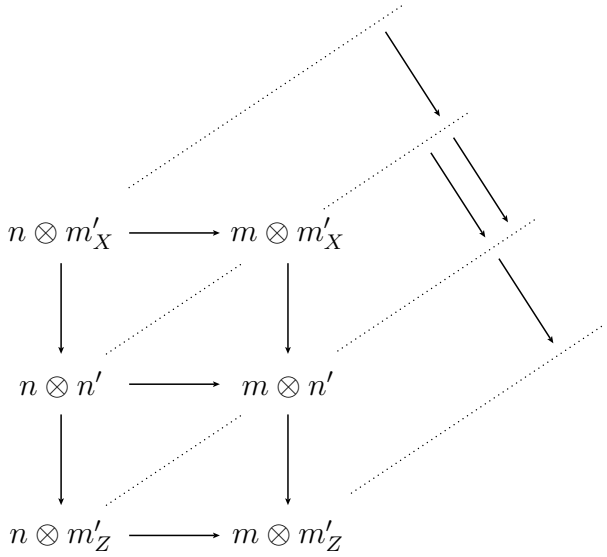
We use the Kunneth formulae to compute the logical operators:

$$
\begin{aligned}
H_1(C_1 \otimes C_2^\top) &= \mathrm{coker}(S_1) \otimes L_2^\top + L_1 \otimes \mathrm{coker}(S_2^\top) \\
|H_1(C_1 \otimes C_2^\top)| &= (m_1 - |\mathrm{im}(S_1)|)|L_2^\top| + |L_1|(n_2 - |\mathrm{im}(S_2^\top)|) \\
&= (m_1 - |\mathrm{im}(S_1^\top)|)|L_2^\top| + |L_1|(n_2 - |\mathrm{im}(S_2)|) \\
&= |\ker(S_1^\top)||L_2^\top| + |L_1||\ker(S_2)| \\
&= |L_1^\top||L_2^\top| + |L_1||L_2|
\end{aligned}
$$

The toric code is obtained from the product of a (classical) repitition code with its dual. The important thing to note is that the parity check matrix (stabilizer generators) is the object of primary importance, not the space of logical operators. To get the toric code we must start with a degenerate parity check matrix $S = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$. Then $|L| = |L^\top| = 1$ and we get the two logical qubits in the product. Using the matrix $S = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$. gives $|L| = 1, |L^\top| = 0$ which gives one logical qubit in the product; this is a surface code.

1.4.3 Product of a classical and a quantum code

This results in two separate quantum codes, as indicated in the following diagram:

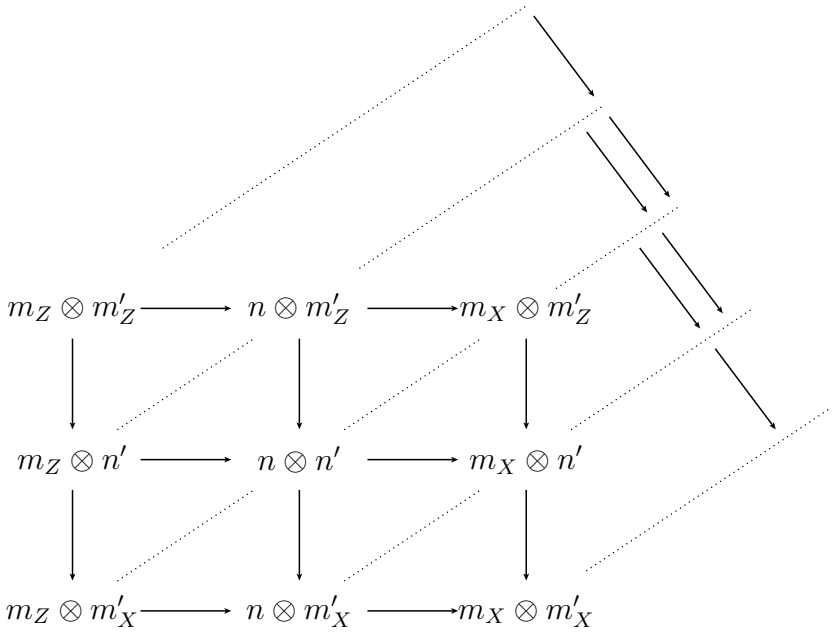


(There are also two classical codes at the endpoints.)

In this way we can generate the 3 (spatial) dimension toric code, as a product of the 2D toric code with the repitition code. Here we see the two resulting codes have sheets and lines for logical operators, one code has x-type sheets and z-type lines, the other code is x-type lines and z-type sheets.

1.4.4 Product of two quantum codes

Continueing this pattern we can generate three different quantum codes as a product of two quantum codes:



For example, the product of the 2D toric code with itself produces the 4D toric code with x and z type sheet operators, thats the code in the middle.

Another example, the middle code of the product Stean x Stean has parameters $[67, 1, 9]$.

1.5 Sums and Pushouts

In any category the sum of two objects A and B is given by an object C and two maps $f : A \to C$ and $g : B \to C$. These maps show how to embedd A and B into their "sum". A further requirement is that C is somehow minimal: any other contender C' for the sum of A and B with "embedding" maps f' and g' must factor uniquely through C.

In the category of sets we take the disjoint union of A and B, similarly in the category of topological spaces. The embedding into the sum is the obvious inclusion map.

In the category of vector spaces, we take the direct sum $A \oplus B$, together with maps $f = I \oplus 0$ and $g = 0 \oplus I$. This extends to the category of chain complexes, which gives the disjoint union of two codes.

If we would like to join objects A and B along some "sub-part", $i : R \to A$, $j : R \to B$ we play the same game but now require f and g to respect i and j, that

is, $f \circ i = g \circ j$. This is known as a "pushout" (of i and j.) In the category of sets, we would take the disjoint union as before, and then identify those elements according to $(fi)(r) \sim (gj)(r), r \in R$.

This identification also works for topological spaces, but for vector spaces we project out the *subspace* defined by $fi - gj$. The notation is then $A \oplus_R B$.

Extended to chain complexes we obtain a general way to "weld" two quantum codes together [30].

TODO...

1.6 Non-CSS codes

TODO...

Chapter 2

Representations of Subsystem Code Hamiltonians

Quantum error correcting codes, in particular *stabilizer codes* [21, 14], aim to protect an encoded quantum state by measuring a collection of commuting observables. When these same observables are used as the terms of a Hamiltonian a dynamical picture emerges: the encoded state now belongs to the (degenerate) groundspace of a system [16]. Questions about protecting this state are now related to questions about the spectrum of this Hamiltonian.

A more general notion of error correction comes from the theory of *subsystem* codes [2, 32]. Here the observables no longer commute, which complicates the error correction procedure. Also, the corresponding Hamiltonian has non-commuting terms, but we may still find integrals of motion that label eigenspaces of the system.

A family of topological subsystem codes, from the code perspective [8, 9, 33], from the cont-mat (integrals of motion!) perspective: [25, 12]

topological subsystem Codes: [33] Structure of 2D Topological Stabilizer Codes: [9] Gauge color codes, gauge fixing: [10] Gauge color codes, single shot: [11]

We elucidate these ideas using representation theory. The group generated by the terms of the Hamiltonian will be called the *gauge group*, and the intuition is that this group acts on itself in much the same way that it acts on states. Indeed, we forget all about the state space and examine the structure of the group itself.

When this approach succeeds we find we can block diagonalize Hamiltonian made

from the Pauli spin operators I X Z and Y, with coupling coefficients ± 1. The blocks are labelled by eigenvalues of integrals of motion generated by the gauge group. These operators are the *stabilizers.* For Hamiltonians such as the transverse-field Ising model, $\mathcal{H} = \sum_i X_i X_{i+1} + Z_i$, this is not much use because the gauge group does not generate many stabilizers (in fact, only one in this case). We are more interested in the situation where the gauge operators generate a large (extensive) number of stabilizers, and we examine two such cases: the 2d-compass model [2] and the Kitaev honeycomb model [27].

2.1 Motivation: graph theory

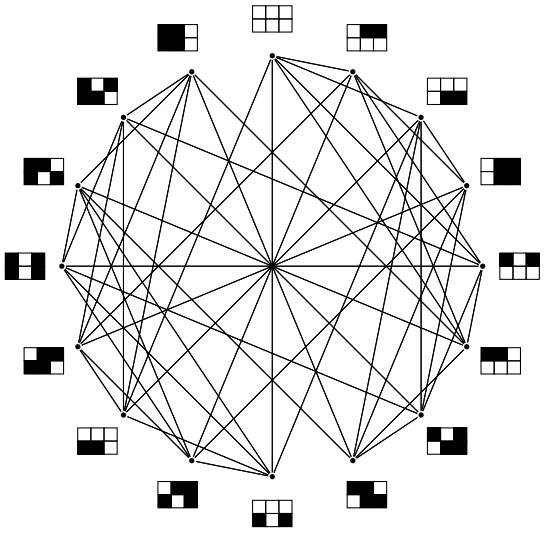


Figure 2.1: The gauge group action on a basis is depicted as a graph.

A *group action* of a group G on a set V is a function η that associates each $g \in G$ with a bijection $\eta(g) : V \to V$. Given such a group action and a set $G_0 \subset G$ we can form a graph as follows. The vertices correspond to the set V and (directed) edges are $\{(v, gv) : v \in V, g \in G_0\}$. If the set G_0 is closed under group inverse then we can consider the graph to be undirected. The complex vector space with basis V is written $\mathbb{C}[V]$. We can now extend η linearly to get a *representation* of the group G:

$$\mathbb{C}[\eta] : G \to GL(\mathbb{C}[V]).$$

We show in Fig 2.1 the action of the the following subset of the 6-qubit Pauli group on the computational basis:

$$G_0 = \left\{ \begin{matrix} XXI,\ I\,II,\ IXX,\ II\,I,\ XIX,\ I\,II,\ ZII,\ IZI,\ IIZ \\ I\,II, XXI, II\,I, IXX, I\,II, XIX, ZII, IZI, IIZ \end{matrix} \right\}$$

The *Cayley graph* of a group G and generating set G_0 is denoted Cayley(G, G_0). It has nodes G and edges $\{(g, hg) : g \in G, h \in G_0\}$. This is the graph associated with the group action on itself. The group action on itself is in some sense the "mother" of all other ways that G can act on a set, and likewise the Cayley graph is the "most relaxed" version of all graphs obtained in this way from G acting on a set.

The adjacency matrix of a graph with N nodes is the N by N matrix A with non-zero entries $A_{ij} = 1$ corresponding to edges (i, j) of the graph. It follows that the adjacency matrix A of Cayley(G, G_0) is the linear operator $A : \mathbb{C}[G] \to \mathbb{C}[G]$ given by

$$A = \sum_{g \in G_0} \rho_{\text{reg}}(g)$$

where ρ_{reg} is the left regular representation of G on $\mathbb{C}[G]$.

Using representation theory we have that $\rho_{\text{reg}} : G \to \mathbb{C}[G]$ decomposes as the dirrect sum of irreducible representations $\rho_k : G \to \text{GL}(V_k)$ and so

$$A = \sum_{g \in G_0} \bigoplus_k \rho_k(g)$$

block diagonalizes A, with each ρ_k possibly appearing multiple times in the direct sum over k.

See [17] and [26] for more details.

2.2 Representations of gauge codes

2.2.1 The Pauli group

The Pauli group $\mathcal{P}_1$ is normally defined as a set of matrices closed under matrix multiplication, but we can define it abstractly as the group generated by the (abstract)

elements $\{\omega, X, Z\}$ with relations as follows:

$$\omega^2 = I,\ X^2 = I,\ Z^2 = I,\ \omega X \omega X = I,\ \omega Z \omega Z = I,\ \text{and } \omega Z X Z X = I,$$

where I is the group idenity. Actually, ω is generated by X and Z, so it is not necessary to include ω in the generating set, but here it simplifies the relations.

To define the *n-qubit Pauli group* $\mathcal{P}_n$, we use the $2n+1$ element generating set $\{\omega, X_1, .., X_n, Z_1, .., Z_n\}$ with relation $\omega^2 = I$ as before, and

$$\begin{gathered} X_i^2 = I,\ Z_i^2 = I,\ \omega X_i \omega X_i = I,\ \omega Z_i \omega Z_i = I,\ \omega Z_i X_i Z_i X_i = I,\ \text{for } i = 1, ...n, \\ Z_i X_j Z_i X_j = I,\ \text{for } i, j = 1, .., n,\ i \neq j. \end{gathered} \tag{2.1}$$

This abstract approach to the definition of a group is known as a group *presentation*. In general, this is a set of generators together with a set of relations satisfied by these generators.

Note that each of the generators squares to the identity, and of these, only ω commutes with every element of $\mathcal{P}_n$. Therefore we will write ω as $-I$, similarly ± 1 will denote the set $\{\omega, I\}$, and $-X$ is ωX, etc.

The subgroup of $\mathcal{P}_n$ generated by the elements $\{X_1, ..., X_n\}$ is denoted $\mathcal{P}_n^X$. These are the X-type elements. Similarly, $\{Z_1, ..., Z_n\}$ generates the subgroup of Z-type elements $\mathcal{P}_n^Z$.

Every element $g \in \mathcal{P}_n$ can be written uniquely as a product $g = \pm g_X g_Z$, where g_X is an X-type operator and g_Z is a Z-type operator. This gives the size of the group as:

$$|\mathcal{P}_n| = 2^{2n+1}.$$

2.2.2 Subgroups of the Pauli group

We now define an *$n-$qubit gauge group* to be any subgroup G of $\mathcal{P}_n$. By choosing a set of generators $G_0 \subset \mathcal{P}_n$,

$$G := \langle G_0 \rangle.$$

We will assume G is not abelian, which implies that $-I \in G$. We also restrict G_0 to only contain Hermitian operators, which is equivalent to requiring that $g^2 = I$ for all

$g \in G_0$. Now let S be the largest subgroup of G not containing $-I$. S is then an abelian subgroup, also known as the *stabilizer* subgroup. G decomposes as a direct product:

$$G = S \times R,$$

where $R \cong P_r$ for some $1 \leq r \leq n$, and $S \cong \mathbb{Z}_2^m$ for $0 \leq m < n$. Therefore,

$$|G| = |S||R| = 2^{m+2r+1}.$$

We call R the *reduced gauge group*. We consider both S and R to be subgroups of G. Let $\phi : R \to P_r$ be a group isomorphism, then $R_0 := \{\phi^{-1}(X_i), \phi^{-1}(Z_i)\}_{i=1,..,r}$ is a set of independant generators of R. We also let S_0 be a set of m independant generators of S.

To find the cosets of G in $\mathcal{P}_n$ we take the group closure of $G - \mathcal{P}_n$; when this is non-empty we only need to add I and $-I$. This is another gauge group, whose reduced gauge group is known as the *logical* operators L, and whose stabilizer subgroup is known as the *error* operators T. Now any coset of G can be written as ltG with $l \in L$ and $t \in T$. The size of T equals the size of S: $|T| = |S| = 2^m$. If we let L_0 be an independant generating set for L then we have the important formula:

$$\frac{1}{2}|L_0| + |S_0| + \frac{1}{2}|R_0| = n. \tag{2.2}$$

2.2.3 Irreducible representations of the Pauli group

We now define the *Pauli representation* of the Pauli group as a group homomorphism:

$$\rho_{\text{pauli}} : \mathcal{P}_n \to \text{GL}(\mathbb{C}[2^n])$$

where $\mathbb{C}[2^n]$ is the 2^n-dimensional state space of n qubits. On the independant generators $\{X_1, .., X_n, Z_1, .., Z_n\}$, ρ_{pauli} is defined as the following tensor product of 2×2 matrices:

$$\rho_{\text{pauli}}(X_i) := \bigotimes_{j=1}^{n} \left\{ \begin{array}{ll} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & \text{for } j \neq i, \\ \\ \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} & \text{for } j = i \end{array} \right\},$$

$$\rho_{\text{pauli}}(Z_i) := \bigotimes_{j=1}^{n} \left\{ \begin{array}{ll} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & \text{for } j \neq i, \\ \\ \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} & \text{for } j = i \end{array} \right\}.$$

Normally the image of ρ_{pauli} is thought of as the Pauli group itself, and we are indeed free to think that way because ρ_{pauli} is a group isomorphism.

Given a group representation $\rho : G \to GL(V)$ the *character* of ρ is a function $\chi_\rho : G \to \mathbb{C}$ given by

$$\chi_\rho(g) = \text{Tr } \rho(g).$$

Given two functions $u, v : G \to \mathbb{C}$ we define the following inner product:

$$\langle u, v \rangle := \frac{1}{|G|} \sum_{g \in G} u(g)\overline{v(g)}.$$

The character of the Pauli representation, $\chi_{pauli} : \mathcal{P}_n \to \mathbb{C}$ is given by:

$$\chi_{pauli}(g) = \sum_{v \in basis} \langle v|\rho_{pauli}(g)|v\rangle = \left\{ \begin{array}{ll} \pm 2^n & \text{if } g = \pm I \\ 0 & \text{otherwise} \end{array} \right.$$

Since $|P_n| = 2^{2n+1}$ it follows that $\langle \chi_{pauli}, \chi_{pauli} \rangle = 1$ and so ρ_{pauli} is an irreducible representation of $\mathcal{P}_n$.

The only other irreps of $\mathcal{P}_n$ are the 1-dimensional irreps $\rho : \mathcal{P}_n \to \mathbb{C}$ defined on the

independant generators as:

$$\rho(X_i) = \pm 1, \quad \rho(Z_i) = \pm 1.$$

So we have 2^{2n} many 1-dimensional irreps, and a single 2^n-dimensional irrep. Summing the squares of the dimensions shows that we have a complete set of irreps of $\mathcal{P}_n$.

2.2.4 Irreducible representations of gauge groups

Although ρ_{pauli} restricted to a gauge group $G \subset \mathcal{P}_n$ serves as a representation of G it is no longer irreducible. Our aim will be to decompose ρ_{pauli} into irreps of G.

The 1-dimensional irreps $\rho : G \to \mathbb{C}$, are now defined by specifying the action of ρ on the independant generators:

$$\rho(h) = \pm 1 \text{ for } h \in S_0, \quad \rho(\phi^{-1}(X_i)) = \pm 1, \quad \rho(\phi^{-1}(Z_i)) = \pm 1.$$

This gives all 2^{m+2r} of the 1-dimensional irreps.

The 2^r-dimensional irreps are given by:

$$\rho(h) = \pm I^{\otimes r} \text{ for } h \in S_0, \quad \rho(\phi^{-1}(X_i)) = X_i, \quad \rho(\phi^{-1}(Z_i)) = Z_i.$$

We are free to choose the signs of the $\rho(h)$ for each $h \in S_0$. Hence there are 2^m many of these irreps. Each such choice corresponds to the choice of a *syndrome* vector $s(h) = \pm 1$, for $h \in S_0$, or alternatively, choice of an element $t \in T$:

$$\rho_t^1(h) = \begin{cases} I^{\otimes r} & \text{if } th = ht \\ -I^{\otimes r} & \text{if } th = -ht \end{cases}$$

Because G decomposes into a direct product $G = S \times \mathcal{P}_r$ we have the following representations:

$$\rho_t(g) = \rho_t^1(h)\rho_{pauli}^r(g'),$$

where $g = hg'$, $h \in S$, $g' \in \mathcal{P}_r$ and ρ_{pauli}^r is the r-qubit Pauli representation. The

character for this representation is:

$$\chi_t(hg') = \rho_t^1(h) \sum_{v \in basis} \langle v|\rho_{pauli}^r(g')|v\rangle = \begin{cases} \pm 2^r \rho_t^1(h) & \text{if } g' = \pm I \\ 0 & \text{otherwise} \end{cases}$$

We have that $|G| = 2^{2r+m+1}$ and so $\langle \chi_t, \chi_t \rangle = 1$ and ρ_t is an irreducible representation of G. We now count the occurances of this representation in ρ_{pauli}^r:

$$\begin{aligned} \langle \chi_{pauli}^r, \chi_t \rangle &= \frac{1}{|G|} \sum_{g \in G} \chi_{pauli}^r(g) \overline{\chi_t(g)} \\ &= \frac{1}{2^{2r+m+1}} \sum_{g=\pm I} 2^n 2^r = \frac{2^{n+1+r}}{2^{2r+m+1}} = 2^k \end{aligned}$$

where k is the number of logical qubits so that $n = r + m + k$.

In summary, the Pauli representation decomposes into 2^m many irreps ρ_t, each with dimension 2^r, and appearing with multiplicity 2^k :

$$\rho_{\text{pauli}} = \bigoplus_{l \in L, t \in T} \rho_t.$$

2.2.5 Symmetry invariant basis

In general, given a representation $\rho : G \to \mathrm{GL}(V)$ and the character of some irreducible representation $\chi : G \to \mathbb{C}$ the following operator $P : V \to V$ projects onto the subspace on which this irreducible representation acts:

$$P := \frac{d}{|G|} \sum_{g \in G} \overline{\chi(g)} \rho(g).$$

where d is the dimension of the irreducible representation. We can use this to calculate projectors onto the irreps ρ_t in ρ_{pauli}:

$$\begin{aligned} P_t &= \frac{d}{|G|}\sum_{g\in G}\overline{\chi_t(g)}\rho_{pauli}(g) \\ &= \frac{d}{|G|}\sum_{h\in S}\sum_{g\in R}\overline{\chi_t(hg)}\rho_{pauli}(hg) \\ &= \frac{d}{|G|}2^{2r}\sum_{h\in S}\rho_t^1(h)\rho_{pauli}(h) \\ &= \frac{1}{2^m}\sum_{h\in S}\rho_t^1(h)\rho_{pauli}(h). \end{aligned}$$

We can also write this as a product of projectors onto the ± 1 eigenspaces of stabilizers $\rho_{pauli}(h)$ for $h \in S$. Choose generators $h_1, ..., h_m$ of S and then the projectors onto the ± 1 eigenspace of $\rho_{pauli}(h_i)$ are

$$P_t^i = \frac{1}{2}\big(I^{\otimes n} \pm \rho_{pauli}(h_i)\big)$$

and we see that

$$P_t = \prod_{i=1,...,m} P_t^i = \frac{1}{2^m}\big(I^{\otimes n} \pm \rho_{pauli}(h_1)\big)...\big(I^{\otimes n} \pm \rho_{pauli}(h_m)\big).$$

This projector will have rank 2^{k+r} and

$$U := \sum_{t\in T} P_t$$

is a unitary transformation that sends physical qubits to encoded qubits.

2.3 The Hamiltonian

The Hamiltonian of interest is an operator $\mathcal{H} : \mathbb{C}[2^n] \to \mathbb{C}[2^n]$:

$$\mathcal{H} := \sum_{g\in G_0} \rho_{\text{pauli}}(g).$$

Using the above decomposition we find:

$$\mathcal{H} = \sum_{g \in G_0} \bigoplus_{l \in L, t \in T} \rho_t(g) = \bigoplus_{l \in L, t \in T} \sum_{g \in G_0} \rho_t(g).$$

We will notate each block as $\mathcal{H}_t := \sum_{g \in G_0} \rho_t(g)$ for each irrep ρ_t appearing in $\mathcal{H}$.

The form of $\mathcal{H}$ is seen to be very similar to the adjacency matrix of Cayley(G, G_0) but instead of the regular representation we are using the Pauli representation.

More generally, we can assign (real valued) weights to each operator in G_0, that is $w : G_0 \to \mathbb{R}$, and

$$\mathcal{H} = \sum_{g \in G_0} w(g) \rho_{\text{pauli}}(g) = \bigoplus_{l \in L, t \in T} \sum_{g \in G_0} w(g) \rho_t(g).$$

2.4 Examples

In the following examples we will make the identification between g and $\rho_{pauli}(g)$. So terms such as Z and X are understood to be the corresponding Pauli linear operators.

2.4.1 4-qubit gauge code

Here we work out a simple example with $n = 4$ that illustrates the above concepts and can be readily calculated by hand. In this example we supress the tensor product symbol when we write operators. We take as gauge operators:

$$G_0 = \{XXII, IIXX, ZIZI, IZIZ\}.$$

Evidently, the stabilizers are generated by $S_0 = \{XXXX, ZZZZ\}$, and we can choose the reduced gauge generators to be $R_0 = \{XXII, ZIZI\}$. The logical operators are generated by $L_0 = \{XIXI, ZZII\}$, and then the errors T_0 corresponding to S_0 will be

$\{ZZZI, IIIX\}$. All of this can be summarized in a table of anti-commuting pairs:

$$\begin{aligned} L_0 = &\quad XIXI \quad ZZII \\ S_0, T_0 = &\quad XXXX \quad ZZZI \\ &\quad ZZZZ \quad IIIX \\ R_0 = &\quad XXII \quad ZIZI \end{aligned}$$

where the number of rows equals n and each entry commutes with the entries on other rows, and anticommutes with the entry on the same row. Indeed, we have a presentation of the group $\mathcal{P}_4$, as it was defined in Eq (2.1). In particular, we have presented R as $\mathcal{P}_1$ and this makes it obvious an isomorphism $\phi : R \to \mathcal{P}_1$ to use:

$$\phi(XXII) := X, \quad \phi(ZIZI) := Z.$$

We now evaluate $\rho(g)$ for each $g \in G_0$ to work out the form of each block $\mathcal{H}_\rho$. Note that $g = g'h$ where $g' \in R$ and $h \in S$ so

$$\rho(g) = \rho(g')\rho(h) = \pm\phi(g').$$

We find the four blocks in the Hamiltonian:

$$\mathcal{H}_t = \sum_{g \in G_0} \rho_t(g) = \rho_t(XXII) + \rho_t(IIXX) + \rho_t(ZIZI) + \rho_t(IZIZ) = X \pm X + Z \pm Z.$$

In the next two sections we repeat this calculation for more complicated gauge groups. The key is to find an isomorphism ϕ (as well as R and S), then the calculation proceeds straightforwardly.

2.4.2 2D compass model

Here we consider the two-dimensional compass model [2]. We coordinatize the qubits on a square lattice of $l \times l$ sites, (i, j) for $1 \leq i, j \leq l$. This gives $n = l^2$. For the single qubit Pauli operators acting on site (i, j) we coordinatize with subscripts ij, with i and

j understood modulo l. The generators of the gauge group are

$$G_0 = \big\{X_{ij}X_{i,j+1},\ Z_{ij}Z_{i+1,j} \text{ for } 1 \le i,j \le l\big\}.$$

We write generators of the reduced gauge group in anti-commuting pairs:

$$R_0 = \big\{X_{i1}X_{ij},\ Z_{1j}Z_{ij} \text{ for } 2 \le i,j \le l\big\}.$$

This makes it clear the isomorphism $\phi : R \to \mathcal{P}_r$ to use, and we again use pairs i,j to coordinatize $\mathcal{P}_r$:

$$\phi(X_{i1}X_{ij}) = X_{i-1,j-1}, \quad \phi(Z_{1j}Z_{ij}) = Z_{i-1,j-1}, \text{ for } 2 \le i,j \le l.$$

The generators for the stabilizers are

$$S_0 = \big\{\prod_{i=1}^{l} X_{ij}X_{i,j+1},\ \prod_{i=1}^{l} Z_{ji}Z_{j+1,i} \text{ for } 1 \le j \le l-1\big\}.$$

The logical operators are generated by $L_0 = \big\{\prod_i X_{i1}, \prod_j Z_{1j}\big\}$. These sets have cardinalities:

$$|G_0| = 2l^2,\ |R_0| = 2(l-1)^2,\ |S_0| = 2(l-1).$$

And we note that 2.2 is satisfied. Now we write down the values of the irreps on the gauge operators. Here we define each irrep using a pair of syndrome vectors s_X and s_Z :

$$\rho(X_{i1}X_{i2}) = X_{i-1,1} \qquad \rho(Z_{1i}Z_{2i}) = Z_{1,i-1} \qquad \text{for } 2 \le i \le l$$

$$\rho(X_{il}X_{i1}) = X_{i-1,l-1} \qquad \rho(Z_{li}Z_{1i}) = Z_{l-1,i-1} \qquad \text{for } 2 \le i \le l$$

$$\rho(X_{ij}X_{i,j+1}) = X_{i-1,j-1}X_{i-1,j} \qquad \rho(Z_{ji}Z_{j,i+1}) = Z_{j-1,i-1}Z_{j,i-1} \qquad \text{for } 2 \le i \le l, 2 \le j < l$$

$$\rho(X_{1j}X_{1,j+1}) = s_X(j-1)\prod_{i=1}^{l-1} X_{i,j-1}X_{ij} \quad \rho(Z_{j1}Z_{j+1,1}) = s_Z(j-1)\prod_{i=1}^{l-1} Z_{j-1,i}Z_{ji} \quad \text{for } 2 \le j < l$$

$$\rho(X_{11}X_{12}) = \prod_{j=1}^{l-1} s_X(j)\prod_{i=1}^{l-1} X_{i1} \qquad \rho(Z_{11}Z_{21}) = \prod_{j=1}^{l-1} s_Z(j)\prod_{i=1}^{l-1} Z_{1i}.$$

Note the transposition symmetry between the X and Z-type operators. We sum all these terms to find the form of the hamiltonian in each block:

$$\mathcal{H}_\rho = \sum_{g \in G_0} \rho(g) = \sum_{1 \le i,j < l} \rho(X_{ij}X_{i,j+1}) + \rho(Z_{ij}Z_{i+1,j}).$$

We note that in [13], they perform a spin transformation of the compass model which also results in a $(l-1) \times (l-1)$ lattice of spins and identical Hamiltonian except for some signs.

2.4.3 Kitaev honeycomb model

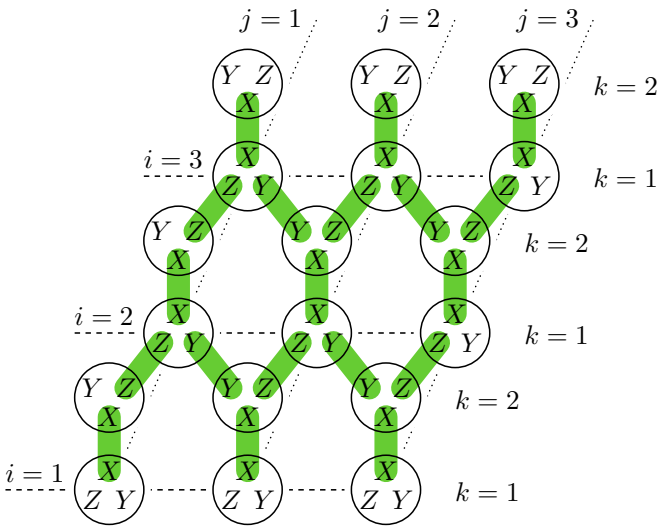


Figure 2.2: Gauge generators have support on the edges of the honeycomb lattice. Qubits here are depicted as circles.

The Kitaev honeycomb model [27] is built from spins on the sites of a hexagonal lattice. The lattice of linear size l has $n = 2l^2$ sites which we coordinatize using integer triples i, j, k with $1 \le j, k \le l$ and $k = 1, 2$. We use periodic boundary conditions so i, j are to be taken modulo l. See figure 2.2. The edges of the lattice are in one-to-one correspondence with the generators G_0:

$$G_0 := \big\{ X_{ij1}X_{ij2},\ Z_{ij2}Z_{i+1,j1},\ Y_{ij1}Y_{i-1,j+1,2} \text{ for } 1 \le i, j \le l \big\}.$$

Note that we make the definition $Y := XZ$ for each site.

Stabilizers are generated from closed strings of gauge operators. For example, each hexagon gives a stabilizer

$$\begin{aligned} h_{ij} :&= X_{ij1}X_{ij2}Z_{ij2}Z_{i+1,j1}Y_{i+1,j1}Y_{i,j+1,2}X_{i,j+1,2}X_{i,j+1,1}Z_{i,j+1,1}Z_{i-1,j+1,2}Y_{i-1,j+1,2}Y_{ij1} \\ &= Z_{ij1}Y_{ij2}X_{i+1,j1}Z_{i,j+1,2}Y_{i,j+1,1}X_{i-1,j+1,2}. \end{aligned}$$

And the two homologically non-trivial loops give stabilizers:

$$h_v := \prod_{i=1}^{l} Y_{i11}Y_{i12}, \quad h_h := \prod_{j=1}^{l} X_{1j2}X_{2j1}.$$

This gives independant stabilizer generators S_0 from each hexagon, less one, as well as h_v and h_h. The number of hexagons is $\frac{1}{2}n$ and so we find $|S_0| = \frac{1}{2}n + 1$. There are no logical operators, so we must have $|R_0| = n - 2$.

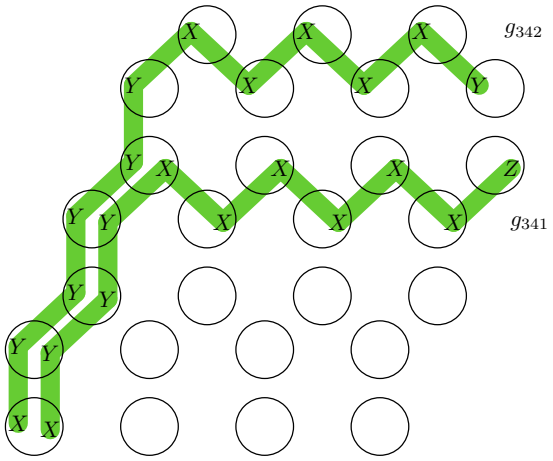


Figure 2.3: Two elements of the set R_0 corresponding to $i = 3$, $j = 4$ and $k = 1, 2$.

Now we construct a set of string operators R_0, one for each site on the lattice, except for the two sites $(1, 1, 1)$ and $(1, 1, 2)$. Each string $g_{ijk} \in R_0$ is constructed as the product of gauge operators along a path starting at $(1, 1, 1)$ and terminating at (i, j, k). See figure 2.3. Each such path is built from two "straight" path segments, first in the i direction and then in the j direction. The paths for operators g_{ij1} and g_{ij2} coincide along the i direction but become disjoint in the j direction: the g_{ij1} path goes around the bottom of the hexagons and the g_{ij2} path goes around the top. With periodic boundary conditions R_0 forms an independant generating set of R of size $n - 2$.

We construct an isomorphism $\phi : R \to \mathcal{P}_r$ by sending elements of R_0 bijectively to the following independant generating set of $\mathcal{P}_r$:

$$\big\{c_{2j} := Z_1 ... Z_{j-1} X_j,\ c_{2j+1} := Z_1 ... Z_{j-1} Y_j \ \text{ for } 1 \leq j \leq r\big\}.$$

The bijection is constrained by setting $\phi(g_{ij1}) := c_{2j'+1}$ and $\phi(g_{ij2}) := c_{2j'}$ where j' is chosen uniquely for each i, j. The c_j are paired Majorana fermion operators [27].

We check this is a group homomorphism by showing that relations satisfied by elements of R_0 are satisfied by their images under ϕ. All such relations are either of the form $g^2 = \ \pm\ I$, $gg' = \pm\, g'g$, or products thereof. So it is sufficient to check squares of elements and commutation relations. Every element of R_0 anticommutes with every other element of R_0, and this is true also of the c_j. Also, $g_{ij1}^2 = -I$ and $g_{ij2}^2 = I$ is preserved by ϕ because $c_{2j}^2 = I$ and $c_{2j+1}^2 = -I$. Finally, ϕ is an isomorphism because it is a bijection of two independant generating sets.

The next step is to write each element of G_0 as a product of reduced gauge operators and stabilizers. The key thing to note is that the product of two operators $g_{ijk}, g_{i'j'k'} \in R_0$ gives a string operator between the sites (i, j, k) and (i', j', k'). And *any* string operator between these two sites can then be generated by using stabilizers to "deform" the string $g_{ijk} g_{i'j'k'}$. For example, taking the product of two operators from R_0 that differ by one path segment gives the following:

$$Z_{ij2} Z_{i+1,j,1} = g_{ij2} g_{i+1,j,1}$$
$$Y_{i+1,j,1} Y_{i,j+1,2} = g_{i+1,j,1} g_{i,j+1,2}$$

We need the homologically non-trivial stabilizers to get these:

$$Z_{lj2} Z_{1j1} = h_v g_{lj2} g_{1j1} \qquad \text{for } \ 2 \leq j \leq l$$

And the $X_{ij1} X_{ij2}$ gauge operators can be generated by the product of $g_{ij1} g_{ij2}$ and the

enclosed hexagon stabilizers:

$$X_{ij1}X_{ij2} = g_{ij1}g_{ij2}\prod_{j'=1}^{j-1} h_{ij'}.$$

The only G_0 operators that are not quadratic in R_0 operators are the five operators that touch either of the sites $(1,1,1)$ or $(1,1,2)$.

So each block in the Hamiltonian is seen to be quadratic in the c_j plus five other Pauli operator terms which we denote as Λ_ρ:

$$\mathcal{H}_\rho = \sum_{ij} \Gamma_{ij}(\rho) c_i c_j + \Lambda_\rho$$

The coefficients Γ_{ij} are dependant on the irrep ρ.

2.5 Outlook

Monomial formalism: [37]

Chapter 3

Error Correction in a Non-Abelian Topologically Ordered System

We consider the theory of two-dimensional topologically ordered systems with anyonic excitations. There are two main approaches to defining these, one being more algebraic and the other more topological.

The algebraic side is known to mathematicians as the study of braided tensor categories, or more specifically, modular tensor categories. This algebraic language appears to be more commonly used in the physics literature [27]. Such algebraic calculations can be interpreted as manipulations of string diagrams [3], or *skeins*, which begins to hint at how knot theory and topology comes into play.

Building from the other direction, one starts with a topological space (of low dimension) and attempts to extract a combinatorial or algebraic description of how this space can be built from joining smaller (simpler) pieces together. This is perhaps more difficult to grapple with. These topologically rooted constructs are known as modular functors, or the closely related topological quantum field theories (TQFT's).

That these two approaches (algebraic versus topological) meet is one of the great miracles of modern mathematics and physics.

Modular functors can be constructed from skein theory. In the physics literature,

this appears as skeins mysteriously growing out of manifolds [31], or as motivated by renormalization group considerations [29]. A further physical motivation is this: if a skein is supposed to correspond to the $(2+1)$-dimensional world-lines of particles as in the Schrödinger picture, modular functors would correspond to the algebra of observables, as in a Heisenberg picture.

In the physics literature, modular functors are explicitly used in [19, 18]. Also, [7] and [28] use the language of modular functors but they call them TQFT's. This is in fact reasonable because a modular functor can be seen as (part of) a TQFT, but is quite confusing to the novice who attempts to delve into the mathematical literature.

The definition of a modular functor appears to be well motivated physically. Unfortunately, there are many such definitions in the mathematical literature [38, 36, 5, 35]. According to [6] section 1.2 and 1.3, there are many open questions involved in rigorously establishing the connection between these different axiomatizations. Furthermore, the concept of modular tensor category has not been established to correspond exactly (bijectively) to any of these modular functor variants. We try not to concern ourselves too much with these details, but merely note these facts as a warning to the reader who may go searching for the "one true formulation" of topological quantum field theory.

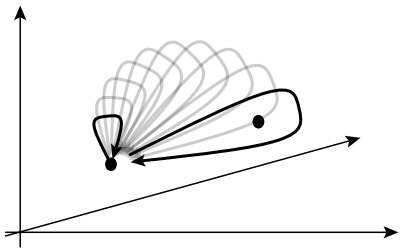
Of the many variants of modular functor found in the literature, one important distinction to be made is the way anyons are labeled. In the mathematical works [36, 5, 35] we see that anyons are allowed to have superpositions of charge states. However, in this work we restrict anyons to have definite charge states, as in [38, 18, 7]. This seems to be motivated physically as such configurations would be more stable citation?.

One of the goals of this work is to sketch how a braided tensor category arises from a modular functor. In the mathematical literature, this is covered in [36, 35, 5] but as we just noted they use a different formulation for a modular functor.

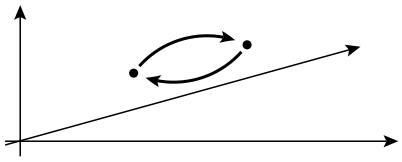
> "People these days have a tendency to actually define a subfactor as being some kind of tensor category. But these guys [subfactors] are the algebras of observables of your physical system, and if you want to ignore them and throw them out then you strongly risk throwing out the baby with the bathwater." - Vaughan Jones [23]

3.1 Topological Exchange Statistics

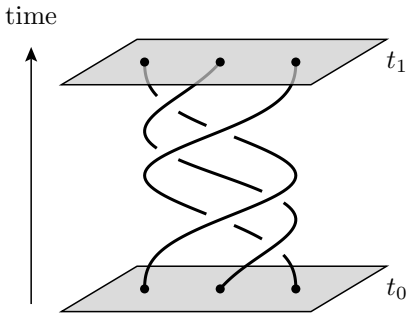
In three spatial dimensions, the process of winding one particle around another is topologically trivial. This is because the path can be deformed back to the identity; there is no obstruction:



The square root of this operation is a swap:

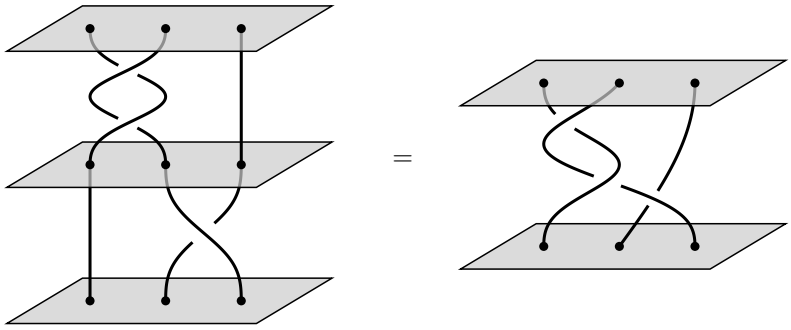


We consider world-lines of particles in two spatial dimensions. Here we show three particles undergoing some exchange and then returning to the original location:

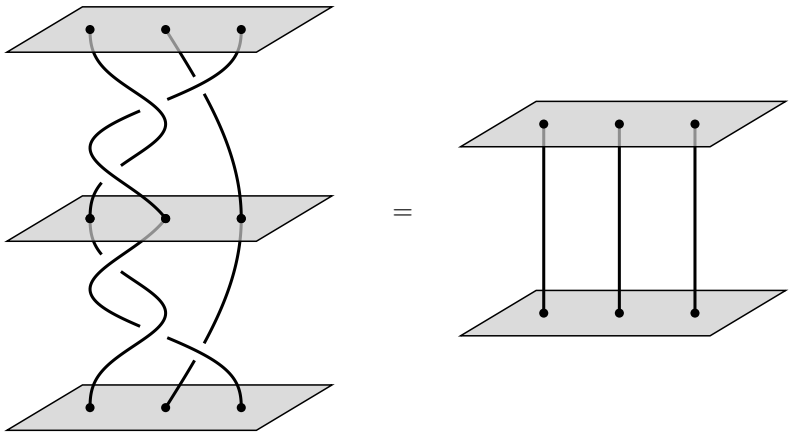


Note that if we allow the particles to move in one extra dimension then we can untangle these braided world lines. The question is, how has the wavefunction of the system changed from t_0 to t_1?

Examining the structure of these processes more closely, we see that we can compose them by sequentially performing two such braids, one after the other:



And, by "reversing time", we can undo the effect of any braid:

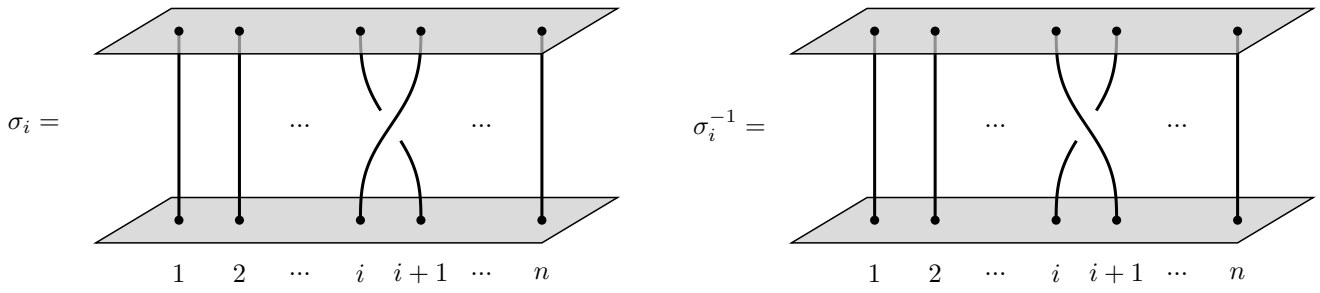


This means we have a group, known as the *braid group.* For n particle worldlines we denote this group as B_n.

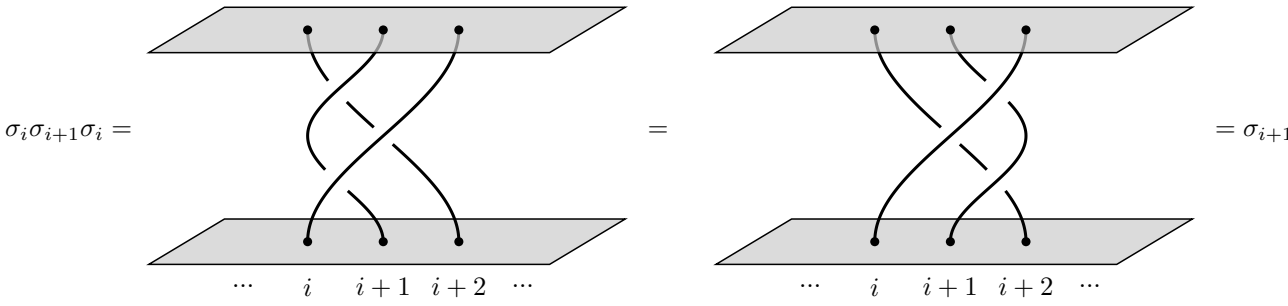
This group can also be described purely algebraically. B_n has $n-1$ generators $\sigma_1, .., \sigma_{n-1}$ that satisfy the following two relations:

$$\sigma_i \sigma_j = \sigma_j \sigma_i \quad \text{for} \quad |i-j| > 1,$$
$$\sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1} \quad \text{for} \quad 1 \leq i \leq n-2.$$

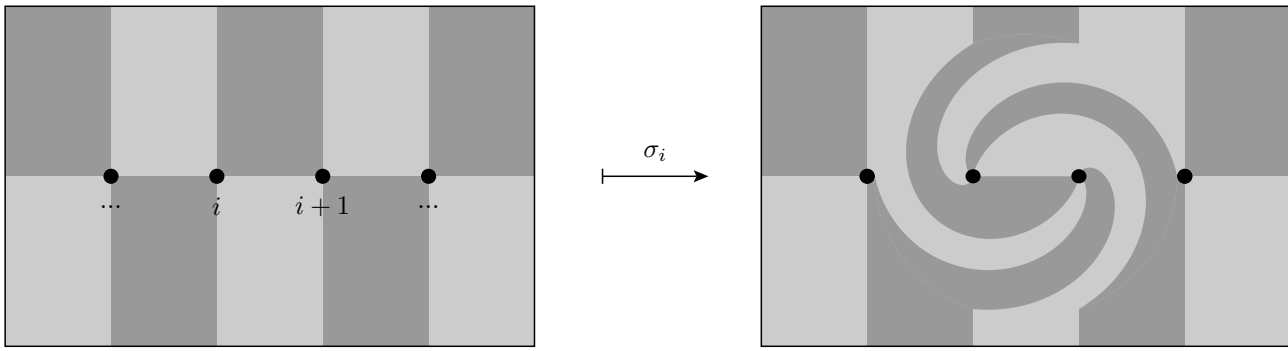
The easiest way to understand these relations is to consider the geometric braids corresponing to the σ_i.



Now we see why $\sigma_i\sigma_j = \sigma_j\sigma_i \quad \text{for} \quad |i-j| > 1$, because the two braids are operating on disjoint strands. To understand the second relation we draw:



There is another important geometric representation of the braid group which is purely two-dimensional.



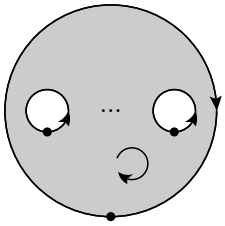
3.2 Modular Functors

We list the axioms for a *2-dimensional unitary topological modular functor*.

For our purposes a *surface* will be a compact oriented 2-dimensional differentiable manifold with boundary. We will not require surfaces to be connected. By a *hole* of M we mean a component of its boundary. Each hole will inherit the manifold orientation, contain a distinguished *base point*, and be labeled with an element from a (fixed) finite set $\mathcal{A}$. This is the set of "anyon labels", and comes equiped with a vacuum element $\mathbb{I}$ and an involution $\widehat{\ }$ such that $\widehat{\mathbb{I}} = \mathbb{I}$. The involution maps an anyon label to the "antiparticle" label.

We will mostly be concerned with planar such surfaces, that is, a *disc* with holes removed from the interior. Do we actually restrict to considering only such surfaces?

Only when we get to standard surfaces and POPs... Such surfaces will be given a clockwise orientation, which induces a counterclockwise orientation on any *interior* hole and a clockwise orientation on the *exterior* hole.



Although it helps to draw such a surface as a disc with holes, we stress that there is no real distinction to be made between interior holes and the exterior hole. That is, a disc with n interior holes is (equivalent to) a sphere with $n+1$ holes. We merely take advantage of the fact that a sphere with at least one hole can be flattened onto the page by "choosing" one of the holes to serve as the exterior hole.

By a *map of surfaces* $f : M \to N$ we mean a diffeomorphism that preserves manifold orientation, hole labels, and base points. If we require that the map preserves base points and so on, are we restricting to maps between manifolds with the same number of holes? Yes, a diffeomorphism is also a homeomorphism, so these manifolds are topologically equivalent. Note that we also deal with maps of various other objects (vector spaces, sets, etc.) but a map of surfaces will have these specific requirements.

Two maps of surfaces $f : M \to N$ and $f' : M \to N$ will be called *isotopic* when one is a continuous deformation of the other. In detail, we have a continuously parametrized family of maps $f_t : M \to N$ for $t \in [0,1]$ such that $f_0 = f$, $f_1 = f'$ and the restriction of f_t to the set of marked points $X \subset \partial M$ is constant: $f_t|_X = f|_X$ for $t \in [0,1]$. Such a family $\{f_t\}_{t\in[0,1]}$ is called an *isotopy* of f. This is an equivalence relation on maps $M \to N$, and the equivalence class of f under isotopy is called the *isotopy class* of f.

A *modular functor* $\mathscr{H}$ associates to every surface M a finite dimensional complex vector space $\mathscr{H}(M)$, called the *fusion space* of M. For each map $f : M \to N$ The modular functor associates a unitary transformation $\mathscr{H}(f) : \mathscr{H}(M) \to \mathscr{H}(N)$ that only depends on the isotopy class of f. Functoriality requires that $\mathscr{H}$ respect composition of maps.

We have the following axioms for $\mathscr{H}$.

Unit axioms. The fusion space of an empty surface is one dimensional, $\mathscr{H}(\phi) \cong \mathbb{C}$. Does a modular functor have to be over $\mathbb{C}$? No, mathematicians do this over any ring k. For M_a a disc with boundary label a we have $\mathscr{H}(M_{\mathbb{I}}) \cong \mathbb{C}$ and $\mathscr{H}(M_a) \cong 0$ for $a \neq \mathbb{I}$. For an annulus $M_{a,b}$ with boundary labels a, b we have $\mathscr{H}(M_{a,\widehat{a}}) \cong \mathbb{C}$ and $\mathscr{H}(M_{a,b}) \cong 0$ for $a \neq \widehat{b}$.

Monoidal axiom. The disjoint union of two surfaces M and N is associated with the tensor product of fusion spaces:

$$\mathscr{H}(M \amalg N) \cong \mathscr{H}(M) \otimes \mathscr{H}(N).$$

This is natural from the point of view of quantum physics, where the Hilbert space of two disjoint systems is the tensor product of the space for each system.

Gluing axiom. Denote a surface M with (at least) two holes labeled a, b as $M_{a,b}$. If we constrain $b = \widehat{a}$ then we may *glue* these two holes together to form a new surface N. To construct N we choose a diffeomorphism from one hole to the other that maps base point to base point and reverses orientation. Identifying the two holes along this diffeomorphism gives the glued surface N. (There is a slight technicality in ensuring that N is then differentiable, but we will gloss over this detail.) The fusion space of $M_{a,\widehat{a}}$ then embeds unitarily in the fusion space of N. Moreover, there is an isomorphism called a *gluing map*:

$$\bigoplus_{a \in \mathcal{A}} \mathscr{H}(M_{a,\widehat{a}}) \xrightarrow{\cong} \mathscr{H}(N).$$

say something about isotopy of gluing

Unitarity axiom. Reversing the orientation of the surface M to form $\overline{M}$ we get the dual of the fusion space: $\mathscr{H}(\overline{M}) = \mathscr{H}(M)^*$. i don't think we use this for anything

Compatibility axioms. Loosely put, we require that the above operations play nicely together, and commute with maps of surfaces. For example, a sequence of gluing operations applied to a surface can be performed regardless of the order (gluing is associative) and we require the various gluing maps for these operations to similarly agree. For another example, we require $\mathscr{H}$ to respect that gluing commutes with disjoint union. I guess we should try to keep a consistent level of precision, and so probably be more explicit here. I agree it is a bit jarring to suddenly switch to being

more informal...

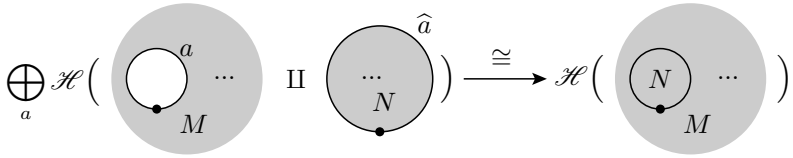
Observables. The seam along which gluing occurs can be associated with an observable as follows. We take a gluing map g and projectors P_a onto the summands in the above direct sum:

$$P_a : \bigoplus_{b \in \mathcal{A}} \mathscr{H}(M_{b,\widehat{b}}) \to \mathscr{H}(M_{a,\widehat{a}}).$$

then the observable will be the set of operators $\{P_a g^{-1}\}_{a \in \mathcal{A}}$. For each $a \in A$ we call the image of P_a a *charge sector* for that observable. Note that in the glued surface N the seam has no preferred orientation (or base point). If we choose an orientation for the seam this corresponds to choosing one of the two boundary components in the original surface $M_{a,\widehat{a}}$. If these boundary components come from disconnected components of $M_{a,\widehat{a}}$ the seam cuts N into two pieces and the orientation chooses an *interior:* following the orientation around the seam presents the interior to the left.

We intentionally confuse the distinction between an observable as a set of operators, and the associated seam along which gluing occurs. Any simple closed curve within the interior of M can serve as the seam along which gluing occurs... If we now take M to be an arbitrary surface, and γ an observable on M, we can write M as the gluing of another surface N... XXX

Consequences of axioms. A common operation is to glue two separate surfaces. We can do this by first taking disjoint union (tensoring the fusion spaces) and then gluing. Here we show this process applied to two surfaces M and N.

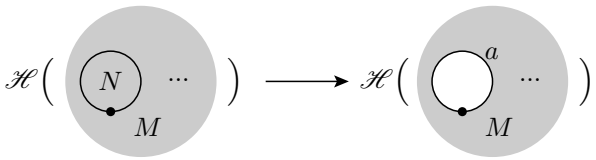


We display the surface N with the $\widehat{a}$ boundary on the outside, to show more clearly how N fits into M. In the glued surface we indicate the placement of M and N and the seam along which gluing occurred, as well as the identification of base points.

We note two other consequences of the axioms. A hole of M labeled with $\mathbb{I}$ can be replaced with a disk (by gluing) and this does not change the fusion space of M. That is, a hole that carries no charge can be "filled-in". And, the dimensionality of the

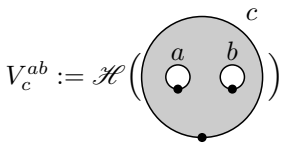
fusion space of a torus is the cardinality of $\mathcal{A}$. This can be seen by gluing one end of an annulus (cylinder) to the other.

Fusion. When a surface can be presented as the gluing of two separate surfaces, we have projectors onto the fusion space of either glued surface:

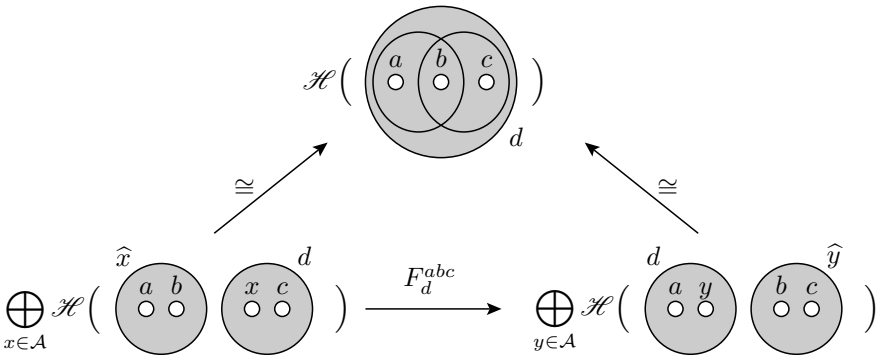


In this case, we define the operation of *fusion* to replace the interior of an observable by a single hole. This is an operation on the manifold itself, and we will only do this when the interior piece is a disc with zero or more holes.

F-move. The fusion space of the disc and annulus are specified by the axioms, and we define the fusion space of the disc with two holes, or *pair-of-pants* as:



The *F-move* is constructed from two applications of a gluing map (one in reverse) as the following commutative diagram shows:



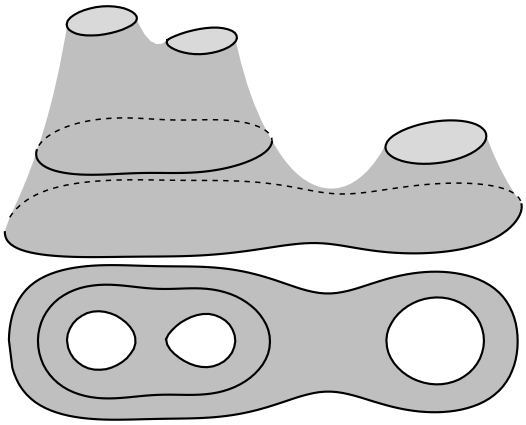
Here we have a surface M_d^{abc} with four labeled boundary components, as well as two separate ways of gluing pairs-of-pants to get M_d^{abc}. We can also write this out in terms of the fusion spaces of pair-of-pants:

$$F_d^{abc} : \bigoplus_{x\in\mathcal{A}} V_{\widehat{x}}^{ab} \otimes V_d^{xc} \to \bigoplus_{y\in\mathcal{A}} V_d^{ay} \otimes V_{\widehat{y}}^{bc}.$$

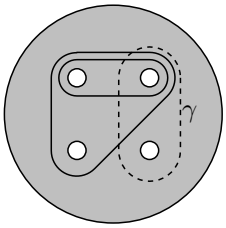
tensor with identity to extend this map under gluing

at this point we specialize to considering only discs with holes

POP decomposition. Given a manifold M which is a disc with at two or more holes, we show how to present M as the gluing of various pair-of-pants. Such an arangement will be termed a *POP decomposition.* (We refer to [22] for more details on this construction, and [20] for a leisurely description of Morse theory.) This will yield a decomposition of $\mathscr{H}(M)$ into a direct sum of fusion spaces of pair-of-pants. The key idea is to choose a "height" function $h : M \to \mathbb{R}$ with some specific properties that allow us to cut the manifold up along level sets of h. First, we need that critical points of h are isolated. This is the defining condition for h to be a *Morse function.* Also, we need that h is constant on ∂M, and the values of h at different critical points are distinct. Now choose a sequence of non-critical values $a_1 < a_2 < ... < a_n$ in $\mathbb{R}$ such that every interval $[a_{i-1}, a_i]$ contains exactly one critical value of h and the image of h lies within $[a_1, a_n]$. Each component of $h^{-1}([a_{i-1}, a_i])$ is then either an annulus, a disc, or a pair-of-pants depending on the index of (any) critical point it contains. We then re-glue any annuli or discs until there are no more of these and we have only pair-of-pants.

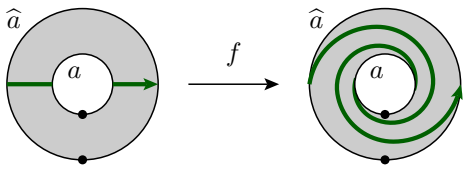


Clearly such a POP decomposition is not unique, and the goal here is to understand how to switch between decompositions, and in particular, given an observable γ, and a given POP decomposition, find a sequence of "moves" such that γ is in the resulting POP decomposition:



One way to achieve this is via Cerf theory, which is the theory of how one may deform Morse functions into other Morse functions and the kind of transitions involved in their critical point structure. This was the approach used in [18]. In this work we use a simpler method, which is essentially the same as *skein theory.* This is the *refactoring theorem.*

Dehn twist. Consider a surface $M_{a,\widehat{a}}$ with two boundary components, a and $\widehat{a}$. Let f be a map $M_{a,\widehat{a}} \to M_{a,\widehat{a}}$ which performs a clockwise 2π full-twist or *Dehn twist.* Here we show the action of f by highlighting the equator of the annulus:

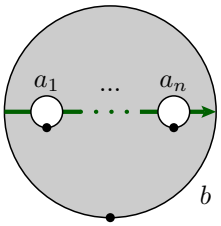


We define the induced map on fusion spaces as $\theta_a := \mathscr{H}(f)$. Because $\mathscr{H}(M_{a,\widehat{a}})$ is one-dimensional this will be multiplication by a complex number which we also write as θ_a.

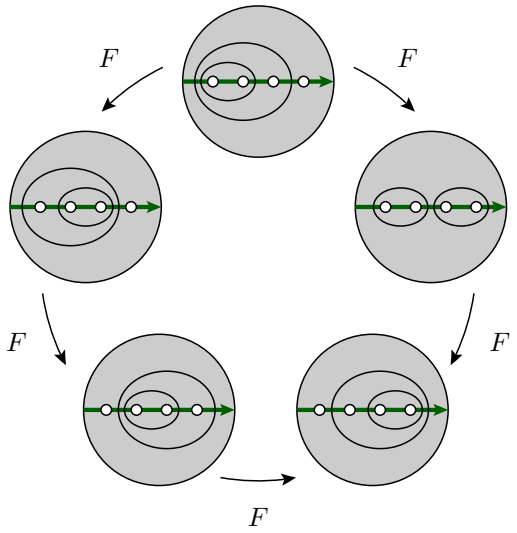
If we now take M to be an arbitrary surface, and γ an observable on M, we can consider a neighbourhood of γ which will be an annulus, and perform a Dehn twist there, which we denote as $f_\gamma : M \to M$. Writing M as a gluing along γ of another manifold $N_{a,\widehat{a}}$, the action of $\mathscr{H}(f_\gamma)$ will decompose as a direct sum over charge sectors:

$$\bigoplus_{a \in \mathcal{A}} \theta_a \mathscr{H}(N_{a,\widehat{a}}).$$

Standard surfaces. For each $n = 0, 1, ...$ and every ordered sequence of anyon labels $a_1, ..., a_n, b$ we choose a *standard surface.* This is a surface with $n+1$ boundary components labeled $a_1, ..., a_n, b$ which we denote $M_b^{a_1 ... a_n}$. For concreteness, we let this surface be the subset of the complex plane defined by $\left\{x \in \mathbb{C} \text{ st. } |x - \frac{1}{2}| \leq \frac{1}{2}\right\}$ with n discs $\left\{x \in \mathbb{C} \text{ st. } |x - \frac{2i-1}{2n}| < \frac{1}{4n}\right\}_{i=1,...n}$ removed. We then label the interior holes $a_1, ..., a_n$ in order of increasing real coordinate, and the exterior hole is labeled b. The base points are placed in the direction of the negative imaginary axis. As a notational convenience we highlight the *equator* of the surface which is the intersection of the real axis with the surface:



Each standard surface comes with a collection of *standard POP decompositions*: these will be POP decompositions where we require each observable to cross the equator twice and have counterclockwise orientation. Up to isotopy, a given standard surface will have only finitely many of these. On a standard surface with three interior holes there are two standard POP decompositions, and one F-move that relates these. On a standard surface with four interior holes there are five standard POP decompositions and five F-moves that relate these. In this case the F-moves themselves satisfy an equation that is enforced by the axioms of the modular functor. This is known as the pentagon equation, which we depict as the following commutative diagram:



We define the vector spaces $V_b^{a_1 \cdots a_n} := \mathscr{H}(M_b^{a_1 \cdots a_n})$. We now choose a basis for each of the $V_b^{a_1 a_2}$ to be $\{v_{b,\mu}^{a_1 a_2}\}_\mu$. For every standard POP decomposition of $M_b^{a_1 \cdots a_n}$, we get a decomposition of $V_b^{a_1 \cdots a_n}$ into direct sums of various $V_{b'}^{a_1' a_2'}$. This then gives a *standard basis* of $V_b^{a_1 \cdots a_n}$ relative to this standard POP decomposition using the corresponding $\{v_{b',\mu}^{a_1' a_2'}\}_\mu$ for each of the $V_{b'}^{a_1' a_2'}$. ...ugh

Note that for any standard surface $M_b^{a_1 \cdots a_n}$ and any choice of k contiguous holes $a_j, ..., a_{j+k-1}$ we can find an observable that encloses exactly these holes in at least one of the standard POP decompositions of $M_b^{a_1 \cdots a_n}$.

Gluing of two standard surfaces is not in general going to produce another standard surface. However, we can ... fix this

Curve diagram. We next study maps from standard surfaces to arbitrary surfaces. The set of all maps from $M_b^{a_1 \cdots a_n}$ to a surface N will be denoted as $\mathrm{Hom}(M_b^{a_1 \cdots a_n}, N)$. We call each such map a *curve diagram*, or more specifically, a curve diagram on N. The reason for this terminology is that we can reconstruct (up to isotopy) any map $f : M_b^{a_1 \cdots a_n} \to N$ from the restriction of f to the equator of $M_b^{a_1 \cdots a_n}$. In other words, we can uniquely specify any map $f : M_b^{a_1 \cdots a_n} \to N$ by indicating the action of f on the equator. We take full advantage of this fact in our notation: any figure of a surface with a "green line" drawn therein is actually notating a diffeomorphism, not a surface!

Any curve diagram will act on a standard POP decomposition of $M_b^{a_1 \cdots a_n}$ sending it to a POP decomposition of N. Surprisingly, the converse of this statement also holds: any POP decomposition of N (a disc with holes) comes from some curve diagram acting on a standard POP decomposition. The proof of this is constructive, and we call this the refactoring theorem below.

show how to glue two curve diagrams together to get another curve diagram

We note in passing two connections to the mathematical literature. Such curve diagrams have been used in the study of braid groups [15], and this is where the name comes from, although our curve diagrams respect the base points and so could be further qualified as "framed" curve diagrams. And, we note the similarity of curve diagrams and associated modular functor to the definition of a planar algebra [24], the main difference being that planar algebras allow for not just two but any even number of curve intersections at each hole.

Z-move. We let z be a diffeomorphism of standard surfaces $z : M_b^{a_1 \cdots a_n} \to M_{a_1}^{a_2 \cdots a_n b}$ that preserves the equator. (Considering the standard surface as a sphere with holes placed uniformly around a great circle, z is seen to be a "rotation".) This acts by pre-composition to send a curve diagram $f \in \mathrm{Hom}(M_b^{a_1 \cdots a_n}, N)$ to $fz \in \mathrm{Hom}(M_{a_1}^{a_2 \cdots a_n b}, N)$. This we call a Z-move of f.

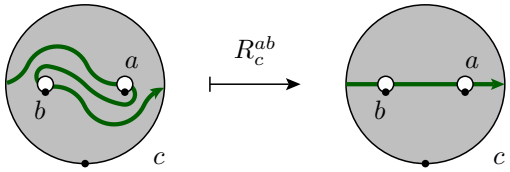
In this way, any curve diagram can be seen as another curve diagram that has a cyclic permutation of the labels of the underlying standard surface.

R-move. Given anyon labels a, b and c, and arbitrary surface N, we now define

the following map of curve diagrams on N :

$$R_c^{ab} : \mathrm{Hom}(M_c^{ab}, N) \to \mathrm{Hom}(M_c^{ba}, N).$$

This map works by taking a curve diagram $f : M_c^{ab} \to N$ to the composition $f\sigma$ where $\sigma : M_c^{ba} \to M_c^{ab}$ is a counterclockwise "half-twist" map that exchanges the a and b holes. Here we show the action of R_c^{ab} on one particular curve diagram:

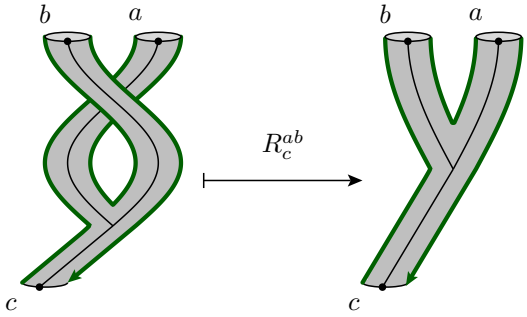


Such an application of R_c^{ab} to a particular curve diagram we call an *R-move.* As noted above, a curve diagram serves to pick out a basis for the fusion space, and the point of this R-move is to switch between different curve diagrams for the same surface. This is highlighted to draw the readers attention to the fact that the R-move does not swap the labels on the holes: the surface itself stays the same.

As we extended Dehn twists under gluing, we can do this for R-moves. Given any pair-of-pants in M ... keep going

When do we start confusing the distinction between topological operations and the corresponding vector space operation via $\mathscr{H}$?

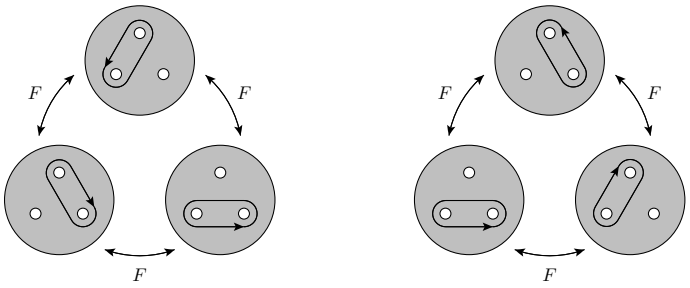
Skeins. The preivious figure can be seen as a "top-down" view of the following three dimensional arrangement:



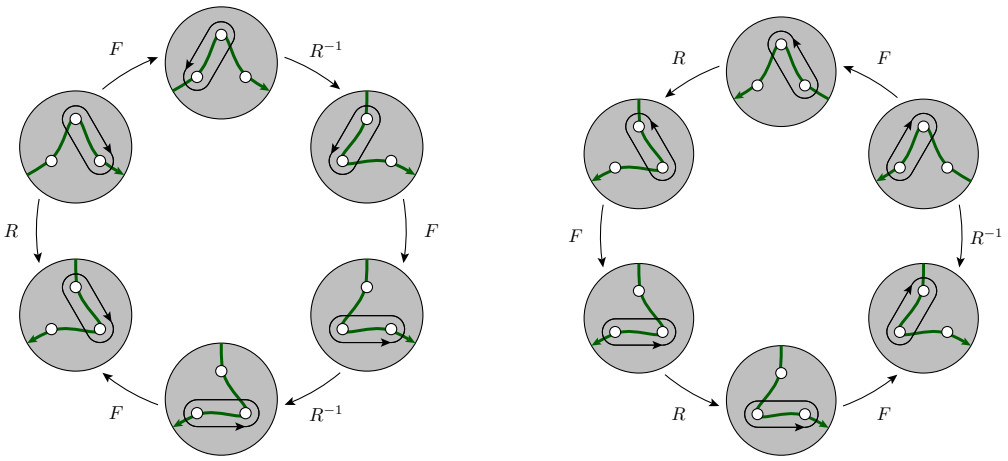
This figure is intended to be topologically the same as the previous flat figure, with the addition of a third dimension, and the c boundary has been shrunk. Also note thin black lines connecting the base points. The black lines do not add any extra structure, they can be seen as a part of the hexagon cut out by the green line and boundary

components. But notice this: the black lines are "framed" by the green lines. These are the ribbons used in skein theory! Note that all the holes are created equal: there is no distinction between "input" holes and "output" holes (as there is with cobordisms or string diagrams.)

Hexagon equation. For a surface with three interior holes there are infinitely many POP decompositions (up to isotopy). These are all made by choosing an observable that encloses two holes. The F-moves allow us to switch between these POP decompositions, and the axioms for the modular functor make these consistent. Here we show two such triangle consistency requirements (they are reflections of each other):



Given a POP decomposition of a surface N there are various curve diagrams on N that produce this decomposition from a standard POP decomposition, and there are certain R-moves will map between these. Once again, these moves must be consistent, and here we note the two "hexagon equations" corresponding to the above two triangles:



Note that we have neglected to indicate the base points here. Given a curve diagram, we can agree that base points occur "to the right" of the image of the equator. But in notating diagrams such as these, there is still an ambiguity: the position of the base points should be the same in each surface in the diagram. If we try to correct for this

post-hoc by rotating individual holes, we will then be correct only up to possible Dehn twist(s) around each hole. We can certainly track these twists if we wanted to, but in the interests of simplicity we do not. This introduces a global phase ambiguity into the calculations.

3.3 Refactoring theorem

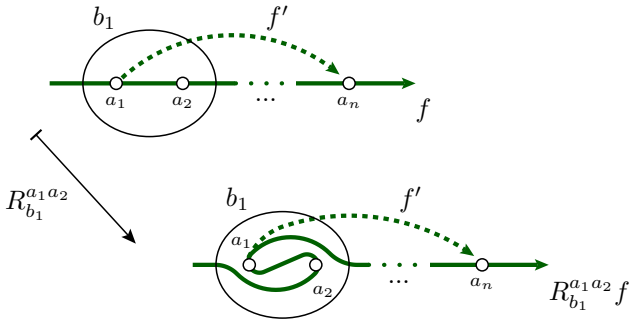
In this section we specialize to considering planar surfaces only. The theorem we are building towards shows that the R-moves act transitively on curve diagrams. We don't much care about transitivity. Should specialize this. That is, given two curve diagrams f and f' on N we can find a sequence of R-moves that transform f to f'. The key idea is to consider the (directed) image of the equator under these curve diagrams. As mentioned previously, this image is sufficient to define the entire diffeomorphism (up to isotopy) and so we are free to work with just this image, or *curve*, and we confuse the distinction between a curve diagram (a diffeomorphism) and its curve.

The construction proceeds by considering adjacent pairs of holes along f' and then acting on the portion of f between these same two holes so that they then become adjacent on the resulting curve. Continuing this process for each two adjacent holes of f' will then show a sequence of R-moves that sends f to f'. still not obvious: what if some such R-move destroys a previously adjacentized pair of holes?

To this end, consider a sequence of holes $a_1, ..., a_n$ appearing sequentially along f such that a_1 and a_n appear sequentially on f'. We may need to apply some Z-moves to f to ensure that a_1 appears sequentially before a_n. Now consider the case such that along f' between these two holes there is no intersection with f. We form a closed path ξ by following the f' curve between a_1 and a_n and then following the f curve in reverse from a_n back to a_1. (Note that to be completely rigorous here we would need to include segments of path contained within boundary components.) The resulting closed path bounds a disc. If ξ has clockwise orientation XX define clockwise XX we apply the following sequence of R-moves to f:

$$R_{b_1}^{a_1 a_2}, R_{b_2}^{a_1 a_3}, ..., R_{b_{n-1}}^{a_1 a_{n-1}}.$$

Depicted here is the first such move:



After each of these R-moves the closed path formed by following the f' curve between a_1 and a_n and then following the R-moved f curve back to a_1 will traverse one less hole, and still bound a disc. After all of the R-moves this path will only touch a_1 and a_n, and bound a disc. Therefore, we have acted on the f curve so that the resulting curve has a_1 and a_n adjacent and the bounded disc gives an isotopy for that segment of the curve.

When the closed curve ξ has anti-clockwise orientation we use the same sequence of R-moves but with R replaced by R^{-1}.

Generalizing further, when the f' curve between a_1 and a_n has (transverse) intersections with f we use every such intersection to indicate a switch between using R and R^{-1}.

This refactoring theorem will be reformulated below as a combinatorial "paperclip algorithm".

Use c instead of f for curve diagrams?

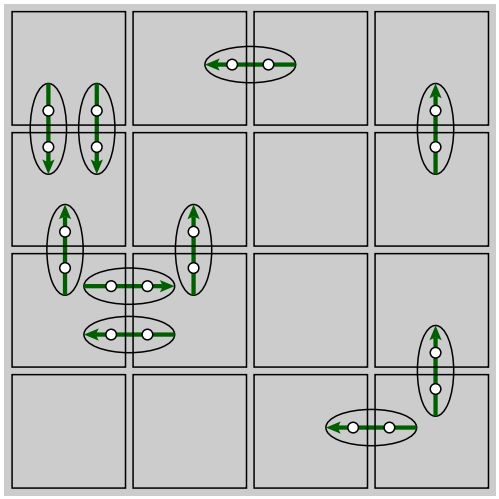
3.4 Physical model

The manifold underlying our system is a torus. We endow this with a $L \times L$ square lattice of observables:

$$\Lambda := \{\gamma_{ij}\}_{i,j=1,\ldots,L}$$

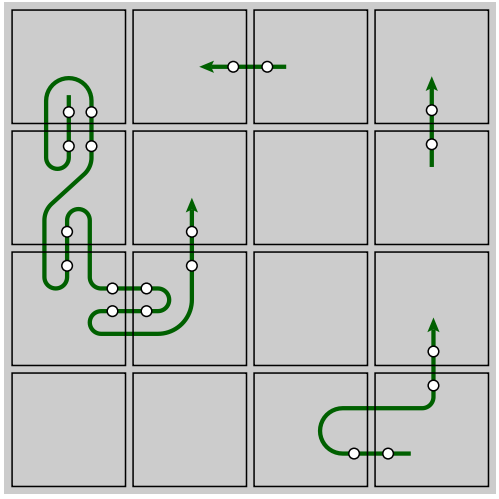
These observables are the physically accessible observables of the noise reduction procedure we call the *decoder*. We call each such γ_{ij} a *tile*. We show a small gap between the tiles but this is not meant to reflect an actual physical gap.

The noise process acts to populate the manifold with a randomly distributed set of pair creation processes, whose size is much smaller than the resolution of the lattice. We model this as a random distribution of pair-of-pants:

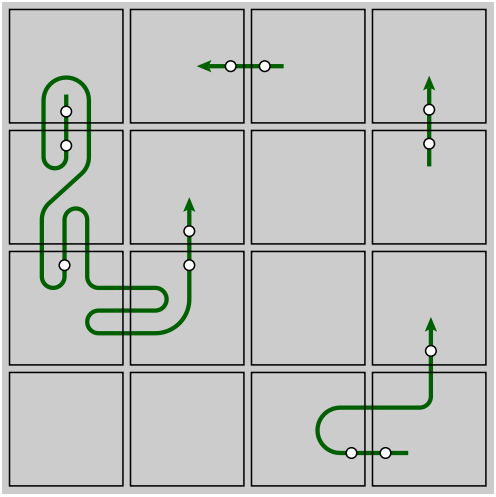


Each such pair will have vacuum total charge and so the observables γ_{ij} will only see pairs that intersect, ie. we need only consider distributing these pairs transversally along edges of the tiles.

In order to compute measurement outcomes for the γ_{ij} we first need to concatenate any two curve diagrams that participate in the same γ_{ij}. Because each curve has vacuum total charge this can be done in an arbitrary way:



Working in the basis picked out by the resulting curve diagrams, we can calculate measurement outcomes for each tile, the result of which is recorded on the original curve:

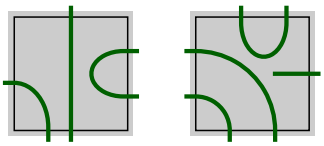


It might be helpful to be a bit more explicit either here or later about exactly how we perform this step, moving charges around with the paperclip algorithm until they are all neighbouring and then we are in a standard basis and can use F-moves to calculate fusion outcomes. good idea

3.5 Combinatorial curve diagrams

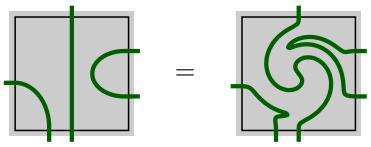
The basic data structure involved in the simulation we term a *combinatorial curve diagram.* Firstly, we will require each curve to intersect the edges of tiles transversally, and in particular a curve will not touch a tile corner.

For each tile in the lattice, we store a combinatorial description of the curve(s) intersected with that tile. Each component of such an intersection we call a *piece-of-curve.*

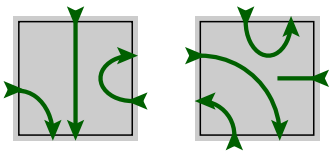


We follow essentially the same approach as taken in [1] to describe elements of a Temperley-Leib algebra, but with some extra decorations. The key idea is to store a *word* for each tile, comprised of the letters $\langle$ and $\rangle$. Reading in a clockwise direction around the edge of the tile from the top-left corner, we record our encounters with each piece-of-curve, writing $\langle$ for the first encounter, and $\rangle$ for the second. We may also encounter a dangling piece-of-curve (the head or the tail), so we use another symbol $*$

for this. The words for the above two tiles will then be $\langle\langle\rangle\rangle\langle\rangle$ and $\langle\rangle * \langle\langle\rangle\rangle$. When the brackets are balanced, each such word will correspond one-to-one with an intersection of a curve in a tile, up to a continuous deformation of the interior of the tile. Ie. the data structure will be insensitive to any continuous deformation of the interior of the tile, but the simulation does not need to track any of these degrees of freedom.



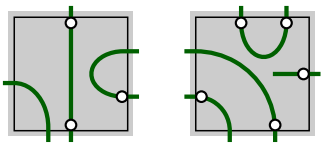
We will also need to record various other attributes of these curves, and to do this we make this notation more elaborate in the paragraphs **(I)**, **(II)** and **(III)** below. Each symbol in the word describes an intersection of the curve with the tile boundary, and so as we decorate these symbols these decorations will apply to such intersection points.



(I) We will record the direction of each piece-of-curve, this will be either an `in` or `out` decoration for each symbol. Such decorations need to balance according to the brackets. The decorated symbols $*_{\texttt{in}}$ and $*_{\texttt{out}}$ will denote respectively either the head or the tail of a curve. The words for the diagrams above now read as $\langle_{\texttt{in}}\langle_{\texttt{out}}\rangle_{\texttt{in}}\rangle_{\texttt{out}}\langle_{\texttt{out}}\rangle_{\texttt{in}}$ and $\langle_{\texttt{in}}\rangle_{\texttt{out}}*_{\texttt{in}}\langle_{\texttt{out}}\langle_{\texttt{in}}\rangle_{\texttt{out}}\rangle_{\texttt{in}}$.

(II) We will record, for each intersection with the tile edge, a numeral indicating which of the four sides of the tile the intersection occurs on. Numbering these clockwise from the top as $1, 2, 3, 4$ we have for the above curves: $\langle_1\langle_2\rangle_2\rangle_3\langle_3\rangle_4$ and $\langle_1\rangle_1 *_2 \langle_3\langle_3\rangle_4\rangle_4$.

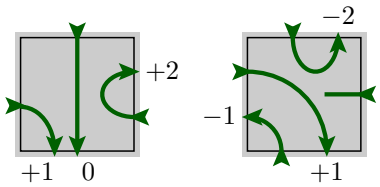
(III) Finally, we will also decorate these symbols with anyons. This will be an index to a leaf of a (sum of) fusion tree(s). This means that anyons only reside on the curve close to the tile boundary, and so we cannot have more than two anyons for each piece-of-curve. The number of such pieces is arbitrary, and so this is no restriction on generality.



In joining tiles together to make a tiling we will require adjacent tiles to agree on their shared boundary. This will entail sequentially pairing symbols in the words for adjacent tiles and requiring that the `in` and `out` decorations are matched. Because the word for a tile proceeds conter-clockwise around the tile, this pairing will always reverse the sequential order of the symbols of adjacent tiles. For example, given the above two tiles we sequentially pair the $\langle_{\texttt{out},2}\rangle_{\texttt{in},2}$ and $\rangle_{\texttt{out},4}\rangle_{\texttt{in},4}$ symbols with opposite order so that $\langle_{\texttt{out},2}\sim\rangle_{\texttt{in},4}$ and $\rangle_{\texttt{in},2}\sim\rangle_{\texttt{out},4}$.

Note that in general this data structure will store many disjoint curve diagrams $c_i : [0,1] \to D_{n_i}$ within a disc D_m where $\sum n_i = m$.

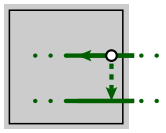
For each piece-of-curve, apart from a head or tail, there is an associated number we call the *turn number*. This counts the number of "right-hand turns" the piece-of-curve makes as it traverses the tile, with a "left-hand turn" counting as -1. (To be more rigorous, we would define this number using the winding number of the simple closed curve formed by the piece-of-curve adjoined to a segment of the boundary of the tile traversed in a clockwise direction.) This number will take one of the values $-2, -1, 0, 1, 2$:



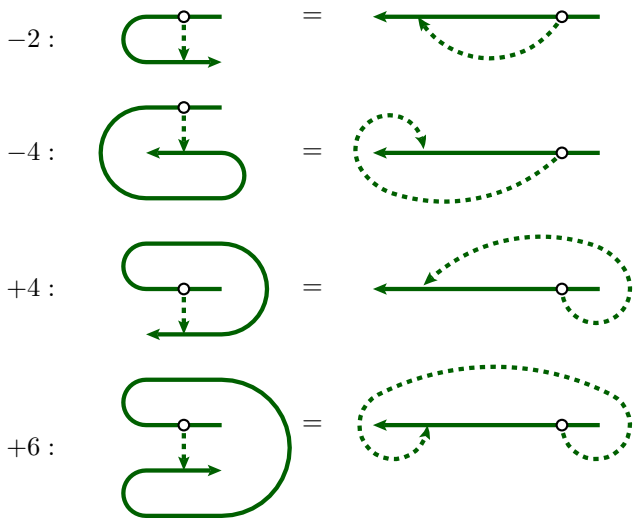
3.6 The paperclip algorithm

Anyons are transported around the lattice by moving them along tile edges. Note that transport here is the same as the refactoring from above. In general, such a transport will intersect with a curve diagram in many places. Each such intersection is transverse, and we use each intersection point to cut the entire transport into smaller paths each of which touch the curve diagram twice. The origin and destination of such an anyon path now splits the curve diagram $c : [0,1] \to D_n$ into three disjoint pieces which we

term *head*, *body* and *tail*, where the head contains the point $c(1)$, the tail contains $c(0)$ and the body is the third piece. These arise with various arrangements, but here we focus on one instructive case, the other cases are similar: transporting along one edge of a tile *forwards* (from tail to head) along a curve diagram:



This arrangement is equivalent (isotopic) to one of four "paperclips", which we distinguish between by counting how many *right-hand turns* are made along the body of the curve diagram. We also show an equivalent (isotopic) picture where the curve diagram has been straightened, and the resulting distortion in the anyon path:



The sequence of anyons along the head, body and tail, we denote as H, B and T, respectively. These sequences have the same order as the underlying curve diagram, and we use H^r, B^r and T^r to denote the same anyons with the reversed order. Using the above diagram, we can now read off the R-moves for each of the four paperclips:

$$-2 : R[B]$$

$$-4 : R[H^r]\ R[H]\ R[B]$$

$$+4 : R[B]\ R[T]\ R[T^r]$$

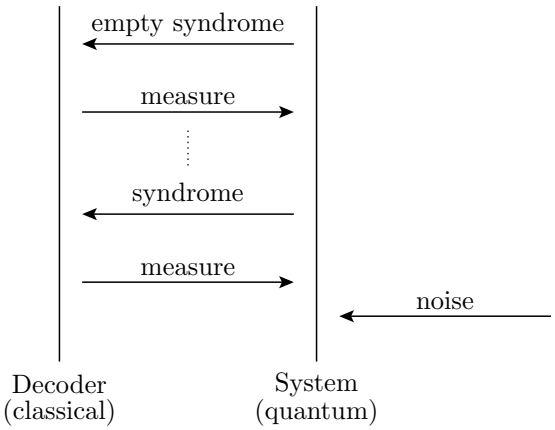
$$+6 : R[H^r]\ R[H]\ R[B]\ R[T]\ R[T^r]$$

where notation such as $R[B]$ is understood as sequentially clockwise braiding around each anyon in B.

That these four paperclips exhaust all possibilities can be seen by considering the winding number of the simple closed curve made by combining the body of the curve diagram with the path followed by the anyon (appropriately reversing direction as needed).

3.7 Decoding algorithm

After the noise process is applied to the system, the error correction proceeds as a dialogue between the decoder and the system. The decoder measures succesively larger and larger regions of the lattice until there are no more charges or a topologically non-trivial operation has occured (an operation that spans the entire lattice.) Here we show this in a process diagram, with time running up the page:

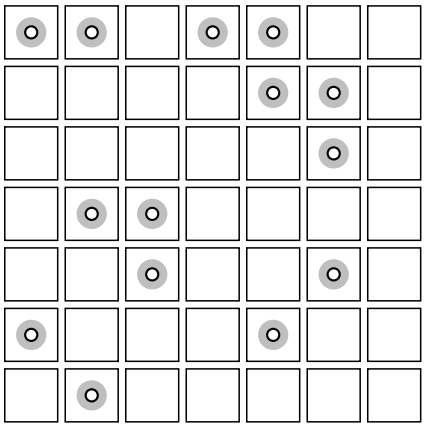


So far we have discussed the simulation of the (quantum) system and now we turn to the decoder algorithm. Here is a pseudo-code listing for this, and we explain each step via an example below.

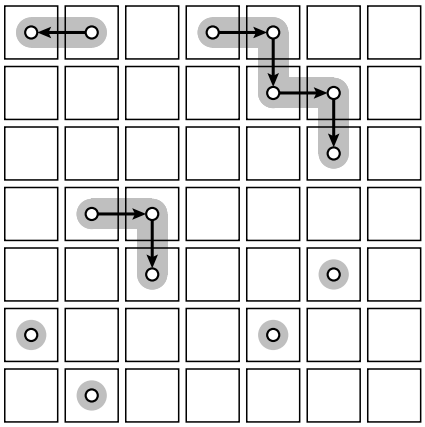
```
1:  def decode():
2:      syndrome = get_syndrome()
3:
4:      # build a cluster for each charge
5:      clusters = [Cluster(charge) for charge in syndrome]
6:
```

```
 7:      # join any neighbouring clusters
 8:      join(clusters, 1)
 9:
10:      while clusters:
11:
12:          # find total charge on each cluster
13:          for cluster in clusters:
14:              fuse_cluster(cluster)
15:
16:          # discard vacuum clusters
17:          clusters = [cluster for cluster in clusters if non_vacuum(cluster)]
18:
19:          # grow each cluster by 1 unit
20:          for cluster in clusters:
21:              grow_cluster(cluster, 1)
22:
23:          # join any intersecting clusters
24:          join(clusters, 0)
25:
26:      # success !
27:      return True
```

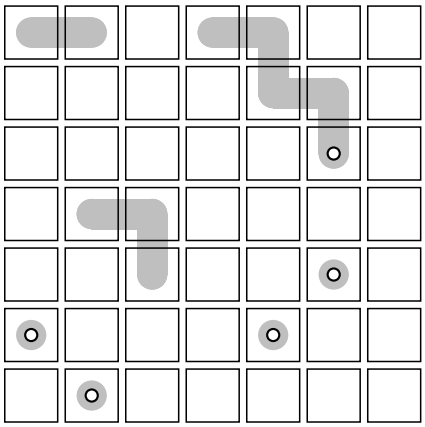
First, we show the result of the initial call to `get_syndrome()`, on line 2. The locations of anyon charges are highlighted in red. For each of these charges we build a `Cluster`, on line 5. Each cluster is shown as a gray shaded area.



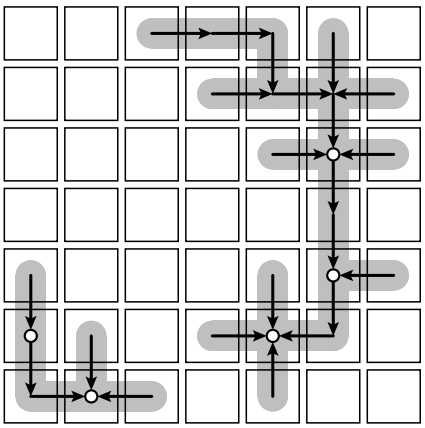
The next step is the call to `join(clusters, 1)`, on line 8, which joins clusters that are separated by at most one lattice spacing. We now have seven clusters:



Each cluster is structured as a rooted tree, as indicated by the arrows which point in the direction from the leaves to the root of the tree. This tree structure is used in the call to `fuse_cluster()`, on line 14. This moves anyons in the tree along the arrows to the root, fusing with the charge at the root.



For each cluster, the resulting charge at the root is taken as the charge of that cluster. Any cluster with vacuum total charge is then discarded (line 17). In our example, we find two clusters with vacuum charge and we discard these. The next step is to grow the remaining clusters by one lattice spacing (line 20-21), and join (merge) any overlapping clusters (line 24).



Note that we can choose the root of each cluster arbitrarily, as we are only interested in the total charge of each cluster.

We repeat these steps of fusing, growing and then joining clusters (lines 10-24.) If at any point this causes a topologically non-trivial operation, the simulation aborts and a failure to decode is recorded. Otherwise we eventually run out of non-vacuum clusters, and the decoder succeeds (line 27). Note that for simplicity we have neglected the boundary of the lattice in this example.

Maybe it is worthwhile to briefly recall the broad structure of our simulation somewhere here to help structure the discussion. I.e. we have first noise creation, then we iterate {syndrome measurement, classical decoding algorithm, transport} until failure or success. I agree the structure needs work.

Bibliography

[1] D. Bacon. Operator quantum error-correcting subsystems for self-correcting quantum memories. *Phys. Rev. A*, 73:012340, Jan 2006.

[2] J. C. Baez and M. Stay. Physics, topology, logic and computation: A rosetta stone. Physics, topology, logic and comp In New Structures for Physics, ed. Bob Coecke, Lecture Notes in Physics vol. 813, Springer, Berlin, 2011, pp. 95-174, 2009.

[3] H. Bombin. Topological subsystem codes. *Physical Review A*, 81(3):032301, 2010.

[4] H. Bombín. Structure of 2d topological stabilizer codes. *Communications in Mathematical Physics*, 327(2):387–432, 2014.

[5] H. Bombín. Gauge color codes: optimal transversal gates and gauge fixing in topological stabilizer codes. *New Journal of Physics*, 17(8):083002, 2015.

[6] H. Bombín. Single-shot fault-tolerant quantum error correction. *Physical Review X*, 5(3):031043, 2015.

[7] H. Bombin, M. Kargarian, and M. Martin-Delgado. Interacting anyonic fermions in a two-body color code model. *Physical Review B*, 80(7):075111, 2009.

[8] W. Brzezicki and A. M. Oleś. Symmetry properties and spectra of the two-dimensional quantum compass model. *Phys. Rev. B*, 87:214421, Jun 2013.

[9] A. R. Calderbank, E. M. Rains, P. W. Shor, and N. J. Sloane. Quantum error correction and orthogonal geometry. *Physical Review Letters*, 78(3):405, 1997.

[10] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill. Topological quantum memory. J. Math. Phys. 43, 4452-4505 (2002)., 2001.

[11] P. Diaconis and M. Shahshahani. Generating a random permutation with random transpositions. *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiet*, 57(2):159–179, 1981.

[12] D. Gottesman. Class of quantum error-correcting codes saturating the quantum hamming bound. *Physical Review A*, 54(3):1862, 1996.

[13] M. Kargarian, H. Bombin, and M. A. Martin-Delgado. Topological color codes and two-body quantum lattice hamiltonians. *New Journal of Physics*, 12(2):025018, 2010.

[14] P. Kaski. Eigenvectors and spectra of cayley graphs. `http://www.tcs.hut.fi/Studies/T-79.300/2002S/esitelmat/kaski_paper_020506.pdf`, 2002.

[15] A. Kitaev. Anyons in an exactly solved model and beyond. *Annals of Physics*, 321(1):2 – 111, 2006. January Special Issue.

[16] K. Michnicki. 3-d quantum stabilizer codes with a power law energy barrier, 2012.

[17] D. Poulin. Stabilizer formalism for operator quantum error correction. *Phys. Rev. Lett.*, 95:230504, Dec 2005.

[18] M. Suchara, S. Bravyi, and B. Terhal. Constructions and noise threshold of topological subsystem codes. *Journal of Physics A: Mathematical and Theoretical*, 44(15):155301, 2011.

[19] J.-P. Tillich and G. Zemor. Quantum ldpc codes with positive rate and minimum distance proportional to $n^{1/2}$, 2009.

[20] M. Van den Nest. A monomial matrix formalism to describe quantum many-body states. *New Journal of Physics*, 13(12):123004, 2011.