Non-Abelian Quantum Codes

Simon David Burton

A thesis submitted in partial fulfilment of requirements for the degree of Doctor of Philosophy

Faculty of Science
The University of Sydney
2016

Abstract

Like their classical counterparts, quantum codes are designed to protect quantum information from noise. From the perspective of information theory one considers the operations required to restore the encoded information given a syndrome which diagnoses the noise. From a more physics perspective, one considers systems whose energetically protected groundspace encodes the information. In this work we show that standard error correction procedures can be applied to systems where the noise appears as non-abelian Fibonacci anyons. In the case of a Hamiltonian with non-commuting terms, we build a theory describing the spectrum of these models, with particular focus on the 3D gauge color code model. Numerics support the conjecture that this model is gapped, which one would expect for a self-correcting quantum memory.

Statement of Originality

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*)

Statement of Authorship Attribution

Chapter 3 of this thesis is available as preprint [27], of which I am the sole author. Chapter 4 is based on collaborative work, available as preprint [28]. The algorithm used was co-designed with the co-authors. I implemented, tested and ran the algorithms.

To Arina

Contents

CHAPTER 1

Introduction

{motivate the research area and summarise key results}

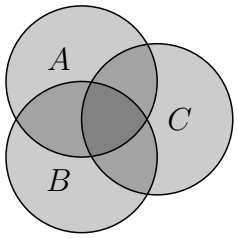
In this chapter we give a brief introduction to the theory of quantum error correcting codes [29, 33]. This will form the foundation for the rest of the thesis, in terms of being the "easy" version of all that follows.

1.1 Homology of a surface

We begin with a consideration of "size", or "counting". To denote the size of something A we write $\mu(A)$. Size is *additive* in the sense of $\mu(A \cup B) = \mu(A) + \mu(B)$ except that A and B may have intersection. In this case we would have counted the size of the intersection twice and so we modify this formula as

$$\mu(A \cup B) = \mu(A) + \mu(B) - \mu(A \cap B).$$

We can continue this idea to find the size of the union of three pieces

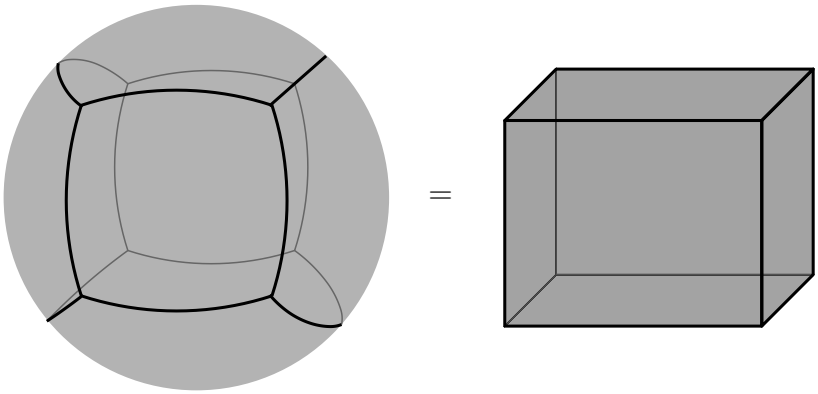


In this case the formula reads:

$$\begin{aligned}\mu(A \cup B \cup C) = \mu(A) + \mu(B) + \mu(C) \\ - \mu(A \cap B) - \mu(A \cap C) - \mu(B \cap C) \\ + \mu(A \cap B \cap C).\end{aligned} \tag{1.1}$$

The point here is the alternating signs: each time we try to count a size we overcount by one intersections worth, subtracting those intersections goes too far in the opposite direction and so we need to add intersections of intersections, and so on.

We now wish to apply this idea to count the size of a sphere. The trick here is to tile the sphere with the faces of a cube:

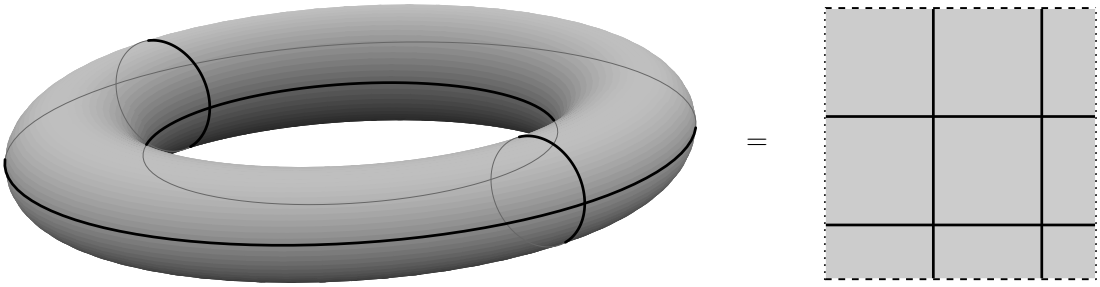


So we have six faces and one might suggest that $\mu(S^2) = 6$ but these are closed faces, so they intersect on their edges, of which we have 12. But these edges intersect at vertices and there are 8 of these. We extend the above formula (1.1) to calculate:

$$\mu(S^2) = 6 - 12 + 8 = 2.$$

So the sphere has "size" two! The magic here is that any other convex polyhedron would give the same answer of two. This, of course, is known as the *Euler characteristic*, and for a sphere this is indeed two. This approach to defining Euler characteristic is discussed in the fascinating book [63].

We repeat this calculation for another surface, a torus.



This time we use four faces, eight edges and four vertices:

$$\mu(S^1 \times S^1) = 4 - 8 + 4 = 0.$$

The Euler characteristic has many equivalent definitions, and we now turn to one of these, which is the idea of a homology. This theory goes back to Poincaré who was trying to deal with topological issues as they arise in complex analysis [65].

We are going to replace sets of things by vector spaces whose basis is the original set. And just to keep things simple we will take our vector spaces over the finite field with two elements $\mathcal{F} = \{0, 1\}$. This has the distinct advantage of eliminating all sign errors!

From the set of faces we form a vector space C_2 with basis the set of faces. Similarly, the one-dimensional pieces are the basis of C_1 and the zero-dimensional pieces are the basis of C_0:

$$\begin{aligned} C_2 &: \text{faces}, \\ C_1 &: \text{edges}, \\ C_0 &: \text{vertices}. \end{aligned}$$

The formula for Euler characteristic now reads:

$$\mu(C_\bullet) = \dim(C_2) - \dim(C_1) + \dim(C_0). \tag{1.2}$$

But now things get more interesting, because we have the following linear operators:

$$C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0. \tag{1.3}$$

These are defined to take the "boundary" of a shape. The operator ∂_2 gives the boundary of a face $f \in C_2$ which is just the sum of edges incident to (contained by) that face:

$$\partial_2(f) = \sum_{\substack{e \in \text{edges}, \\ e \sim f}} e.$$

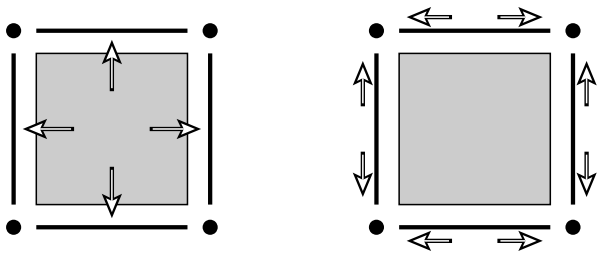
where we use $\sim$ to indicate incidence, and we extend ∂_2 to all of C_2 by linearity. Similarly, ∂_1 is defined to take an edge to the sum over its vertex endpoints:

$$\partial_1(e) = \sum_{\substack{v \in \text{vertices}, \\ v \sim e}} v.$$

Now with a small amount of thought one finds that

$$\partial_1 \circ \partial_2 = 0.$$

This is because each vertex around a face gets counted twice, and this is zero in $\mathcal{F}$-linear arithmetic.



In other words, the boundary of the boundary is empty! Or equivalently,

$$\text{im}(\partial_2) \subset \ker(\partial_1).$$

The subspace $\ker(\partial_1)$ of C_1 will be sums of edges that form closed loops, and we call these *cycles*. The subspace $\text{im}(\partial_2)$ is the space of *boundaries*. So the above formula says the space of boundaries is contained within the space of cycles. This allows us to define the following quotient, known as the first *homology* group:

$$H_1 := \ker(\partial_1)/\text{im}(\partial_2).$$

These are the cycles modulo boundaries.

With a bit more work we can extend the above sequence (1.3) to

$$0 \xrightarrow{\partial_3} C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} 0$$

so that $\partial_i \circ \partial_{i+1} = 0$ for $i = 0, 1, 2$ and then define

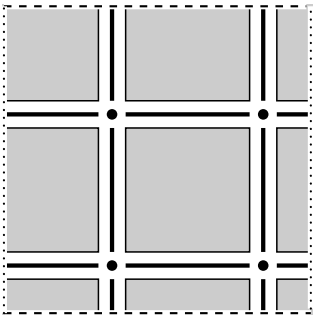
$$H_i := \ker(\partial_i)/\mathrm{im}(\partial_{i+1}), \quad \text{for} \quad i = 0, 1, 2.$$

Now we have a new formula for the Euler characteristic,

$$\mu(H_\bullet) = \dim(H_2) - \dim(H_1) + \dim(H_0), \tag{1.4}$$

which follows from an application of the rank-nullity theorem {What's this?} to equation (1.2).

Going back to the torus example, with four faces, eight edges and four vertices:



we see that the sum over all faces in C_2 has zero boundary,

$$\partial_2(\sum_{f \in \text{faces}} f) = 0$$

and this is the only vector with zero boundary so that that $\dim(H_2) = 1$. The space H_1 is two dimensional, with representative cycles given by a vertical or horizontal loop of edges. Finally, the space H_0 is one dimensional: these are single points. Putting this together we get

$$\mu(S^1 \times S^1) = \dim(H_2) - \dim(H_1) + \dim(H_0) = 1 - 2 - 1 = 0,$$

which agrees with our previous calculation for the Euler characteristic of the torus.

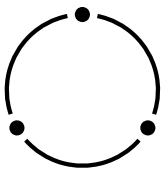
1.2 Classical and quantum codes

This is all great but what does it have to do with quantum codes? Well, before we talk about the quantum case, it is worth first going over what we mean by a code in the classical sense of the word. We wish to communicate a single bit of information, but the communication channel we use suffers from random noise. This noise acts to randomly toggle bits. One way to mitigate against this effect is to just send multiple copies of each bit we wish to communicate. For example, we send either 000 or 111. Once again it is useful to think of this as a three dimensional vector over $\mathcal{F}$. When the message is received any noise can be diagnosed using the check matrix, $S : \mathcal{F}^3 \to \mathcal{F}^3$:

$$S = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}.$$

The codewords belong in the kernel of this operator. Any failure to be in the kernel is presumed to come from a noise process. Notice that this matrix is rank degenerate.

There is a reason for this: we can view it as the boundary operator for the homology of a circle!



In this case, there is one bit for each of the three edges, and the S matrix will record a boundary vertex between non-identical bits.

$$0 \longrightarrow C_1 \xrightarrow{S=\partial_1} C_0 \longrightarrow 0.$$

Thinking of the finite field $\mathcal{F} = \{0, 1\}$ as a classical bit would suggest that the passage to quantum codes involves taking superpositions over these two bit values. Indeed this is what we do. The two dimensional complex Hilbert space that we get is known as a *qubit*:

$$\mathbb{C}[\mathcal{F}] = \{\alpha|0\rangle + \beta|1\rangle, \quad \alpha, \beta \in \mathbb{C}\}.$$

Notice how we put the $\mathcal{F}$-linear values inside the ket. Taking $n-$fold tensor products of a qubit corresponds to superpositions over n dimensional $\mathcal{F}$-linear values. This basis we call the *computational basis.*

Using this basis, we write matrices for the two important operators, Pauli X and Z:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

These two operators generate the (real) single qubit *Pauli group* $\mathcal{P}_1$. We call these *bitflip* and *phaseflip* operators, respectively. For the n-qubit Pauli group $\mathcal{P}_n$ we take n-fold tensor products of I, X, and Z, where I is the single qubit identity operator. We suppress the tensor symbol in such products, for example writing XII for $X \otimes I \otimes I$.

Now we take the classical repetition code and try the following quantum version:

$$\alpha|0\rangle + \beta|1\rangle \mapsto \alpha|000\rangle + \beta|111\rangle.$$

To detect any single bitflip error, such as XII, IXI or IIX we measure the *check* operators ZZI, IZZ. The outcome of such measurements we call a *syndrome* because these serve to diagnose an error process.

However, bitflip errors are not the only unitary operators that we would like to detect. Indeed, any single bit phaseflip operator, ZII, IZI or IIZ has the effect

$$\alpha|000\rangle + \beta|111\rangle \mapsto \alpha|000\rangle - \beta|111\rangle$$

which goes unnoticed by our syndrome measurements. Effectively we still have a classical code.

In order to move towards the solution to this problem, we examine more closely the action of the check operators. Given any state

$$|\psi\rangle = \alpha|000\rangle + \beta|111\rangle$$

we have

$$g|\psi\rangle = |\psi\rangle$$

for $g \in \{III, ZZI, IZZ, ZIZ\}$. In other words, $|\psi\rangle$ is *stabilized* by the group generated by ZZI and IZZ. This motivates the following definition. A *stabilizer code* is specified by a commutative subgroup S of $\mathcal{P}_n$ such that $-I \notin S$. We define the subgroup $\mathcal{P}_n^X$ to be generated by n-fold tensor products of the I and X operators. Similarly, $\mathcal{P}_n^Z$ is generated by n-fold tensor products of the I and Z operators.

We now make the restriction that the generators of the stabilizer group come from either $\mathcal{P}_n^X$ or $\mathcal{P}_n^Z$. This is known as a *CSS stabilizer code* [47, 29].

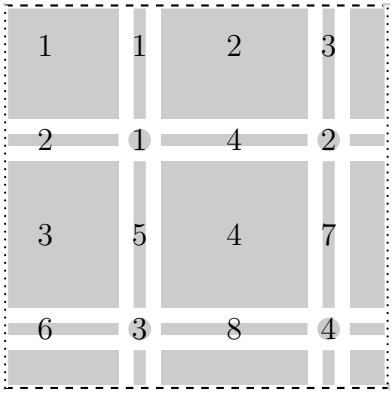
Both of $\mathcal{P}_n^X$ and $\mathcal{P}_n^Z$ are abelian and isomorphic as groups to the n-fold product of $\mathbb{Z}_2$. But $\mathbb{Z}_2$ is more than an abelian group, it's also a field, which we have been notating as $\mathcal{F}$. In this way, we identify these groups with the n-dimensional vector space over the field $\mathcal{F}$:

$$\mathcal{P}_n^X \cong \mathcal{F}^n, \quad \mathcal{P}_n^Z \cong \mathcal{F}^n.$$

Using this identification, the commutativity of operators $u \in \mathcal{P}_n^X$ and $v \in \mathcal{P}_n^Z$ is equivalent to the $\mathcal{F}$-linear inner product of u and v being zero.

So the theory of CSS stabilizer codes becomes amenable to the theory of finite dimensional vector spaces. But there's more than this. It turns out that such a stabilizer code is essentially equivalent to a homology!

We show how this works by using the above example of torus homology. This example is known as the *Kitaev toric code* [33]. Here we separately number the faces, edges and vertices as



Using this ordering we write the boundary operators as the following matrices, with zero entries indicated by dots:

$$S_X = \partial_2^\top = \begin{pmatrix} 1 & 1 & 1 & . & . & 1 & . & . \\ 1 & . & 1 & 1 & . & . & . & 1 \\ . & 1 & . & . & 1 & 1 & 1 & . \\ . & . & . & 1 & 1 & . & 1 & 1 \end{pmatrix},$$

$$S_Z = \partial_1 = \begin{pmatrix} 1 & 1 & . & 1 & 1 & . & . & . \\ . & 1 & 1 & 1 & . & . & 1 & . \\ 1 & . & . & . & 1 & 1 & . & 1 \\ . & . & 1 & . & . & 1 & 1 & 1 \end{pmatrix}.$$

The rows of $\partial_2^\top$ become the X type generators of the stabilizer group, and the rows of ∂_1 are Z type generators. It follows that the homology condition $\partial_1 \partial_2 = 0$ is exactly the commutativity requirement $S_Z S_X^\top = 0$ for a stabilizer code. Writing m_X for the

rows of S_X and m_Z for the rows of S_Z we have the following sequence:

$$\mathcal{F}^{m_X} \xrightarrow{S_X^\top} \mathcal{F}^n \xrightarrow{S_Z} \mathcal{F}^{m_Z}. \tag{1.5}$$

The S_Z operators detect bitflip errors $u \in \mathcal{F}^n$ via $\mathcal{F}$-linear multiplication on the left:

$$v = S_Z u.$$

This vector is the *syndrome* vector. We now dissect the space $\mathcal{F}^n$ according to the kernel of S_Z. Writing $\mathcal{F}^n$, the space of bitflip operators, as a direct sum:

$$\mathcal{F}^n = \ker(S_Z) \oplus T_X$$

where the kernel of S_Z are the *undetectable errors*, or cycles. Everything else is in the space T_X, which are the *detectable errors*. The undetectable errors contain the X type stabilizers, or boundaries, which don't effect the codespace. Also in the kernel of S_Z are the X type logical operators, which we write as L_X. These operators are cycles that are not boundaries, and so they represent elements of the homology H_1.

We already know $S_Z S_X^\top = 0$ so that vectors in the row space of S_X are undetectable by S_Z. Note that S_X and S_Z are rank degenerate matrices, so we make the non-degenerate matrices $\tilde{S}_X$ and $\tilde{S}_Z$ by deleting rows:

$$\tilde{S}_X = \begin{pmatrix} 1 & 1 & 1 & . & . & 1 & . & . \\ 1 & . & 1 & 1 & . & . & . & 1 \\ . & 1 & . & . & 1 & 1 & 1 & . \end{pmatrix}, \quad \tilde{S}_Z = \begin{pmatrix} 1 & 1 & . & 1 & 1 & . & . & . \\ . & 1 & 1 & 1 & . & . & 1 & . \\ 1 & . & . & . & 1 & 1 & . & 1 \end{pmatrix}.$$

We find a right inverse to $\tilde{S}_Z$ and form the matrix T_X:

$$\tilde{S}_Z T_X^\top = I$$

where I here is the appropriate $\mathcal{F}$-linear identity. This T_X matrix has as rowspace the *detectable errors:*

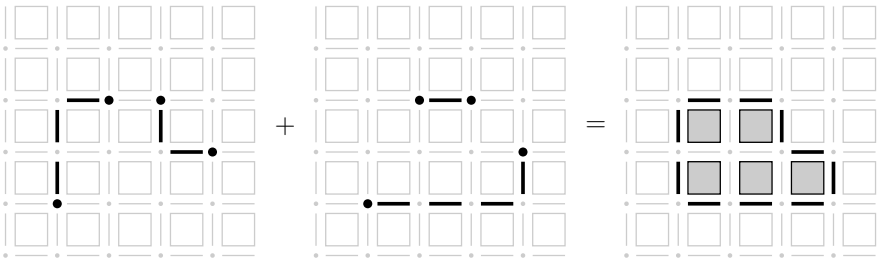
$$T_X = \begin{pmatrix} . & . & . & . & 1 & . & . & 1 \\ . & . & . & . & . & . & 1 & . \\ . & . & . & . & . & . & . & 1 \end{pmatrix}.$$

The rows of this matrix, together with those from $\tilde{S}_X$, form a six dimensional subspace of $\mathcal{F}^n$. The other two dimensions are spanned by operators L_X such that $L_X^\top S_Z = 0$:

$$L_X = \begin{pmatrix} 1 & . & . & . & 1 & . & . & . \\ . & 1 & . & 1 & . & . & . & . \end{pmatrix}.$$

Together this forms an (L, S, T)-decomposition of the CSS stabilizer code S [37, 90]. {What is the point of this?}

To see the error correction process more vividly, we expand the code dimensions. Here we show a tiling of the torus, with $m_X = 5 \times 5$ tiles. There are two edges per tile, so this code has $n = 50$ qubits:

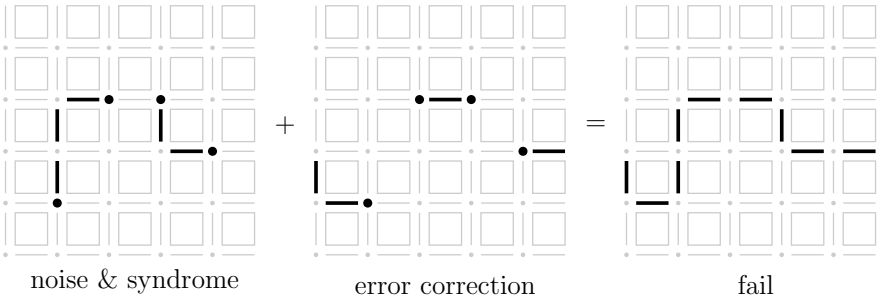


We show a noise process that acts by bitflip errors and the resulting syndrome. The noise process acts on qubits, this is a vector in $\mathcal{F}^n$:

$$c \in \mathcal{F}^n = C_1.$$

So the error process is represented as some collection of edges. The syndrome operator S_Z gives the boundary of these edges, shown as black vertices. The error correction procedure takes these boundary vertices as input and attempts to reconstruct the most likely collection of edges with this boundary. This then is the operator $c' \in \mathcal{F}^n$ that is applied to correct the error. Note that $c + c'$ is a cycle because the vertices of c and c' cancel out. If the resulting operator $c + c'$ is in the image of S_X, ie. a boundary, then the error correction has succeeded.

Otherwise, $c + c'$ is not a boundary and represents a non-trivial operator in H_1 and will therefore alter the encoded qubits {If this is the first place you introduce a homologically non-trivial operator you need to explain why it corresponds to a non-trivial logical operator.} :



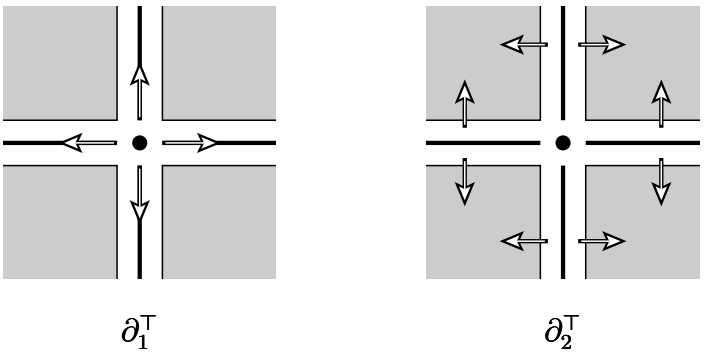
To detect phase-flip errors, we can perform a change of basis on each qubit ... {fixme} this swaps the matrices of the X and Z operators and so the above sequence (1.5) becomes:

$$\mathcal{F}^{m_Z} \xrightarrow{S_Z^\top} \mathcal{F}^n \xrightarrow{S_X} \mathcal{F}^{m_X}$$

and we can repeat the same error correction analysis that we did for bitflip errors.

... {fixme}

Given any homology $\partial_1 \partial_2 = 0$ we get another homology by taking the transpose: $\partial_2^\top \partial_1^\top = 0$.



1.3 The energetic viewpoint

So far we have been considering how to protect quantum information using the framework of error correction. An alternative perspective arises by considering energetic protection. This works by considering the stabilizer generators G_0 as the terms of a Hamiltonian:

$$H = \sum_{g \in G_0} g.$$

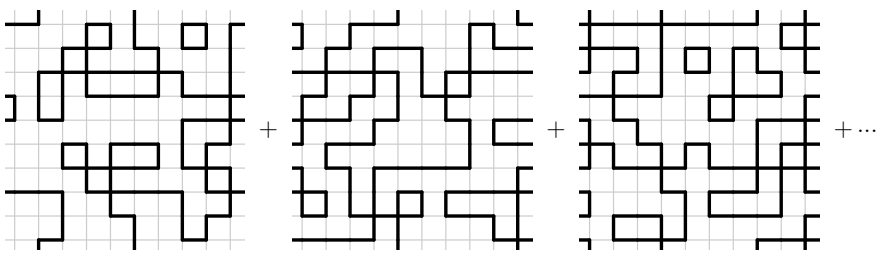
Note that in this thesis we use a neg-Hamiltonian convention, so that the groundspace belongs to the largest eigenvalue of H. Because all the terms in H commute, we can label the eigenspaces of this Hamiltonian uniquely by the eigenvalues of the stabilizers. The groundspace is the simultaneous $+1$ eigenspace of the stabilizers and therefore corresponds exactly to the stabilized codespace above.

In terms of error correction, we think of noise processes as being diagnosed by a syndrome. But here effects of noise are now interpreted energetically, as particle creation. Any bitflip error "creates" particles at the vertex endpoints. In other words, the syndrome is interpreted as a collection of particles. These particles are called *anyons* because of their unusual exchange statistics.

We can write down elements of a basis for the groundspace by summing over the orbit of the stabilizer code,

$$\sum_{g_X \in S_X} g_X |l_X\rangle,$$

where l_X is any logical bitflip operator, written as a computational basis element (inside the ket). Such a basis element looks like a so-called string-net condensate [66]:



It is clear from this picture that the state is stabilized and hence belongs in the groundspace of H : the S_Z stabilizers act as $+1$ on this state because there are no vertex endpoints, and the S_X stabilizers act to permute the terms of the sum.

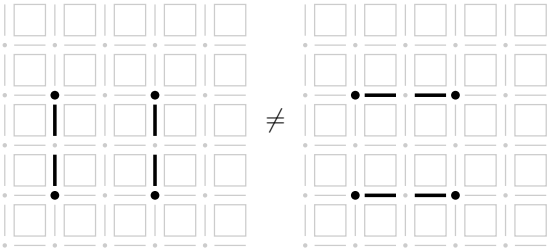
1.4 Two roads to Non-abelian codes

There are two approaches to non-abelian codes explored in this thesis. The first involves relaxing the commutativity of the Hamiltonian terms. These are the gauge code Hamiltonians discussed in chapter 2. While these Hamiltonians are no longer easily diagonalizable, we still find the stabilizers playing an important role. In particular, we generalize the (L, S, T)-decomposition to these Hamiltonians, and show how this relates to the string-net condensation picture. Building states by summing over the orbit of the terms of the Hamiltonian is the basic idea behind group representation theory.

With commuting Hamiltonian terms it is easy to find the spectral gap, which is the difference between the groundspace eigenvalue and the first excited eigenvalue. In chapter 2 we make progress understanding the spectrum of the non-commuting gauge code Hamiltonians, with particular attention payed to the gap.

The second approach to non-abelian codes can be understood from an algebraic topology perspective. A practitioner of these arts would likely describe the fundamental group of a topological space as being the non-abelian version of its homology. And this is indeed closely related to the theory of anyons and modular functors which we describe in chapter 3. Then in chapter 4, we go on to show how error correction can be simulated in these systems.

In the abelian theory the following two processes are equivalent (homologous), but for general anyon theories this is not the case:



While chapter 4 relies on chapter 3, chapter 2 is independent of these.

1.5 Philosophy

It seems that physics has a long history of surprising encounters with advanced mathematical concepts, long after the mathematicians themselves have finished being excited by them. This would suggest the following algorithm for success in theoretical physics: {This is verging on too informal for a thesis. }

1. look at what mathematicians were getting excited about several decades ago,
2. ???
3. profit!

This is somewhat the philosophy of the present thesis. The downfall of this is perhaps that some concepts are elucidated in an overly technical manner. However, the author feels this approach to be useful as it makes contact with a shared mathematical language.

1.6 Summary of thesis

{we gonna say some stuff}

1.7 Acknowledgement

It is a great privilege to work with deeply intelligent people. In this I am honoured to have been able to collaborate with Courtney Brell and Steven Flammia. Also, I could not have finished this research without the generous and patient ear of my supervisor, Andrew Doherty. His ability to understand my blabbering, even when I did not, saved me many times.

CHAPTER 2

Representations and Spectra of Gauge Code Hamiltonians

2.1 Introduction

Physicists often like to solve Hamiltonians using a change of basis, or spin transform. But we can also work with transformations on the level of a group of operators, and later on figure out the spin transform (if needed). This is in line with the thinking of Gottesman who proposed a Heisenberg picture for quantum computing [48]. Here states are specified by the operators that act on them, instead of expicitly working with the states themselves. This is often harder than just manipulating the states themselves, but when it works it can yield new perspectives on the dynamics of the system. This is also the philosophy of category theory, where the goal is to lift information about elements of some mathematical object up to the level of the operators (morphisms) on the objects themselves. However, forgetting about the meaning of the symbols in this way leaves one with the question: "What is an operator?"

From the perspective of a quantum code these are the things that we use to diagnose errors and perform error correction. We can also interpret these operators as the terms of a Hamiltonian, whose groundspace corresponds to the energetically protected codespace. In the case of mutually commuting operators we can easily diagonalize the Hamiltonian, but for other systems of interest this does not hold.

The mathematicians have a name for this question "what is an operator?" This is known as *representation theory.* We examine three different notions of such representations, with a view to extracting spectral information about the Hamiltonian. Group theory representations give a block diagonalization of the Hamiltonian: these are the *irreducible representations* and in our case can be labelled by stabilizer eigenvalues. The coarser tool of Perron-Frobenius theory [75, 44, 8] gives information about the spectral layout of these blocks in the case of CSS gauge codes. At the finest level, the operators in each of these blocks form a semisimple Lie algebra and ideals in this algebra correspond to tensor products of representations.

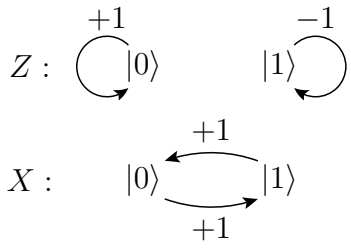
While there are some hints of this theory in the literature [6, 90] here we spell out in detail how this works and much more. Partly it's because these models are new and we don't have many examples.

Using all of these tools we perform exact diagonalization on some large instances of the 3-dimensional gauge color code Hamiltonian [14, 15, 64]. These numerics support the conjecture that these models are gapped, which in turn lends weight to the pos-

sibility that these may be self-correcting quantum memories. Having a constant gap (bound from below) is part of the story of topologically ordered phases [62, 25].

2.1.1 Some motivating examples

Example 1. We start our journey considering a two-dimensional state space. This space is blessed with two basis vectors $|0\rangle$ and $|1\rangle$. The Z and X operators act on these states as:



From this picture we can see that Z acts by *stabilizing* the state $|0\rangle$ and anti-stabilizing the $|1\rangle$ state. The Z operator has been reduced to two operators each acting on a one dimensional subspace: $Z = +1 \oplus -1$. The X operator serves to "bitflip" the state between these two subspaces.

But what happens if we get confused and end up swapping the X and Z operators? We would like to see the X operator as stabilizing / anti-stabilizing two subspaces, together with the Z operator as bitflipping between these. The trick is to consider the *orbits* of the operator we hope to act as a stabilizer. In this case there is only one orbit, $|0\rangle + |1\rangle$ and indeed, the Z operator bitflips this to another state $|0\rangle - |1\rangle$ that is anti-stabilized by X.

We are going to be considering Hamiltonians built from summing operators of this form. In this paper we use a "neg-Hamiltonian" convention, to save complicating expressions with negative signs. The ground space corresponds to the *largest* eigenvalue.

So building a Hamiltonian from a single X or Z term, we find the ground space as the stabilized space by summing over the orbit of that term. The other operator, which we call the *adjacent operator*, acts to bitflip between the eigenspaces.

Example 2. To further elucidate this idea we turn to another example, which is a Hamiltonian built from three commuting and independent operators:

$$H = XXI + IXX + ZZZ.$$

Starting with $|000\rangle$ the terms of the Hamiltonian generate an orbit given by

$$\text{Orbit}(|000\rangle) = \{|000\rangle, |011\rangle, |110\rangle, |101\rangle\}.$$

Notice that the ZZZ term fixes all these states. Summing over this orbit we get the following ground state:

$$\text{GS} = |000\rangle + |011\rangle + |110\rangle + |101\rangle.$$

We select three adjacent operators ZII, IIZ, and IXI, one for each of the stabilizer

operators:

$$ZII : \text{GS} \mapsto |000\rangle + |011\rangle - |110\rangle - |101\rangle \text{ anti-stabilized by } XXI,$$
$$IIZ : \text{GS} \mapsto |000\rangle - |011\rangle + |110\rangle - |101\rangle \text{ anti-stabilized by } IXX,$$
$$IXI : \text{GS} \mapsto |010\rangle + |001\rangle + |100\rangle + |111\rangle \text{ anti-stabilized by } ZZZ.$$

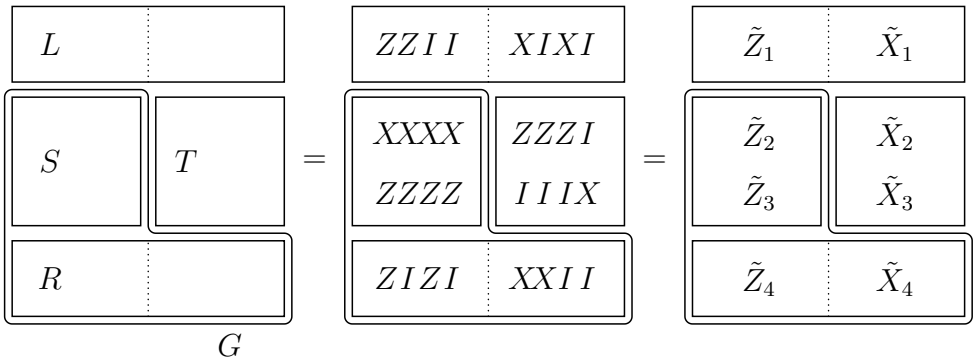
These adjacent operators form an abelian group of order $2^3 = 8$ and by applying each element of this group to the ground state we get a basis of our state space, which we call a *symmetry invariant basis.*

The adjacent operators are not unique in general. For this example we could have also chosen IZZ, ZZI, XXX.

Example 3. Now we consider a four qubit example:

$$H = XXII + IIXX + ZIZI + IZIZ.$$

This time the terms of the Hamiltonian do not generate an abelian group. We will call this group, as generated by the terms in the Hamiltonian, the *gauge group*, G. The *stabilizer* subgroup of G will be the elements of G that commute with every other element in G. By inspection we see this group is generated by $S_0 = \{XXXX, ZZZZ\}$. We can extend these generators to a complete independent generating set for G using the operators $R_0 = \{XXII, ZIZI\}$. These R_0 operators generate the *reduced gauge group* R. The operators adjacent to S_0 we call the *error operators* T_0. We choose $T_0 = \{ZZZI, IIIX\}$. (Once again, these are not unique.) The *logical operators* are the n-qubit Pauli operators outside of the group G that commute with G. In this case they are generated by $L_0 = \{XIXI, ZZII\}$. All of this can be summarized in a table of adjacent pairs:



where the number of rows equals n, each operator commutes with operators on other rows, and anticommutes with the operator on the same row. There is no need to include any phases ($\pm I$) in these tables because phases commute with everything. If we take all the entries in the left column we get the operators $\{ZZII, XXXX, ZZZZ, ZIZI\}$. These generate an abelian group that stabilizes the state $|\psi\rangle = |0000\rangle + |1111\rangle$. Let r be the gauge operator $XXII$ adjacent to the stabilizer $ZIZI$, The state $|\psi\rangle$ then lies in the $G-$orbit

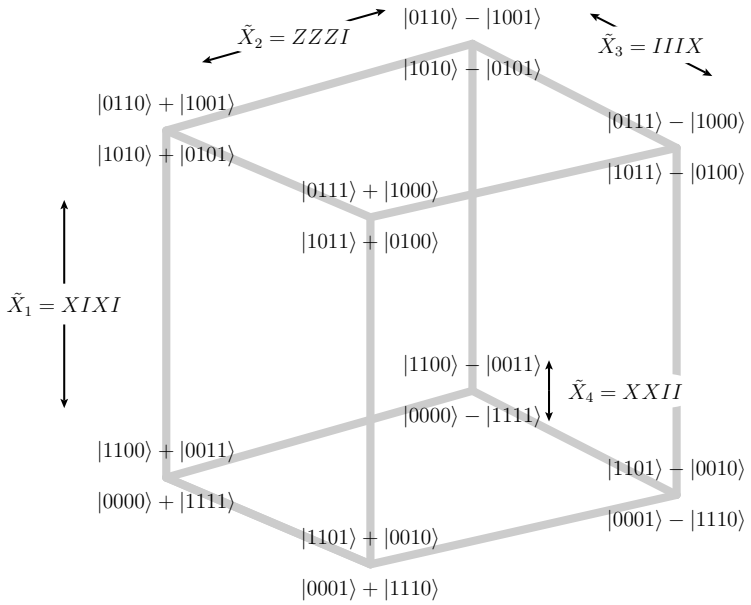
$$\{|\psi\rangle, r|\psi\rangle\} = \{|0000\rangle + |1111\rangle, |1100\rangle + |0011\rangle\}.$$

We use the T_0 operators $t_1 = ZZZI$ and $t_2 = IIIX$ to list three other $G-$orbits:

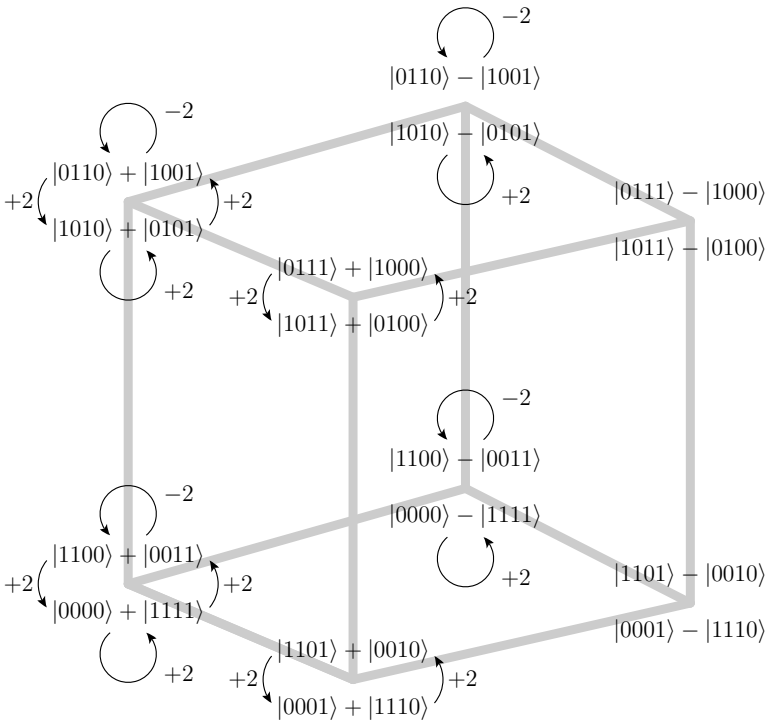
$$\{t_1|\psi\rangle, t_1 r|\psi\rangle\}, \{t_2|\psi\rangle, t_2 r|\psi\rangle\}, \{t_1 t_2|\psi\rangle, t_1 t_2 r|\psi\rangle\}.$$

So now we have sixteen vectors, forming an orthogonal basis for the state space. This is the symmetry invariant basis for this Hamiltonian.

We can now arrange these basis vectors on the vertices of a four dimensional hypercube, such that each dimension corresponds to one of the adjacent $\tilde{X}_i$ bitflip operators. Such an arrangement has a cartesian product structure which induces a tensor product decomposition of the original state space that corresponds to the $\tilde{X}_i$:



The Hamiltonian acts on states by left multiplication. Because this action is a sum of gauge group elements, it will decompose into blocks, one for each $G-$orbit. We depict this action as a weighted graph, where we omit edges with zero weight:



Equivalently we use this basis to write the matrix for the Hamiltonian in block

diagonal form:

$$H = \begin{pmatrix} 2(X+Z) & 0 & 0 & 0 \\ 0 & 2X & 0 & 0 \\ 0 & 0 & 2Z & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \otimes I$$

This block diagonal form will be worked out for general gauge group G below and summarized in Section 2.2.6.

2.2 Group representations

2.2.1 The Pauli group

The Pauli group $\mathcal{P}_1$ is normally defined as a set of matrices closed under matrix multiplication,[1] but we can define it abstractly as the group generated by the (abstract) elements $\{\omega, X, Z\}$ with relations as follows:

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

where I is the group identity. 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. This group has eight elements, and is isomorphic to the dihedral group D_4, the symmetry group of a square.

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

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

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 $\pm I$ will denote the set $\{\omega, I\}$, and $-X$ is ωX, etc.

We write the group commutator as $[[g, h]] := ghg^{-1}h^{-1}$ and note the important commutation relation:

$$[[Z_i, X_j]] = \begin{cases} -I & \text{if } i = j, \\ I & \text{if } i \neq j. \end{cases}$$

If we take an arbitrary $g \in \mathcal{P}_n$ written as a product of the generators, it follows that we can rewrite this product uniquely as $g = \pm g_1 ... g_n$ where each g_i is one of I, Z_i, X_i or $X_i Z_i$ for $i = 1, .., n$. Therefore, the size of the Pauli group is

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

The subgroup of $\mathcal{P}_n$ generated by the elements $\{X_1, ..., X_n\}$ is denoted $\mathcal{P}_n^X$. These

[1] The original definition of the Pauli group also includes an imaginary unit i, which we do not include. So perhaps this should be called the *real* Pauli group.

are the X-type elements. Similarly, $\{Z_1, ..., Z_n\}$ generates the subgroup of Z-type elements $\mathcal{P}_n^Z$.

2.2.2 Subgroups of the Pauli group

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

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

Because G is not abelian, it follows 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 $R \cong \mathcal{P}_r$ for some $1 \leq r \leq n$, and $S \cong \mathbb{Z}_2^m$ for $0 \leq m < n$. Therefore,

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

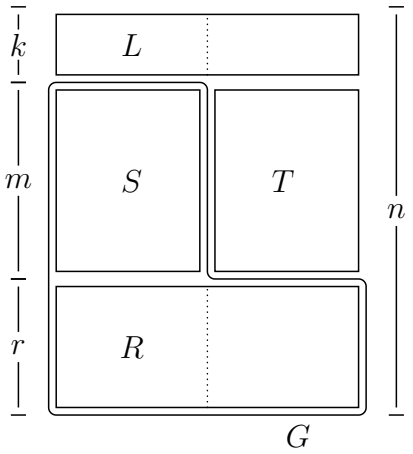
We call R the *reduced gauge group*. We consider both S and R to be subgroups of G. Let $\phi : R \to \mathcal{P}_r$ be a group isomorphism, then $R_0 := \{\phi^{-1}(X_i), \phi^{-1}(Z_i)\}_{i=1,..,r}$ is a set of independent generators of R. We also let S_0 be a set of m independent generators of S.

To find the cosets of G in $\mathcal{P}_n$ we take the group closure of $\mathcal{P}_n - G$; 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 independent generating set for L then we have the important formula:

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

$$= k + m + r \tag{2.3}$$

We summarize the information in this section in a table of Pauli group elements arranged in two columns and n rows:



Here we show the $2n$ generators of $\mathcal{P}_n$ arranged so that each row contains a pair of generators, where each such generator anti-commutes with the operator on the same

row and commutes with all the other operators in the table. Note that this is exactly the definition of the Pauli group via a presentation given in the previous section. Furthermore, the table shows $2k$ generators of L, m generators each for S and T, and $2r$ generators of R. The gauge group G encloses R and S, and one can immediately see how L and T also form a gauge group.

2.2.3 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 independent generators $\{X_1, .., X_n, Z_1, .., Z_n\}$, ρ_{pauli} is defined as the following tensor product of 2×2 matrices:

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

$$\rho_{\text{pauli}}(Z_i) := \bigotimes_{j=1}^{n} \begin{cases} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} & \text{for } j = i \\ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & \text{for } j \neq i \end{cases}$$

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:

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

Since $|\mathcal{P}_n| = 2^{2n+1}$ 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 independent 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. Sum-

ming the squares of the dimensions shows that we have a complete set of irreps of $\mathcal{P}_n$.

2.2.4 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 independent 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 occurrences 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 :

$$\rho_{\text{pauli}} = \bigoplus_{t \in T} \rho_t \otimes I^{\otimes k}$$

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.

2.2.6 The Hamiltonian

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

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

Using the above decomposition we find:

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

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

Fact 0: The Hamiltonian is block diagonalized, with blocks indexed by operators t in the abelian group T and multiplicity 2^k :

$$H = \bigoplus_{t \in T} H_t \otimes I^{\otimes k}.$$

More generally, we can assign real valued weights $J_g \in \mathbb{R}$ to each operator $g \in G_0$,

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

In other words, using weights does not change the block structure of H.

In the following sections we will forget the distinction between g and $\rho_{\text{pauli}}(g)$, so terms such as Z and X can be understood as the corresponding Pauli linear operators.

2.3 Applications

We now use the tools built so far to analyze two examples of gauge code Hamiltonians. The above procedure is not entirely automatic, it relies on extracting the isomorphism ϕ, but when this can be made to work it works surprisingly well.

2.3.1 The 2D compass model

Here we consider the two-dimensional compass model [5]. 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

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

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

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

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

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

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 $k+m+r = n$ 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 :

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

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

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

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

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

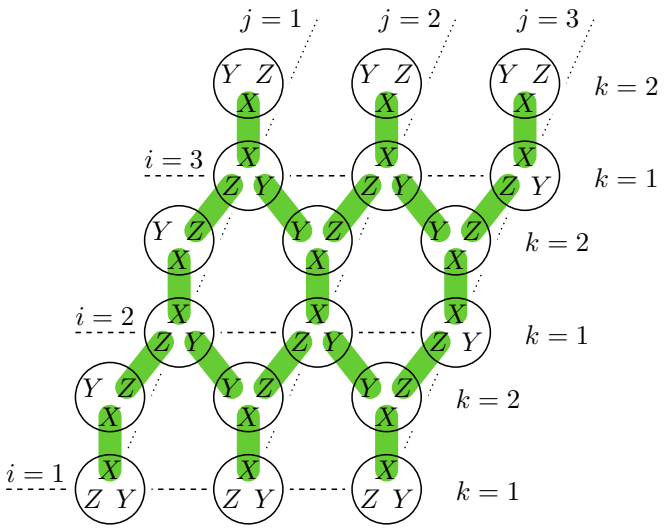
Note the transposition symmetry between the X and Z-type operators. We sum all these terms to find the form of the Hamiltonian in each block:

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

We note that in [26], they perform a spin transformation of the compass model which also results in an $(l-1) \times (l-1)$ lattice of spins and identical Hamiltonian up to some signs.

2.3.2 The Kitaev honeycomb model

The Kitaev honeycomb model [60] 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. Gauge generators have support on the edges of the honeycomb lattice, and we depict qubits here as circles:



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.[2] 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:

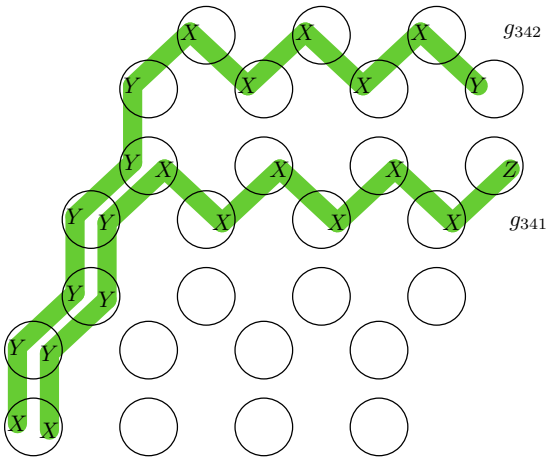
$$h_v := \prod_{i=1}^{l} Y_{i11}Y_{i12}, \quad h_h := \prod_{j=1}^{l} X_{1j2}X_{2j1}.$$

This gives independent 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 $|R_0| = n - 2$.

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).

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

[2]This operator Y is not Hermitian, but it only appears in a tensor pair in the Hamiltonian and so these terms will be Hermitian.



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 independent 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 independent generating set of $\mathcal{P}_r$:

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

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 [56, 60].

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 independent 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:

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

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:

$$H_\rho = \sum_{ij} \Gamma_{ij}(\rho)c_i c_j + \Lambda_\rho$$

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

In [59] they introduce a similar set of mutually anti-commuting string operators R_0.

2.4 Symplectic representations

This is a way of "brute-forcing" the representations when we cannot find a way of writing them down in a closed form expression. For finite systems this yields an algorithm that is efficiently implementable.

In this section, and the remainder of this chapter, we restrict our attention to gauge groups formed from terms in $\mathcal{P}_n^X \cup \mathcal{P}_n^Z$. We call these *CSS gauge codes.* We next turn to a discussion of the symplectic structure of these operators.

Let $\mathcal{F}$ denote the finite field with two elements 0 and 1. Both $\mathcal{P}_n^X$ and $\mathcal{P}_n^Z$ are abelian groups, and can be identified with the additive group structure of the n dimensional vector space over $\mathcal{F}$:

$$\mathcal{P}_n^X \cong \mathcal{F}^n, \quad \mathcal{P}_n^Z \cong \mathcal{F}^n.$$

We do this in the obvious way by sending X_i to the basis vector with 1 in the $i-$th entry, and similarly for each Z_i. We also identify the computational basis of our statespace $\mathbb{C}[2^n]$ with $\mathcal{F}^n$ in the obvious way:

$$\mathbb{C}[2^n] \cong \mathbb{C}[\mathcal{F}^n].$$

This has the potential to be very confusing, and so where appropriate we use X and Z subscripts.

X-type operators act on the $\mathbb{C}[\mathcal{F}^n]$ basis vectors using $\mathcal{F}$ addition:

$$g_X \in \mathcal{P}_n^X \cong \mathcal{F}^n, \quad g_X : v \longmapsto g_X + v$$

Z-type operators act on the $\mathbb{C}[\mathcal{F}^n]$ basis vectors using $\mathcal{F}$ inner product:

$$g_Z \in \mathcal{P}_n^Z \cong \mathcal{F}^n, \quad g_Z : v^\top \longmapsto g_Z v^\top$$

This is an $\mathcal{F}$ scalar, just zero or one. We think of this as a "syndrome". This suggests that actually these Z-type operators live in the dual vector space $\mathcal{F}_n$. Because of the underlying symmetry (and notational confusion) between the X and Z-type operators, we make the convention that by default all our $\mathcal{F}$-vectors come as row vectors (ie. dual vectors). This means we use the transpose operator $^\top$ to indicate a primal (column) vector.

It doesn't make sense to add an X-type operator and a Z-type operator:

$$g_Z + g_X \quad \textit{don't do this!!!}$$

but it does make sense to take the inner product:

$$g_Z g_X^\top = g_X g_Z^\top.$$

This is an $\mathcal{F}$ scalar which gives the commutator of the two operators.

An $\mathcal{F}$-linear operator such as $A : \mathcal{F}^n \to \mathcal{F}^m$ acts on the left as $u^\top \mapsto Au^\top$. It also acts on dual vectors as $A : \mathcal{F}_m \to \mathcal{F}_n$ which corresponds to acting on the right: $v \mapsto vA$. We call the rowspace of A the *span* and denote it as

$$\langle A \rangle = \{vA | v \in \mathcal{F}_m\}$$

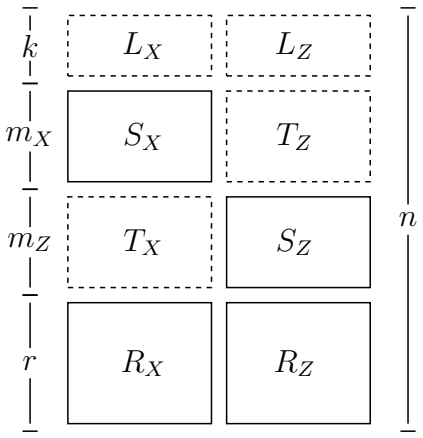
The kernel of A is defined as

$$\ker(A) = \{u^\top | u^\top \in \mathcal{F}^n, \quad Au^\top = 0\}.$$

We wish to use this language to decompose a CSS gauge group G. First we write the gauge group generators in terms of X-type and Z-type operators:

$$G_0 = G_X \cup G_Z.$$

Following the theory from the previous section, we are going to rewrite the gauge group generators as a union of stabilizer generators $S_0 = S_X \cup S_Z$ and reduced gauge generators $R_0 = R_X \cup R_Z$. Similarly, the error operators will be split into X and Z type operators T_X and T_Z and finally the logical operators L_X and L_Z. We summarize all of these sets in the following table:



The solid rectangles indicate operators that span the X and Z parts of the gauge group, and the dashed rectangles indicate operators that do not live inside the gauge group.

We consider each of these blocks $L_X, L_Z, S_X, T_Z, T_X, S_Z, R_X, R_Z$, as well as G_X, G_Z, as either a set of $\mathcal{F}_n$ vectors (the rows) or as an $\mathcal{F}$-linear operator. For example, we write $u \in R_X$ to mean u is a row of the matrix R_X.

We first find the stabilizers S_Z. These are built out of $\mathcal{F}_n$ vectors from the span of

G_Z that commute with the rows of G_X :

$$\begin{aligned}\langle S_Z \rangle &= \{\ vG_Z \mid vG_Z G_X^\top = 0, \quad v \in F_{|G_Z|}\ \} \\ &= \{\ vG_Z \mid v^\top \in \ker(G_X G_Z^\top)\ \}.\end{aligned}$$

The generators (rows of S_Z) are then extracted from this span by row reduction. We swap the role of X and Z to find S_X.

Once we have the stabilizers, in order to complete the above table as a presentation of the Pauli group we solve the following $\mathcal{F}$-linear block matrix equation,

$$\begin{pmatrix} L_X \\ S_X \\ T_X \\ R_X \end{pmatrix} \begin{pmatrix} L_Z \\ T_Z \\ S_Z \\ R_Z \end{pmatrix}^\top = I,$$

subject to the restriction that the rows of R_X lie in the span of G_X and the rows of L_X do not. Similarly for R_Z and L_Z. This set of 16 equations is quadratic in the unknown variables and so it is not obvious how to proceed, but it turns out a systematic way can be found.

We begin by finding L_Z. These operators satisfy the following *homology* condition:

$$l_Z \in L_Z \ \text{ is given by } \ l_Z^\top \in \ker(G_X) \ \mod \ \langle S_Z \rangle.$$

In other words, L_Z is formed from a basis for the kernel of G_X row-reduced using S_Z. To be more specific we take any direct sum decomposition

$$\mathcal{F}_n = \langle S_Z \rangle \oplus V$$

then the operation of mod $\langle S_Z \rangle$ is the projection onto V. We can explicitly write such a projector as the $n \times n$ matrix given by

$$P_Z = I + A^\top S_Z$$

where the matrix A is the $m_Z \times n$ matrix consisting of the leading 1's in any row-reduction of S_Z. We define P_X similarly for the operation of mod $\langle S_X \rangle$.

To find L_X we solve the following $\mathcal{F}$-linear system:

$$\begin{pmatrix} L_Z \\ G_Z \end{pmatrix} L_X^\top = \begin{pmatrix} I \\ 0 \end{pmatrix}$$

The reduced gauge group matrix R_X is found as a row-reduction of $G_X P_X$. We cannot merely set R_Z to be $G_Z P_Z$ because we also require $R_X R_Z^\top = I$. Instead we define the auxiliary matrix $\widetilde{R}_Z$ to be a row-reduction of $G_Z P_Z$.

The error operators T_X are then found as the solution to the $\mathcal{F}$-linear system:

$$\begin{pmatrix} L_Z \\ S_Z \\ \widetilde{R}_Z \end{pmatrix} T_X^\top = \begin{pmatrix} 0 \\ I \\ 0 \end{pmatrix}$$

And then the operators T_Z solve the $\mathcal{F}$-linear system:

$$\begin{pmatrix} L_X \\ S_X \\ T_X \\ R_X \end{pmatrix} T_Z^\top = \begin{pmatrix} 0 \\ I \\ 0 \\ 0 \end{pmatrix}$$

Finally at this point R_Z is given as the solution to

$$\begin{pmatrix} L_X \\ S_X \\ T_X \\ R_X \end{pmatrix} R_Z^\top = \begin{pmatrix} 0 \\ 0 \\ 0 \\ I \end{pmatrix}$$

Note that R_Z and $\widetilde{R}_Z$ have identical span and so we have $R_Z T_X^\top = 0$.

We call this array of eight $\mathcal{F}$-linear matrices an (L, S, T, R)-decomposition of the gauge group. In general this will not be unique for any given gauge group.

2.4.1 The Hamiltonian

The complex Hilbert state space of our Hamiltonian has 2^n dimensions and we write this space as $\mathbb{C}[2^n]$. This notation is meant to suggest that we are forming a $\mathbb{C}$ vector space using 2^n "points" as basis vectors. Working in the computational basis, we do indeed have 2^n such points; these are the elements of $\mathcal{F}_n$. And so we make the identification

$$\mathbb{C}[2^n] \cong \mathbb{C}[\mathcal{F}_n].$$

In other words, we are labeling our basis vectors with elements of $\mathcal{F}_n$ and therefore such notation as

$$\langle u|H|v\rangle$$

with $u, v \in \mathcal{F}_n$ makes sense. We will make further use of this below, by writing $\mathcal{F}$-vector space computations inside the Dirac brackets.

Returning to the code (L, S, T, R)-decomposition above, given the Pauli operator $t \in T$ such that $t = t_X t_Z$ (in $\mathcal{P}_n$) we get a basis for the irrep ρ_t:

$$\{|vR_X + t_X\rangle \text{ such that } v \in \mathcal{F}_r\}.$$

In other words, the basis of the irrep ρ_t is an affine subspace of $\mathcal{F}_n$. Each such affine subspace is indexed by an element of $\mathcal{F}_r$ and all of these are translates of each other, so we make the following identification:

$$\mathbb{C}[\{vR_X + t_X\}_{v \in \mathcal{F}_r}] \cong \mathbb{C}[\mathcal{F}_r].$$

This will allow us to write the components of each block H_t of the Hamiltonian as $\langle u|H_t|v\rangle$ for $u, v \in \mathcal{F}_r$. We make this identification of affine subspaces not out of laziness but because it will help us to compare each of the Hamiltonian blocks H_t below. {I'd suggest you introduce Ht and Htx,tz more prominently. }

Important: The computational basis identifies basis vectors of $\mathbb{C}[2^n]$ with elements of a finite vector space $\mathcal{F}_n$:

$$\mathbb{C}[2^n] \cong \mathbb{C}[\mathcal{F}_n].$$

The (L, S, T, R)-decomposition naturally decomposes $\mathcal{F}_n$ into 2^{m_Z+k} affine subspaces:

$$\{vR_X + t_X + l_X\}_{v\in\mathcal{F}_r}$$

for each $t_X \in \langle T_X\rangle, l_X \in \langle L_X\rangle$. Each such affine subspace forms a basis for the irreducible blocks H_{t_X,t_Z} of H, and can be naturally identified with $\mathcal{F}_r$:

$$H_{t_X,t_Z} : \mathbb{C}[\mathcal{F}_r] \to \mathbb{C}[\mathcal{F}_r].$$

We now wish to understand the action of the gauge group on each of its irreps. Starting with the $t_X, t_Z = 0, 0$ irrep, this is where each of the stabilizers has a trivial action. In $\mathcal{F}^n$ this corresponds to the additive action of the zero vector.

States $u \in \langle R_X\rangle$ can be built from a vector matrix product

$$u = vR_X$$

with $v \in \mathcal{F}_r$. Since $R_X R_Z^\top = I$ we can write $v = uR_Z^\top$. Each $g_X \in G_X$ acts on u to give

$$\begin{aligned} u_1 &= (u + g_X) \bmod \langle S_X\rangle \\ &= (u + g_X)P_X \\ &= (vR_X + g_X)P_X. \end{aligned}$$

writing $u_1 = v_1 R_X$ we then have

$$\begin{aligned} v_1 &= (vR_X + g_X)P_X R_Z^\top \\ &= v + g_X R_Z^\top. \end{aligned}$$

So we have that working in the computational basis, the action of the X part of the gauge group in the $t_X, t_Z = 0, 0$ irrep is to send $v \in \mathcal{F}_r$ to $v + g_X R_Z^\top$. In summary, we have the following contributions from the G_X terms of the Hamiltonian:

$$\langle v|H_{0,0}|v + g_X R_Z^\top\rangle \mathrel{+}= 1, \quad \text{for } g_X \in G_X, v \in \mathcal{F}_r$$

where we use the $+=$ notation because there may be other contributions to the same component. These terms will always be off the diagonal unless g_X is a stabilizer.

The action of the G_Z gauge group contributes to the diagonal of H. These diagonal terms apply a kind of "potential energy" penalty to the basis states that depends on the *syndrome* vector:

$$\text{syndrome}(u) = G_Z u^\top$$

for $u^\top \in \mathcal{F}^n$. This is an $\mathcal{F}$-vector that has one component for each row of G_Z. Writing $|G_Z|$ for the number of these rows, and using a *weight* function w that just counts the number of non-zero entries in any $\mathcal{F}$-vector we have the following contributions to the

Hamiltonian:

$$\langle v|H_{0,0}|v\rangle \mathrel{+}= |G_Z| - 2w(G_Z R_X^\top v^\top),$$

for $v \in \mathcal{F}_r$.

Adding up all of the above we have in summary,

$$H_{0,0} = \sum_{\substack{v\in\mathcal{F}_r \\ g_X\in G_X}} |v + g_X R_Z^\top\rangle\langle v| + \sum_{v\in\mathcal{F}_r} \left(|G_Z| - 2w(G_Z R_X^\top v^\top)\right) |v\rangle\langle v|.$$

For any $t_X \in \langle T_X\rangle$ the Hamiltonian block $H_{t_X,0}$ has components indexed by basis vectors:

$$u = vR_X + t_X$$

this means that the G_X gauge terms have the same effect on $H_{t_X,0}$ as $H_{0,0}$ and only the diagonal has changed:

$$H_{t_X,0} = \sum_{\substack{v\in\mathcal{F}_r \\ g_X\in G_X}} |v + g_X R_Z^\top\rangle\langle v| + \sum_{v\in\mathcal{F}_r} \left(|G_Z| - 2w(G_Z R_X^\top v^\top + G_Z t_X^\top)\right) |v\rangle\langle v|.$$

The general form of each Hamiltonian block is:

$$\begin{aligned} H_{t_X,t_Z} = &\sum_{\substack{v\in\mathcal{F}_r \\ g_X\in G_X}} \eta(t_Z g_X^\top)\ |v + g_X R_Z^\top\rangle\langle v| \\ &+ \sum_{v\in\mathcal{F}_r} \left(|G_Z| - 2w(G_Z R_X^\top v^\top + G_Z t_X^\top)\right) |v\rangle\langle v|. \end{aligned}$$

Here we use η to send $t_Z g_X^\top$ which is an $\mathcal{F}$ value to the multiplicative subgroup $\{\pm 1\}$ of $\mathbb{C}$:

$$\eta(0) = 1,\ \eta(1) = -1.$$

The $\eta(t_Z g_X^\top)$ term is a kind of parity check that picks up one phase flip for (some of) the X type stabilizers found in g_X. This works because T_Z is a left inverse of $S_X^\top$. The $t_Z \in \langle T_Z\rangle$ selects which X type stabilizers act as -1 in this irrep.

In summary, we have the complete representation theory for CSS gauge code Hamiltonians.

2.5 Gapless 1D models

In this section we briefly introduce two important one dimensional models that fit into the CSS gauge code framework.

The XY-model [77] lives on a one dimensional chain of n qubits. We write the gauge group generators as

$$G_0 = \{X_i X_{i+1}, Z_i Z_{i+1} \ \text{ for } \ i = 1, ..., n\}$$

with periodic boundary conditions. For n even this model has no logical operators, one X-type stabilizer and one Z-type stabilizer. With n odd, there are no stabilizers and $k = 1$. Normally the gauge operators are written as $\{X_i X_{i+1}, Y_i Y_{i+1}$ for $i = 1, ..., n\}$ but note that there is an automorphism of the Pauli group that sends G_0 to these operators. For n even, this model is exactly solvable, and the gap goes to zero as the system size grows [67].

For the one dimensional transverse field Ising model, we have

$$G_0 = \{X_i, Z_i Z_{i+1} \quad \text{for} \quad i = 1, ..., n\},$$

with periodic boundary conditions. This model has one X-type stabilizer and no logical operators. This model is also exactly solvable, with gap going to zero as the system size grows [77].

2.6 Perron-Frobenius theory

{You need to make it clearer how much of this is newly defined and how much is developing previous work. Is the Perron-Frobenius construction your own?}

A CSS gauge code is *self-dual* when the X and Z type gauge generators are equal:

$$G_X = G_Z.$$

The XY-model is an example of a self-dual CSS gauge code. A CSS gauge code is *weakly self-dual* when there is a permutation P on the set of n qubits that induces equality of the gauge generators:

$$G_X P = G_Z,$$

where we write P as an $n \times n$ permutation matrix. The compass model is then weakly self-dual when we transpose the square lattice of $l \times l$ qubits.

Any stabilizer that acts as -1 in a given block of the Hamiltonian we will call a *frustrated stabilizer* (with respect to the given block.) Similarly, a *satisfied stabilizer* acts as $+1$ in a given block of the Hamiltonian. We will call a vector *stabilized* if it is a $+1$ eigenvector of every stabilizer in the given gauge group.

We now turn to another notion of reducibility, coarser than the group theoretic reducibility. One way to understand this is via graph theory. Given a CSS gauge code Hamiltonian H, we see that the diagonal terms (working in the computational basis) come from the Z operators and the off-diagonal terms come from X type operators. We think of the Z operators as potential energy, and the X operators as kinetic terms. This suggests the following definition. We define a graph Γ with vertices the 2^n computational basis elements, and edges:

$$|v\rangle \mapsto g_X|v\rangle, \quad \text{for all} \quad v \in \mathcal{F}_n, g_X \in G_X.$$

These are undirected edges because $g_X^2 = I$. We also add weighted loops corresponding to the Z-type gauge operators:

$$|v\rangle \mapsto \sum_{g_Z \in G_Z} g_Z|v\rangle, \quad \text{for all} \quad v \in \mathcal{F}_n.$$

In this way we can consider H and Γ interchangeably, as either a matrix or a graph. An irreducible matrix is one whose corresponding graph is connected. The examples with diagrams in section 2.1.1 are meant to illustrate this graph picture.

The off-diagonal entries of H are all positive. If we use a spectral shift operator, a constant multiple of the identity $+cI$, we find that $H + cI$ is a non-negative matrix. We call such a matrix *Stoquastic* [20].

Lemma 2.1 {Given a ...} The matrix Γ decomposes into irreducible stoquastic

matrices Γ_{t_X} indexed by $t_X \in \langle T_X \rangle$ with multiplicity 2^k:

$$\Gamma = \bigoplus_{\substack{t_X \in \langle T_X \rangle \\ l_X \in \langle L_X \rangle}} \Gamma_{t_X}.$$

Proof: because why? ∎

Lemma 2.2 {Given a ...} For each $t_X \in \langle T_X \rangle$ the largest eigenvalue of Γ_{t_X}, $\lambda_1(\Gamma_{t_X})$ is non-degenerate, and is associated with an eigenvector $v_1(\Gamma_{t_X})$ with positive components.

Proof: By the previous Lemma, we can apply the Perron-Frobenius theorem to the matrices Γ_{t_X}. ∎

A basis for each Γ_{t_X} is given by a coset of G_X in $\mathcal{F}_n$:

$$\left\{ |vS_X + uR_X + t_X\rangle \right\}_{v \in \mathcal{F}_{m_X}, u \in \mathcal{F}_r}.$$

Lemma 2.3 For any weakly self-dual gauge code Hamiltonian H, every groundstate of H is stabilized.

Proof: We will construct a basis for the groundspace consisting of vectors that are stabilized. The set of top eigenvectors

$$\{v_1(\Gamma_{t_X})\}_{t_X \in \langle T_X \rangle}$$

must contain a basis for the groundspace of H. Let such a basis vector be $v_1(\Gamma_{t_X})$ for some t_X. ... ∎

Proposition 2.4 For any weakly self-dual gauge code Hamiltonian H we have:

$$\lambda_1(H) = \lambda_1(H_{0,0})$$

and for $t_X \in \langle T_X \rangle, t_Z \in \langle T_Z \rangle$ with $t_X \neq 0$ or $t_Z \neq 0$,

$$\lambda_1(H) > \lambda_1(H_{t_X,t_Z}).$$

Proof: Consequence of previous lemma. ∎

Notice that $H_{0,0}$ is also irreducible stoquastic (in the computational basis) and so has non-degenerate groundspace, but it appears with multiplicity 2^k within the Hamiltonian H and this accounts for the degeneracy of the groundspace of H.

The next goal is to search for the second eigenvalue of H, $\lambda_2(H)$.

Lemma 2.5 Given a gauge code Hamiltonian H and $t_X \in \langle T_X \rangle$, the blocks $H_{t_X,0}$ are stoquastic.

Proof: because why? ∎

Proposition 2.6 For a weakly self-dual gauge code Hamiltonian H,

$$\lambda_1(H_{t_X,0}) < \lambda_1(H_{t_X,t_Z}) \quad \text{and}$$
$$\lambda_1(H_{0,t_Z}) < \lambda_1(H_{t_X,t_Z}),$$

where $t_X \neq 0$ and $t_Z \neq 0$.

Proof: because why? ∎

Therefore, to find the spectral gap of a weakly self-dual gauge code Hamiltonian, we need only examine the top two eigenvalues of $H_{0,0}$ and the top eigenvalue of $H_{t_X,0}$ for each $t_X \in T_X$. We summarize this in the theorem:

Theorem 2.7 For a weakly self-dual gauge code Hamiltonian H, the spectral gap is given by:

$$\min\Big\{\lambda_1(H_{0,0}) - \lambda_2(H_{0,0}), \min_{t_X \in \langle T_X \rangle} \{\lambda_1(H_{0,0}) - \lambda_2(H_{t_X,0})\}\Big\}$$

Proof: Combine Proposition 2.4 with Proposition 2.6. ∎

2.7 The gauge color code model

We now turn to the central animal that motivated the theory developed in this chapter.

The three dimensional gauge color code [14, 15, 64] is a self-dual CSS gauge code. It is based on the following geometric construction known as a *colex* [17]. We begin with a tetrahedron and subdivide it into finitely many convex 3-dimensional polytopes, or *bodies*. Each body has a boundary consisting of 0-dimensional cells which we call *vertices*, 1-dimensional cells called *edges* and 2-dimensional cells called *faces*. By a *cell* we mean any of these 0,1,2 or 3-dimensional convex polytopes. Any two bodies in this tetrahedral subdivision will have either empty intersection or otherwise intersect on a common vertex, edge or face. When the intersection is on a face these two bodies are called *adjacent*. Two vertices in the same edge will also be called adjacent. Each body is colored by one of four *colors*, either taken to be red, green, blue, yellow or otherwise an element of the set $\{1, 2, 3, 4\}$. The four exterior triangular faces of the bounding tetrahedron are called *regions,* the intersection of two regions is called a *border* and the intersection of three regions is called a *corner.* A cell not contained within any region is called an interior cell.

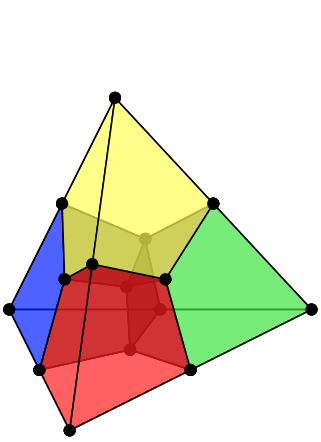
This colored cellulation is required to have the following further properties:

1. Adjacent bodies have different colors.

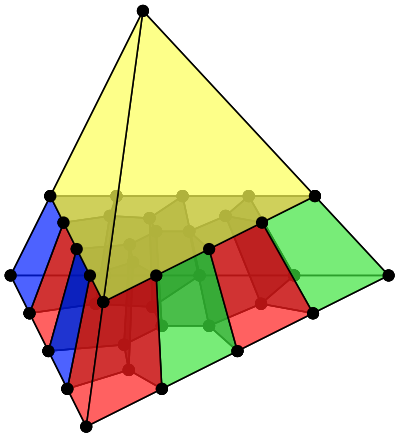
2. Each region has a unique color such that no bodies intersecting that region has that color.

3. All vertices are adjacent to four other vertices, except for the corner vertices which are adjacent to three other vertices.

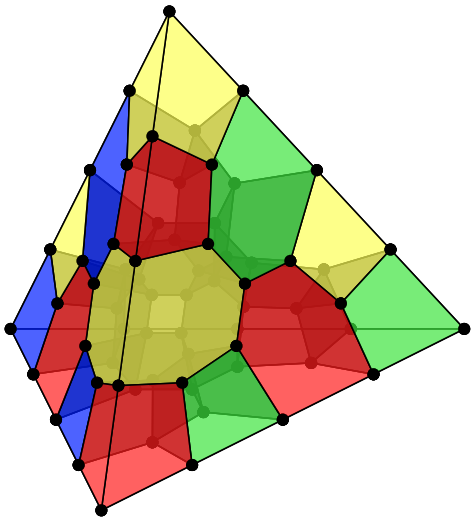
Here we show some instances of this construction, along with the colors of the unobscured bodies. Each instance is labeled by the number of vertices n.



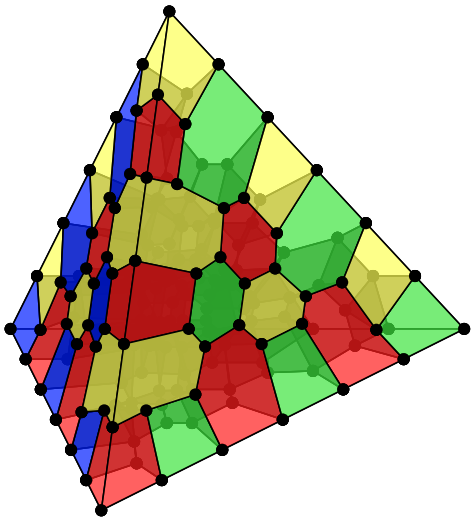
$n = 15$



$n = 39$



$n = 65$



$n = 175$

We note the following consequences of the above conditions. Every face supports an even number of vertices. We now think of each region as corresponding to a "missing" body. Each edge is then contained in the boundary of three bodies. This means we can associate a unique color to each edge, which is also the color of two bodies intersecting a vertex of the edge. That is, each edge joins two bodies of the same color. Each face bounds two bodies, and so we color each face with the two colors of these bodies.

Using this cellulation we now construct the gauge code. Qubits are associated with the n vertices. We associate operators to other cells, or union of cells, by using the contained vertices as support. {Is this correct? All the operators here are built from faces, not cells. } Because this is a self-dual code, the same goes for both the X and Z type operators. The $X/Z-$type gauge group is generated by $X/Z-$type operators supported on each face. The $X/Z-$type stabilizer group is generated by $X/Z-$type operators supported on each body. There is one $X/Z-$type logical operator and these are generated by $X/Z-$type operators supported on any region.

2.8 The orbigraph

Performing numerics on small gauge code models makes it evident that there is a great deal more symmetry than we have found using the above techniques. In particular, the

components of eigenvectors have many identical values. This motivates the following exploration of the symmetry of the Hamiltonian.

A *graph automorphism* is a permutation matrix P such that

$$P^T H P = H.$$

The set of all such graph automorphisms form a group $\mathcal{A}$. The goal here is to find eigenvectors of H that are invariant under the action of $\mathcal{A}$. Such an eigenvector will have components that are constant on the orbits of $\mathcal{A}$, so therefore it will live naturally on the vector space whose basis is these orbits. We can do better than this, and actually project the graph itself down onto this space. We do this by constructing a new graph which we call the *orbigraph*. The matrix for the orbigraph is written $H/\mathcal{A}$ and acts on the vector space with basis consisting of the orbits of $\mathcal{A}$. It follows that the components of $H/\mathcal{A}$, indexed by a pair of $\mathcal{A}-$orbits i and j, is defined by

$$(H/\mathcal{A})_{ij} = |\{g \in G \text{ s.t. } gv \in j\}| \text{ where } v \in i.$$

In other words, $(H/\mathcal{A})_{ij}$ counts the number of gauge group elements that sends any particular element v of the $\mathcal{A}-$orbit i to the $\mathcal{A}-$orbit j.

Here is a simple example. We take as gauge group

$$G = \{XII, IXI, IIX, ZII, IZI, IIZ\}.$$

The Hamiltonian $H = \sum_{g \in G} g$ has matrix:

$$H = \begin{pmatrix} 3 & 1 & 1 & 1 & . & . & . & . \\ 1 & 1 & . & . & 1 & 1 & . & . \\ 1 & . & 1 & . & 1 & . & 1 & . \\ 1 & . & . & 1 & . & 1 & 1 & . \\ . & 1 & 1 & . & -1 & . & . & 1 \\ . & 1 & . & 1 & . & -1 & . & 1 \\ . & . & 1 & 1 & . & . & -1 & 1 \\ . & . & . & . & 1 & 1 & 1 & -3 \end{pmatrix}$$

where we indicate zero entries with a dot. In this case, the automorphism group is $\mathcal{A} = S_3$, there are four $\mathcal{A}-$orbits, and

$$H/\mathcal{A} = \begin{pmatrix} 3 & 3 & . & . \\ 1 & 1 & 2 & . \\ . & 2 & -1 & 1 \\ . & . & 3 & -3 \end{pmatrix}.$$

In this case the automorphism group $\mathcal{A}$ of the graph is the same as the automorphism group $Aut(G)$ of the gauge group, but in general it is possible that $Aut(G) < \mathcal{A}$.

Note that the orbigraph is no longer Hermitian, and in general will not even be normal.

We can also write $H/\mathcal{A} = QHP$ using the following two matrices for P and Q :

$$Q = \begin{pmatrix} 1 & . & . & . & . & . & . & . \\ . & 1 & . & . & . & . & . & . \\ . & . & . & . & 1 & . & . & . \\ . & . & . & . & . & . & . & 1 \end{pmatrix}, \quad P = \begin{pmatrix} 1 & . & . & . \\ . & 1 & . & . \\ . & 1 & . & . \\ . & 1 & . & . \\ . & . & 1 & . \\ . & . & 1 & . \\ . & . & 1 & . \\ . & . & . & 1 \end{pmatrix}.$$

The idea is that each column of P sums over an orbit, and each row of Q chooses one member of each orbit.

In general, we will apply this idea to each of the blocks $H_{t_X,0}$. From **Fact 2** above, and using the fact that we are summing over a trivial representation of $\mathcal{A}$ we have the following:

> **Fact 3:** The spectrum of the orbigraph of $H_{t_X,0}$ contains the ground eigenvalue of $H_{t_X,0}$.

{Please prove this.}

2.8.1 The compass model

For the next example we take the $l = 3$ compass model. $H_{0,0}$ acts on a 16 dimensional space. The order of $\mathcal{A}$ is 72, and we find three $\mathcal{A}-$orbits. The orbigraph method can be applied in the case where the Hamiltonian weights for X and Z are uniform as w_X and w_Z. We separate the X and Z terms of the orbigraph to show how this works:

$$H_{0,0}/\mathcal{A} = \begin{pmatrix} . & 9 & . \\ 1 & 4 & 4 \\ . & 6 & 3 \end{pmatrix} + \begin{pmatrix} 9 & . & . \\ . & 1 & . \\ . & . & -3 \end{pmatrix} = \begin{pmatrix} 9 & 9 & . \\ 1 & 5 & 4 \\ . & 6 & . \end{pmatrix}$$

This can be solved analytically and we find $\lambda_1 = 4 + 2\sqrt{13} \cong 11.21110255$. Keep in mind that the original state space has dimension $2^9 = 512$ so we have come a long way down to 3.

By exact numerical diagonalization we get the spectrum of $H_{0,0}$ and note that the orbigraph lifts all degeneracy as well as missing some excited eigenspaces:

λ	$H_{0,0}$ degeneracy	$H_{0,0}/\mathcal{A}$ degeneracy
11.2111025509	1	1
6.0	1	1
2.0	4	
0.0	4	
-3.21110255093	1	1
-4.0	4	
-6.0	1	

The reason we miss some excited spaces is that they do not contain any trivial irrep of $\mathcal{A}$. Note that we miss the eigenspace with $\lambda = -6$ even though this is one dimensional. It must contain some other non-trivial one dimensional irrep. If we want to make an

orbigraph for these other spaces we would construct the orbigraph by summing over orbits using different characters of $\mathcal{A}$ (these are *momenta* in the abelian terminology). See [35] chapter 5 for more details.

2.8.2 The gauge color code model

The smallest gauge color code has $n = 15$ qubits, G_0 has 18 each of X/Z-type gauge operators, and 4 each of X/Z-type stabilizer generators. $H_{0,0}$ acts on a 64 dimensional space, and $\mathcal{A}$ has order 720. We find 7 orbits:

$$H_{0,0}/\mathcal{A} = \begin{pmatrix} 18 & 18 & . & . & . & . & . \\ 3 & 12 & 15 & . & . & . & . \\ . & 6 & 6 & 12 & . & . & . \\ . & . & 9 & . & 9 & . & . \\ . & . & . & 12 & -6 & 6 & . \\ . & . & . & . & 15 & -12 & 3 \\ . & . & . & . & . & 18 & -18 \end{pmatrix}$$

The eigenvalue equation results in the recurrence relation:

$$\lambda a_k = 3k a_{k-1} + (18 - 6k) a_k + (18 - 3k) a_{k+1},$$

which has largest solution $\lambda_1 = 18\sqrt{2} \cong 25.4558441$.

Numerics give the full spectrum of $H_{0,0}$ and we note that the orbigraph lifts all degeneracy as well as preserving every eigenvalue:

λ	$H_{0,0}$ degeneracy	$H_{0,0}/\mathcal{A}$ degeneracy
25.4558441227	1	1
16.9705627485	6	1
8.48528137424	15	1
0.0	20	1
-8.48528137424	15	1
-16.9705627485	6	1
-25.4558441227	1	1

The second eigenvalue of H comes from a recurrence relation in two variables which has solution $\lambda_2 = 9\sqrt{2} + 3\sqrt{10} \cong 22.21475504$. So the gap for this Hamiltonian is

$$\lambda_1 - \lambda_2 = 9\sqrt{2} - 3\sqrt{10} \cong 3.24108908.$$

2.8.3 A table of orbigraphs

Here we tabulate the order of the graph automorphism group $\mathcal{A}$ of $H_{t_X,0}$ and the resulting orbigraph sizes, which is the number of $\mathcal{A}-$orbits. We use the software library `nauty`[68] for computing graph automorphisms.

model	n	r	t_X	$\lvert\mathcal{A}\rvert$	$\lvert\mathcal{A}-\text{orbits}\rvert$	$\lvert\text{Aut(code)}\rvert$
1D XY	9	8	0	18	23	18
	10	8	0	200	10	20
	11	10	0	22	63	22
	12	10	0	288	36	24
2D compass	9	4	0	72	3	36
	9	4		12	4	
	16	9	0	128	24	64
	16	9		16	48	
	25	16	0	200	430	100
	25	16		20	3418	
3D compass	27	22	0	216	20609	
	27	22		72	60283	
3D gauge color	15	6	0	720	7	$\lvert S_4\rvert = 24$
	15	6		36	16	
	39	18	0	36	14400	$\lvert\mathbb{Z}_3\rvert = 3$
	65	32	0			$\lvert\mathbb{Z}_4\rvert = 4$

We also show the order of Aut(code) which is the automorphism group of the gauge code, defined as follows. Elements of this group act by permuting the n qubits. Such an action is given by the $\mathcal{F}$-linear permutation matrices P_n, P_Z and P_X, such that the following $\mathcal{F}$-linear equations hold:

$$P_X G_X P_n = G_X,$$
$$P_Z G_Z P_n = G_Z.$$

This implies that the action of Aut(code) commutes with the Hamiltonian and preserves eigenvalues of the stabilizers. Therefore this group action restricts to an action on $H_{0,0}$.

Mystery: the graph automorphism group is often bigger, sometimes much bigger, than the automorphism group of the underlying code. So where is the extra symmetry coming from?

A crucial hint is provided by the fact that these graph symmetries respect the symplectic structure in the following sense. Recall that we defined graph symmetries via permutation matrices P such that $P^\top H P = H$. In other words, P is a permutation on the set of basis vectors $\mathcal{F}^n$. It turns out that not only are these maps $\mathcal{F}$-linear, but they also preserve syndromes. By this we mean we can find an $\mathcal{F}$-linear map Q such that the following diagram commutes:

$$\begin{array}{ccc} \mathcal{F}^n & \xrightarrow{G_z} & \mathcal{F}^{|G_Z|} \\ {\scriptstyle P}\downarrow & & \downarrow{\scriptstyle Q} \\ \mathcal{F}^n & \xrightarrow{G_z} & \mathcal{F}^{|G_Z|} \end{array}$$

That P has this extra $\mathcal{F}$-linear behaviour is not true in general, but it holds for the orbigraphs in the above table.

All this suggests a further examination of the commutation structure of the code. Indeed, this is captured by the notion of a Lie algebra, which we turn to next.

2.9 Lie algebra representations

We now turn to a finer notion of representation theory, being the representation theory of semi-simple Lie algebras. We outline the theory of representations of semi-simple Lie algebras [45] and show how this applies to CSS gauge code Hamiltonians.

An abstract *Lie algebra* $\mathfrak{g}$ is a vector space together with a bilinear form:

$$[\ ,\ ] : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$$

such that $[A, B] = -[B, A]$ and $[A, [B, C] + [B, [C, A]] + [C, [A, B]] = 0$.

A *representation* of a Lie algebra $\mathfrak{g}$ on a vector space V is a linear map

$$\rho : \mathfrak{g} \to \mathrm{GL}(V)$$

that sends the abstract bracket to the concrete one:

$$\rho([A, B]) = \rho(A)\rho(B) - \rho(B)\rho(A).$$

A Lie algebra requires us to "forget" about multiplication of operators, and only allow the taking of brackets (and linear combinations.) This is not as crazy as it may at first seem. The fundamental calculation in the theory of quantum stabilizer codes is actually a Lie algebra calculation. Consider a state $|\psi\rangle$ that is stabilized by some operator $s \in S$:

$$s|\psi\rangle = |\psi\rangle,$$

We wish to understand the effect of an error operator $t \in T$ on our state $|\psi\rangle$, where we have $st = -ts$:

$$st|\psi\rangle = ts|\psi\rangle + [s, t]|\psi\rangle = -ts|\psi\rangle = -t|\psi\rangle,$$

which shows that $t|\psi\rangle$ is a -1 eigenvalue of s. The key point here is that nowhere did we need to multiply (compose) two operators, it was all done using the bracket.

We continue the analysis of CSS gauge code Hamiltonians. The terms of the Hamiltonian block H_{t_X,t_Z} form a Lie algebra which we denote $\mathfrak{g}_{t_X,t_Z}$. The basis for this Lie algebra is formed from all iterated brackets of the terms in H_{t_X,t_Z}. This is a concrete Lie algebra, or in other words, it comes with a representation on the vector space $\mathbb{C}[\mathcal{F}_r]$.

The simplest such example of this is the one qubit Lie algebra which is generated by X and Z. This will have basis $\{X, Z, 2XZ = [X, Z]\}$ and so is a three dimensional Lie algebra. In fact, it is isomorphic to $\mathfrak{sl}_2(\mathbb{C})$ the Lie algebra of traceless two by two matrices. Notice that we do not include I in these algebras as this is associated to the multiplicative (group) structure of the operators. Moreover we never need consider Hamiltonians with such terms as these just shift the spectrum by a constant. Notice also that if we try to build a larger Lie algebra from taking iterated brackets of the n-qubit operators $\{X_i, Z_i\}$ we still only get a direct sum of n copies of $\mathfrak{sl}_2(\mathbb{C})$. So while the group generated by products of $\{X_i, Z_i\}$ is the whole Pauli group $\mathcal{P}_n$, as a Lie algebra we get something finer. This reflects the fact that these terms break up into n commuting "pieces", which motivates the following definition.

An *ideal* of a Lie algebra $\mathfrak{g}$ is a Lie subalgebra $\mathfrak{h} \subset \mathfrak{g}$ that "consumes" all the other elements of $\mathfrak{g}$:

$$[A, B] \in \mathfrak{h}, \quad \text{for all} \quad A \in \mathfrak{h}, B \in \mathfrak{g}.$$

In general the structure of the Lie algebra $\mathfrak{g}_{t_X,t_Z}$ will be more complicated than the corresponding group structure. But we do have the following:

The Lie algebra $\mathfrak{g}_{t_X,t_Z}$ is a semi-simple Lie algebra.

This follows from the characterization of semi-simple Lie algebras as those having no nonzero abelian ideals. Any such ideal would correspond to the existence of stabilizers, and we already got rid of these.

We also have a ready made *Cartan subalgebra* of $\mathfrak{g}_{t_X,t_Z}$. This is the algebra $\mathfrak{h}_{t_X,t_Z}$ generated by the $Z-$type terms of H_{t_X,t_Z}. The *weight spaces* are the simultaneous eigenspaces of the operators in the Cartan subalgebra. These eigenspaces are labeled by what we called syndromes previously, and these are all one dimensional because the span of R_X does not intersect the kernel of R_Z. Therefore the representation of $\mathfrak{g}_{t_X,t_Z}$ on $\mathbb{C}[\mathcal{F}_r]$ is irreducible.

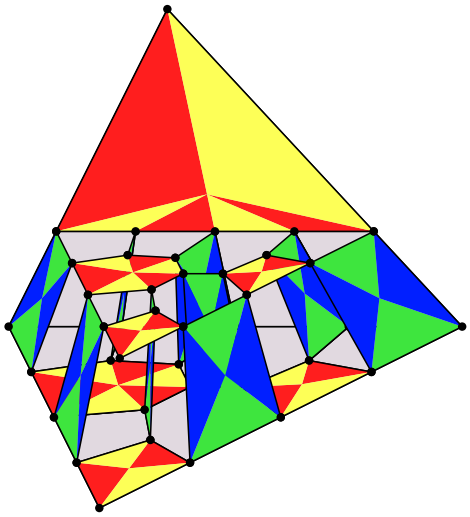
Note that a decomposition of a Lie algebra into disjoint (apart from zero) ideals will give a direct sum decomposition of the Lie algebra. Also, any irreducible representation of a direct sum of Lie algebras can be considered as the tensor product of irreducible representations of the individual summands. This is key to the numerical algorithms below: we examine the ideals generated by the terms in the Hamiltonian block H_{t_X,t_Z}. Each such ideal corresponds to a direct summand of $\mathfrak{g}_{t_X,t_Z}$ and so the spectrum of H_{t_X,t_Z} can be written as a sum over spectra of smaller gauge code Hamiltonians corresponding to each ideal.

2.9.1 Ideal structure of gauge codes

Ideals are generated by anti-commuting operators, and so to find these ideals we search for a partition of the gauge group operators such that operators from different partitions commute.

The XY-model has gauge group terms X_iX_{i+1}, Z_iZ_{i+1} for $i = 1, ..., n$. When $n = 2k$ is even these terms generate two Lie algebra ideals. For $i = 1, ..., k$ the terms $X_{2i}X_{2i+1}$ and $Z_{2i+1}Z_{2i+2}$ generate one ideal, and the other ideal comes from switching X and Z.

We next examine the Lie algebra ideal structure of the gauge color code. Two faces operators, of X and Z type, will anti-commute only when they intersect on a single vertex. This only happens when such faces have disjoint coloring. Here we show an example of this in the $n = 39$ model:



There are three of these arrangements, each corresponding to the three ways of partitioning the set of colors into two sets of two. It follows that $\mathfrak{g}_{t_X,t_Z}$ is the direct sum

of 6 disjoint ideals, and specifically, that each Hamiltonian term in H_{t_X,t_Z} lies in a single one of these ideals. This result is crucial for obtaining the exact diagonalization numerical results below.

2.9.2 Lie algebra classification

Here we explicitly compute decompositions of $\mathfrak{g}_{0,0}$ into the direct sum of simple Lie algebras. First we review the classificiation of simple Lie algebras.

Let n be the rank of a simple lie algebra $\mathfrak{g}$. This is the dimension of the Cartan subalgebra $\mathfrak{h}$.

The simple Lie algebras are classified into four infinite series A_n, B_n, C_n, D_n as well as five other exceptional Lie algebras that we will not need.

The A_n series can be constructed as $\mathfrak{sl}_{n+1}$ which are the traceless $(n+1)\times(n+1)$ matrices. Therefore the algebra dimension is n^2+2n.

For $n \geq 2$ the B_n series comes from the Lie algebras $\mathfrak{so}_{2n+1}$. These can be constructed as $(2n+1)\times(2n+1)$ matrices in block form

$$\begin{pmatrix} P & Q & T \\ R & S & U \\ V & W & 0 \end{pmatrix}$$

with Q, R anti-symmetric and $P^\top = -S, T = -W^\top, U = -V^\top$. This algebra therefore has dimension $2n^2+n$.

For $n \geq 3$ the C_n series comes from the Lie algebras $\mathfrak{sp}_{2n}$. These can be constructed as $2n \times 2n$ matrices in block form

$$\begin{pmatrix} P & Q \\ R & S \end{pmatrix}$$

with Q, R symmetric and $P^\top = -S$. It follows that this algebra has dimension $2n^2+n$.

For $n \geq 4$ the D_n series is $\mathfrak{so}_{2n}$. These can be constructed as $2n \times 2n$ matrices in block form

$$\begin{pmatrix} P & Q \\ R & S \end{pmatrix}$$

with Q, R anti-symmetric and $P^\top = -S$. Therefore this algebra has dimension $2n^2-n$.

2.9.3 A table of gauge code Lie algebras

Using brute-force computation we now find the dimension of the ideals in $\mathfrak{g}_{0,0}$ and therefore which simple Lie algebra these correspond to. Note that here we switch back to using n to denote the number of qubits, which in general is not the rank of the Lie algebra.

model	n	r	t_X	$\mathfrak{g}_{t_X,0}$
1D XY	9	8	0	D_9
	10	8	0	$D_5 \oplus D_5$
	11	10	0	D_{11}
	12	10	0	$D_6 \oplus D_6$
1D Ising	$4, ..., 16$	$n-1$	0	D_n
2D compass	9	4	0	A_{15}
	16	9	0	D_{256}
3D gauge color	15	6	0	$6A_1$
	39	18	0	$6A_7$
	65	32	0	$4A_{31} \oplus 2A_{63}$

This verifies the ideal decompositions we found in the previous section, and also corroborates the large amount of symmetry found with the orbigraph method. For example, the S_6 symmetry of the $n = 15$ gauge color code corresponds to permutations of the six A_1 ideals.

There is a remaining mystery of where the extra $\mathbb{Z}_2$ symmetry of the 2D compass model is coming from. This symmetry was found with the orbigraph method in section 2.8.3. It would be interesting if this symmetry turns out to be an element of the Weyl group of the Lie algebra $\mathfrak{g}_{t_X,0}$.

2.10 Numerical results

Here we show tables for the first and second eigenvalues of the compass and gauge color code models. These results are obtained using exact diagonalization methods. For each instance we indicate the groundspace eigenvalue λ_1 which is obtained from $H_{0,0}$. Then we list the second eigenvalue of $H_{0,0}$ as well as the first eigenvalue of $H_{t_X,0}$ for $t_X \neq 0$. The weight of the corresponding frustrated stabilizer is $w(s_Z)$. The eigenvalue closest to $\lambda_1(H_{0,0})$ is marked with a tick, along with the value of the gap, $\lambda_1 - \lambda_2$. We only show the results for a single frustrated stabilizer generator, as it was confirmed numerically that adding further frustrated stabilizers never produces a better candidate for λ_2. (This involved performing exact diagonalization on the top eigenvalues of every $H_{t_X,0}$ block in the Hamiltonian.) Also, we only show non-isomorphic stabilizer generators, under the lattice symmetry of the model. We use the iterative solvers in software library `SLEPc` [50] to find these eigenvalues.

2D compass code model

n	t_X	$w(s_Z)$	λ_1	λ_2 ?	gap
16	0		19.012903	16.335705	
		8		18.369300 ✓	0.643603
25	0		29.076200	27.597280	
		10		28.624004 ✓	0.452196
36	0		41.410454	40.585673	
		12		41.094532 ✓	0.315922

Such numerics for the 2D compass model have been previously found using similar methods [26].

3D compass code model

n	t_X	$w(s_Z)$	λ_1	λ_2 ?	gap
27	0		60.295471	58.382445	
		18		59.757677 ✓	0.53779

3D gauge code model

n	t_X	$w(s_Z)$	λ_1	λ_2 ?	gap
15	0		25.455844	16.970563	
		8		22.214755 ✓	3.241089
65	0		104.076026	99.014097	
		8		100.429340	
		12		100.585413	
		12		101.602340	
		18		102.382483 ✓	1.693543
175	0		267.197576	264.250644	
		8		263.171190	
		8		263.324858	
		8		263.340832	
		12		264.269635	
		12		264.617135	
		12		264.745548	
		18		264.843629	
		18		265.413935	
		18		265.754772	
		24		266.148188 ✓	1.04939

The gap of the 3D gauge color code is clearly far more robust than the other models, see Figure 2.1. It does decrease with n, but note also that the stabilizers in the code are also growing, up to weight 24. To emphasize this point we show in Figure 2.2 the ground eigenvalues of all of these blocks $H_{t_X,0}$. For larger codes in this family the stabilizers do not get bigger than weight 24. It is not clear to what extent these results are representative of larger code sizes, but we can already see from Figure 2.2 evidence that the weight of the frustrated stabilizer generator plays a more important role than the size of the code itself.

There are two main points to make about these numerics. The first is that the gap of the compass model is decreasing much faster than the gap in the gauge color model. In fact, there is strong evidence [34] that the gap of the compass model tends to zero as the lattice size grows. The second point to make is that the gap always corresponds to frustrating a stabilizer ($t_X \neq 0$.) Moreover, the stabilizer that gives rise to the gap is the one with largest weight. This is a crucial connection to make because the stabilizers of the compass model grow with the linear size of the model while those of the gauge color model do not need to grow beyond a constant bound. This would suggest that if this is the mechanism for gapless behaviour that the gauge color model may be gapped.

The above numerics reach the limit of presently available computational resources. To find eigenvalues for each Hamiltonian block $H_{t_X,0}$ we need to operate on wavefunctions with 2^r real coefficients. The iterative solvers in `SLEPc` need to store at least two, but ideally more, of these wavefunctions. For $r = 32$ this is about 32 Gigabytes for one wavefunction (using double precision coefficients) and so this value for r is roughly the upper limit on these numerical techniques. Without decomposing the gauge color

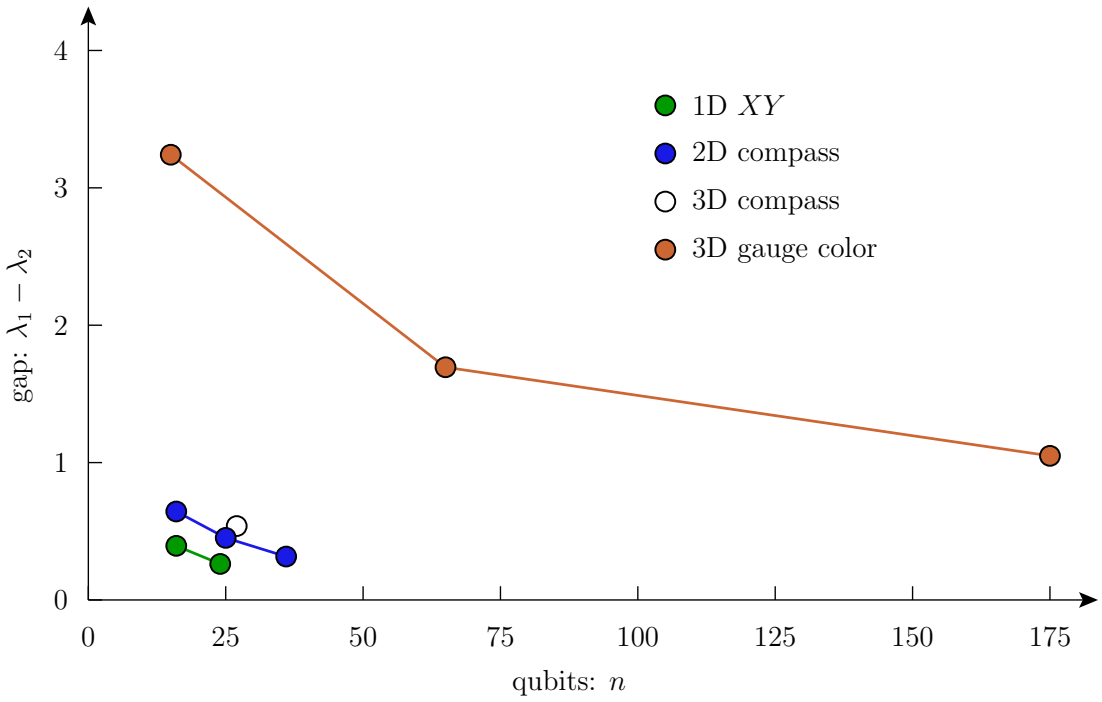


Figure 2.1: The spectral gap of four different gauge code Hamiltonians, versus the number of qubits n. The gap is defined as the difference between the ground eigenvalue and the first excited eigenvalue. These results are obtained by exact diagonalization.

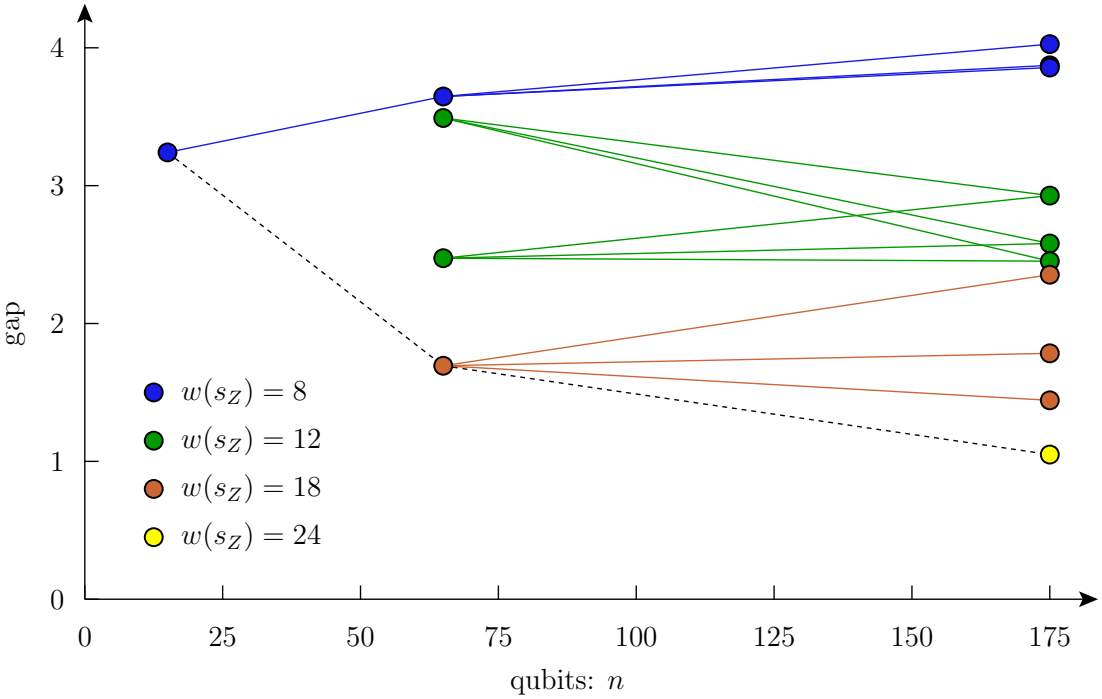


Figure 2.2: Here we show the spectral gap for each Hamiltonian block $H_{t_X,0}$ of the 3D gauge color code models of sizes $n = 15$, $n = 65$ and $n = 175$. This gap is defined as $\lambda_1(H) - \lambda_1(H_{t_X,0})$. Each point is colored according to the weight of the frustrated stabilizer generator.

code into six disjoint ideals it would be impossible to obtain the results for the $n = 175$ code, as this code has $r = 94$.

2.11 Cheeger cuts

In this final section of Chapter 2 we give some heuristic arguments for why the size of the stabilizers is related to the gap of the code.

The Perron-Frobenius structure theory places strong constraints on the first and second eigenvectors of Γ_{t_X} : the first eigenvector has all positive entries, and therefore all vectors orthogonal to the first eigenvector will have both positive and negative entries. In general, the set of edges of Γ_{t_X} where such a vector changes sign we call a Cheeger cut [30, 31]. (We ignore the possibility that this vector may have zero entries.) The Cheeger cut associated to the second eigenvector is particularly important, and we next show an example of how this cut relates to the gap.

2.11.1 The double well model is gapless

We consider a linear graph Hamiltonian with a "double-well" potential. This does not correspond to any gauge code Hamiltonian. The state space will be d dimensional with basis vectors numbered $|1\rangle, ..., |d\rangle$. We take $H = A + U$ with

$$A_{ij} = \begin{cases} 1 & \text{if } |i-j| = 1, \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad U_{ij} = \begin{cases} 2 & \text{if } i = j = 1 \text{ or } i = j = n, \\ 0 & \text{otherwise.} \end{cases}$$

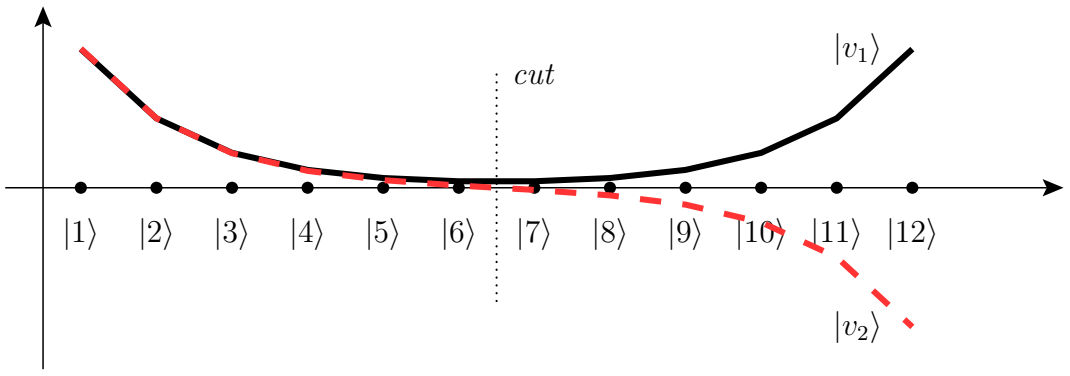
A here is a kind of transition matrix, and U is a diagonal potential energy term.

For $d \gg 1$, the largest eigenvalue is $\lambda_1 \cong \frac{5}{2}$. The corresponding eigenvector $|v_1\rangle$ has all positive components that decay exponentially away from the well sites at $|1\rangle$ and $|d\rangle$:

$$\langle i|v_1\rangle \cong 2^{i-1}\langle 1|v_1\rangle \quad \text{for} \quad i \ll \frac{d}{2}.$$

For the second eigenvalue, λ_2 we also have $\lambda_2 \cong \frac{5}{2}$ and indeed, as d grows the gap $\lambda_1 - \lambda_2 \to 0$ and so this model is gapless.

Here we depict the wavefunctions for the first two eigenvectors for a system with $d = 12$:



The simplest way to show this model is gapless is using a variational argument. Any another vector $|u\rangle$ that is orthogonal to the groundspace vector will have $\langle u|H|u\rangle \leq \lambda_2$. To construct a candidate for $|u\rangle$ partition the basis vectors into two parts:

$$\Gamma = \Gamma_A \cup \Gamma_B$$

and write $|v_1\rangle = |v_A\rangle \oplus |v_B\rangle$ as well as Hamiltonian with this decomposition as

$$H = \begin{pmatrix} H_{AA} & H_{AB} \\ H_{BA} & H_{BB} \end{pmatrix}.$$

Now let

$$|u\rangle = |v_A\rangle \oplus -|v_B\rangle$$

And then

$$\begin{aligned} \lambda_2 \geq \langle u|H|u\rangle &= \langle v_A|H_{AA}|v_A\rangle + \langle v_B|H_{BB}|v_B\rangle - \langle v_B|H_{BA}|v_A\rangle - \langle v_A|H_{AB}|v_B\rangle \\ &= \lambda_1 - 4\langle v_B|H_{BA}|v_A\rangle. \end{aligned}$$

So if we can show that $\langle v_B|H_{BA}|v_A\rangle$ tends to zero we are done. This term involves the dynamical coupling between the groundstate wavefunction along the cut between A and B. To succeed we must find such a cut where the wavefunction is small. In general this appears to be quite difficult, even though in the models we are considering numerics show that not only is the wavefunction small away from potential wells but it is exponentially small.

2.11.2 The cut and symmetry

We now study the cut associated to the second eigenvector of a weakly self-dual gauge Hamiltonian H, and relate this to the stabilizers of the code. The key realization is that Γ_{t_X} is like the double well potential above, but now we have 2^{m_X} such wells, that is, one for every $s_X \in \langle S_X\rangle$. This is clear from examining the basis vectors for Γ_{t_X}. These are

$$|vS_X + uR_X + t_X\rangle, \quad \text{where} \quad v \in \mathcal{F}_{m_X}, u \in \mathcal{F}_r$$

and those that satisfy the most G_Z terms are precisely those with $u = 0$.

We already know this is either the second eigenvector of $H_{0,0}$ or otherwise the first eigenvector of $H_{t_X,0}$ for some $t_X \neq 0$. To relate this to the Perron-Frobenius theory we note the decomposition:

$$\Gamma_{t_X} = \bigoplus_{t_Z \in \langle T_Z\rangle} H_{t_X,t_Z}.$$

This gives the spectral decomposition of each graph Γ_{t_X} in terms of "momenta" t_Z.

We focus on Γ_0. This must contain the second eigenvector of H by weak self-duality of the code. X type stabilizers $s_X \in S_X$ act on the $0, t_Z$ irreps in Γ_0 by ± 1 according to the commutator $[[s_X, t_Z]]$. Suppose the second eigenvector of H lives in H_{0,t_Z} for $t_Z \neq 0$. Let $s_X \in S_X$ with $[[s_X, t_Z]] = -1$. Then we must have an odd number of Cheeger cuts on every Γ_0 path between $|v\rangle$ and $s_X|v\rangle$ for all basis vectors $|v\rangle$, that is, $v \in \langle S_X\rangle \oplus \langle R_X\rangle$.

In a similar vein, if the second eigenvector of H lives in $H_{0,0}$ then we must have an even number of Cheeger cuts on every Γ_0 path between $|v\rangle$ and $s_X|v\rangle$ for all stabilizers $s_X \in S_X$ and basis vectors $|v\rangle$.

In summary, the idea is that large stabilizers lead to widely separated well potentials and hence gapless behaviour, while stabilizers of bounded weight force the cuts to appear close to the wells and hence maintain a gap. Even though numerics show the wavefunction becoming exponentially small away from well potentials, it is also exponentially wide. So making these arguments rigorous appears to be difficult.

The following fact would appear to be true under certain conditions, but is not at all true for example when T is trivial:

Proto-fact: For a sufficiently "well-behaved" weakly self-dual gauge code Hamiltonian H

$$\begin{aligned}\lambda_2(H) &= \min_{t_X \neq 0} \lambda_1(H_{t_X,0}) \\ &= \min_{t_Z \neq 0} \lambda_1(H_{0,t_Z}).\end{aligned}$$

Indeed, contrary to this proto-fact we suspect that $H_{0,0}$ will not be gapped in the generic case. Numerics suggest that there is no lower bound on the gap of randomly constructed stabilizer-less gauge code Hamiltonians. Perhaps double well behaviour can still be imitated even without stabilizers: merely having a large region of almost-stabilizer behaviour (large shallow well) could be enough to send the gap to zero.

There are also results that state that generic local Hamiltonians are gapless [71].

2.11.3 Cheeger inequalities

We saw above how the Cheeger cut gives a variational ansatz for building a second eigenvector to the Hamiltonian and hence an upper bound on the gap.

In this section we show how the Cheeger cut also yields a lower bound on the gap.

In [43], they derive the following Cheeger inequality by considering bi-partitions of the graph. We will do the same, but using matrix block notation.

Let v_2 be a second eigenvector, $Hv_2 = \lambda_2 v_2$ and $||v_2|| = 1$. We bi-partition the space so that v_2 has (vector) blocks:

$$v_2 = \begin{pmatrix} x \\ y \end{pmatrix}$$

with $x \geq 0$ and $y \leq 0$, component-wise. Let the blocks of H under the same partition be:

$$H = \begin{pmatrix} A & C \\ C^\top & B \end{pmatrix}.$$

If we denote $\lambda_1(A)$ as the top eigenvalue of A and $\lambda_1(B)$ as the top eigenvalue of B, then

$$\begin{aligned}\lambda_2 = v_2^\top H v_2 &= x^\top A x + 2x^\top C y + y^\top B y \\ &\leq x^\top A x + y^\top B y \ \leq \ ||x||^2 \lambda_1(A) + ||y||^2 \lambda_1(B) \\ &\leq \min(\lambda_1(A), \lambda_1(B)) \ \leq \ \lambda_1.\end{aligned}$$

Defining the following constant as a maximization over all bi-partitions of H :

$$\nu(H) := \max_{A,B} \ \min(\lambda_1(A), \lambda_1(B))$$

the above calculation shows that

$$\lambda_2 \leq \nu(H) \leq \lambda_1.$$

2.12 Summary

{Needs a concluding section summarising results and discussing open questions and potential research directions. }

In section 1 we apply general principles of group representation theory to block diagonalize Hamiltonians. The terms of the Hamiltonian generate a group (by multiplication), and we also have a .... {fixme}

In summary, we have the complete representation theory for CSS gauge code Hamiltonians.

analyse spectrum of weakly self-dual CSS gauge code Hamiltonians using Perron-Frobenius theory.

It is clear from numerics that nothing as simple as the proto-fact holds. Roughly speaking, larger stabilizers lead to gapless behaviour, but

more models need to be considered, numerics obtained, including codes chosen from random ensembles.

The Cheeger inequalities are a first attempt to relate the gap to the geometric structure of the graph Γ, in particular the size of stabilizers. This analysis is too weak to show any result, and so further work is needed. If this approach is to succeed, a much sharper bound on the shape of the wavefunction is needed.

The perturbation theory for stabilizer Hamiltonians has good results, we would also want something like this for gapped gauge code Hamiltonians, to establish physicality.

We leave as an open problem to extend the representation theory of CSS gauge codes in section 2.4 to arbitrary gauge codes (not necessarily satisfying the CSS condition). We do not need this in the present work.

CHAPTER 3

A Short Guide to Anyons and Modular Functors

To the working physicist, anyon theory is meant to describe certain quasi-particle excitations occurring in two dimensional topologically ordered systems. A typical calculation using this theory will involve operations such as $\otimes$ to combine anyons, F_d^{abc} to re-associate such combinations, and R_c^{ab} to commute or braid these anyons. Although there is a powerful string-diagram notation that greatly assists these manipulations, we still appear to be operating on particles arranged on a one-dimensional line, algebraically ordered from left to right. The obvious question is, where is the other dimension? The topological framework for considering these anyons as truly living in a two dimensional space is known as a modular functor, or topological quantum field theory. In this work we show how the apparently one-dimensional algebraic anyon theory is secretly the theory of anyons living in a fully two-dimensional system. The mathematical literature covering this secret is vast, and we try to distill this down into something more manageable.

3.1 Overview

In this work we describe 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 fusion tensor categories, or more specifically, modular tensor categories. This algebraic language appears to be more commonly used in the physics literature, such as the well cited Appendix E of Kitaev [60]. Such algebraic calculations can be interpreted as manipulations of string diagrams [7], or *skeins*. These strings encode connections in the algebra, for example the Einstein summation convention. But they also faithfully encode the twisting that occurs when anyons are re-ordered or braided around each other. And these kinds of spatial relationships are fruitfully studied using topological methods.

Working 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. 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

surprises of modern mathematics and physics.

Modular functors can be constructed from skein theory. In the physics literature, this appears as skeins growing out of manifolds [76], or as motivated by renormalization group considerations [66]. 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 Refs. [42, 41]. Also, Refs. [12] and [61] 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 [81, 80, 9, 79]. According to [10] section 1.2 and 1.3, there are several open questions involved in rigorously establishing the connection between these different axiomatizations. In particular, what physicists call anyon theory, and mathematicians call a 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.

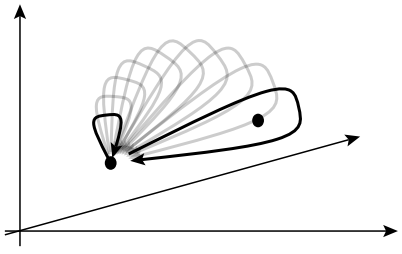
Of the many variants of modular functor found in the literature, one important distinction to be made is the way anyons are labeled. In the mathematical works [80, 9, 79] we see that anyons are allowed to have superpositions of charge states. However, in this work we restrict anyons to have definite charge states, as in [81, 41, 12]. This seems to be motivated physically as such configurations would be more stable. {This justification seems a little vague. Could you make it better justified by making contact with the literature on superselection rules?}

The main goal of the present work is to sketch how a braided fusion tensor category arises from a modular functor. In the mathematical literature, this is covered in [80, 79, 9] but as we just noted they use a different formulation for a modular functor.

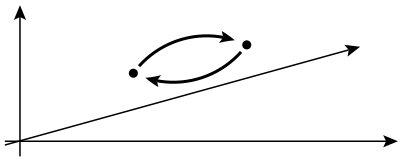
3.2 Topological Exchange Statistics

In this section we begin with the familiar question of particle exchange statistics in three dimensions, whose answer is bosons and fermions. We then show how in restricting the particles to two dimensions many more possibilities arise. Our focus will be on the close connection between the algebraic and the topological viewpoints, aiming to motivate the definition of a modular functor given in the next section.

In three spatial dimensions, the process of winding one particle around another, a *monodromy*, 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:



For identical particles this is a *symmetry* of the system. Continuing with this line of thought leads to consideration of the *symmetric group* on n letters, S_n. This group is generated by the $n-1$ swap operations $s_1, ..., s_{n-1}$ that obey the relations

$$\begin{aligned} s_i^2 &= 1, \\ s_i s_j &= s_j s_i \quad \text{for} \quad |i-j| > 1, \\ s_i s_{i+1} s_i &= s_{i+1} s_i s_{i+1} \quad \text{for} \quad 1 \leq i \leq n-2. \end{aligned}$$

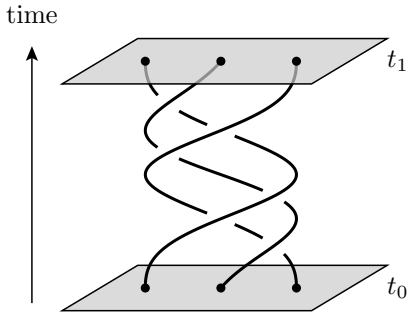
Writing the Hilbert space of the system as V we would then expect S_n to act on this space via unitary transformations $U(V)$:

$$\mathscr{H} : S_n \to U(V).$$

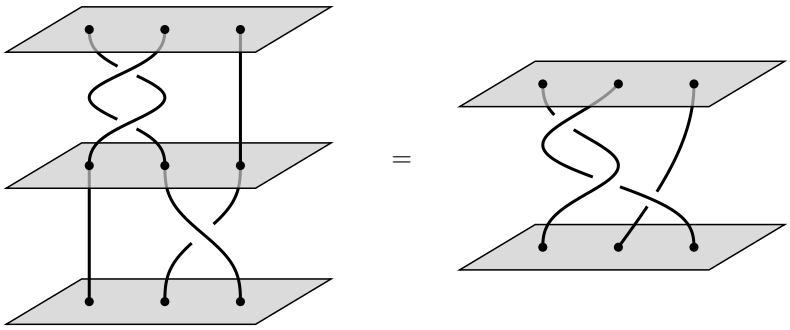
This is the first example of the kind of functor we will be talking about. In this case it is a group representation; $\mathscr{H}$ is a homomorphism between two groups.

When the above monodromy is constrained to two dimensions we can no longer deform this process to the identity:

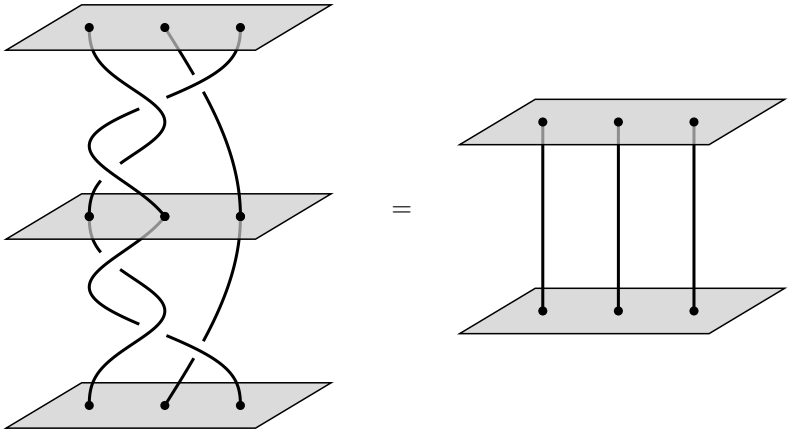
and so in two dimensions we cannot expect a swap to square to the identity. To see this more clearly, we must examine the entire (2+1)-dimensional *world-lines* of these particles. For an example, here we show the world-lines of three particles undergoing an exchange and then returning to their original positions:



Note that if we allow the particles to move in one extra dimension then we can untangle these braided world lines. The question is now, what is the group that acts on the state space 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 shows that these processes do form a group, known as the *braid group*. For n particle world-lines we denote this group as B_n. For identical particles this group acts as symmetries of the state space:

$$\mathscr{H} : B_n \to U(V).$$

What we have given is a topological description of the group B_n. More formally, we can describe these braid world lines as paths in the *configuration space* of n points. This space is defined as the product of a two dimensional space M for each point, minus the subspace where points overlap:

$$\mathcal{C}_n = \big(\prod_1^n M\big) - \Delta, \quad \Delta = \{(x_1, ..., x_n) | x_i = x_j \ \text{ for some } \ i \neq j\}.$$

Because we are considering identical particles (so far) we use the *unlabelled configuration space* $\mathcal{UC}_n$ which is the quotient of $\mathcal{C}_n$ by the natural action of the permutation group S_n :

$$\mathcal{UC}_n = \mathcal{C}_n / S_n.$$

The *geometric braid group* as we have so far informally described it can now be rigorously defined as the fundamental group:

$$B_n = \pi_1(\mathcal{UC}_n, x)$$

where x is some reference configuration in $\mathcal{UC}_n$. See Ref. [46] for further discussion.

This group also has a purely algebraic description via generators and relations, as was shown by Artin in 1947, [4, 13]. In this description, B_n is generated by $n-1$

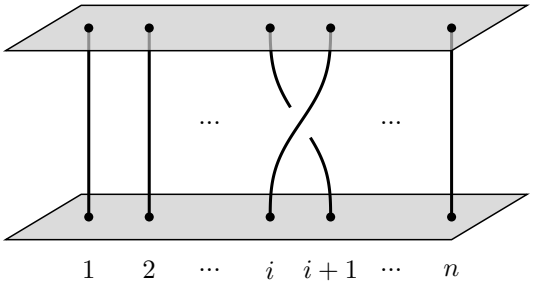
elements $\sigma_1, .., \sigma_{n-1}$ that satisfy the following relations:

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

These relations are the same as for S_n above, except we do not require $\sigma_i^2 = 1$.

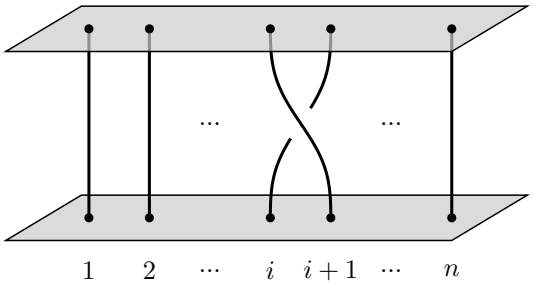
It is easy to see that the geometric braid group satisfies these relations. Here we show the braids corresponding to the generators σ_i :

$\sigma_i =$



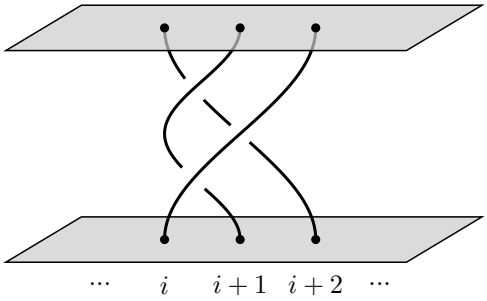
with inverses:

$\sigma_i^{-1} =$

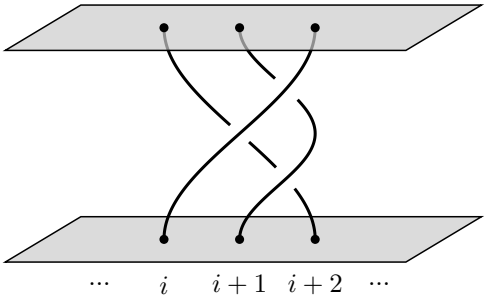


Note that $\sigma_i\sigma_j = \sigma_j\sigma_i \quad \text{for} \quad |i-j| > 1$ because the two braids are operating on disjoint world-lines. The second relation is also easy to see:

$\sigma_i\sigma_{i+1}\sigma_i =$



$=$



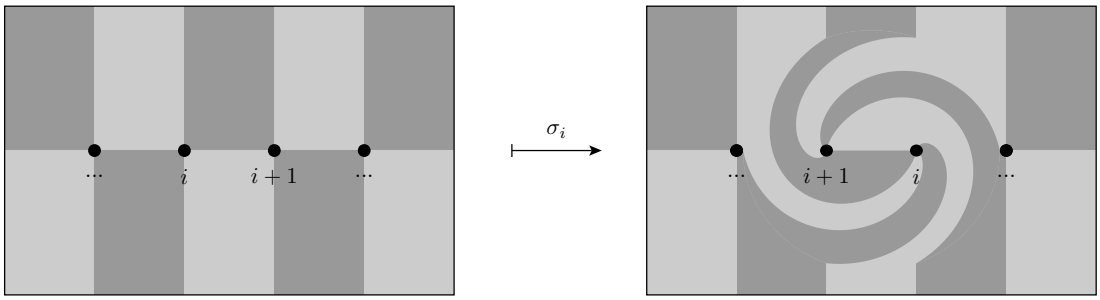
$= \sigma_{i+1}\sigma_i\sigma_{i+1}$

There is another important geometric representation of the braid group which is purely two-dimensional. We pick an n-point finite subset $Q_n \subset M$ and consider diffeomorphisms $f : M \to M$ that map Q_n to itself. The *mapping class group* of M relative

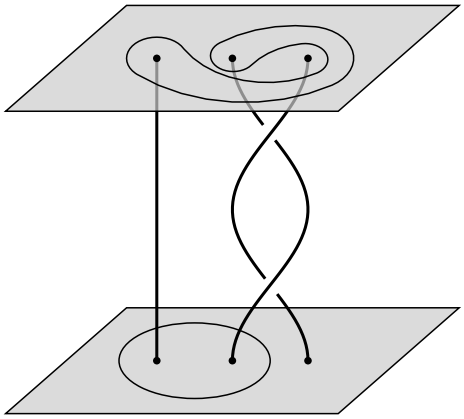
to Q_n is the set of all such diffeomorphisms up to an equivalence relation $\sim_{\text{iso}}$:

$$MCG(M, Q_n) = \{f : M \to M \text{ such that } f(Q_n) = Q_n\}/\sim_{\text{iso}}$$

The equivalence relation $\sim_{\text{iso}}$ is called *isotopy* which allows for any continuous deformation of $f : M \to M$ that fixes each point in Q_n. Each generator σ_i of the braid group is found in $MCG(M, Q_n)$ as a *half-twist* that swaps two points $i, i+1 \in Q_n$:



In order to show the action of this half-twist we have decorated the manifold with a a checkered pattern, but there is a more important object that lives on the manifold itself. An *observable* is a simple closed curve in M that does not intersect Q_n. Such a closed curve is called an observable because these will be associated to measurements of the total anyonic charge on the interior of the curve. The importance of understanding the braid group as identical to the mapping class group is now manifest: whereas geometric braids act on states as in a Schrödinger picture, elements of the mapping class group act on the observables as in a Heisenberg picture.



This diagram should give the reader some idea as to why these two definitions of the braid group are equivalent, but the actual proof of this is somewhat involved. We cite Ref. [58] for an excellent contemporary account that fills in these gaps.

The definition given above for the geometric braid group and the mapping class group make sense for any two dimensional manifold M but for concreteness we consider M to be a flat disc. The corresponding algebraic definition of the braid group will in general be altered depending on the underlying manifold M.

So far we have been studying the exchange statistics for n identical particles. Without this restriction, one needs to constrain the allowed exchange processes so as to preserve particle type. For example, if all particles are different we would use the *pure braid group* PB_n. The geometric description of this group is as the fundamental group of the labelled configuration space

$$PB_n = \pi_1(\mathcal{C}_n, x).$$

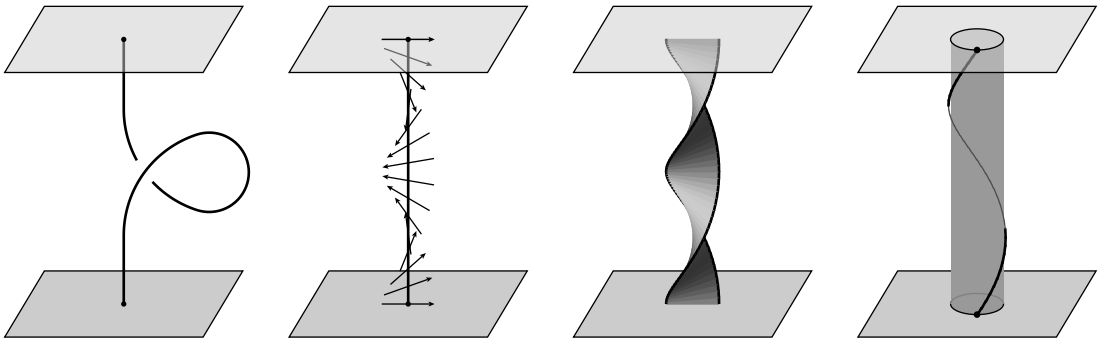
Loops in $\mathcal{C}_n$ correspond to braids where each world-line returns to the point it started from. The algebraic description of this group is somewhat complicated and we omit this. In terms of the mapping class group, we can describe PB_n as the *pure mapping class group*:

$$PMCG(M, Q_n) = \{f : M \to M \text{ such that } f(x) = x \text{ for } x \in Q_n\}/\sim_{\text{iso}} .$$

One further complication arises when particles have a rotational degree of freedom: ie., they can be rotated by 2π in-place and this effects the state of the system. To capture this action, we use the *framed braid group* FB_n. This group can be presented algebraically using the same generators and relations as for the braid group B_n, along with "twist" generators θ_i for $i = 1, .., n$. These must satisfy the further relations

$$\begin{aligned} \theta_i\theta_j &= \theta_j\theta_i \\ \theta_i\sigma_j &= \sigma_j\theta_i \text{ if } i < j \text{ or } i \geq j+2 \\ \theta_{i+1}\sigma_i &= \sigma_i\theta_i \\ \theta_i\sigma_i &= \sigma_i\theta_{i+1}. \end{aligned}$$

Here we show four geometric approaches to representing a twist. Any person that has struggled to untangle their headphone cable will immediately see what is going on here.



On the left we have a loop; it is not a braid because it travels backwards in time. If we pull on this loop to make it straight, we introduce a twist. This is shown in the next figure, where we show a *framing* which is a non-degenerate vector field along the world-lines of a braid. The initial and final vectors in the vector field must be the same. By non-degenerate we mean that the vector field is everywhere non-zero and non-tangent to the world-line. In the next figure we show a *ribbon*: instead of point particles we have short one-dimensional curves in M. On the right the particle is represented as a boundary component (a hole) of M with a distinguished point. In this picture the world-line looks like a tube.

All these representations of twists carry essentially the same information. In the sequel we will stick to thinking of particles as boundary components because this fits well with the way we are formulating observables as simple closed curves in M.

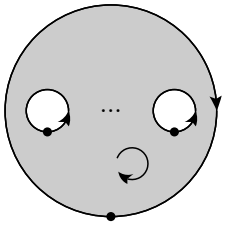
At this point in the narrative we are close to our next destination. All of these considerations, of framed or unframed, labelled or not, with possibly different underlying manifolds, together with observables, is mean to be captured by the formalism of a modular functor which we turn to next.

3.3 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 connected 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 equipped 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. 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. 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. The weaker notion of homotopy of maps will not be used here, but for the maps we use it turns out that homotopy is equivalent to isotopy. Furthermore, we can weaken the requirement that maps be differentiable, because every continuous map $f : M \to N$ is (continuously) isotopic to a differentiable map [39].

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, $\mathscr{H}(\phi) \cong \mathbb{C}$. 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 image of these holes in N we call a *seam*. 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*:

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

and this isomorphism depends only on the isotopy class of the seam in N.

Unitarity axiom. Reversing the orientation of the surface M to form $\overline{M}$ we get the dual of the fusion space. That is, we have the following isomorphism:

$$\mathscr{H}(\overline{M}) \xrightarrow{\cong} \mathscr{H}(M)^*.$$

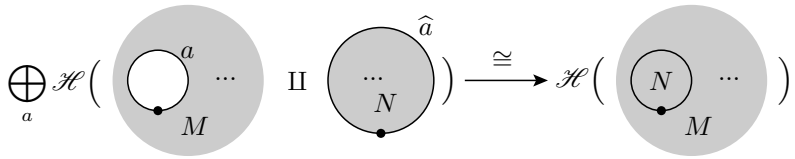
Compatibility axioms. Loosely put, we require that the above operations play nicely together, and commute with maps of surfaces. For example, a sequence of gluing operations applied to a surface can be performed regardless of the order (gluing is associative) and we require the various gluing maps for these operations to similarly agree. For another example, we require $\mathscr{H}$ to respect that gluing commutes with disjoint union.

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.

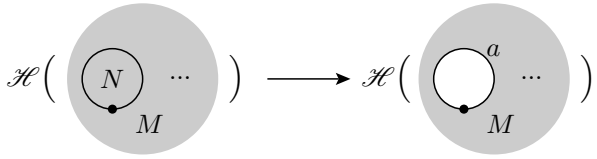
Consequences of axioms. A common operation is to glue two separate surfaces. We can do this by first taking disjoint union (tensoring the fusion spaces) and then gluing. Here we show this process applied to two surfaces M and N.



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.

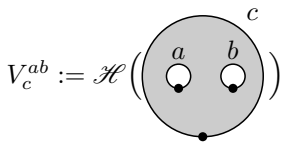
We note two other consequences of the axioms. A hole of M labeled with $\mathbb{I}$ can be replaced with a disk (by gluing) and this does not change the fusion space of M. That is, a hole that carries no charge can be "filled-in". And, the dimensionality of the fusion space of a torus is the cardinality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

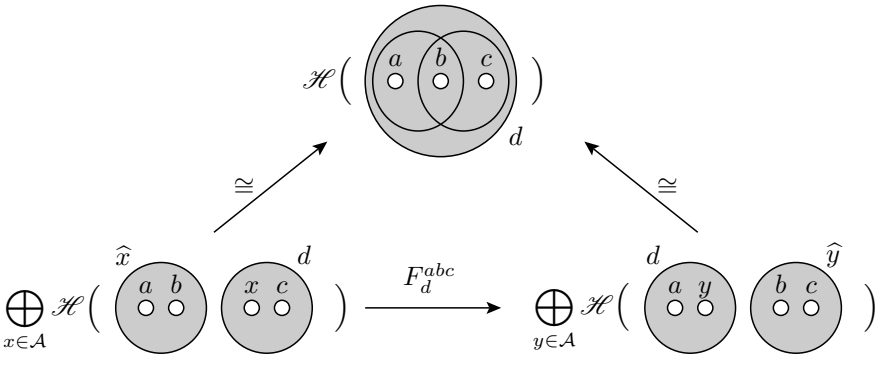


In this case, we define the operation of *fusion* to replace the interior of an observable by a single hole. This is an operation on the manifold itself, and we will only do this when the interior piece is a disc with zero or more holes.

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:

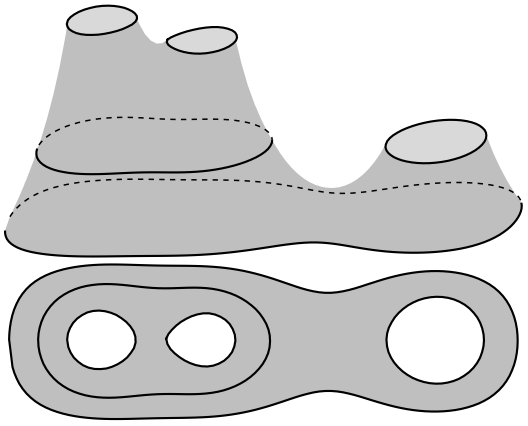


Here we have a surface M_d^{abc} with four labeled boundary components, as well as two separate ways of gluing pairs-of-pants to get M_d^{abc}. We can also write this out in terms of the fusion spaces of pair-of-pants:

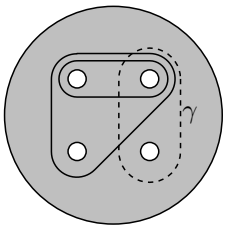
$$F_d^{abc} : \bigoplus_{x\in\mathcal{A}} V_{\widehat{x}}^{ab} \otimes V_d^{xc} \to \bigoplus_{y\in\mathcal{A}} V_d^{ay} \otimes V_{\widehat{y}}^{bc}.$$

By using gluing, and summing over charge sectors, we can extend this operation to apply where any of the boundary components a, b, c or d are merely seams in a larger manifold.

POP decomposition. Given a manifold M which is a disc with two or more holes, we show how to present M as the gluing of various pair-of-pants (POP). Such an arrangement will be termed a *POP decomposition.* (We refer to Ref. [54] for more details on this construction, and Ref. [46] 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.

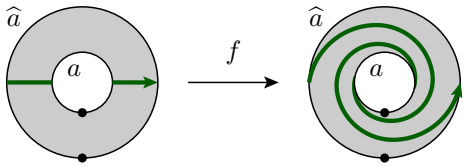


Clearly such a POP decomposition is not unique, and the goal here is to understand how to switch between decompositions, and in particular, given an observable γ, and a given POP decomposition, find a sequence of "moves" such that γ is in the resulting POP decomposition:



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 Ref. [41]. In this work we use a simpler method, which is essentially the same as skein theory. This is the refactoring theorem that we describe below.

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, {Is this the origin of the superselection rule in the theory?} 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:

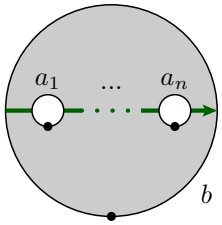
$$\bigoplus_{a\in\mathcal{A}} \theta_a \mathscr{H}(N_{a,\widehat{a}}).$$

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 $M_b^{a_1 \dots a_n}$.

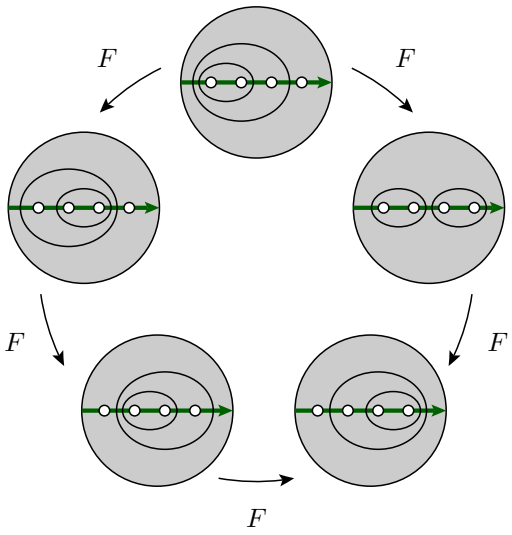
For concreteness we define this surface using the following expression for a closed disc less n open discs:

$$\left\{(x,y) \in \mathbb{R}^2 \text{ st. } \left|(x,y) - \left(\frac{1}{2}, 0\right)\right| \leq \frac{1}{2}\right\} - \left\{(x,y) \in \mathbb{R}^2 \text{ st. } \left|(x,y) - \left(\frac{2i-1}{2n}, 0\right)\right| < \frac{1}{4n}\right\}_{i=1,\dots n}.$$

We then label the interior holes $a_1, ..., a_n$ in order of increasing x coordinate, and the exterior hole is labeled b. The base points are placed in the direction of negative y coordinate. As a notational convenience we highlight the *equator* of the surface which is the intersection of the x 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 an immediate consequence of the way we have defined F-moves. This is known as the pentagon equation, which we depict as the following commutative diagram:



We are now starting to confuse the notation for the topological space M and the fusion space $\mathscr{H}(M)$. Also, each of these F-moves is referring to an isomorphism that is block decomposed according to the charge sectors of the indicated observables.

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 $V_{b'}^{a_1' a_2'}$.

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 $M_b^{a_1\cdots a_n}$.

We next show how to glue the exterior hole of a standard surface M to an interior hole of another standard surface N. To do this within $\mathbb{R}^2$ we rescale and translate the two surfaces so that the exterior hole of M coincides with the interior hole of N. {Illustrate this with a figure?} At this point the union is not in general going to produce another standard surface, and so we remedy this by applying any isotopy within $\mathbb{R}^2$ that fixes the x-axis while moving the surface to a standard surface.

Curve diagram. We next study maps from standard surfaces to arbitrary surfaces. The reader should think of this as akin to choosing a basis for a vector space. 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. (This reconstruction works because a simple closed curve on a sphere cuts the sphere into two discs.) 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.

The operation of gluing of surfaces can be extended to gluing of curve diagrams as long as we are careful with the way we identify along the seam: the identification map

needs to respect the equator of the curve diagram. (Keep in mind that a curve diagram is really a map of surfaces, and so gluing two such maps involves two separate gluing operations.)

We note in passing two connections to the mathematical literature. Such curve diagrams have been used in the study of braid groups [32], 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 [55], the main difference being that planar algebras allow for not just two but any even number of curve intersections at each hole.

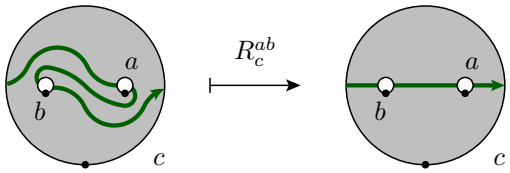
Z-move. We let z be a diffeomorphism of standard surfaces $z : M_b^{a_1 \dots a_n} \to M_{a_1}^{a_2 \dots a_n b}$ that preserves the equator. (Considering the standard surface as a sphere with holes placed uniformly around a great circle, z is seen to be a "rotation".) This acts by pre-composition to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

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

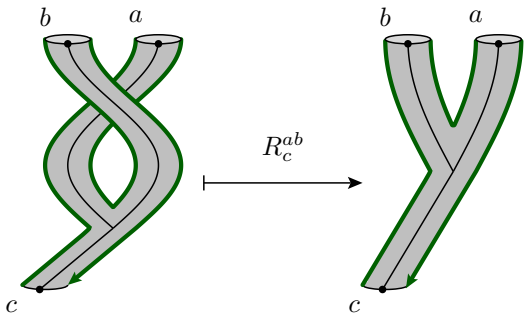
This map works by taking a curve diagram $f : M_c^{ab} \to N$ to the composition $f\sigma$ where $\sigma : M_c^{ba} \to M_c^{ab}$ is a counterclockwise "half-twist" map that exchanges the a and b holes. Here we show the action of R_c^{ab} on one particular curve diagram:



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.

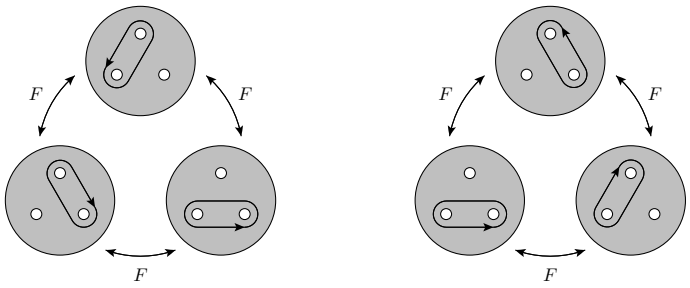
As we extended Dehn twists under gluing, and F-moves under gluing, we also do this for R-moves.

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

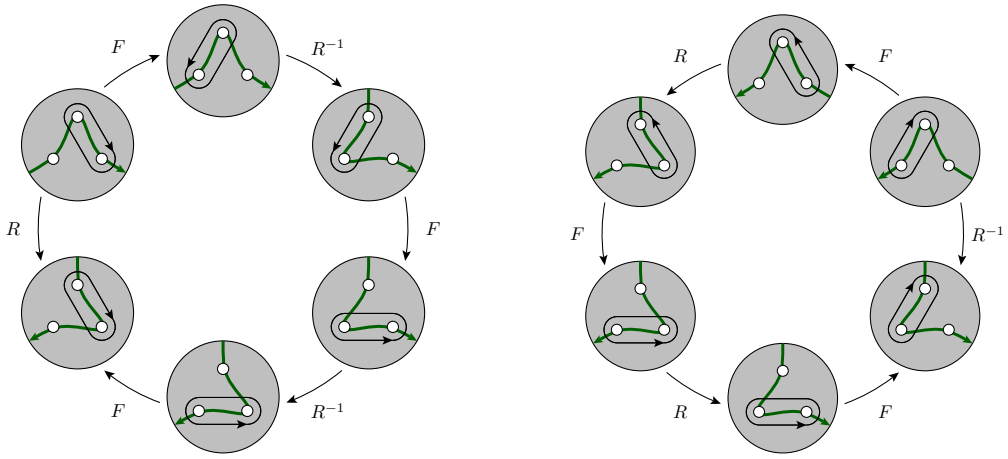


This figure is intended to be topologically the same as the previous flat figure, with the addition of a third dimension, and the c boundary has been shrunk. Also note thin black lines connecting the base points. The black lines do not add any extra structure, they can be seen as a part of the hexagon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

The reason why we mention the pentagon and hexagon equations is that these become important in an algebraic description of the theory. Because we have defined everything in terms of a modular functor we get these equations "for free".

3.4 Refactoring theorem

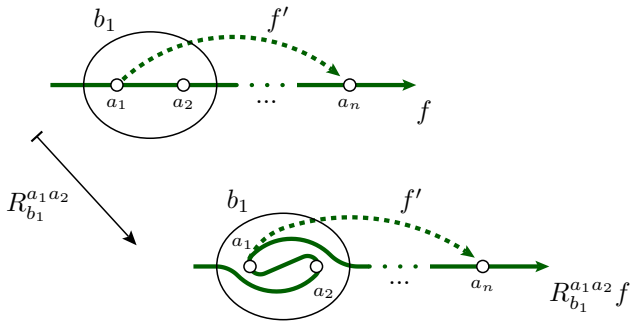
In this section we specialize to considering planar surfaces only. The theorem we are building towards shows that the R-moves act transitively on curve diagrams. By transitive we mean that given two curve diagrams f and f' on N we can find a sequence of R-moves that transform f to f'. The key idea is 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'.

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 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}.

We continue in this way moving backwards (from head to tail) sequentially applying this procedure.

3.5 Summary

The formalism of modular functors is needed to justify and explain the simulation used in the next chapter. Anyon simulations in the physics literature are typically conceived of as one-dimensional arrays of anyons, but for our simulation we really do need to consider these anyons as living in a two-dimensional system, and also to account for observables that wind arbitrarily around the anyons.

In the next chapter we will use as underlying manifold for our system a torus. As we have seen the modular functor axioms force the dimensionality of the fusion space of a torus to be equal to the number of charges. This can be seen by gluing one end of a cylinder (annulus) to the other, and applying the gluing axiom. This fusion space contains the protected quantum information.

The noise processes will be modelled by replacing small patches of the torus with pair-of-pants that have vacuum total charge. Physically accessible observables will be associated with fixed "tiles" on the torus, whose observation outcomes will serve to diagnose the noise process.

In order to calculate the probabilities for these observation outcomes we will need to be able to move curve diagrams around until the observables of the curve diagram (these are the standard POP decompositions) contain the desired observable. That we can always do this (at least for planar surfaces) was shown by the refactoring theorem. This will be re-formalized in a combinatorial way below as the *paperclip algorithm*.

CHAPTER 4

Error Correction in a Non-Abelian Topologically Ordered System

The work in this chapter is based on the collaboration [28].

In this chapter we apply the methods of Chapter 3 to develop a simulation of noise processes in a two dimensional system with non-abelian anyons.

Topologically ordered quantum systems in two dimensions show great promise for long-term storage and processing of quantum information [62, 33, 72]. The topological features of such systems are insensitive to local perturbations [21, 22, 69], and they have quasiparticle excitations exhibiting anyonic statistics [85]. These systems can be used as quantum memories [62, 33] or to perform universal topological quantum computation [42, 72].

Quantum error correction is vital to harnessing the computational power of topologically ordered systems. When coupled to a heat bath at any non-zero temperature, thermal fluctuations will create spurious anyons that diffuse and quickly corrupt the stored quantum information [74]. Thus, the passive protection provided by the mass gap at low temperature must be augmented by an *active* decoding procedure.

In order to efficiently classically simulate an error-correction protocol for a topologically ordered quantum memory, it is necessary to simulate the physical noise processes, the decoding algorithm, and the physical recovery operations. Decoding algorithms are typically designed to run efficiently on a classical computer, but there is generally no guarantee that the noise and recovery processes should be classically simulable. Because of this, almost all of the sizable research effort on active quantum error correction for topological systems has focused on the case of abelian anyons [33, 36, 37, 83, 82, 38, 19, 16, 89, 3, 84, 53, 23, 86, 40, 2], which can be efficiently simulated due to the fact that they cannot be used for quantum computation.

Recent investigations have begun to explore quantum error correction for non-abelian anyon models [24, 87, 51, 88, 52]. Nonabelian anyon models are especially interesting because braiding and fusion of these anyons in general allows for the implementation of universal quantum computation. However, the initial studies of error-correction in non-abelian anyon systems have focused on specific models, such as the Ising anyons [24, 52] and the so-called Φ-Λ model [87, 51] that, while non-abelian, are not universal for quantum computation. The general dynamics of these particular anyon models is known to be efficiently classically simulable, a fact that was exploited to enable efficient simulation of error correction in these systems. When considering more general anyon models, their ability to perform universal quantum computation

would seem a significant barrier to their simulation on a classical computer. While simulation of general dynamics does indeed seem intractable, we argue that the kinds of processes that are typical of thermal noise are sufficiently structured to allow for their classical simulation in the regimes where we expect successful error correction to be possible. This insight allows us to simulate the noise and recovery processes for a quantum code based on a universal anyon model.

Concretely, we consider quantum error correction in a two-dimensional system with Fibonacci anyons, a class of non-abelian anyons that are universal for quantum computation [42, 72]. Fibonacci anyons are experimentally motivated as the expected excitations of the $\nu = \frac{12}{5}$ fractional quantum Hall states [78], and can be realized in several spin models [66, 18, 57, 73] and composite heterostructures [70]. Any of these physical systems could be used to perform universal topological quantum computation, and can be modelled by our simulations. Natural sources of noise from thermal fluctuations or external perturbations will be suppressed by the energy gap but must still be corrected to allow for scalable computation.

We use a flexible phenomenological model of dynamics and thermal noise to describe a system with Fibonacci anyon excitations. Within this model, we apply existing general topological error-correction protocols, and simulate the successful preservation of quantum information encoded in topological degrees of freedom. Topological quantum computation protocols using non-abelian anyons typically implicitly assume the existence of an error-correction protocol to correct for diffusion or unwanted creation of anyons. The ability to simulate the details of how and when these techniques succeed on finite system sizes has not previously been available, and so our results are the first explicit demonstration that such a scheme will be successful when applied to a universal topological quantum computer.

4.1 Fibonacci anyons

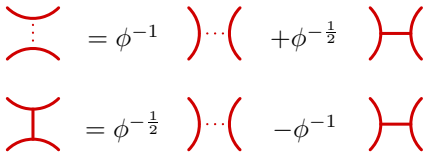
The defining difference between abelian and non-abelian anyon theories is that in an abelian theory particle content alone uniquely determines the outcome of joint charge measurements. In contrast, outcomes for non-abelian charge measurements depend on the history of the particles as well as their type. We consider a system supporting non-abelian Fibonacci anyon excitations, denoted by τ. Two such anyons can have total charge that is either τ or $\mathbb{I}$ (vacuum), or any superposition of these, and so the fusion space in this case is 2-dimensional.

For Fibonacci anyons, the non-trivial R and F moves are

$$R_{\mathbb{I}}^{\tau\tau} = \mathrm{e}^{\frac{-4\pi i}{5}} \quad , \quad R_{\tau}^{\tau\tau} = \mathrm{e}^{\frac{3\pi i}{5}} \quad , \quad F_{\tau}^{\tau\tau\tau} = \begin{pmatrix} \phi^{-1} & \phi^{-\frac{1}{2}} \\ \phi^{-\frac{1}{2}} & -\phi^{-1} \end{pmatrix} ,$$

where the matrix is given in a basis labelled $(\mathbb{I}, \tau)$ and $\phi = \frac{1+\sqrt{5}}{2}$ is the golden ratio.

In terms of skeins we write the F-moves as:



The solid lines represent Fibonacci world-lines. The dotted lines represent vacuum

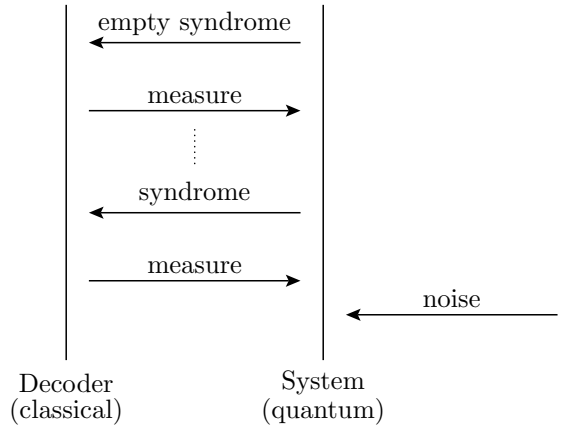


Figure 4.1: The simulation of a single round of error correction involves two components: the classical decoder, and the quantum system. After the noise process has been applied to the system, the simulation proceeds as a dialogue between the decoder and the system. Decoding is successful when all charges have been eliminated, resulting in an empty syndrome.

charges, and we are free to include these lines or not. We leave these anyon paths as undirected because Fibonacci anyons are self-inverse. The non-trivial R-moves are:

For more details of the Fibonacci anyon theory, see e.g. Ref. [72] and references therein.

4.2 Physical model

We consider encoding quantum information in the fusion space of a torus. As we saw in the previous chapter, the dimension of this space is equal to the number of charges (the cardinality of $\mathcal{A}$). For Fibonacci anyons, we have two charges, $\mathbb{I}$ and τ, so this gives a fusion space equal to one qubit. In this work we do not consider the logical operators that act on the encoded state, but merely note that it is sufficient to protect the state if all operations remain local. [1]

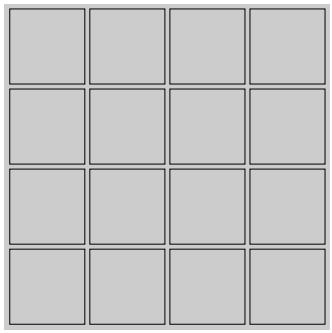
We endow the torus with an $L \times L$ square lattice of observables [2]:

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

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

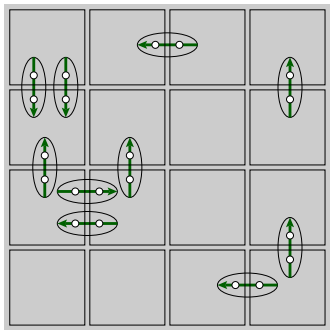
[1] For an example of a *non-local* operation we create a pair of charges and take one of the charges around a non-contractible loop of the torus.

[2] We defined the observables associated to a modular functor in section 3.3.



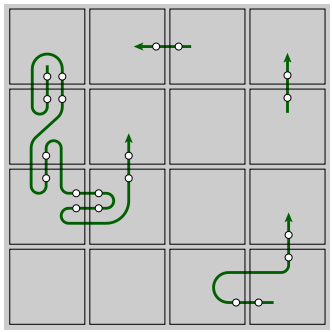
We also use these observables to construct an idealized Hamiltonian of the form $H = -\sum_{\gamma\in\Lambda} |\mathbb{I}\rangle\langle\mathbb{I}|_\gamma$, where $|\mathbb{I}\rangle\langle\mathbb{I}|_\gamma$ is the projector to vacuum charge at tile γ. Therefore, the ground space of the model has vacuum total charge on each tile. Typical thermal noise processes would act to create charges locally on the manifold. This has the effect of populating the manifold with a randomly distributed set of pair creation processes, whose size is much smaller than the resolution of the lattice. In the case of abelian anyons, this agrees with prior work (such as Ref. [33]). For non-abelian anyons we could also consider other noise processes such as braiding or hopping of anyons, but for simplicity we don't consider this here. [3]

We model this noise by randomly replacing small patches of the surface with pair-of-pants:



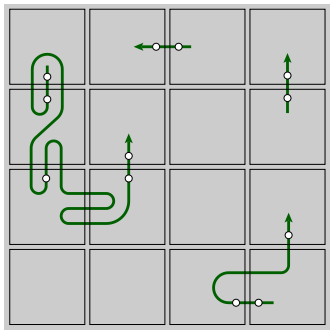
Each such pair will have vacuum total charge and so the observables γ_{ij} will only see pairs that intersect, ie. we need only consider distributing these pairs transversally along edges of the tiles.

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:

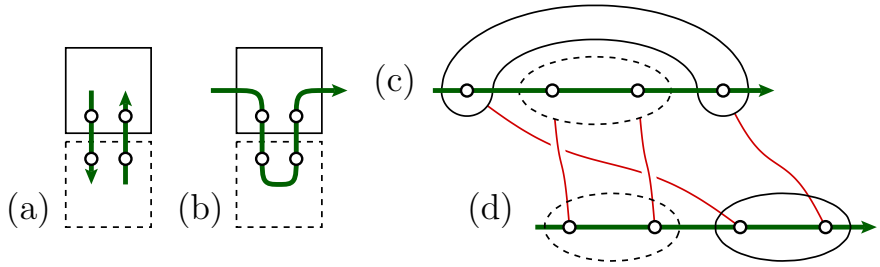


[3] It was seen in Ref. [24] that the pair-creation-only setting was sufficient to capture the qualitative features of an error-correction simulation for the Ising anyons and we have no reason to expect that this would change when considering the Fibonacci anyon model.

Working in the basis picked out by the resulting curve diagrams, we can calculate measurement probabilities for each tile. The measurement outcomes are then randomly sampled, and the results recorded on the original curve:



As another example of the kind of calculation that needs to be performed, here we zoom-in and show an example involving just two tiles (inside a larger tiling):



In (a) we have two pair-creation events, each crossing the boundary of a tile. {The problem we have is...} One observable (tile) is solid black, and the other observable is dashed. In (b) we join the participating curve diagrams arbitrarily into a single curve diagram. The curve diagram (c) is the same as (b), we just pulled the curve straight (an isotopy). Now we braid anyons around each other until all charges within a tile are neighbors on the curve, as in (d). The red lines then correspond to the worldlines for these braids. The state (d) is equivalent to the original state (a), but now the observables we need (the solid and dashed black lines) are at most a few $F-$moves away.

If the reader finds this confusing, that's because it *is* confusing. This is why so much care and detail was taken in chapter 3. The systematic implementation of these calculations relies on all the machinery developed in chapter 3, in particular the refactoring theorem of section 3.4. The data structure we will use is called a *combinatorial curve diagram* and is described in section 4.4.1. The algorithm that operates on this data is called the *paper-clip algorithm* and will be described in section 4.4.2. This data structure and algorithm is a combinatorial version of the theory of modular functors suitable for implementation on a classical computer.

We consider this treatment of anyon dynamics to be a phenomenological model in that it neglects any microscopic details of the system. This is consistent with the principles of topologically ordered systems and anyonic physics, where the key universal features describing the anyon model correspond to large length-scale physics, while the microscopic physics plays a less important (and non-universal) role. Note also that our analysis applies equally well to either a continuum setting, or to discrete lattice models supporting anyonic excitations.

In order to perform a logical error on our code, a noise process must have support on a a region of the manifold that is non-local, (cannot be contracted). These correspond

```
 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
```

Figure 4.2: Pseudo-code listing for the (classical) decoding algorithm.

to processes in which anyonic charge is transported around a non-trivial loop before annihilating to vacuum. We do not consider these processes as we regard the decoding to have failed if any such non-local process would occur.

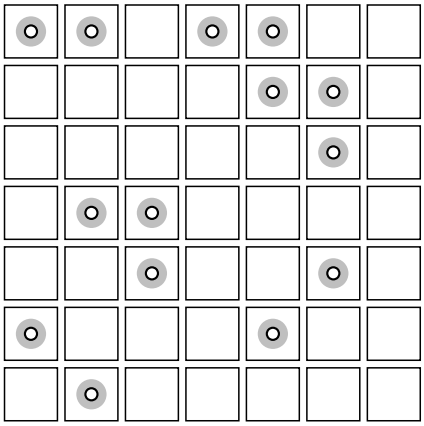
4.3 Decoding algorithm

After the noise process is applied to the system, the error correction proceeds as a dialogue between the decoder and the system. In this section we describe the decoder side of this dialogue, the quantum side is much more difficult and is discussed in the following sections. 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.) In Figure 4.1 we show this in a process diagram, with time running up the page. Note that unlike the case of abelian anyons, the decoder cannot determine the charge of each cluster given only the syndrome information (as in Ref. [19]), and so must repeatedly physically query the system to measure these charges.

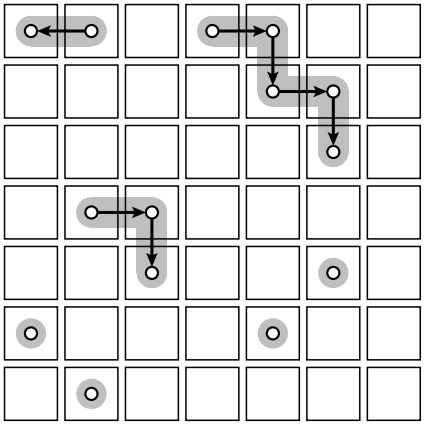
In Figure 4.2 we show a pseudo-code listing for the decoder algorithm, and we explain each step via an example below. This decoder is based on a hierarchical clustering

algorithm [49, 88], and follows a similar strategy to the hard-decision renormalization group decoder [19].

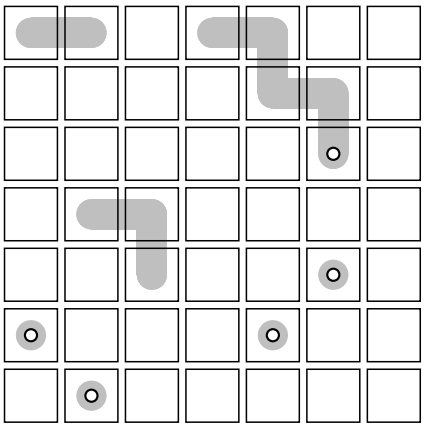
First, we show the result of the initial call to `get_syndrome()`, on line 2. The locations of anyon charges are indicated by the thick black circles. For each of these charges we build a `Cluster`, on line 5. Each cluster is shown as a gray shaded area in the following diagrams.



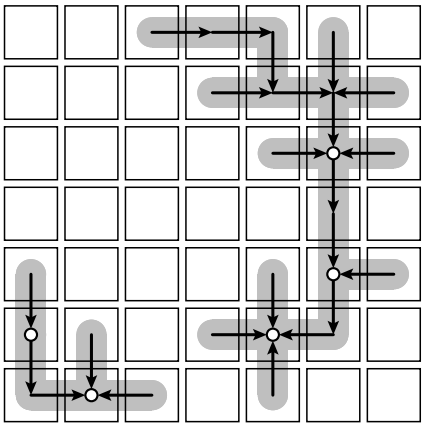
The next step is the call to `join(clusters, 1)`, on line 8, which joins clusters that are separated by at most one lattice spacing. We now have seven clusters:



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). For simplicity we have neglected the boundary of the lattice in this example.

4.4 Simulation of the quantum system

It is known that the process of braiding and fusing Fibonacci anyons is sufficient to implement universal quantum computing, so at first sight it seems foolish to try to simulate this using classical computers.

However, it turns out to indeed be possible, up to some reasonably large system sizes. The reasons why are outlined in a heuristic manner as follows. The decoding threshold (see Figure 4.3) occurs far below the bond percolation threshold. Below this percolation threshold, error processes decompose into separate components of average size $O(\log(L))$ and variance $O(1)$ [11]. Each such component can be simulated separately, ie. each component corresponds to a tensor factor of the Hilbert space of the system (this is the monoidal axiom, Section 3.3). Because the average size of such a component is $O(\log(L))$ the corresponding fusion space will have dimension $O(\text{poly}(L))$.

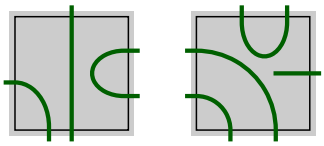
Working against this system separability is the action of the decoder, which will tend to join (fuse) nearby anyons and thereby also at times connect separate components. All is not lost, however, as this act of fusing also reduces the number of anyons needed to be simulated.

At some point, with high enough error rate, the combined action of noise plus decoder will become exponentially difficult to simulate. Nevertheless, we are able to simulate error-correction in the regime around the error-correction threshold for linear lattice sizes up to $L = 128$. See Figure 4.3 below.

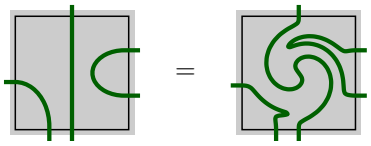
We now turn to a description of the data-structures and algorithm used to simulate the quantum system. This relies heavily on the theory developed in Chapter 3.

4.4.1 Combinatorial curve diagrams

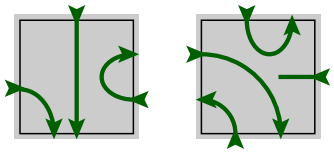
The basic data structure involved in the simulation of the quantum system we term a *combinatorial curve diagram.* This is essentially a combinatorial formulation of the theory of modular functors presented in Chapter 3. The sole purpose of this formulation is to be able to simulate fusion outcomes for each of the observables in the lattice $\Lambda := \{\gamma_{ij}\}_{i,j=1,\ldots,L}$. For each tile in this 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 Ref. [1] to describe elements of a Temperley-Leib algebra, but with some extra decorations. Firstly, we will require each curve to intersect the edges of tiles transversally, and in particular a curve will not touch a tile corner. 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.



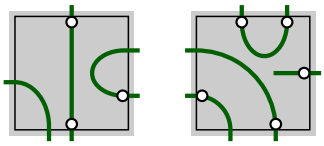
We will also need to record various other attributes of these curves, and to do this we make this notation more elaborate in the paragraphs **(I)**, **(II)** and **(III)** below. Each symbol in the word describes an intersection of the curve with the tile boundary, and so as we decorate these symbols 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 $\langle_1\rangle_1 *_2 \langle_3\langle_3\rangle_4\rangle_4$.

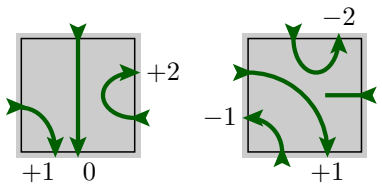
(III) Finally, we will also decorate these symbols with anyons. This will be an index to a leaf of a (sum of) fusion tree(s). This means that anyons only reside on the curve close to the tile boundary, and so we cannot have more than two anyons for each piece-of-curve. The number of such pieces is arbitrary, 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 counter-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 $\sum n_i = m$.

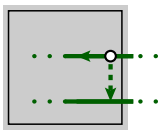
For each piece-of-curve, apart from a head or tail, there is an associated number we call the *turn number*. This counts the number of "right-hand turns" the piece-of-curve makes as it traverses the tile, with a "left-hand turn" counting as -1. (To be more rigorous, we would define this number using 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 $-2, -1, 0, 1, 2$:



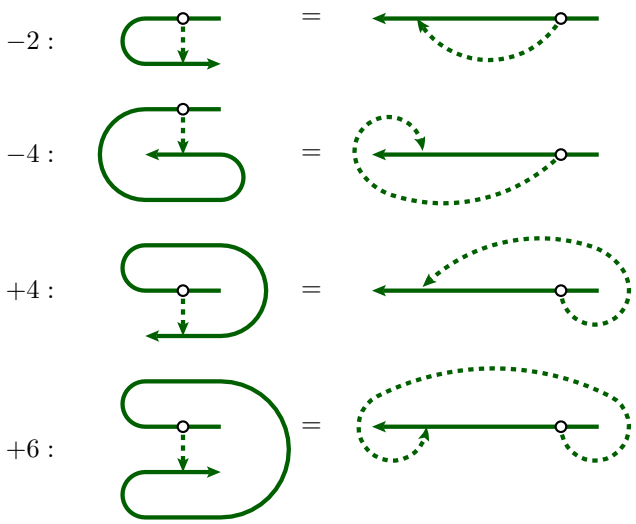
4.4.2 The paperclip algorithm

In the description of the refactoring theorem in Section 3.4 we thought of R-moves as acting on the basis of the system as in the Heisenberg picture. Now we switch to an equivalent perspective and consider R-moves as transport of anyon charges as in a Schrodinger picture. The anyons will be transported around the lattice by moving them along tile edges. 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:

$$
\begin{aligned}
-2 &: R[B] \\
-4 &: R[H^r]\ R[H]\ R[B] \\
+4 &: R[B]\ R[T]\ R[T^r] \\
+6 &: R[H^r]\ R[H]\ R[B]\ R[T]\ R[T^r]
\end{aligned}
$$

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).

4.5 Numerical results

The results of simulating the noise plus decoder are shown in Figure 4.3. We run a monte-carlo simulation, for various lattice sizes L, and error rate per edge t_{sim}. The

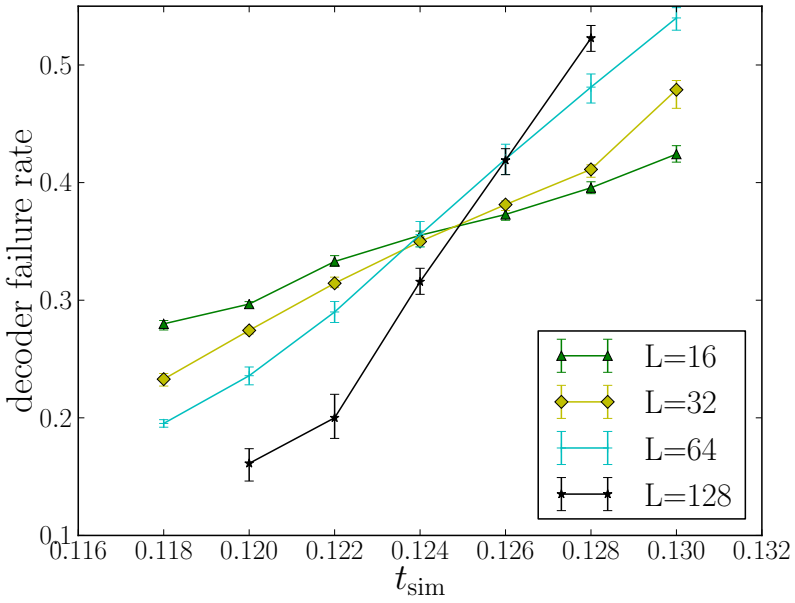


Figure 4.3: Results of monte-carlo sampling of the decoding simulation for various lattice sizes L and expected pair-creation events per edge t_{sim}.

number of pair-creation processes accross each edge of the lattice is sampled from a poisson process, and t_{sim} indicates the expected value of this number.

The decoder failure rate shows a clear threshold at around $t_{\mathrm{sim}} = 0.125 \pm 0.003$. Below this threshold errors are increasingly suppressed as the lattice size grows.

4.6 Summary

{we did stuff}

CHAPTER 5

Conclusions

{brief chapter summarising all results}

Bibliography

[1] S. Abramsky. Temperley-lieb algebra: from knot theory to logic and computation via quantum mechanics. In L. K. G. Chen and S. Lomonaco, editors, *Mathematics of Quantum Computing and Technology*, pages 415–458. Taylor & Francis, 2007.

[2] R. S. Andrist, J. R. Wootton, and H. G. Katzgraber. Error thresholds for abelian quantum double models: Increasing the bit-flip stability of topological quantum memory. *Phys. Rev. A*, 91:042331, Apr 2015.

[3] H. Anwar, B. J. Brown, E. T. Campbell, and D. E. Browne. Fast decoders for qudit topological codes. *New J. Phys.*, 16(6):063038, jun 2014.

[4] E. Artin. Theory of braids. *Annals of Mathematics*, pages 101–126, 1947.

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

[6] D. Bacon and A. Casaccino. Quantum error correcting subsystem codes from two classical linear codes. *arXiv preprint quant-ph/0610088*, 2006.

[7] J. Baez and M. Stay. *Physics, topology, logic and computation: a Rosetta Stone*. Springer, 2010.

[8] J. C. Baez and J. Biamonte. Quantum techniques for stochastic mechanics. *arXiv preprint arXiv:1209.3632*, 2012.

[9] B. Bakalov and A. A. Kirillov. *Lectures on tensor categories and modular functors*, volume 21. American Mathematical Society Providence, 2001.

[10] B. Bartlett, C. L. Douglas, C. J. Schommer-Pries, and J. Vicary. Modular categories as representations of the 3-dimensional bordism 2-category. *arXiv preprint arXiv:1509.06811*, 2015.

[11] M. Z. Bazant. Largest cluster in subcritical percolation. *Phys. Rev. E*, 62:1660–1669, Aug 2000.

[12] M. E. Beverland, R. König, F. Pastawski, J. Preskill, and S. Sijher. Protected gates for topological quantum field theories. *arXiv preprint arXiv:1409.3898*, 2014.

[13] J. Birman. Braids, links and mapping class groups. *Annals of Math. Studies*, 82, 1974.

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

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

[16] H. Bombin, G. Duclos-Cianci, and D. Poulin. Universal topological phase of two-dimensional stabilizer codes. *New J. Phys.*, 14(7):073048, 2012.

[17] H. Bombin and M. Martin-Delgado. Exact topological quantum order in d= 3 and beyond: Branyons and brane-net condensates. *Physical Review B*, 75(7):075103, 2007.

[18] N. E. Bonesteel and D. P. DiVincenzo. Quantum circuits for measuring Levin-Wen operators. *Phys. Rev. B*, 86:165113, Oct 2012.

[19] S. Bravyi. Subsystem codes with spatially local generators. *Phys. Rev. A*, 83:012320, Jan 2011.

[20] S. Bravyi, D. P. Divincenzo, R. Oliveira, and B. M. Terhal. The complexity of stoquastic local hamiltonian problems. *Quantum Information & Computation*, 8(5):361–385, 2008.

[21] S. Bravyi, M. Hastings, and S. Michalakis. Topological quantum order: stability under local perturbations. *J. Math. Phys.*, 51:093512, Jan 2010.

[22] S. Bravyi and M. B. Hastings. A short proof of stability of topological order under local perturbations. *Comm. Math. Phys.*, 307(3):609–627, Jan 2011.

[23] S. Bravyi, M. Suchara, and A. Vargo. Efficient algorithms for maximum likelihood decoding in the surface code. *Phys. Rev. A*, 90:032326, Sep 2014.

[24] C. G. Brell, S. Burton, G. Dauphinais, S. T. Flammia, and D. Poulin. Thermalization, error-correction, and memory lifetime for ising anyon systems. *Phys. Rev. X*, 4:031058, 2014.

[25] B. J. Brown, D. Loss, J. K. Pachos, C. N. Self, and J. R. Wootton. Quantum memories at finite temperature. *Rev. Mod. Phys.*, 88:045005, Nov 2016.

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

[27] S. Burton. A Short Guide to Anyons and Modular Functors. 2016.

[28] S. Burton, C. G. Brell, and S. T. Flammia. Classical simulation of quantum error correction in a fibonacci anyon code. *arXiv preprint arXiv:1506.03815*, 2015.

[29] 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.

[30] J. Cheeger. A lower bound for the smallest eigenvalue of the laplacian. *Problems in analysis*, pages 195–199, 1970.

[31] F. R. Chung. *Spectral graph theory*. Number 92. American Mathematical Soc., 1997.

[32] P. Dehornoy. *Why are braids orderable?* Panoramas et synthèses - Société mathématique de France. Société Mathématique de France, 2002.

[33] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill. Topological quantum memory. *J. Math. Phys.*, 43(9):4452–4505, 2002.

[34] J. Dorier, F. Becca, and F. Mila. Quantum compass model on the square lattice. *Physical Review B*, 72(2):024448, 2005.

[35] H. S. Dragos M. Cvetkovic, Michael Doob. *Spectra of graphs. Theory and application*. Pure and Applied Mathematics. Academic Press, 3rd revised edition, 1980.

[36] G. Duclos-Cianci and D. Poulin. Fast decoders for topological quantum codes. *Phys. Rev. Lett.*, 104:050504, 2010.

[37] G. Duclos-Cianci and D. Poulin. A renormalization group decoding algorithm for topological quantum codes. In *Information Theory Workshop (ITW), 2010 IEEE*, pages 1–5, 2010.

[38] G. Duclos-Cianci and D. Poulin. Fault-tolerant renormalization group decoder for Abelian topological codes. *Quant. Inf. Comput.*, 04 2013.

[39] B. Farb and D. Margalit. *A Primer on Mapping Class Groups (PMS-49)*. Princeton University Press, 2011.

[40] A. G. Fowler. Minimum weight perfect matching of fault-tolerant topological quantum error correction in average $o(1)$ parallel time. *Quant. Inf. Comput.*, 15:0145–0158, 2015.

[41] M. H. Freedman, A. Kitaev, and Z. Wang. Simulation of topological field theories by quantum computers. *Communications in Mathematical Physics*, 227(3):587–603, 2002.

[42] M. H. Freedman, M. Larsen, and Z. Wang. A modular functor which is universal for quantum computation. *Communications in Mathematical Physics*, 227(3):605–622, 2002.

[43] S. Friedland and R. Nabben. On cheeger-type inequalities for weighted graphs. *Journal of Graph Theory*, 41(1):1–17, 2002.

[44] F. G. Frobenius. Über matrizen aus nicht negativen elementen. 1912.

[45] W. Fulton and J. Harris. *Representation theory: a first course*, volume 129. Springer Science & Business Media, 2013.

[46] R. Ghrist. *Elementary applied topology*. Createspace, 2014.

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

[48] D. Gottesman. The Heisenberg representation of quantum computers. In S. P. Corney, R. Delbourgo, and P. D. Jarvis, editors, *Group22: Proceedings of the XXII International Colloquium on Group Theoretical Methods in Physics*, pages 32–43, Cambridge, MA, 1999. International Press.

[49] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer, 2009.

[50] V. Hernandez, J. E. Roman, and V. Vidal. SLEPc: A scalable and flexible toolkit for the solution of eigenvalue problems. *ACM Trans. Math. Software*, 31(3):351–362, 2005.

[51] A. Hutter, D. Loss, and J. R. Wootton. Improved hdrg decoders for qudit and non-abelian quantum error correction. *New J. Phys.*, 17(3):035017, 2015.

[52] A. Hutter and J. R. Wootton. Continuous error correction for ising anyons. 2015.

[53] A. Hutter, J. R. Wootton, and D. Loss. Efficient markov chain monte carlo algorithm for the surface code. *Phys. Rev. A*, 89:022326, Feb 2014.

[54] N. V. Ivanov. Mapping class groups. In R. Sher and R. Daverman, editors, *Handbook of Geometric Topology*, pages 523–633. Elsevier Science, 2001.

[55] V. F. Jones. Planar algebras, i. *arXiv preprint math/9909027*, 1999.

[56] P. Jordan and E. P. Wigner. About the pauli exclusion principle. *Z. Phys*, 47(631):14–75, 1928.

[57] E. Kapit and S. Simon. Three- and four-body interactions from two-body interactions in spin models: A route to abelian and non-abelian fractional chern insulators. *Phys. Rev. B*, 88:184409, Nov 2013.

[58] C. Kassel, O. Dodane, and V. Turaev. *Braid Groups*. Graduate Texts in Mathematics. Springer New York, 2010.

[59] G. Kells, J. K. Slingerland, and J. Vala. Description of kitaev's honeycomb model with toric-code stabilizers. *Phys. Rev. B*, 80:125415, Sep 2009.

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

[61] A. Kitaev and J. Preskill. Topological entanglement entropy. *Physical review letters*, 96(11):110404, 2006.

[62] A. Y. Kitaev. Fault-tolerant quantum computation by anyons. *Ann. Phys.*, 303(1):2–30, 2003.

[63] D. A. Klain and G.-C. Rota. *Introduction to geometric probability*. Cambridge University Press, 1997.

[64] A. Kubica, B. Yoshida, and F. Pastawski. Unfolding the color code. *New Journal of Physics*, 17(8):083026, 2015.

[65] S. Lefschetz. The early development of algebraic topology. *Bulletin of the Brazilian Mathematical Society*, 1(1):1–48, 1970.

[66] M. A. Levin and X.-G. Wen. String-net condensation: A physical mechanism for topological phases. *Phys. Rev. B*, 71:045110, Jan 2005.

[67] E. Lieb, T. Schultz, and D. Mattis. Two soluble models of an antiferromagnetic chain. *Annals of Physics*, 16(3):407–466, 1961.

[68] B. D. McKay and A. Piperno. Practical graph isomorphism, {II}. *Journal of Symbolic Computation*, 60(0):94 – 112, 2014.

[69] S. Michalakis and J. P. Zwolak. Stability of frustration-free Hamiltonians. *Comm. Math. Phys.*, 322(2):277–302, 2013.

[70] R. S. K. Mong, D. J. Clarke, J. Alicea, N. H. Lindner, P. Fendley, C. Nayak, Y. Oreg, A. Stern, E. Berg, K. Shtengel, and M. P. A. Fisher. Universal topological quantum computation from a superconductor-abelian quantum hall heterostructure. *Phys. Rev. X*, 4:011036, Mar 2014.

[71] R. Movassagh. Generic local hamiltonians are gapless. *arXiv preprint arXiv:1606.09313*, 2016.

[72] C. Nayak, S. H. Simon, A. Stern, M. Freedman, and S. Das Sarma. Non-Abelian anyons and topological quantum computation. *Rev. Mod. Phys.*, 80(3):1083, 2008.

[73] G. Palumbo and J. K. Pachos. Non-abelian chern-simons theory from a hubbard-like model. *Phys. Rev. D*, 90:027703, Jul 2014.

[74] F. Pastawski, A. Kay, N. Schuch, and J. I. Cirac. Limitations of Passive Protection of Quantum Information. *Quant. Inf. Comput.*, 10:580, 2010.

[75] O. Perron. Zur theorie der matrices. *Mathematische Annalen*, 64(2):248–263, 1907.

[76] R. N. C. Pfeifer, O. Buerschaper, S. Trebst, A. W. W. Ludwig, M. Troyer, and G. Vidal. Translation invariance, topology, and protection of criticality in chains of interacting anyons. *Phys. Rev. B*, 86:155111, Oct 2012.

[77] P. Pfeuty. The one-dimensional ising model with a transverse field. *ANNALS of Physics*, 57(1):79–90, 1970.

[78] J. K. Slingerland and F. A. Bais. Quantum groups and non-abelian braiding in quantum hall systems. *Nucl. Phys. B*, 612(3):229–290, 2001.

[79] U. Tillmann. S-structures for k-linear categories and the definition of a modular functor. *Journal of the London Mathematical Society*, 58(1):208–228, 1998.

[80] V. G. Turaev. *Quantum invariants of knots and 3-manifolds*, volume 18. Walter de Gruyter, 1994.

[81] K. Walker. On wittens 3-manifold invariants. *preprint*, 1991.

[82] D. S. Wang, A. G. Fowler, C. D. Hill, and L. C. L. Hollenberg. Graphical algorithms and threshold error rates for the 2d color code. *Quant. Inf. Comput.*, 10(9):780–802, 2010.

[83] D. S. Wang, A. G. Fowler, A. M. Stephens, and L. C. L. Hollenberg. Threshold error rates for the toric and surface codes. *Quant. Inf. Comput.*, 10:456, 2010.

[84] F. H. E. Watson, H. Anwar, and D. E. Browne. A fast fault-tolerant decoder for qubit and qudit surface codes. 2014.

[85] F. Wilczek. *Fractional Statistics and Anyon Superconductivity*. World Scientific, Singapore, 1990.

[86] J. R. Wootton. A simple decoder for topological codes. *Entropy*, 17(4):1946–1957, apr 2015.

[87] J. R. Wootton, J. Burri, S. Iblisdir, and D. Loss. Decoding non-Abelian topological quantum memories. *Phys. Rev. X*, 4:011051, 2014.

[88] J. R. Wootton and A. Hutter. Active error correction for abelian and non-abelian anyons. 2015.

[89] J. R. Wootton and D. Loss. High threshold error correction for the surface code. *Phys. Rev. Lett.*, 109(16):160503, 2012.

[90] B. Yoshida and I. L. Chuang. Framework for classifying logical operators in stabilizer codes. *Physical Review A*, 81(5):052302, 2010.