Quantum error correction thresholds for non-Abelian Turaev-Viro codes

Alexis Schotte

Supervisors: Prof. Dr. Frank Verstraete
Prof. Dr. Jutho Haegeman

Dissertation submitted in fulfillment of the requirements for the degree of
Doctor in Science: Physics

Department of Physics and Astronomy
Faculty of Sciences

2016-2021

Doctoral Committee

Preface

This dissertation is the culmination of four years of doctoral research. During these years, I have had the opportunity to work on two research projects that were intimately related, but still quite different. The common thread in these projects is their connection to topological order and the use of tensor network representations of many-body quantum states in the process of solving the problem at hand.

The first project consisted of studying non-Abelian topological phase transitions in Levin-Wen string-net models using a variational class of tensor-network states. This research was performed in collaboration with Jose Carrasco, Bram Vanhecke, Laurens Vanderstraeten, Jutho Haegeman, Frank Verstraete and Julien Vidal. The resulting article was published in Ref. [1].

The second project concerns the use of Levin-Wen models as quantum error correcting codes and the simulation of their error correction thresholds. It combines insights from topological quantum field theory with tensor network techniques in order to obtain the first error correction threshold for a two-dimensional error correcting code (of qubits) subject to generic noise, for which universal quantum computation can be performed within its code space. This research was performed in collaboration with Guanyu Zhu, Lander Burgelman and Frank Verstraete. After three years of work and about 180 000 CPU hours worth of simulations, my collaborators and I have detailed our findings on this topic in Ref. [2].

[1]   Alexis Schotte et al. "Tensor-network approach to phase transitions in string-net models". In: *Physical Review B* 100.24 (2019), p. 245125.

[2]   Alexis Schotte et al. "Quantum error correction thresholds for the universal Fibonacci Turaev-Viro code". In: *arXiv preprint arXiv:2012.04610* (2020).

This dissertation will cover the research performed in Ref. [2]. For completeness, the content of Ref. [1] is provided in Appendix E.

Acknowledgment

The past four years and a half have been an quite a journey, which ultimately resulted in this dissertation. All of this could not have been possible without the help and support of many different people. I, therefore, wish to take a moment to thank all of them for their direct or indirect contributions to this work and to my well-being while producing it.

First and foremost, I would like to thank the people of Belgium and Europe for granting me the opportunity of doing doctoral research. Being able to spend four years solely on thinking has been an incredible privilege for which I will forever be grateful.

My doctoral research has been both more pleasant and more productive thanks to the many great people that surrounded me during these years. In particular, I would like to thank my supervisors, professor Frank Verstraete and professor Jutho Haegeman for their trust and guidance along the way. Thanks to their enthusiasm and ceaseless encouragements, I was able to persevere even when results initially seemed to stay absent, and eventually managed to complete this research project.

I also thank my colleagues from the *quantum group* in Ghent for the amazing atmosphere and the great number of interesting discussions, both on physics and all else. Inge, for her ability to handle the chaos that is the administration of said research group with remarkable calmness and efficacy. And my various coauthors: Jose Carrasco, Bram Vanhecke, Laurens Vanderstraeten, Julien Vidal, and in particular Guanyu Zhu and Lander Burgelman for the fruitful collaborations.

Many of the numerical simulations were performed on the Stevin high-performance computing infrastructure of the Flemish Supercomputer Center, funded by the Research Foundation Flanders (FWO) and the Flemish Government. I am grateful for their amazing support, and the helpful answers to all my questions.

I would like to express my gratitude to all members of my doctoral committee for taking their time to read this thesis, and for their genuine interest, questions and valuable suggestions.

Life would have been utterly boring, if it wasn't for all the friends I have met. I am grateful to all of them for enriching my life, each of them in their own and unique way. To the close friends that I kept from my studies in Leuven: Samuel, Bram, Koen, Bob, Vincent, Simon and Jeroen, for the hours upon hours of pointless discussions on all possible topics, resulting in many laughs and the occasional acquired wisdom. To all those in the Lindy Hop community in Belgium and beyond, for the countless hours of absolute joy and excitement, and for creating a magical atmosphere where people can connect on a level that I've yet to experience elsewhere. To Juliette, for being an amazing friend. And to Delphine, for her exhilarating exuberance. Due to the rather unfortunate combination

of writing a thesis and a global pandemic, I haven't seen many of you in ages. I can't wait for that to change.

Finally, I would like to thank my family. My father, the original Dr. A. Schotte, for always believing in me and for inspiring me to pursue my ambitions. My mother, for her incredible devotion to help me overcome my struggles and reach my potential. When anything involving numbers seemed like an insurmountable task for me 20 years ago, who would have thought I would ultimately end up pursuing a doctoral degree in theoretical physics? I certainly didn't, but thanks to my parents, it turned out I could. My brother and sister, for their profound impact on who I became as a person. My grandparents, for their unconditional love and support. My parents in law, for accepting me into their family as if they had known me forever. Illa, for being a good girl (occasionally, let's not exaggerate here), and for the many walks providing the opportunity to clear my head. And, of course, Sofie, for making my life better in more ways than I ever thought possible. Thank you for being by my side and supporting me, especially during those final months when finishing my thesis in the midst of a global pandemic. Of all 7.8 billion people who walk the Earth, there is none with whom I would rather have spend the seemingly never ending lockdown. I do not know where life will take us, but as long as we're together, I am certain the road will be fascinating.

<div align="right">

Alexis Schotte
January 2021

</div>

Nederlandstalige samenvatting

Achtergrond: kwantumcomputers en foutcorrectie

Doordat transistoren steeds kleiner worden en we stilaan in het regime komen waarbij kwantummechanische effecten een belangrijk rol beginnen te spelen, is het duidelijk dat we binnenkort de limiet zullen bereiken van wat mogelijk is met klassieke computers. In de jaren 80, is het idee ontstaan om die kwantummechanische effecten te benutten om buiten de grenzen van klassieke computers te treden. Het resultaat is een nieuw soort computer, genaamd de *kwamtumcomputer*. Deze heeft fundementele verschillen ten opzichte van een klassieke computer. De kwantumcomputer maakt namelijk gebruik van *kwantumbits* (qubits) in plaats van klassieke bits. Door de principes van de kwantummechanica te gebruiken kan een kwantumcomputer bepaalde berekeningen uitvoeren die onmogelijk zijn voor zelfs de krachtigste klassieke supercomputers.

Een belangrijk voorbeeld hiervan is het simuleren van kwantumveeldeeltjessystemen. De simulatie van deze systemen zou kunnen leiden tot een enorme vooruitgang in kwantum-chemie, vastestoffysica en bijvoorbeeld ook in farmaceutisch onderzoek. Deze uitgebreide mogelijkheden hebben ertoe geleid dat er grote belangstelling is voor kwantumcomputers in zowel de academische wereld als in the tech industrie. Het bouwen van een kwantum-computer blijkt echter een bijzonder grote uitdaging. Een van de grootste hindernissen is het feit dat elke onvermijdbare interactie met de omgeving zorgt voor kleine foutjes die de berekening kunnen verstoren en tot een verkeerd resultaat kunnen leiden. Zulke fouten ko-men ook voor in klassieke computers, maar voor kwantumcomputers is de situatie een stuk ingewikkelder. Ten eerste zijn fouten in klassieke computers beperkt tot "bit-flips"(waarbij een bit van 0 naar 1 wordt veranderd of vice versa) terwijl deze in kwantumcomputers de toestand kunnen transformeren naar een continuüm aan foutieve toestanden. Ten tweede beïnvloeden metingen in kwantummechanica de toestand van het systeem dat gemeten wordt. Hierdoor is het een erg delicate taak om fouten in een kwantumcomputer te detec-teren zonder deze daardoor te verstoren. Bovendien stelt het "no-cloning" theorema dat het onmogelijk is om kwantumtoestanden gewoonweg te kopiëren als "back-up".

In 1995 ontdekte Peter Shor de eerste *kwantumfoutcorrigerende code*. Dit is een me-thode waarmee de kwantumtoestand van "logische" qubits beschermd kan worden tegen fouten door deze op te slaan in de toestand van een groter aantal fysieke qubits. Sinds-dien is kwantumfoutcorrectie (QEC) uitgegroeid tot een bloeiend onderzoeksdomein. In het algemeen bestaat een QEC procedure uit drie verschillende stappen: eerst wordt de kwantumtoestand van de N logische qubits geëncodeerd in een groter aantal fysieke qu-bits. De resulterende toestand zal zich steeds bevinden in een 2^N-dimensionale deelruimte genaamd de *coderuimte* van de totale fysieke Hilbert ruimte. Deze fysieke qubits worden dan blootgesteld aan ruis (in de vorm van willekeurige fouten op individuele qubits) die de toestand uit de coderuimte zullen halen. Hierna wordt er een reeks van metingen uit-gevoerd (genaamd *syndrome measurements*), om uit te zoeken wat voor fouten gebeurd

zijn. Zoals reeds vermeld, moet men hierbij erg voorzichtig zijn om de geëncodeerde informatie niet te verstoren. Op basis van de resultaten van deze "syndrome" metingen, zal een algoritme uiteindelijk beslissen welke stappen moeten worden ondernomen om de oorspronkelijke toestand te herstellen. Voor deze laatste stap bestaan verschillende strategieën (*decoders*), die elk een verschillende mate van bescherming bieden tegen bepaalde soorten ruis. Deze verschillende *decoders* worden vaak vergeleken op basis van hun *foutcorrectiedrempel*. Dit is een getal dat aanduidt wat de maximale intensiteit is van de ruis waartegen een bepaalde decoder bescherming biedt.

Een veelbelovende klasse van foutcorrigerende codes zijn *topologische codes*. Dit zijn codes waarbij de logische kwantumtoestand, die men tracht te beschermen tegen fouten, wordt opgeslagen in de grondtoestandsruimte van een tweedimensionaal topologisch kwantumsysteem. Topologische kwantumsystemen vertonen *topologische kwantumfasen van materie*, die niet beschreven kunnen worden door het gebruikelijke paradigma van symmetriebreking in de klassieke fysica. Deze systemen worden gekenmerkt door een ontaarding van de grondtoestandsruimte en door het feit dat ze onconventionele excitaties vertonen die zich noch als bosonen noch als fermionen gedragen. Deze bijzondere excitaties worden *anyonen* genoemd. De topologische systemen waar ze in voorkomen kunnen onderverdeeld worden in twee categorieën. Deze categorieën worden gebaseerd op de manier waarop een paar anyonen kan samensmelten tot een nieuw anyon. Voor *Abelse* modellen is de uitkomst van zo'n samensmelting steeds uniek, terwijl er bij *niet-Abelse* modellen meerdere mogelijke uitkomsten zijn. Hoewel deze laatste soort ingewikkelder is, leidt het wel tot interessante implicaties. Men kan namelijk voor sommige niet-Abelse modellen de taken van een universele kwantumcomputer uitvoeren door paren van anyon te creëren, ze rond elkaar te winden volgens bepaalde patronen, en ze dan terug samen te smelten.

In topologische codes zorgen lokale fouten ervoor dat er paren van deze anyonen ontstaan. De correctieprocedure bestaat er dan in deze deeltjes op de juiste manier samen te brengen zodat ze elkaar opheffen en het systeem naar de juiste grondtoestand terugkeert. Er zijn verschillende gekende topologische codes. De meest bekende is de *surface code*, een Abelse code die door Kitaev werd geïntroduceerd in 1997 en, omwille van zijn hoge foutcorrectiedrempel, momenteel wordt gebruikt in een aantal experimentele opstellingen om de eerste kleinschalige kwantumcomputers te maken.

Naast het beschermen van de logische qubits tegen lokale fouten, vereist een werkende universele kwantumcomputer ook de mogelijkheid om een *universele verzameling van operaties* uit te voeren op deze logische qubits. Dit moet bovendien gebeuren op een manier waarbij ze te allen tijde beschermd blijven tegen fouten. Dit blijkt een grote uitdaging voor de meeste foutcorrigerende codes. In veel gevallen is er een grote extra kost verbonden aan het veilig kunnen uitvoeren van dergelijke universele kwantumoperaties. Voor sommige niet-Abelse topologische codes kunnen universele berekeningen echter worden uitgevoerd zonder dat men ooit de coderuimte moet verlaten, waardoor dit dus ook intrinsiek veilig gebeurt (en zonder extra kost). Zulke codes zijn daarom een aantrekkelijk alternatief voor de huidige aanpak op basis van Abelse codes, maar tot voor kort was er zeer weinig geweten over hun foutcorrectiedrempel (m.a.w. of ze ook werkelijk voldoende bescherming kunnen bieden tegen realistische hoeveelheden ruis).

Het vinden van foutcorrectiedrempels voor universele niet-Abelse codes

In de laatste jaren is veel vooruitgang geboekt in het onderzoek naar niet-Abelse topologische codes. Men is er in geslaagd om voor sommige codes deze error tresholds te schatten.

Dit onderzoek maakte gebruik van fenomenologische modellen die een versimpelde weergave bieden van niet-Abelse codes. In plaats van te werken met een microscopisch model van fysieke qubits die zijn blootgesteld aan ruis, werd er gewerkt met modellen waarin fouten worden voorgesteld als het ontstaan van paren van anyonen en als de beweging van deze anyonen. Hoewel dit een sterke aanwijzing is dat niet-Abelse codes ook hoge foutcorrectiedrempels hebben, ontbraken er tot op heden resultaten over foutcorrectiedrempels voor meer realistische ruis op het niveau van fysieke qubits in een *microscopische* beschrijving van een universele niet-Abelse code. Deze thesis beschrijft de eerste succesvolle poging om dit te veranderen.

We definiëren hiervoor een nieuwe topologische foutcorrigerende code die gebaseerd is op string-net modellen. Dit is een familie van topologische kwantumsystemen die werd ontdekt door Levin en Wen in 2005 en waarvan wordt gedacht dat ze alle topologische fasen kan beschrijven. Specifiek gebruiken wij een aangepaste versie van het Fibonacci string-net model, dat anyonische excitaties bevat die universeel zijn.

Door gebruik te maken van inzichten uit topologische kwantumveldentheorie, kunnen we ingewikkelde kwantumtoestanden van de fysieke qubits eenvoudig voorstellen aan de hand van superposities van *fusiediagrammen* van anyonen. Bovendien stellen deze inzichten ons ook in staat om af te leiden hoe de lading van anyonische excitaties gemeten kan worden, wat overeenkomt met de eerder beschreven *syndrome measurements*. We geven daarna een gedetailleerde beschrijving van hoe deze metingen kunnen worden uitgevoerd als een reeks van elementaire operaties en metingen op een klein aantal qubits. Ook alle ingrepen die nodig zijn in de correctieprocedure worden op deze manier beschreven.

Het grootste obstakel in het simuleren van deze foutcorrigerende code is het bepalen van de anyonen die onstaan door individuele fouten op fysieke qubits. We maken hiervoor gebruik van zogenaamde *tensornetwerktoestanden*. Dit is een theoretische methode voor het beschrijven van kwantumveeldeeltjessystemen die gebaseerd is op recente inzichten in kwantumverstrengeling, een eigenschap die intristiek is aan kwantummechanische systemen. Tensornetwerktoestanden laten ons toe om kwantumveeldeeltjessystemen bij lage energie zeer efficiënt te modelleren en zijn daarom uitermate geschikt om de zeer complexe toestanden van de fysieke qubits te beschrijven nadat er door lokale fouten groepen van anyonische quasideeltjes zijn ontstaan. Dit stelt ons in staat om fouten op qubit-niveau te beschrijven aan de hand van collectieve eigenschappen van de excitaties die zij creëren.

Gewapend met deze kennis en voortbouwend op het eerder vermelde onderzoek met fenomenologische modellen, construeren we een methode voor een volledige simulatie van de foutcorrectie in de ontwikkelde code. Deze simulatie beschrijft hoe de kwantumtoestand van het systeem verandert door de invloed van ruis op qubits door de impact van de metingen die worden uitgevoerd en door de operaties die worden toegepast tijdens de correctieprocedure. De microscopische ruis wordt hierbij voorgesteld als een model van willekeurige Pauli-operatoren die worden toegepast op individuele qubits. Deze methode wordt vervolgens gebruikt om de allereerste schatting van de foutcorrectiedrempel in een volledig microscopische universele niet-Abelse foutcorrigerende code uit te voeren.

Contents

1 | Introduction

This dissertation studies the use of Levin-Wen models as quantum error correcting codes and the simulation of their error correction thresholds. In doing so, three different themes in physics are combined: topological order, tensor networks, and quantum error correction. Before giving a more technical introduction to the research presented in this dissertation, we provide a brief introduction to these three topics to familiarize the reader with the broader context wherein this dissertation should be understood.

1.1 Topological order and anyons

For a long time, it was believed that the Landau-Ginzburg theory of symmetry breaking [1] described all possible phases of matter. In the late 1980s, however, the study of chiral spin states [2, 3] in the context of high temperature superconductivity, revealed that these states contain a new kind of order that can not be attributed to symmetry breaking. Due to the connection with topological quantum field theory, the new type of order was named *topological order*. While experiments showed that chiral spin states did not provide the right description of high temperature superconductivity, it was found that the similarity between these states and fractional quantum Hall (FQH) states [4, 5] makes it possible to use the theory of topological order to describe different FQH states, confirming that the theory was realized in real physical systems.

Several definitions of topological order exist [6]. In 2+1 dimensions, most of them are built around the following two key features [7]. The first one is a robust topological ground space degeneracy. Here, *robust* means that this degeneracy is stable against generic perturbations of the Hamiltonian [8, 9] and has a stable energy gap (in the thermodynamical limit). Note that since it is robust against *any* local perturbation (given that the system is large enough), this includes perturbations that break any possible symmetry. Meanwhile *topological* refers to the fact that this degeneracy typically depends on the topology of space, e.g., on the genus of a closed surface as in Fig. 1.1.

The second defining property is the non-Abelian geometric phase of topologically degenerate ground states [10, 11, 12], which is a unitary matrix U associated to a closed path of Hamiltonians. For example, for a system defined on a torus, there are two such paths


g=0   g=1   g=0
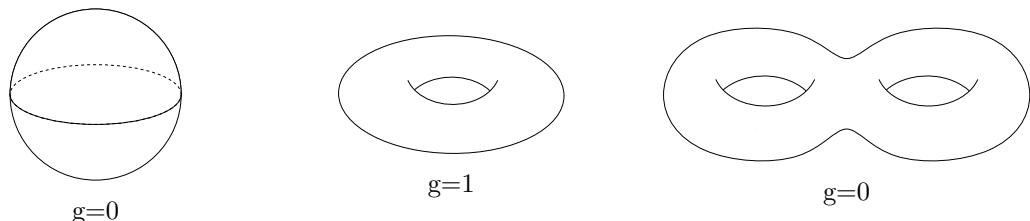
Figure 1.1: Orientable surfaces with genus $g = 0, 1, 2$.

associated to shearing and squeezing the torus. The transformation of the ground space under these actions is given by the matrices S and T, whose quantum numbers uniquely define the topological phase of the system.

Besides the two properties listed above, topologically ordered systems in 2+1 dimensions are characterized by the low-energy excitations that appear on top of the degenerate ground space. Contrary to the situation arising in three spatial dimensions[1], exchanging a pair of identical particles in two spatial dimensions may result in any[2] phase $e^{i\theta}$, rather than only ± 1. The special cases $\theta = 0, \pi$ correspond to bosons and fermions, respectively. Particles with other values of θ are called *anyons* [14]. Their complete mathematical description relies on *category theory*, which will be discussed in Sec. 2.1. For a pedagogical introduction on anyon models, the reader is referred to Ref. [15].

1.2 Quantum computing and quantum error correction

In the 1980s the idea originated that quantum mechanics could be used as a resource to simulate other quantum mechanical systems and to perform computations that are intractable for classical computers [16, 17]. In particular, inherent quantum properties such as interference and entanglement can be exploited for information processing purposes, resulting in a novel computing paradigm. Since then, quantum computing has grown into a very active field of research [15, 18] and the experimental efforts to realize a scalable quantum computer are well on their way.

Broadly speaking, a quantum computer should be understood as a device that can perform the following operations on a quantum state (which is typically a system of qubits) in a controlled fashion:

- **Initialization**: one must be able to initialize the system in some known initial quantum state (e.g., $|000\rangle$ for a system of 3 qubits).

- **Unitary evolution**: by performing a circuit of unitary multi-qubit gates, the system is evolved according to some unitary operator. In order to realize any generic unitary with arbitrary precision, a *universal* gate set (consisting of a few 1 and 2-qubit gates) is required.

- **Measurement**: measure the quantum states in some basis.

Noise induced by interactions with the environment and errors caused by imperfect hardware cause a serious threat to an ongoing quantum computation. Indeed, the quantum information manipulated by a quantum computer is extremely fragile, since it is encoded in nonlocal correlations between different parts of the system. Therefore, it is crucial that this quantum information is protected adequately.

To understand the nature of this challenge we will start by illustrating how *classical* information, consisting of bits, can be protected from errors. On classical information, the only possible errors are bit-flips ($0 \leftrightarrow 1$). Classical information can be protected against such errors by using a classical *error correcting code*. This is a method in which a number of *logical* bits (containing the information that we wish to protect) are encoded in a greater number of *physical* bits. The simplest example of a classical error correcting code is the

---

[1]Exhanging a pair of particles twice in 3 spatial dimensions results in a path which can always be untangled and shrunk to a trivial path. Hence, the phase that is picked up when exchanging a pair has to square to 1.

[2]Technically, these exchange phases must always be a root of unity [13].

repetition code, in which one logical bit is replaced by 3 copies of itself:

$$0 \mapsto 000,$$
$$1 \mapsto 111.$$

In case one of these three bits, for example the first one, is affected by an error, this transforms these states to

$$000 \to 100,$$
$$111 \to 011.$$

Clearly, as long as more than one of the physical bits is affected, we can recover the initial information using

$$100, 010, 001 \to 000,$$
$$011, 101, 110 \to 111.$$

If the probability for an error is p on each of the three bits, the probability that we fail to recover the initial information correctly is $P_{\text{fail}} = 3p^2 - 2p^3$. Hence, when $3p^2 - 2p^3 < p$ or $p < 1/2$, the three bit repetition code improves the reliability of the encoded information. We can play the same game with N bits instead of 3 to further improve the reliability. For large N, and with $p = \frac{1}{2} - \epsilon$, one finds that

$$P_{\text{fail}} \sim \mathrm{e}^{-N\epsilon^2},$$

meaning that for any $p < 1/2$, we can recover the initial information with an arbitrary degree of certainty by increasing N.

Similar to their classical counterparts, quantum error correcting codes protect the state of logical qubits by encoding it in the state of a higher number of qubits[3]. Contrary to classical information however, protecting quantum information against noise is a highly nontrivial task due to the following difficulties which arise in the quantum case:

- **A continuum of possible errors:** While errors on classical information consist exclusively of bit-flips, errors on qubits are unitary transformations. To protect quantum information against errors, we must be able to protect it against *any* unitary transformation, rather than some discrete set of possible errors.

- **Measurements disturb the state:** Detecting the occurrence of errors (in order to know what correction to apply) must happen without destroying the encoded information.

- **No cloning:** Due to the no-cloning theorem, one cannot simply protect quantum information by making additional copies of it.

In general, quantum error correcting codes encode the state of k logical qubits using n physical qubits. The information is thus encoded in a 2^k-dimensional subspace, called the *code space*, of the 2^n-dimensional physical Hilbert space. Errors on the physical qubits, will generally[4] take their total quantum state out of the code space. Errors are detected

---

[3]In general, there is no need to restrict to two level systems as the fundamental building blocks.

[4]Note that errors which do not take the state out of the code subspace map one logical state to another and are thus undetectable.

through measurements (called *syndrome measurements*), which must be carefully chosen as to not destroy the encoded information. Based on their outomes (called the *syndrome*), an appropriate sequence of unitary gates is applied to return the state to the code space. The error correction procedure succeeds if the combined action of the noise, measurement and correction operations is trivial inside the code space.

The first example of a quantum error correcting code was found by Shor in 1995 [19]. It encodes one logical qubit using 9 physical qubits which are grouped in three clusters:

$$|0\rangle \;\mapsto\; |\bar{0}\rangle \equiv \frac{1}{2\sqrt{2}}\Big(|000\rangle + |111\rangle \Big)\Big(|000\rangle + |111\rangle \Big)\Big(|000\rangle + |111\rangle \Big),$$

$$|1\rangle \;\mapsto\; |\bar{1}\rangle \equiv \frac{1}{2\sqrt{2}}\Big(|000\rangle - |111\rangle \Big)\Big(|000\rangle - |111\rangle \Big)\Big(|000\rangle - |111\rangle \Big),$$

and in general: $\alpha |0\rangle + \beta |1\rangle \mapsto \alpha |\bar{0}\rangle + \beta |\bar{1}\rangle$. Errors are detected in two steps. First, the observables $\sigma_z \otimes \sigma_z \otimes \mathbb{1}$ and $\mathbb{1} \otimes \sigma_z \otimes \sigma_z$ are measured within each of the three clusters. Note that this does not give us any information about the encoded state, which is crucial in order to avoid destroying the superposition $\alpha |\bar{0}\rangle + \beta |\bar{1}\rangle$. This information is sufficient to determine which, if any, of the three qubits within a cluster is flipped compared to the other two. We can then apply a σ_x operator to that qubit to flip it back. Second, the 6-qubit observables $\sigma_x^{\otimes 6} \otimes \mathbb{1}^{\otimes 3}$ and $\mathbb{1}^{\otimes 3} \otimes \sigma_x^{\otimes 6}$ are measured. This allows us to detect which, if any, of the clusters has a different sign than the others. This can then be corrected by applying $\sigma_z^{\otimes 3}$ to said cluster.

A generic unitary error, up to a physically irrelevant global phase, can be expanded as

$$U = \mathrm{e}^{\mathrm{i}(\varepsilon_x\sigma_x + \varepsilon_y\sigma_y + \varepsilon_z\sigma_z)} \approx \mathbb{1} + \mathrm{i}\varepsilon_x\sigma_x + \mathrm{i}\varepsilon_y\sigma_y + \mathrm{i}\varepsilon_z\sigma_z + O(|\varepsilon|^2)\,.$$

Upon performing the syndrome measurements described above, such a generic unitary will collapse to a bit-flip error (σ_x), a phase-flip error σ_z or both ($\sigma_y = \mathrm{i}\sigma_x\sigma_z$) with a probability of order $|\varepsilon|^2$. Hence, this code also provides protection against generic unitary errors, rather than only against bit-flip and phase-flip errors. For recovery to fail, at least two bit-flip errors must happen within a single cluster, or a phase-flip error must happen withing at least two clusters. If the unitary U was applied to every qubit, the probability of such events is of the order $|\varepsilon|^4$, which confirms that when $|\varepsilon|$ is small enough, the Shor code improves the reliability of the encoded quantum information.

Shortly after the discovery of the first quantum error correcting codes [19, 20, 21], it was found that quantum error correction is possible even when the recovery procedure itself involves imperfect gates [22, 23, 24]. Schemes with this property are called *fault-tolerant* and their existence confirms that quantum computation is indeed possible despite the fact that physical quantum gates will never be perfect. The only requirement is that they are *close enough* to the perfect case.

In the early 2000s, Kivaev showed that it is possible to simulate (and hence perform) universal quantum computation by braiding anyons [25]. This scheme, known as *topological quantum computation* has the advantage that operations performed through braiding are *inherently* fault-tolerant by construction. The simplest model of anyons which allows for universal quantum computation through braiding is the Fibonacci model [26].

The degenerate ground space of topological systems (see Sec. 1.1) can be used as the code space of an error correcting code. Local errors will then create pairs of anyonic excitations, which must be brought back together and annihilated in order to return to

the code space. The stable energy gap of topological systems means that they do offer a level of passive error protection, which can be combined with active error correction (i.e., actively fusing pairs of anyonic excitations) for nonzero temperatures in order to offer a high level of protection against thermal noise. Experimental realizations of topological systems, however, are not practical for the construction of quantum computers, which are more likely to use qubits as a fundamental resource.

Fortunately, the techniques of topological quantum error correction can be used without relying on any experimental realization of topologically phases. One can encode the logical state in a ground state of a local topologically ordered Hamiltonian defined on a system of qubits (regardless of the actual Hamiltonian governing the time evolution of those qubits). This Hamiltonian operator can then be measured actively such that errors result in the creation of pairs of (quasiparticle) anyonic excitations. One can then apply the machinery of topological error correction to fuse and annihilate these quasiparticle excitations. The most famous and simultaneously simplest example of a topological code is the surface code (or the associated toric code) [25, 27].

As a way to provide some intuition on the principles at play in topological codes, we conclude this section by giving a brief introduction on the toric code. The toric code is defined on a square lattice with the periodic boundary conditions of a torus. The code space is then defined as the ground space of the following Hamiltonian:

$$H = -\sum_v Q_v - \sum_p B_p \,, \tag{1.1}$$

where the sums run over all vertices v and all plaquettes p, and the operators Q_v and B_p are defined as

$$Q_v = \bigotimes_{e\in v} \sigma_z^e = \sigma_z \underset{\sigma_z}{\overset{\sigma_z}{+}} \sigma_z \,,$$

$$B_p = \bigotimes_{e\in p} \sigma_x^e = \sigma_x \underset{\sigma_x}{\overset{\sigma_x}{\square}} \sigma_x \,.$$

For an $L \times L$ square lattice with periodic boundary one has $n = 2L^2$ edges, L^2 vertices and L^2 plaquettes. However, because of the torus topology, not all B_p and Q_v operators are independent: one has $\prod_{p'\neq p} B_{p'} = B_p$, and similarly: $\prod_{v'\neq v} Q_{v'} = Q_v$. The groundspace of Eq. (1.1) must, therefore, satisfy $L^2 - 2$ independent[5] constraints, meaning that it is four-dimensional (thus encoding the state of $k = 2$ logical qubits).

If we think of the qubit states $|0\rangle$ and $|1\rangle$ as "non-occupied" and "occupied" edges, respectively, then the vertex stabilizers Q_v enforce the condition that only an *even* number of occupied edges are allowed to meet in every vertex. The condition $\prod_p B_p |\psi_0\rangle = |\psi_0\rangle$ means that ground states of Eq. (1.1) are superpositions of different configurations of loops. If we denote a classical configuration of loops on the lattice by C (and the corresponding quantum state by $|C\rangle$), then we can define the following functions:

$$L_1(C) = \#(\text{occupied horizontal edges on a single column}) \quad \text{mod } 2 \,,$$
$$L_2(C) = \#(\text{occupied vertical edges on a single row}) \quad \text{mod } 2 \,.$$

We can now associate the logical states $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$ with the following basis of

---

[5]Note that $[Q_v, B_p] = [Q_v, Q_{v'}] = [B_p, B_{p'}] = 0$.
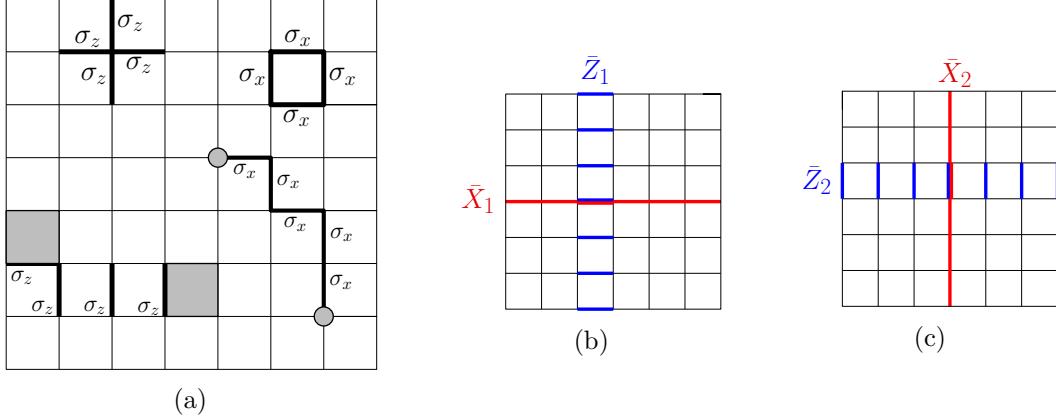
Figure 1.2: (a) Excitations in the toric code. The shaded disks and squares represent violated vertex and plaquette operators, respectively. The (dual) lines connecting each pair represent the string of errors that created them. (b,c) The logical operators on the toric code.

the code space:

$$|\psi_{00}\rangle = \frac{1}{\mathcal{N}} \sum_{C: \substack{L_1(C)=0 \\ L_2(C)=0}} |C\rangle,$$

$$|\psi_{01}\rangle = \frac{1}{\mathcal{N}} \sum_{C: \substack{L_1(C)=0 \\ L_2(C)=1}} |C\rangle,$$

$$|\psi_{10}\rangle = \frac{1}{\mathcal{N}} \sum_{C: \substack{L_1(C)=1 \\ L_2(C)=0}} |C\rangle,$$

$$|\psi_{11}\rangle = \frac{1}{\mathcal{N}} \sum_{C: \substack{L_1(C)=1 \\ L_2(C)=1}} |C\rangle,$$

where $\mathcal{N}$ is a normalization factor. One can easily verify that the logical operator $\bar{Z}_1 \equiv \sigma_z \otimes \mathbb{1}$ corresponds to a *dual* string of σ_z operators winding along the torus vertically, as indicated on Fig. 1.2b. Analogously, the logical operator $\bar{Z}_2 \equiv \mathbb{1} \otimes \sigma_z$ corresponds to a dual string of σ_z operators winding along the torus horizontally, as indicated on Fig. 1.2c. Meanwhile, the logical operators $\bar{X}_1 \equiv \sigma_x \otimes \mathbb{1}$ and $\bar{X}_2 \equiv \mathbb{1} \otimes \sigma_x$ correspond to a string of σ_x operators winding along the torus.

We can now consider what happens when the system is subjected to errors in the form of random Pauli operators. When a string of σ_x errors are applied, this will result in a pair of violated vertex operators Q_v at its endpoints (see Fig. 1.2a). Such pairs of excitations can always be removed by applying an appropriate string of σ_x operators in order to recombine them. The specific correction to remove a pair of excitations not unique. The only requirement is that the correction and error paths are homologically equivalent, i.e., that the joint path of the original string or errors and the correction forms a contractible loop (does not wind around the torus). Indeed, whenever the combined path of error and correction forms a homologically trivial cycle, their combined action is identical to a product of plaquette operators. On the other hand, if their combined path is homologically non-trivial, their combined action results in the application of a logical

operator ($\bar{X}_1$, $\bar{X}_2$ or $\bar{X}_1\bar{X}_2$), which corrupts the encoded information. A similar situation arises for σ_z errors, where a *dual* string of errors results in a pair of violated plaquette operators B_p at its endpoints.

Correcting the encoded information thus amounts to finding the correct paths along which to fuse pairs of excitations, such that the combined path of all errors and correction operators does not percolate around the torus. It can be shown that when the probability p of an error on each physical qubit stays below a certain threshold p_c, a good correction strategy will result in the probability P_{fail} of causing a logical error to decrease exponentially with the system size L. In the optimal case, the error correction threshold is $p_c \approx 0.109$ [27].

1.3 Tensor networks

During the previous two decades, tensor network states [28, 29, 30] have emerged as a vital tool for both the theoretical and numerical study of strongly correlated quantum many-body systems. Their strength originates in the fact that they can be used to describe low-energy states of interacting systems in terms of local entanglement degrees of freedom, which allows for a very efficient representation of these states despite the exponential scaling of the quantum many-body Hilbert space.

Consider a quantum many-body system of N particles on sites $s = 1, \ldots, N$ which all have a local Hilbert space of dimension d. In general, the wave function of such a system can be written as

$$|\psi\rangle = \sum_{i_1, i_2, \ldots, i_N} C^{i_1 i_2 \ldots i_N} |i_1\rangle |i_2\rangle \ldots |i_N\rangle, \tag{1.2}$$

where $\{|i_s\rangle \,|\, i_s = 1, \ldots, d\}$ is a basis for the local Hilbert space of the particle on site s. Since the total Hilbert space has dimension d^N, a generic state requires us to specify d^N complex coefficients $C^{i_1 i_2 \ldots i_N}$. As N increases, describing generic states in the total Hilbert space quickly becomes intractable.

In many cases however, the phyical states of interest contain some additional structure which permits a more efficient representation. In particular, it is believed that low energy eigenstates of gapped Hamiltonians with local interactions obey an area-law scaling of the entanglement entropy [31].

The coefficient $C^{i_1 i_2 \ldots i_N}$ in Eq. (1.2) can be thought of as a tensor with N d-dimensional indices. Since this tensor completely determines the quantum state[6], one can represent the state graphically as follows:

$$|\psi\rangle = \boxed{\qquad C \qquad} . \tag{1.3}$$

Tensor network states are states for which this tensor is decomposed as a contraction of smaller tensors:

$$\boxed{\qquad C \qquad} = \boxed{A^{(1)}} - \boxed{A^{(2)}} \cdots \cdots \boxed{A^{(N)}} . \tag{1.4}$$

The lines connecting the different tensors represent sums over shared indices, which are called *virtual indices*. One can interpret them as representing the entanglement between

---

[6]That is, with a fixed choice for the basis of the local Hilbert spaces.

different parts of the system. The number of possible values for a virtual index is called its *bond dimension.*

For generic states, the bond dimension of these virtual indices will scale exponentially with the system size. For states obeying an area law for the entanglement entropy, however, one can show that the bond dimension of the virtual indices scales at most polynomially with the system size. Hence, this allows for a much more efficient representation of the wave function than the one used in Eq. (1.2). The structure of the decomposition in (1.4) depends on the nature of the state under consideration. One of the most famous class of tensor network states are *matrix product states* (MPS) [32, 33]. This class of tensor network states was conceived to represent ground states of translation invariant quantum spin chains. Their structure is as follows:

$$(1.5)$$

The state in Eq. (1.2) can then be written as

$$|\psi\rangle = \sum_{i_1,i_2,\ldots,i_N} \mathrm{Tr}\left[A^{i_1}A^{i_2}...A^{i_N}\right] |i_1\rangle |i_2\rangle ... |i_N\rangle \,, \tag{1.6}$$

where $\{A^i \,|\, i = 1,\ldots,d\}$ is a set of $D \times D$ matrices.

A similar construction exists for a large class of many-body operators called *matrix product operators* (MPOs):

$$O = \sum_{i_1,\ldots,i_N} \sum_{j_1,\ldots,j_N} \mathrm{Tr}\left[A^{i_1}_{j_1}...A^{i_N}_{j_N}\right] |i_1...i_N\rangle \langle j_1...j_N| \,, \tag{1.7}$$

or, equivalently,

$$(1.8)$$

This class of tensor network states has been generalized to two or more dimensions, yielding *projected entangled pair states* (PEPS) [34]. Similar to MPS states, they consist of tensors that map a set of virtual degrees of freedom to physical degrees of freedom. These tensors are arranged according to the lattice on which the spin system is defined and their virtual indices are then contracted. For a square lattice, the resulting state looks as follows:

$$(1.9)$$

1.4 This dissertation

The biggest theoretical challenge in achieving scalable quantum computation is the construction of more efficient schemes for quantum error correction and fault-tolerance [35, 36]. Topological quantum error correcting codes (QECC), with the most famous representative being the surface code [25, 27, 37, 38, 39], are among the most promising candidates for near-term implementation due to their geometric locality which makes them well suited for practical implementation using state-of-the-art hardware technology such as superconducting or semiconducting qubits, ion traps, silicon photonics, NV centers, cold atoms, just to name a few. Surface codes allow for high quantum error correction thresholds, but have the drawback that one needs a very large overhead of magic states to make the scheme universal for quantum computation [36, 39, 40].

In order to overcome this limitation, one has to consider topological codes which allow for non-Clifford logical gates. One approach in this direction is to consider stabilizer codes in higher dimensions, which allow for non-Clifford transversal gates [41, 42, 43]. Notably, it was discovered recently that one can draw upon the advantages of such higher dimensional models using only a two-dimensional system of qubits by using time to emulate a third dimension [44, 45]. A perpendicular direction is to look beyond the stabilizer formalism and consider non-Abelian codes in 2D which are universal for quantum computation without the need for magic-state distillation [46]. In this dissertation, we follow this second path, as we believe that this is a more natural setting for hardware platforms currently being pursued.

Most conventional topological error correcting codes fall within the framework of the stabilizer formalism [47] and admit quasiparticle excitations which can be characterized as Abelian anyons. However, braiding of these excitations does not allow for a universal gate set. Similarly, double semion topological codes [48, 49], while going beyond the stabilizer code class, exhibit Abelian topological order and are not universal. In order to achieve universal topological quantum computation, a necessary condition is to use systems with a more intricate topological order allowing for non-Abelian anyonic excitations, which have the property that the fusion of two anyons can yield several outcomes [46]. In particular, braiding of Fibonacci anyons can be used to realize a fault-tolerant universal gate set [26]. Several lattice models supporting this non-Abelian topological order have been proposed, such as Kitaev's quantum double models [25] or the string-net models of Levin and Wen [50]. While their ground space is still defined as the simultaneous eigenspace of a set of mutually commuting local check operators, their description falls beyond the stabilizer formalism.

In recent years, there has been significant progress in the study of quantum error correction and decoding for non-Abelian topological codes with a non-stabilizer structure, including numerical estimates of their error thresholds [51, 52, 53, 54, 55]. These works, however, assume the existence of a protected anyonic fusion space and only consider phenomenological noise models on this space, while not specifying a concrete underlying microscopic quantum mechanical spin model suitable for the description of realistic quantum computer implementations.

In this work, we remedy this shortcoming by considering a non-Abelian error correcting code consisting of generic qubits subjected to depolarizing noise. We focus on one of the simplest of those codes, the Turaev-Viro code constructed from the Fibonacci string-net model [50]. Building on the pioneering work of König, Kuperberg and Reichardt [56] and of Bonesteel and DiVincenzo [57], we construct a set of measurements and quantum gates which map arbitrary qubit errors to the Turaev-Viro subspace, and make crucial use of the framework of tensor networks for simulating the error correction process, giving rise to

surprisingly high quantum error correction thresholds. Using a clustering decoder and a fixed-rate sampling noise model, we have obtained a 4.7% threshold for the code subjected to depolarizing noise, and a 7.3% threshold for pure dephasing noise. These numbers are comparable to the code-capacity error threshold of the surface code, which is around 10% for i.i.d. bit-flip or phase-flip noise [27, 58, 59, 83].

Before giving an overview of the content of this dissertation, let us provide a brief review of the history and current status of the Turaev-Viro codes. After Witten [60] and Atiyah's [61] early introduction of the formalism of topological quantum field theory (TQFT) in the 1980s, Turaev and Viro introduced a path integral in terms of a discrete state-sum describing a wide class of 2+1D topological quantum field theories. This led to new quantum invariants of 3-manifold generalizing Witten's quantum invariants in 1992 [62]. Around 1997, Kitaev had the crucial insight that the problem of constructing quantum error correcting codes is effectively equivalent to the one of constructing quantum spin systems providing a Hamiltonian realization of such topological field theories. He introduced a class of quantum double models [25], of which the simplest Abelian version $\mathcal{D}(\mathbb{Z}_2)$ is the well-known toric code. The non-Abelian codes in the quantum double family can be used to implement universal fault-tolerant logical gate sets without magic state distillation. In 2005, Levin and Wen generalized and categorified Kitaev's quantum doubles by introducing string-net models [50], which provide a local Hamiltonian realization of all Turaev-Viro TQFTs. These models capture all non-chiral topological orders (Abelian and non-Abelian) in 2D, including topological orders of the toric code and Kitaev's quantum double model as specific examples. In 2010, König, Kuperberg and Reichardt [56] studied a class of Levin-Wen models with a modular input category from the point of view of error correction and called them *Turaev-Viro codes*. They developed the tube algebra for this modular case and defined a complete basis of the anyonic excitations. In 2012, Bonesteel and DiVincenzo proposed the quantum circuits to measure the vertex and plaquette projectors in the Fibonacci Levin-Wen model [57] and hence made the first step towards practical implementation of Turaev-Viro codes with ordinary qubits by devising an error detection scheme.

Several technical difficulties had to be solved however to turn their error detecting scheme into an error correcting one. First, the original string-net model proposed by Levin and Wen [63] does not admit an easy microscopic description of all types of anyonic excitations in the corresponding topological phases, but only the fluxons (plaquette excitations). As we will see, a single vertex error in this model can bring the system out of the string-net subspace such that the created excitations are no longer anyons as in the case of phenomenological anyon models [51, 54, 55], making error correction and decoding quite challenging. For that purpose, an extended string-net model was defined on a tailed lattice [64, 65] (see also Ref. [66] where tail qubits were introduced for the purpose of incorporating charged and dyonic excitations in topological phases). In the same work [64, 65], a scheme to trap vertex errors and a tadpole swapping scheme to trap plaquette errors were introduced.

In this work, we adopt this tailed-lattice construction and use a similar strategy to trap local vertex errors. We introduce a measurement scheme in terms of tube algebras or *tube operators* [67, 68] whose outcomes contain more syndrome information than the ones reported in Refs. [64] and [65]. This tube algebra enables the definition of an anyonic fusion basis, which can be used to effectively describe the system evolution in the simpler language of an anyon model. We then use the tensor network description of those tube algebras to convert microscopic noise processes such as Pauli errors into anyon-creation processes. The last step needed to calculate error correcting thresholds then consists of simulating the error correcting process.

It is tempting to think that an efficient classical simulation of the error correction process for the Fibonacci Turaev-Viro code is impossible, since braiding Fibonacci anyons is universal for quantum computation. This issue has been addressed in Ref. [55] for a phenomenological Fibonacci anyon model (in which the physical degrees of freedom are anyons as opposed to qubits) and it was demonstrated that it is possible to simulate the error correction threshold with a polynomial complexity. This is because, unlike the quantum computation process where the computational anyons are braided along topologically nontrivial worldlines, the worldlines of noise-created anyons in the error correction process are topologically trivial most of the time. As long as the system is below the percolation threshold corresponding to anyon generation, this classical simulation can be performed in an efficient way. Our work extends the applicability of that result to the case where the physical degrees of freedom are qubits subject to arbitrary noise processes and, hence, allows us to determine error thresholds through classical simulations.

Overview

There are in essence two main achievements detailed in this dissertation. The first is the construction of a non-Abelian topological quantum error correcting code consisting of regular qubits and the design of a complete protocol for error detection and correction in this code. The second is the classical simulation of this error correction procedure using tensor network techniques resulting in an estimate for the associated error correction threshold for a microscopic noise model of Pauli errors.

For the reader's convenience, this dissertation is split up in three parts. The first part consists of one chapter, which provides some technical background on ribbon graphs and on their relation to the Levin-Wen model. Most important equations here, will be repeated in the subsequent parts in order to keep them more or less self contained and improve their readability. Hence, the reader may skip this chapter if they wish, and read it selectively instead when referred back to from later chapters. The second part consists of three chapters which are dedicated to the construction of the error correcting code, the design of a complete error correction scheme including all necessary measurement and recovery cicuits, and the detailed definitions of several decoding schemes. Finally, the third part considers the classical simulation of the error correction process for a depolarizing noise model and presents the numerical results obtained from said simulations.

The flow of this dissertation is as follows:

Chapter 2 introduces the theoretical framework of *ribbon graphs* which will be used extensively throughout this dissertation. We start with a very condensed overview of the necessary category-theoretic definitions and proceed by defining the ribbon graph Hilbert space for an input category $\mathcal{C}$. Largely following Ref. [56], we then discuss its most important properties. By introducing a basis for this Hilbert space, known as the *anyonic fusion basis*, ribbon graph states can be interpreted in terms of anyonic fusion states of the doubled category $\mathcal{DC}$. We define an operator algebra, known as the *tube algebra* [67], whose central idempotents form projectors on the different topological sectors of the doubled category. This tube algebra, together with the interpretation of ribbon graph states as doubled anyonic fusion states, form the guiding principle throughout the remainder of our discussion. We conclude this chapter by illustrating how the Levin-Wen model naturally arises in the context of a lattice realization of the ribbon graph Hilbert space.

Chapter 3 contains a complete and self-contained definition of the extended string-

net code together with a discussion of its most imporant properties. Our starting point is the Fibonacci Levin-Wen string-net model [50] of qubits arranged on a hexagonal lattice, defined from the algebraic data of the Fibonacci unitary fusion category (UFC). We adopt an extension of the string-net model that serves as the basis for an error correcting code. This extension has a twofold motivation. On the one hand, we need a way of correcting violations of the ribbon graph branching rules that can be induced by generic errors at the level of the lattice qubits. Moreover, we require a concise way of characterizing the excitation spectrum in terms of anyonic charges by defining the action of the tube algebra idempotents in the bulk of the lattice model. Both of these requirements can be met by adding an additional "tail edge" to each plaquette, inspired by the constructions introduced in Refs. [64], [65] and [66] for similar reasons.

These considerations then lead to a model of qubits arranged on the edges of a tailed hexagonal lattice on the torus whose fourfold degenerate ground space serves as a topological quantum memory and effectively encodes two logical qubits, and whose excited states can be interpreted as fusion states of doubled Fibonacci anyons. By generalizing the torus setup (genus = 1) to a higher-genus surface, one can scale up the number of logical qubits, which grows approximately linearly with the genus. One can hence perform universal quantum computation via topological operations corresponding to the elements of the mapping class group of the high-genus surface, which can be generated by Dehn twists [26, 56, 69].

After defining the code as the ground space of the extended string-net model, we study its anyonic excitations. We formulate the action of the tube algebra on lattice qubits, and introduce the lattice realization of the anyonic fusion basis of Chapter 2.

Chapter 4 contains detailed definitions of all protocols for error detection and correction. Generic errors on the lattice qubits can cause violations of the string-net (ribbon graph) branching rules. Such a violation can be interpreted as a string ending in a vertex of the lattice, resulting in a qubit state that lies outside of the string-net subspace, which can therefore not be captured as an anyonic fusion state. Circuits for detecting vertex violations were first introduced in Ref. [57]. We define local unitary circuits that can correct an arbitrary combination of vertex errors by pulling the corresponding string end onto the tail edge of the associated plaquette. These vertex correction circuits are similar to those used in Refs. [64] and [65]. After returning the system to the string-net subspace in this way, we define circuits for syndrome extraction, again guided by the concept of the tube algebra. Measuring the idempotents of the tube algebra in each plaquette reveals the location and charge of all anyonic excitations in the system, yielding the error syndrome.

Utilizing the ribbon graph formalism, we naturally arrive at a local unitary circuit which can perform these charge measurements. Equipped with this protocol for syndrome extraction, we are left with the task of recovery, which consists of moving excitations on the lattice and fusing them to back to the anyonic vacuum, thereby returning the system to the code space. Similar to the Abelian case, a logical error occurs when an anyon is wound along a nontrivial cycle of the torus in this process. Building on and extending previous works [56, 64, 65, 69, 70, 71], we design protocols for the necessary recovery operations at the level of the qubits.

Chapter 5 describes the decoding procedure itself, which entails deciding which recovery operations should be carried out for a given error syndrome. We rely on recent advances in error correction for non-Abelian anyon models [51, 52, 55] and tailor the decoders introduced there to our specific code, as well as further design new decoders for our purpose. The main difficulty that arises for the non-Abelian case is the fact that error

correction has to proceed in an iterative fashion because of indeterminacy of fusion outcomes for non-Abelian anyons. Specifically, we adapt a clustering decoder to our setting and further design a fusion-aware iterative matching decoder.

These considerations conclude our discussion of the code definition and associated error correction protocol, giving a complete scheme for error correction in an extended Fibonacci Turaev-Viro code.

In **Chapter 6**, we study the construction of PEPS representations of string-net ground states. We then generalize this construction to obtain an explicit tensor network representation of generic anyonic fusion states that appear as excitations in our model. These PEPS representations will prove essential for the classical simulation of the error correction process, as they enable us to calculate overlaps matrix elements of generic microscopic qubit errors.

In **Chapter 7**, we finally move on to our second main result: the classical simulation of the error correction procedure and the estimate of the error correction threshold. After some general comments on the classical simulation of non-Abelian quantum error correction, we define a fixed-rate sampling model of depolarizing noise. The main problem to be tackled here is the question of how to determine what distribution of anyonic excitations is generated by Pauli errors acting at the level of the lattice qubits. The complex description of the excited anyonic fusion states in terms of qubit states makes this a highly nontrivial task however. By using the PEPS representations of these states, constructed in Chapter 6, we quantitatively analyze the effect of Pauli noise on the code, which effectively allows to translate a physical error rate at the level of the qubits to an anyon generation rate. We can hence simplify the classical simulation of the decoding problem to the simulation of noise-driven dynamics of anyonic fusion states, which is infinitely more feasible than directly simulating the full microscopic model itself in the qubit basis. Having overcome this main difficulty, we adapt recently developed techniques for simulating non-Abelian error correction [55] to the hexagonal geometry and doubled Fibonacci excitations relevant to our model. We conclude this chapter by giving a complete overview of all necessary steps in the classical simulation of error correction in our code.

Chapter 8 contains the result obtained using the Monte Carlo simulations detailed in Chapter 7 for the decoders introduced in Chapter 5. In all cases a very distinct threshold behavior is visible. We then briefly compare these result to those of the surface code with similar assumptions. To the best of our knowledge, our results provide the first threshold estimate for a two-dimensional error correcting code of qubits which is universal for topological quantum computation, without the need for additional non-topological operations.

Finally, in **Chapter 9**, we summarize the results and discuss some possible future research directions. These include the adaptations of our error correction protocols to a full fault-tolerant scheme and investigating planar string-net codes with suitable boundary conditions.

Part I

Technical background

2 | Ribbon graphs

The extended string-net model which we will define in Chapter 3 is best introduced in the context of topological quantum field theory. In particular, it can be understood as a local Hamiltonian realization of the Turaev-Viro TQFT. Hence, the associated error correcting code should be thought of as a microscopic realization of a Turaev-Viro code [56] on a system of qudits. In this chapter, we discuss the ribbon graph Hilbert space originating from the Turaev Viro TQFT. We introduce a basis of this Hilbert space which allows us to interpret ribbon graph configurations in terms of anyonic fusion states. We then discuss the tube algebra which emerges naturally in this context. We conclude by illustrating how the Levin-Wen string-net model emerges naturally when constructing a lattice realization of the ribbon graph Hilbart space.

2.1   Category theory primer

The mathematical underpinning op topological order and anyon models, is category theory. Category theory in itself is a broad field of mathematics, which has been studied intensively for several decades, For the purpose of this work however, we do not need the full mathematical machinery of category theory. Instead, we will give a condensed overview of the algebraic data of unitary fusion categories and unitary modular tensor categories, which are needed for defining our model in the remainder of this section.

A *unitary fusion category* (UFC) $\mathcal{C}$ is defined by a finite set of simple objects

$$\{a_1, a_2, \ldots a_N\}, \tag{2.1}$$

and a collection of algebraic data for this set. The core of this data is formed by the fusion algebra

$$a \times b = \sum_c N_{ab}^c c, \tag{2.2}$$

where $N_{ab}^c \in \mathbb{N}$ are called the fusion coefficients. The fusion algebra satisfies the following associativity condition:

$$\sum_e N_{ab}^e N_{ec}^d = \sum_f N_{bc}^f N_{af}^d. \tag{2.3}$$

We define

$$\delta_{ab}^c = \begin{cases} 0 & \text{if } N_{ab}^c = 0, \\ 1 & \text{otherwise.} \end{cases} \tag{2.4}$$

Among the simple objects, there must be a unique unit element $\mathbf{1}$, satisfying $N_{1a}^b = N_{a1}^b = \delta_{a,b}$ for all $a, b \in \mathcal{C}$. For each $a \in \mathcal{C}$, there is a unique element $a^* \in \mathcal{C}$ satisfying $(a^*)^* = a$, and $N_{ab}^{\mathbf{1}} = N_{ba}^{\mathbf{1}} = \delta_{a^*,b}$.

A UFC associates to every fusion (resp. splitting) vertex, an N_{ab}^c-dimensional vector space V_{ab}^c (resp. V_c^{ab}) over $\mathbb{C}$. For simplicity, we will restrict to the multiplicity-free case, i.e.: $N_{ab}^c = \delta_{ab}^c$.

The fusion associativity condition (2.3) then implies that the fusion or splitting vector spaces corresponding to different orderings of fusion or splitting, are isomorphic:

$$\bigoplus_e V_{ab}^e \otimes V_{ec}^d \simeq \bigoplus_f V_{bc}^f \otimes V_{af}^d. \tag{2.5}$$

The linear map between those two vector spaces is given by a 6-index object called the F-symbol:

$$\tag{2.6}$$

The F-symbol is only defined for allowed fusion vertices. We use the convention that it is zero outside this subspace:

$$F_{cdf}^{abe} = \delta_{abe}\delta_{e^*cd}\delta_{adf}\delta_{bcf^*}F_{cdf}^{abe}, \tag{2.7}$$

where $\delta_{abc} = \delta_{ab}^{c^*}$. The F-symbol must satisfy the following consistency condition, called the *pentagon equation*:

$$F_{e^*dl^*}^{cfg^*}F_{e^*lk^*}^{baf^*} = \sum_h F_{g^*ch^*}^{baf^*}F_{e^*dk^*}^{hag^*}F_{k^*dl^*}^{cbh^*}. \tag{2.8}$$

Each simple object $a \in \mathcal{C}$, we define a constant $d_a \in \mathbb{R}$ called the *quantum dimension*:

$$d_a = \frac{1}{|F_{aa^*\mathbf{1}}^{aa^*\mathbf{1}}|} \, . \tag{2.9}$$

The quantum dimensions satisfy

$$d_a d_b = \sum_k N_{ab}^c d_c \, , \qquad d_\mathbf{1} = 1 \, , \quad \text{and} \quad d_{a^*} = d_a \, . \tag{2.10}$$

The *total quantum dimension* $\mathcal{D}$ is defined as

$$\mathcal{D} = \sqrt{\sum_a d_a^2} \, . \tag{2.11}$$

Finally, the F-symbol, when viewed as a matrix $[F_{cd}^{ab}]_{fe} = F_{cdf}^{abe}$, must be unitary on the subspace on which it is defined:

$$[(F_{cd}^{ab})^{-1}]_{ef} = [(F_{cd}^{ab})^\dagger]_{ef} = \left([F_{cd}^{ab}]_{fe}\right)^* \, , \tag{2.12}$$

$$\sum_f \left(F_{cdf}^{abe'}\right)^* F_{cdf}^{abe} = \delta_{e,e'} \delta_{abe} \delta_{e^*cd} \, .$$

A *unitary ribbon category* (URC) is obtained by adding the notion of braiding and twists to a UFC. Braiding is a unitary operation between fusion spaces that corresponds to a counterclockwise exchange:

$$R^{ab} : \ V_{ab}^c \to V_{ba}^c \ : \ \includegraphics \mapsto \includegraphics = R_c^{ab} \includegraphics \, . \tag{2.13}$$

Note that since we are working in the multiplicity-free case, R_c^{ab} is simply a complex phase. It's inverse corresponds to a clockwise exchange:

$$(R^{ba})^{-1} : \ V_{ab}^c \to V_{ba}^c \ : \ \includegraphics \mapsto \includegraphics = \left(R_c^{ba}\right)^* \includegraphics \, . \tag{2.14}$$

Similar to the pentagon equation (2.8) for the F-symbol, the R-symbol must obey consistency conditions called the *hexagon equations*:

$$R_e^{ca} F_{db^*g}^{c^*a^*e} R_g^{cb} = \sum_f F_{db^*f}^{a^*c^*e} R_d^{cf} F_{dc^*g}^{b^*a^*f} \, , \tag{2.15}$$

$$(R_e^{ac})^* F_{db^*g}^{c^*a^*e} (R_g^{bc})^* = \sum_f F_{db^*f}^{a^*c^*e} (R_d^{fc})^* F_{dc^*g}^{b^*a^*f} \, . \tag{2.16}$$

In a URC, every simple object a is assigned a *topological phase* θ_a, used to "untwist" a ribbon:

$$\includegraphics = \theta_a \includegraphics \, , \qquad \includegraphics = (\theta_a)^* \includegraphics \, . \tag{2.17}$$

The topological phase is related to the R-symbol as follows:

$$\theta_a = \left(R_{\mathbf{1}}^{aa^*} \right)^* . \tag{2.18}$$

A *Unitary Modular Tensor Category*, is a unitary ribbon category for which the R-matrices satisfy a certain nondegeneracy condition. We will not need the technical definition of modularity. For more details, we refer the reader to Ref. [72].

2.1.1   The Fibonacci category

Throughout this paper, we will place a special focus on the Fibonacci category. The Fibonacci category contains two simple objects:

$$\{\mathbf{1}, \tau\}, \tag{2.19}$$

with only one nontrivial fusion rule

$$\tau \times \tau = \mathbf{1} + \tau . \tag{2.20}$$

The pentagon equation (2.8) with these fusion rules only has one solution that satisfies the unitarity condition (2.12). The only nontrivial F-matrix is

$$[F_{\tau\tau}^{\tau\tau}] = \begin{pmatrix} \phi^{-1} & \phi^{-\frac{1}{2}} \\ \phi^{-\frac{1}{2}} & -\phi^{-1} \end{pmatrix}, \tag{2.21}$$

where $\phi = \frac{\sqrt{5}+1}{2}$ is the golden ratio. For all other combinations of indices, F_{kln}^{ijm} is either 1 or 0, depending on whether or not the corresponding indices in Eq. (2.7) satisfy the branching rules.

The quantum dimensions are

$$d_{\mathbf{1}} = 1 , \qquad d_{\tau} = \phi , \tag{2.22}$$

and the total quantum dimension is then given by

$$\mathcal{D} = \sqrt{1 + \phi^2} = \sqrt{2 + \phi} = \sqrt{\sqrt{5}\phi} . \tag{2.23}$$

The hexagon equation admits two solutions, which are related by complex conjugation. We will use the following:

$$R_{\mathbf{1}}^{\tau\tau} = e^{\frac{4\pi i}{5}} , \qquad R_{\tau}^{\tau\tau} = e^{\frac{-3\pi i}{5}} , \tag{2.24}$$

the other R-symbols allowed by the fusion rules are $R_a^{\mathbf{1}a} = R_a^{a\mathbf{1}} = 1$ for any a.

Finally, the modular $\mathcal{S}$-matrix is

$$\mathcal{S} = \frac{1}{\sqrt{2 + \phi}} \begin{pmatrix} 1 & \phi \\ \phi & 1 \end{pmatrix} . \tag{2.25}$$

2.2   The ribbon graph Hilbert space

The *ribbon graph Hilbert space* is the Hilbert space associated to a surface by the Turaev-Viro TQFT defined by a unitary fusion category $\mathcal{C}$ [56]. Even though it is defined for any unitary fusion category, we consider the special case where the input category $\mathcal{C}$ is a

unitary modular tensor category. Furthermore, in addition to the conditions (2.7), (2.8) and (2.12) that are satisfied for every UMTC, we impose the following three conditions for the F-symbols:

$$F_{kln}^{ijm} = F_{lkn^*}^{jim} = F_{jin}^{lkm^*} = F_{k^*nl}^{imj} \frac{v_m v_n}{v_j v_l} , \qquad (2.26)$$

$$F_{j^*jk}^{ii^*1} = \frac{v_k}{v_i v_j} \delta_{ijk} , \qquad (2.27)$$

$$\left(F_{kln}^{ijm} \right)^* = F_{jkm^*}^{lin} . \qquad (2.28)$$

The first of these conditions is known as tetrahedral symmetry, the second one is a normalization condition and the third condition is an alternative formulation of the unitarity condition Eq. (2.12).

Let Σ be a compact orientable surface with a boundary, where we place a single marked point on each connected component of the boundary. A ribbon graph is a labeled, directed graph embedded into Σ, with internal vertices of degree two and three, and a vertex of degree one at each marked boundary point of Σ. The ribbons are labeled with the labels of $\mathcal{C}$. Reversing the direction of an edge in the ribbon graph corresponds to conjugating the edge label. Edges assigned the trivial label 1 can be added to or removed from a given ribbon graph, as the vacuum label denotes the absence of a ribbon. We require that each internal vertex satisfies the fusion rules, i.e.: that $\delta_{ijk} = 1$ for every vertex with incoming lines i, j and k.

The *ribbon graph Hilbert space* $\mathcal{H}_\Sigma$ is the space of formal linear combinations of ribbon graphs, modulo the following relations:

$$(2.29)$$

$$= \delta_{j1} d_i , \qquad (2.30)$$

$$= \sum_n F_{kln}^{ijm} \quad \qquad (2.31)$$

A convenient relation, known as the *bubble bursting* equation, follows from Eqs. (2.31) and Eq. (2.27):

$$= \frac{v_i v_j}{v_k} \delta_{kk'} \qquad (2.32)$$

Another convenient relation that follows from Eq. (2.27) is

$$= \sum_k \frac{v_k}{v_i v_j} \delta_{ijk^*} \qquad (2.33)$$

From here on, in order to simplify the notation and the graphical representations, we will assume that $\mathcal{C}$ is self-dual, meaning $i = i^*$ for all labels $i \in \mathcal{C}$. This allows us to drop

the orientations in the ribbon diagrams. The generalization to non-self-dual categories is straight-forward. A much more elaborate treatment of the ribbon graph Hilbert space, including the non-self-dual case, can be found in Ref. [56]. However, for our purpose, this simplified and condensed summary is sufficient.

We now consider the case where Σ is the n-punctured sphere, $\Sigma_n = S^2 \setminus (A_1 \cup \cdots \cup A_n)$, where each A_i represents a disk. To each hole we assign a single marked boundary point $p \in \partial A_i$. A labeling ℓ of Σ_n associates to each marked boundary point p a label $\ell(p) \in \mathcal{C}$. The ribbon graph Hilbert space $\mathcal{H}_{\Sigma_n}$ can then be decomposed as

$$\mathcal{H}_{\Sigma_n} = \bigoplus_\ell \mathcal{H}^\ell_{\Sigma_n} \,, \tag{2.34}$$

where $\mathcal{H}^\ell_{\Sigma_n}$ is the subspace spanned by ribbon graphs with an edge labeled by $\ell(p)$ connected to p, for every marked boundary point p.

We can define a basis for $\mathcal{H}_{\Sigma_n}$ based on a *pants decomposition* of the surface Σ_n. A pants decomposition is associated to a rooted binary tree with $n-1$ leaves, where the root and leaves each correspond to a hole in Σ_n. Its internal vertices of degree three correspond to *pants segments*, each isomorphic with Σ_3, while its edges correspond to *cylindrical segments*, each isomorphic with Σ_2. An example of a pants decomposition of Σ_4 is depicted in Fig. 2.1. For a given pants decomposition, a basis for $\mathcal{H}_{\Sigma_n}$ is obtained
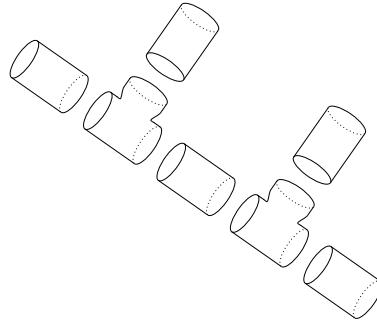
Figure 2.1: A standard pants decomposition for the sphere with four punctures Σ_4.

by fixing the ribbon graph on each leaf to

$$\tag{2.35}$$

on each internal segment and on the root to

$$\tag{2.36}$$

and finally connecting the ribbons in each pant segment. The set of all allowed label assignments of this ribbon graph, constitutes a basis for $\mathcal{H}_{\Sigma_n}$. An inner product on $\mathcal{H}_\Sigma$ can now be defined by setting these basis states to be orthonormal. This basis is called a *computational basis*, since it provides a way of encoding $\mathcal{H}_{\Sigma_n}$ into $5n-6$ qudits. A

more general construction, based on specific triangulations of Σ, exists and gives rise to an entire family of computational bases for $\mathcal{H}_{\Sigma_n}$ [56]. These different computational bases can be related using the equivalence rules (2.29), (2.30) and (2.31), ensuring that the inner product they define is unique.

2.3 Dehn twists and braid moves on $\mathcal{H}_\Sigma$

Consider the set $Q \in \Sigma$ of all marked boundary points. Diffeomorphisms $f : \Sigma \to \Sigma$ that map the Q onto itself, define an action on $\mathcal{H}_\Sigma$. Two special types of such transformations, known is Dehn twists and braid moves, are of particular interests to us, because of their relation with the topological properties of anyon models (i.e.: the R-matrices and topological phases appearing in Sec. 2.1.

A *Dehn twist* corresponds to a 2π-counterclockwise twist along a simple closed curve on Σ_n. If we consider the non-contractible curve γ,

$$
\tag{2.37}
$$

then a Dehn twist $D(\gamma)$ along this curve acts as

$$
D(\gamma) : \qquad \mapsto \qquad , \tag{2.38}
$$

or, schematically

$$
D(\gamma) : \qquad \mapsto \qquad .
$$
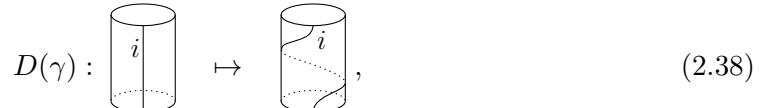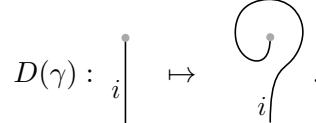
A *braid move* acts on two holes of a pair of pants in Σ_n and is defined as a π-counterclockwise twist along a simple closed curve on Σ_n enclosing the two holes, followed by a π-twist in the opposite direction on each of the legs corresponding to the two holes,

$$
\mapsto \qquad , \tag{2.39}
$$
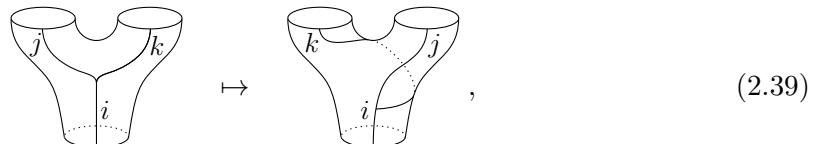
or, schematically

$$
\mapsto \qquad .
$$
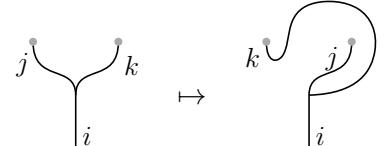
The action of both Dehn twists and braid moves on $\mathcal{H}_\Sigma$ follows by linearly extending their action on the computational basis ribbon graphs to the full Hilbert space.

2.4 The anyonic fusion basis

States described using the computational basis constructed above, do not transform cleanly under the action of the mapping class group generators, making them inconvenient operationally. Below, we will define an orthonormal basis that simultaneously diagonalizes

all Dehn twists and behaves nicely under braid moves. While the construction itself is tedious, this basis will reveal much more of the mathematical structure of $\mathcal{H}_\Sigma$, and will prove invaluable for the rest of this work. In particular, this construction will reveal the relation between states in $\mathcal{H}_\Sigma$ and the Drinfeld center or categorical double $\mathcal{DC}$ of the input category $\mathcal{C}$.

It is important to note that the categorical double of a unitary fusion category is always braided, meaning that one does not need to specify braiding properties for the input category in order to have emergent anyon statistics. In the special case we are considering however, that where the input category $\mathcal{C}$ is itself modular, the Drinfeld center has a special structure. For a modular input category $\mathcal{C}$ the categorical double $\mathcal{DC}$ is isomorphic to the direct product of $\mathcal{C}$ and its reverse category $\bar{\mathcal{C}}$: $\mathcal{DC} \cong \mathcal{C} \otimes \bar{\mathcal{C}}$ [73]. The reverse category $\bar{\mathcal{C}}$ is obtained from $\mathcal{C}$ as follows: for each label $a \in \mathcal{C}$, there is a corresponding label $\bar{a} \in \bar{C}$ which is the same as a, except for its chirality, which is opposite: $\theta_{\bar{a}} = (\theta_a)^*$. Furthermore the R-matrix is replaced by $R_{\bar{a}}^{\bar{b}\bar{c}} = (R_a^{cb})^*$, which one can interpret as swapping the meaning of clockwise and counterclockwise. The doubled category $\mathcal{DC} \cong \mathcal{C} \otimes \bar{\mathcal{C}}$ then has elements $\{a \otimes \bar{b} \mid a \in \mathcal{C}, \bar{b} \in \bar{\mathcal{C}}\}$, which we will denote as $a\bar{b} \equiv a \otimes \bar{b}$, fusion rules

$$\delta_{a\bar{a}'b\bar{b}'c\bar{c}'} = \delta_{abc}\,\delta_{\bar{a}'\bar{b}'\bar{c}'} = \delta_{abc}\,\delta_{a'b'c'}\,, \tag{2.40}$$

and numerical data

$$\theta_{a\bar{a}'} = \theta_a \theta_{\bar{a}'} = \theta_a (\theta_{a'})^*\,, \tag{2.41}$$

$$R_{a\bar{a}'}^{b\bar{b}'c\bar{c}'} = R_a^{bc} R_{\bar{a}'}^{\bar{b}'\bar{c}'} = R_a^{bc} (R_{a'}^{c'b'})^*\,, \tag{2.42}$$

$$F_{c\bar{c}'d\bar{d}'f\bar{f}'}^{a\bar{a}'b\bar{b}'e\bar{e}'} = F_{cdf}^{abe} F_{c'd'f'}^{a'b'e'}\,. \tag{2.43}$$

In the following, we will exploit this structure, in order to make the connection between $\mathcal{H}_\Sigma$ and the Drinfeld center of $\mathcal{C}$ explicit.

A *vacuum line*, denoted by a dashed line, is defined as a weighted superposition of ribbons with different labels,

$$\left.\vdots\right. = \frac{1}{\mathcal{D}} \sum_i d_i \left.\right|_i\,. \tag{2.44}$$

On can easily show that vacuum loops have the property that all other ribbons can freely pass over them:

$$\tag{2.45}$$

where the hashed area inside the vacuum loop denotes a generic ribbon graph configuration. This represents a local identity, where the ribbon graphs inside the hashed area and outside the diagram are assumed to be the same for the left- and right-hand side. Intuitively, vacuum loops render the region they contain invisible from ribbon graph configurations outside. Another useful identity is

$$\tag{2.46}$$

which means we can pull one vacuum loop out of another.

Fix a pants decomposition for Σ_n associated to some rooted binary tree T, and fix a labeling ℓ of the marked boundary points like in Eq. (2.34). A labeling of T assigns an anyon label from the input category $\mathcal{C}$ to every edge of T. Such a labeling of T is said to be fusion-consistent if the fusion rules of $\mathcal{C}$ are satisfied at every internal vertex of T. A pair of labelings of T is said to be boundary-consistent with ℓ if for each marked boundary point p, the labels a and b assigned to the corresponding leaf of T by the respective labelings of T satisfy

$$\delta_{ab\,\ell(p)} = 1. \tag{2.47}$$

A pair of fusion-consistent labelings d of T that is boundary-consistent with ℓ is called a *ℓ-consistent doubled anyonic fusion diagram.*

The *anyonic fusion basis* for $\mathcal{H}^\ell_{\Sigma_n}$ is then an orthonormal basis indexed by ℓ-consistent doubled anyonic fusion diagrams. We still haven't defined the ribbon graph states $|\ell, d\rangle \in \mathcal{H}^\ell_{\Sigma_n}$, that correspond to the ℓ-consistent doubled anyonic fusion diagram d. We will do this by first constructing a three-dimensional ribbon graph on the thickened surface $\Sigma_n \times [-1, 1]$, and then reducing it to a regular ribbon graph on Σ_n.

Think of each of the labelings in d as a ribbon graph, and embed these in the two-dimensional slices $\Sigma_n \times \{1\}$ and $\Sigma_n \times \{-1\}$ of the thickened surface, respectively. Place the marked boundary points of Σ_n in $\Sigma_n \times \{0\}$, and add a vacuum loop in $\Sigma_n \times \{0\}$ around each of the boundary components. Then close off the diagram by adding a ribbon graph edge in $\Sigma_n \times \{0\}$ labeled by $\ell(p)$ connected to every boundary point p, and connect the corresponding edges of the graphs in $\Sigma_n \times \{1\}$ and $\Sigma_n \times \{-1\}$ to this new edge, *inside* the vacuum loop around p,

where the vertical direction represents the additional coordinate. Boundary consistency of d ensures that the vertex created by this closing off of the diagram is a valid ribbon graph vertex. Finally, in order to reduce the resulting diagram to a state in $\mathcal{H}^\ell_{\Sigma_n}$, visualize it as a two-dimensional ribbon graph with crossings in Σ_n (this requires slightly offsetting $\Sigma_n \times \{1\}$ and $\Sigma_n \times \{-1\}$, the convention for the offset only affects the phase of $|\ell, d\rangle$). These crossings can be resolved using the R-symbol of the input category by combining (2.13), (2.31) and (2.27) into

The state $|\ell, d\rangle \in \mathcal{H}^\ell_{\Sigma_n}$ is defined as the resulting superposition of ribbon graphs in Σ_n that is obtained by resolving all crossings in this way. The three-dimensional ribbon graphs introduced here can be manipulated in the same way as regular ribbon graphs in Σ_n, and it is sometimes useful to perform manipulations in the three-dimensional picture before reducing the ribbon graph back to two dimensions. It was shown in Appendix A of Ref. [56] that the anyonic fusion basis is a complete orthonormal basis for $\mathcal{H}^\ell_{\Sigma_n}$. A basis for the entire Hilbert space $\mathcal{H}_{\Sigma_n}$ is then obtained by taking the union of the bases for all values of ℓ according to (2.34)

The anyonic fusion basis states of $\mathcal{H}_{\Sigma_2}$ take the form

$$|k, \ell, a, b\rangle = \quad \equiv \quad . \tag{2.50}$$

Reducing this to a two-dimensional ribbon graph yields

$$= \frac{1}{\mathcal{D}} v_a v_b \sum_{\alpha,\beta} v_\alpha v_\beta \sum_{\gamma,\delta} d_\gamma d_\delta R_\gamma^{a\alpha} R_\delta^{\alpha b} G_{\alpha a b}^{k\delta\gamma} G_{b\alpha\ell}^{\beta a\delta} G_{a\gamma\delta}^{k\beta\alpha} \quad , \tag{2.51}$$

where the six-index G-symbol is defined as

$$= v_\lambda v_\mu v_\nu G_{\lambda\mu\nu}^{ijk} \quad . \tag{2.52}$$

From Eqs. (2.26), (2.28), (2.31) and (2.32), it then follows that, for an input category describing a self-dual anyon model,

$$G_{\lambda\mu\nu}^{ijk} = \frac{1}{v_k v_\nu} F_{\lambda\mu\nu}^{ijk} . \tag{2.53}$$

For the general case of $\mathcal{H}_{\Sigma_n}$ with a standard pants decomposition we get anyonic fusion basis states of the form

$$|\vec{\ell}, \vec{a}, \vec{b}, \vec{c}, \vec{d}\rangle = \quad . \tag{2.54}$$

Note that the doubled ribbons were crossed before attaching them to the root ℓ_n. This is a matter of convention, and was done to give it the same geometry as the other leaves in the pants decomposition. The vacuum loop around the root puncture of the pants decomposition is unnecessary in principle, but was added here to further symmetrize the expression between all punctures[1]. These choices are particularly convenient for the case of self-dual input categories that we are considering here.

---

[1] The inclusion is this additional vacuum loop should be accompanied by a factor $1/D$ in order to ensure proper normalization. We choose not to write it explicitly to avoid additional clutter. Unless mentioned otherwise, proper normalization is always assumed for anyonic fusion basis states.

Reducing basis state (2.54) to a two-dimensional ribbon graph, is done by first using Eqs. (2.33) and (2.14) to reduce all doubled interior lines to single lines, which gives

$$|\vec{\ell}, \vec{a}, \vec{b}, \vec{c}, \vec{d}\rangle = \sum_{\vec{k}, \vec{l}} \left(\prod_{x=1}^{n} \frac{v_{k_x}}{v_{a_x} v_{b_x}} \right) \left(\prod_{y=1}^{n-3} \frac{v_{l_y}}{v_{c_y} v_{d_y}} \right) \left(R_{k_n}^{b_n a_n} \right)^* \tag{2.55}$$

The leaf segments can then be reduced as in (2.51), while the pants segments in (2.55) can be reduced as follows

$$= v_a v_b v_c v_d v_e v_f \sum_{\gamma, \delta} d_\gamma d_\delta R_\gamma^{ad} G_{def}^{kb\delta} G_{dae}^{c\delta\gamma} G_{\gamma dc}^{j\delta a} G_{\delta ba}^{ijk} \tag{2.56}$$

Combining all of this, one finds

$$|\vec{\ell}, \vec{a}, \vec{b}, \vec{c}, \vec{d}\rangle = \sum_{\vec{\alpha}, \vec{\beta}, \vec{k}} X_{\vec{\alpha}, \vec{\beta}, \vec{k}, \vec{l}}^{\vec{a}, \vec{b}, \vec{c}, \vec{d}, \vec{\ell}} \tag{2.57}$$

where the coefficients $X_{\vec{\alpha}, \vec{\beta}, \vec{k}, \vec{l}}^{(\vec{a}, \vec{b}, \vec{c}, \vec{d}, \vec{\ell})}$ follow from Eqs. (2.55), (2.51) and (2.56).

The basis states of the anyonic fusion basis for a given pants decomposition, can be interpreted as fusion states of anyons from the doubled category $\mathcal{DC}$. Indeed, one can easily verify that the topological phases and the R-matrix match with (2.41) and (2.68), respectively, by computing the action of a Dehn twist and of a braid move. Furthermore the bases corresponding to different pants decompositions are related by the correct doubled F-symbol (2.43). These calculations are performed in detail in Ref. [56]. For completeness, we include them (for the self-dual case) below.

2.4.1 The action of Dehn twists, braiding, and recoupling on the anyonic fusion basis

Consider a cylindrical segment in the pants decomposition (for which we have constructed the anyonic fusion basis), with labels a and b assigned to it by an anyonic fusion basis state

$|\ell, d\rangle$. The three-dimensional ribbon graph in $\Sigma_2 \times [-1, 1]$ corresponding to this cylindrical segment, can be represented as the thickened disk

$$(2.58)$$

where the shaded region denotes the inner component of $(\partial\Sigma_2) \times [-1, 1]$. Note that we have applied (2.45) in order to pull a vacuum loop from the shaded region. This vacuum loop can be thought of as originating at one of the boundary components inside the shaded region, using (2.46). For simplicity we will stop drawing the outer boundary of the cylindrical segment. A Dehn-twist along a non-contractible curve γ around the cylinder acts on this three-dimensional ribbon graph in exactly the same way as it would on a regular ribbon graph (2.38), giving

$$(2.59)$$

where the resulting state was simplified by moving the ribbons to $\Sigma_n \times \{0\}$ and pulling them through the vacuum loop, and using (2.17) to remove the resulting kinks. The above identity shows that the anyonic fusion basis diagonalizes Dehn twists, and that the corresponding eigenvalues are the topological spins of the doubled anyons given in Eq. (2.41).

Now consider a pants segment, with corresponding labels a, b and c, and a', b' and c' in $|\ell, d\rangle$. The resulting three-dimensional ribbon graph is

$$(2.60)$$

where we have again pulled a vacuum loop from one of the shaded regions. A braid move on the two interior holes acts in the same way it would on a regular ribbon graph (2.39), giving

$$(2.61)$$

where we have again pulled the upper and lower ribbons through the vacuum loop, and have used (2.13) and (2.14) to remove the resulting kinks. A quick glance at (2.42) confirms that the is again consistent with the interpretation of fusion basis state as fusion states of doubled anyons.

Finally we investigate the relation between anyonic fusion basis states associated with different pants decompositions of Σ_n. For this we can simply use Eq. (2.31) on both the top and the bottom ribbon graph independently in the three-dimensional picture. For Σ_4 this gives the relation

$$
\begin{array}{c}\end{array}
= \sum_{f,f'} F^{abe}_{cdf} F^{a'b'e'}_{c'd'f'} \qquad\qquad , \qquad (2.62)
$$

which is indeed the correct F-matrix for of the doubled theory $\mathcal{DC}$ given in Eq. (2.43).

2.4.2 Higher-genus surfaces

The definition of the anyonic fusion basis can be extended to surfaces of a higher genus. Consider a surface Π_n with genus g, and n punctures. We start by noting that Π_n homeomorphic to the $(n+g)$-punctured sphere Σ_{n+g} with a handle (a punctured torus) glued g punctures. This is known as a handle decomposition.

As before, we start by fixing a pants decomposition of Σ_{n+g} corresponding to a rooted binary tree T of our choosing. Anyonic fusion basis states of $\mathcal{H}_{\Pi_n}$ are then labeled with a labeling ℓ of the n marked boundary points, a doubled anyonic fusion diagram d of $n + g$ anyons (including the root), and a set $\jmath$ of g doubled anyon labels which we will call the *handle labels*. As before, the doubled anyonic fusion diagram d must be ℓ-consistent. In addition to this, it must also be consistent with the handle labels $\jmath$, by which we mean that for a handle $h \in \{1, \ldots, g\}$ with handle label $\jmath(h) = b_+\overline{b_-}$, the doubled anyonic label $a_+\overline{a_-}$ of the corresponding leaf of T must satisfy $\delta_{a_+b_+b_+} = \delta_{a_-b_-b_-} = 1$

The state $|\ell, d, \jmath\rangle \; \mathcal{H}_{\Pi_n}$ is then constructed similarly as the anyonic fusion basis states on the punctured sphere. Start by constructing the three-dimensional ribbon graph corresponding to the doubled fusion diagram d on $\Sigma_{n+g} \times [-1, 1]$. In the n regular punctures, the diagram is closed off according to the boundary labeling ℓ using (2.48). In the remaining g punctures, the diagram is closed off using one of the following ribbon configurations:

$$
\begin{array}{c}\end{array}
\qquad , \qquad
\begin{array}{c}\end{array}
\qquad , \qquad (2.63)
$$

where a_+ and a_- are the labels of the corresponding edge in the doubled fusion diagram, and b_+ and b_- are the handle labels of the corresponding handle h.

Finally, this three-dimensional ribbon graph in $\Pi_n \times [-1, 1]$ is reduced to a regular ribbon graph in Π_n as before, using Eq. (2.49) to remove any crossings. The two possible

choices for the handle configuration in (2.63) result in two nonequivalent bases. The unitary operator relating them can be found in Ref. [56]. For $a_+ = a_- = \mathbf{1}$, it is given by the S-matrix of the doubled category $\mathcal{C} \otimes \bar{\mathcal{C}}$. Note that Eqs. (2.59) and (2.61) are still satisfied, meaning that the resulting ribbon graph states indeed transform appropriately under the action of the mapping class group.

On a torus, the anyonic fusion basis states (with the first handle choice and $\jmath = e\bar{f}$) take the following form:

$$|\vec{\ell}, \vec{a}, \vec{b}, \vec{c}, \vec{d}, e, f\rangle = \qquad\qquad\qquad\qquad\qquad\qquad , \qquad (2.64)$$

where the gray rectangle represents the periodic boundary conditions of a torus.

The subspace of $\mathcal{H}_{\Pi_n}$ where all punctures have the doubled anyon label $1\bar{1}$ is called the *anyonic vacuum*, and is isomorphic to $\mathcal{H}_{\Pi_{n=0}}$. It follows from (2.63), that the anyonic vacuum on a surface with nonzero genus is degenerate. For a torus, this degeneracy is precisely the number of anyon labels in $\mathcal{C} \otimes \bar{\mathcal{C}}$, as can be deduced from (2.64). The vacuum subspace on a torus is spanned by the different possible handle labels. These states are locally indistinguishable since the loops around the handle can be continuously deformed in an arbitrary way without changing the resulting state.

Note that in the case where the doubled anyons associated to the punctures of the torus have a trivial total charge (corresponding to $c_{n-2} = d_{n-2} = \mathbf{1}$ above), the handle labels in the corresponding anyonic fusion states have no influence on braiding operations on these anyons, since the loops around the handle (the lines labeled with e and f above) can always be pulled through a group of anyons with a trivial total charge without changing the state. In practice, when working with states with a trivial total charge, one can largely ignore the precise value of the handle loop labels, since they can not be affected by local operations on $\mathcal{H}_\Pi$. The only operations that can modify the handle labels are those involving paths that are homotopically nonequivalent to the ribbons connected to the punctures. An example of such an operation is a clockwise exchange of two punctures along a path that is homotopically nonequivalent to the ribbons connecting these punctures in the anyonic fusion basis (e.g.: when using the basis in (2.64), exchanging the first two punctures along a path that crosses the vertical gray boundary). Subspaces with fixed handle loop labels $\jmath = e\bar{f}$, are not invariant under such operators. Indeed, describing the outcome of this exchange (in the original basis), will yield a superposition in the loop labels e and f. An extensive treatment of fusion spaces of anyons on a torus can be found in Ref. [74], the "outside" and "inside" bases introduced there, correspond to the first and second handle choices (depected in (2.63)), respectively.

2.5 The tube algebra

Let Σ_A and Σ_B be two surfaces, and let Σ be the surface obtained by gluing these surfaces together along one or more boundary components, in such a way that the marked boundary

points are matched. Ribbon graphs on σ_A and Σ_B with matching edge-labels in the glued boundary components, can then be glued together to form a ribbon graph on Σ. In this way, we can associate to every element $s \in \mathcal{H}_{\Sigma_B}$ a linear map $\hat{s} : \mathcal{H}_{\Sigma_A} \to \mathcal{H}_\Sigma$. The result of acting with $\hat{s}$ on state $t \in \mathcal{H}_{\Sigma_B}$ is a linear combination of ribbon graphs comprising s and t that have matching boundary labels. The map

$$\hat{} : \mathcal{H}_{\Sigma_B} \to \widehat{\mathcal{H}_{\Sigma_B}} \tag{2.65}$$

is an isomorphism between vector spaces, hence we can interpret ribbon graphs as both states and operators interchangeably. Since the composition of operators in $\widehat{\mathcal{H}_{\Sigma_2}}$ is again an operator in $\widehat{\mathcal{H}_{\Sigma_2}}$, they define an operator algebra, known as Ocneanu's *tube algebra* [67]. The computational basis of $\mathcal{H}_{\Sigma_2}$ then corresponds to the basis $\{O_{k\alpha l\beta}\}$ of $\widehat{\mathcal{H}_{\Sigma_2}}$ with

$$O_{kl\alpha\beta} = \quad \alpha \begin{pmatrix} l \\ \beta \\ k \end{pmatrix} . \tag{2.66}$$

Consider the operators

$$\mathcal{P}^{a\bar{b}}_{kl} \equiv \frac{1}{\mathcal{D}} \frac{v_a v_b}{v_k} \quad a \begin{pmatrix} l \\ \\ k \end{pmatrix} b \ , \tag{2.67}$$

corresponding to (a rescaling of) the anyonic fusion basis states (2.50) of $\widehat{\Sigma_2}$. One can show that stacking two such anyonic fusion basis states yields

$$\begin{pmatrix} a' \\ k' \quad k \quad a \quad \ell \\ b' \quad b \end{pmatrix} = \mathcal{D}\, \delta_{aa'} \delta_{bb'} \frac{v_k}{v_a v_b} \quad \begin{pmatrix} a \\ k' \quad \ell \\ b \end{pmatrix} . \tag{2.68}$$

Hence, the operators $\mathcal{P}^{a\bar{b}}_l \equiv \mathcal{P}^{a\bar{b}}_{ll}$ are the simple idempotents of the tube algebra:

$$\mathcal{P}^{a\bar{b}}_l \equiv \frac{1}{\mathcal{D}} \frac{v_a v_b}{v_l} \quad a \begin{pmatrix} l \\ \\ l \end{pmatrix} b \ . \tag{2.69}$$

Similarly, the operators $\mathcal{P}^{a\bar{b}}_{kl}$ with $k \neq l$ are nilpotents.

In general the simple idempotents Eq. (2.69) do not commute with all elements from the tube algebra. In particular, if any nilpotents $\mathcal{P}^{a\bar{b}}_{kl}$ ($k \neq l$) exist for some pair of labels $a\bar{b}$, then these will not commute with the simple idempotents $\mathcal{P}^{a\bar{b}}_l$ since Eq. (2.68) implies $\mathcal{P}^{a\bar{b}}_l \mathcal{P}^{a\bar{b}}_{kl} = \mathcal{P}^{a\bar{b}}_{kl}$ and $\mathcal{P}^{a\bar{b}}_{kl} \mathcal{P}^{a\bar{b}}_l = 0$. For every pair of labels $a\bar{b}$, one can construct a unique central idempotent as follows:

$$\mathcal{P}^{a\bar{b}} = \sum_{l \,|\, \delta_{abl}=1} \mathcal{P}^{a\bar{b}}_l . \tag{2.70}$$

These central idempotents project on the different anyon superselection sectors in a puncture: for a state $\psi \in \mathcal{H}_{\Sigma_n}$ that is a superposition of basis states (2.54), stacking the central idempotent $\mathcal{P}^{a\bar{b}}$ onto the ith hole results in the state where we only keep those basis states in the superposition with anyon label $a\bar{b}$ associated to the ith hole. Note that since these projectors commute with all elements of the tube algebra, the anyon label of a puncture

is stable against local deformations of the ribbon graph (obtained by locally acting with tube algebra elements). Stacking the irreducible idempotent $\mathcal{P}_l^{a\bar{b}}$ on the ith hole instead, further limits the resulting superposition to states with boundary label l in the ith hole.

The simple idempotents can be expressed in the basis of basic tube operators Eq. (2.66):

$$\mathcal{P}_l^{a\bar{b}} = \sum_{\alpha\beta} P_{\alpha\beta}^{(abl)} O_{ll\alpha\beta} \,, \tag{2.71}$$

where the coefficients $P_{\alpha\beta}^{(abl)}$ follow from (2.51)

$$P_{\alpha\beta}^{(abl)} = \frac{1}{\mathcal{D}^2} \frac{d_a d_b}{v_l} v_\alpha v_\beta \sum_{\gamma,\delta} d_\gamma d_\delta R_\gamma^{a\alpha} R_\delta^{\alpha b} G_{\alpha ab}^{l\delta\gamma} G_{b\alpha l}^{\beta a\delta} G_{a\gamma\delta}^{l\beta\alpha} \,. \tag{2.72}$$

For the Fibonacci input category (FIB), the doubled category is known as the doubled Fibonacci category (DFIB):

$$\text{DFIB} = \text{FIB} \times \text{FIB}^* = \{\mathbf{1}, \tau\} \times \{\bar{\mathbf{1}}, \bar{\tau}\} \,. \tag{2.73}$$

The central idempotents corresponding to these 4 anyon types are given by

$$\mathcal{P}^{\mathbf{1}\bar{\mathbf{1}}} = \frac{1}{\mathcal{D}^2} \left(O_{\mathbf{1111}} + \phi O_{\mathbf{11}\tau\tau} \right) \,, \tag{2.74}$$

$$\mathcal{P}^{\mathbf{1}\bar{\tau}} = \frac{1}{\mathcal{D}^2} \left(O_{\tau\tau\mathbf{1}\tau} + \mathrm{e}^{4\pi\mathrm{i}/5} O_{\tau\tau\tau\mathbf{1}} + \sqrt{\phi}\mathrm{e}^{-3\pi\mathrm{i}/5} O_{\tau\tau\tau\tau} \right) \,, \tag{2.75}$$

$$\mathcal{P}^{\tau\bar{\mathbf{1}}} = \frac{1}{\mathcal{D}^2} \left(O_{\tau\tau\mathbf{1}\tau} + \mathrm{e}^{-4\pi\mathrm{i}/5} O_{\tau\tau\tau\mathbf{1}} + \sqrt{\phi}\mathrm{e}^{3\pi\mathrm{i}/5} O_{\tau\tau\tau\tau} \right) \,, \tag{2.76}$$

$$\mathcal{P}^{\tau\bar{\tau}} = \frac{1}{\mathcal{D}^2} \left(\phi^2 O_{\mathbf{1111}} - \phi O_{\mathbf{11}\tau\tau} + \phi O_{\tau\tau\mathbf{1}\tau} + \phi O_{\tau\tau\tau\mathbf{1}} + \frac{1}{\sqrt{\phi}} O_{\tau\tau\tau\tau} \right) \,. \tag{2.77}$$

where $\phi = (1+\sqrt{5})/2$ is again the golden ratio. The last entry decomposes into two simple idempotents: $\mathcal{P}^{\tau\bar{\tau}} = \mathcal{P}_{\mathbf{1}}^{\tau\bar{\tau}} + \mathcal{P}_{\tau}^{\tau\bar{\tau}}$ with

$$\mathcal{P}_{\mathbf{1}}^{\tau\bar{\tau}} = \frac{1}{\mathcal{D}^2} (\phi^2 O_{\mathbf{1111}} - \phi O_{\mathbf{11}\tau\tau}) \,, \tag{2.78}$$

$$\mathcal{P}_{\tau}^{\tau\bar{\tau}} = \frac{1}{\mathcal{D}^2} \left(\phi O_{\tau\tau\mathbf{1}\tau} + \phi O_{\tau\tau\tau\mathbf{1}} + \frac{1}{\sqrt{\phi}} O_{\tau\tau\tau\tau} \right) \,. \tag{2.79}$$

It is important to note that both the $+1$ eigenstates of $\mathcal{P}_{\mathbf{1}}^{\tau\bar{\tau}}$ and those of $\mathcal{P}_{\tau}^{\tau\bar{\tau}}$, should be interpreted as containing a $\tau\bar{\tau}$ anyon. It follows from Eq. (2.68) that their respective $+1$ eigenspaces are related through the nilpotent operators

$$\mathcal{P}_{\mathbf{1}\tau}^{\tau\bar{\tau}} = \mathrm{e}^{-3\pi\mathrm{i}/10} \frac{\phi}{\mathcal{D}} O_{\mathbf{1}\tau\tau\tau} \,, \qquad \text{and} \qquad \mathcal{P}_{\tau\mathbf{1}}^{\tau\bar{\tau}} = \mathrm{e}^{3\pi\mathrm{i}/10} \frac{\sqrt{\phi}}{\mathcal{D}} O_{\tau\mathbf{1}\tau\tau} \,, \tag{2.80}$$

as follows

$$\mathcal{P}_{\mathbf{1}\tau}^{\tau\bar{\tau}} \mathcal{P}_{\mathbf{1}}^{\tau\bar{\tau}} = \mathcal{P}_{\tau}^{\tau\bar{\tau}}, \qquad \mathcal{P}_{\tau\mathbf{1}}^{\tau\bar{\tau}} \mathcal{P}_{\tau}^{\tau\bar{\tau}} = \mathcal{P}_{\mathbf{1}}^{\tau\bar{\tau}}. \tag{2.81}$$

2.6 The Levin-Wen model as a lattice realization of $\mathcal{H}_\Sigma$

We now turn to the realization of the ribbon graph Hilbert space $\mathcal{H}_{\Sigma_n}$ as the ground space of a lattice spin model. We will consider surfaces Σ_n containing n punctures and no other boundary components.

Let $\mathcal{T}$ be a triangulation of Σ_n, and denote its dual graph by Λ. Then Λ is a connected graph with vertices of degree three (each corresponding to a triangle in $\mathcal{T}$). Note that we take Λ to contain boundary edges for each hole, corresponding to edges in $\mathcal{T}$ that lie along the boundary of the respective holes. We now modify the Λ as follows: for each hole, remove all but one of the boundary edges, and then remove the resulting vertices of degree two by identifying the two edges for each of them. The are then left with a lattice containing exactly one boundary edge for each hole, and vertices of degree 3.

A qudit with local Hilbert space $\mathcal{H}_e = \mathbb{C}^N$ is placed on each edge e of Λ, where N is the number of anyon labels in the input category $\mathcal{C}$. We choose an orthonormal basis $\{|i\rangle_e\}$ for this local Hilbert space, where each basis element is labeled by an anyon type of $\mathcal{C}$. This gives a lattice spin system with a Hilbert space $\mathcal{H} \equiv (\mathcal{H}_e)^{\otimes E} = (\mathbb{C}^N)^{\otimes E}$, where E is the number of edges in Λ.

For every vertex v in Λ, we define the following vertex operator:

$$Q_v = \sum_{ijk} \delta_{ijk} |ijk\rangle \langle ijk| , \qquad (2.82)$$

where i, j and k are the labels of the qubits associated the three edges connected to vertex v. One can easily see that all Q_v operators commute with one another, meaning that they can be simultaneously diagonalized. We denote the simultaneous $+1$ eigenspace of all Q_v by $\mathcal{H}_{\text{s.n.}} \equiv \{|\psi\rangle \in \mathcal{H} \,|\, \forall v : Q_v |\psi\rangle = |\psi\rangle\}$, and will refer to it as the *string-net subspace* of $\mathcal{H}$. Let P be the number of plaquettes in Λ, and let $\Delta \simeq \Sigma_{n+P}$ be the surface obtained by placing a puncture at the center of each plaquette in Σ_n. The states in $\mathcal{H}_{\text{s.n.}}$ can then be regarded as (superpositions of) ribbon graphs on Δ by embedding the lattice Λ in surface Δ. More precisely, one can show that the string-net subspace is isomorphic to $\bigoplus_\ell \mathcal{H}_{\Sigma_{n+P}}^{(\ell,1^P)}$, which is the subspace of $\mathcal{H}_\Delta$ defined by the condition that all P holes corresponding to plaquettes in Λ must have a trivial boundary label.

For each plaquette p of the lattice Λ, we now introduce a plaquette operator which corresponds to adding a vacuum loop (devided by the total quantum dimension) around the puncture in Σ_{n+P}, that corresponds to the said plaquette:

$$B_p \left| \begin{array}{c} \substack{m_r \; j_r \quad m_{r-1}} \\ j_1 \quad \vdots \quad m_3 \\ \substack{m_1 \; j_2 \; j_3 \\ m_2} \end{array} \right\rangle = \frac{1}{\mathcal{D}} \left| \begin{array}{c} \substack{m_r \; j_r \quad m_{r-1}} \\ j_1 \quad \vdots \quad m_3 \\ \substack{m_1 \; j_2 \; j_3 \\ m_2} \end{array} \right\rangle = \frac{1}{\mathcal{D}^2} \sum_s d_s \left| \begin{array}{c} \substack{m_r \; j_r \quad m_{r-1}} \\ j_1 \; \fbox{s} \quad m_3 \\ \substack{m_1 \; j_2 \; j_3 \\ m_2} \end{array} \right\rangle . \qquad (2.83)$$

Its action on $\mathcal{H}_{\text{s.n.}}$ can be determined by repeatedly using (2.31) to pull the interior loop onto the lattice, giving

$$B_p = \frac{1}{\mathcal{D}^2} \sum_s d_s O_p^s , \qquad (2.84)$$

with

$$O_p^s \left| \begin{array}{c} \substack{m_r \; j_r \quad m_{r-1}} \\ j_1 \quad \vdots \quad m_3 \\ \substack{m_1 \; j_2 \; j_3 \\ m_2} \end{array} \right\rangle = \sum_{k_1,..,k_r} \left(\prod_{\nu=1}^r F_{sk_\nu k_{\nu+1}}^{m_\nu j_{\nu+1} j_\nu} \right) \left| \begin{array}{c} \substack{m_r \; k_r \quad m_{r-1}} \\ k_1 \quad \vdots \quad m_3 \\ \substack{m_1 \; k_2 \; k_3 \\ m_2} \end{array} \right\rangle . \qquad (2.85)$$

Note that it follows form the F-symbol condition (2.7) that $B_p = 0$ outside the subspace where all involved vertices v satisfy the vertex condition imposed by Q_v, meaning B_p is properly defined on the entire Hilbert space $\mathcal{H}$.

Using the identity (2.46), one can see that the operators B_p are in fact projectors. They are in fact the lattice realization of the central idempotent $\mathcal{P}^{1\bar{1}}$ of the tube algebra

defined in (2.70). Furthermore, because B_p corresponds to adding a vacuum loop around a puncture inside plaquette p, all B_p operators commute. Hence, $\{Q_v\} \cup \{B_p\}$ is a set of commuting projectors.

We can now introduce the following Hamiltonian on $\mathcal{H}$:

$$H_\Lambda = -\sum_v Q_v - \sum_p B_p\,, \tag{2.86}$$

for which the ground space $\mathcal{H}_\Lambda$ is precisely the simultaneous $+1$ eigenspace of all Q_v and B_p operators. We claim that $\mathcal{H}_\Lambda$ is isomorphic to $\mathcal{H}_{\Sigma_n}$. Indeed, since adding a vacuum loop around a puncture allows any ribbons to be pulled across it (effectively hiding the presence of the said puncture), the $+1$ eigenspace of B_p inside $\mathcal{H}_{\text{s.n.}}$ is isomorphic to $\bigoplus_\ell \mathcal{H}_{\Sigma_{n+P-1}}^{(\ell, 1^{P-1})}$. By this reasoning, one finds that the simultaneous $+1$ eigenspace of all Q_v and B_p operators is isomorphic to $\mathcal{H}_{\Sigma_n}$.

A state in the ribbon graph Hilbert space $\mathcal{H}_{\Sigma_n}$ can be explicitly mapped to the ground space $\mathcal{H}_\Lambda$ of H_Λ with the map

$$\Gamma_\Lambda : \mathcal{H}_{\Sigma_n} \to \mathcal{H}_\Lambda$$

who's action is defined as follows:

- Given a state in $\mathcal{H}_{\Sigma_n}$, continuously deform the ribbons in an arbitrary way to avoid all (imaginary) punctures at the centers of the plaquettes of Λ. This deformation yields a state in $\bigoplus_\ell \mathcal{H}_\Delta^{(\ell, 1^P)}$, which can be identified with a lattice state in the string-net subspace. This lattice state is then an eigenstate with eigenvalue $+1$ of all the vertex operators Q_v.

- Apply the total projector $B = \prod_p B_p$ to this lattice state to map it to the ground space of the string-net Hamiltonian H_Λ.

It was shown in Ref. [56] that this map defines an isomorphism between the ribbon graph Hilbert space $\mathcal{H}_{\Sigma_n}$ and the string-net ground space $\mathcal{H}_\Lambda$, which confirms our claim that $\mathcal{H}_\Lambda \simeq \mathcal{H}_{\Sigma_n}$.

For the case where where $n = 0$, (2.86) is precisely the Levin-Wen Hamiltonian introduced in Ref. [50], which is conjectured to describe all doubled topological phases. It's important to note that the braiding properties of the input category $\mathcal{C}$ are not required to define the Levin-Wen model, which is defined for any unitary fusion category satisfying the additional rules (2.27), (2.26) and (2.28). In fact, the original construction has since been generalized in order to drop these additional requirements, for example see Ref. [75].

Part II

Error correction with the extended string-net code

3 | The extended string-net code

From the discussion on the ribbon graph Hilbert space and its realization as the ground space of the Levin-Wen model in Chapter 2, one may have inferred the following: States of lattice qudits that respect the fusion rules of a category $\mathcal{C}$ in all vertices, can be understood as superpositions of fusion states containing at most $|P|$ anyons in $\mathcal{DC}$, where $|P|$ is the number of plaquettes. However, reality is a bit more restrictive: only those anyons in $\mathcal{DC}$ that can correspond to a trivial boundary label can appear in the fusion states above.

One can modify the Levin-Wen model in order to remove this restriction [66], such that any fusion state of $|P|$ doubled anyons can be realized inside the string-net subspace. This modification consists of including one *tail edge* in every plaquette as shown in Fig. 3.1, in order to allow for strings to end inside plaquettes.
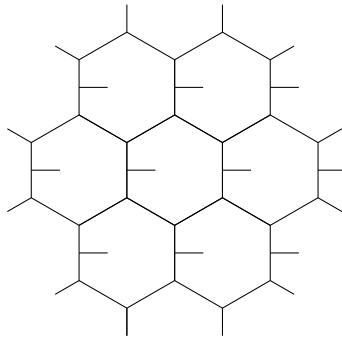
Figure 3.1: The tailed honeycomb lattice.

In this chapter, we consider this *extended string-net model* as the basis for a quantum error correcting code. We start by defining it starting from the algebraic data of a unitary fusion category. We then discuss code deformations, the action of the tube algebra, and the anyonic fusion basis for the string-net subspace.
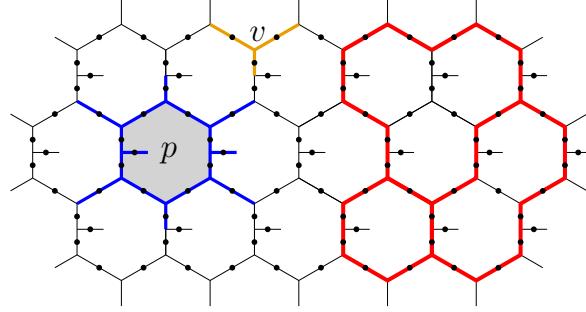
Figure 3.2: Qudits arranged on the tailed honeycomb lattice. The support of B_p and Q_v are indicated in blue and orange, respectively. The red edges represent a valid string-net configuration of qubits in the $|1\rangle$ state when using the Fibonacci input category.

3.1 Definition of the code

The *extended string-net code* is a microscopic realization of a Turaev-Viro code [56]. Its code space is defined as the ground space of the *extended* Levin-Wen string-net model [66]. This is a microscopic model of qudits situated on the edges of a tailed trivalent lattice Λ, obtained by modifying the Levin-Wen Hamiltonian [50] to accommodate one additional "tail edge" in every plaquette as shown in Fig. 3.2. The code space is hence denoted by $\mathcal{H}_\Lambda$. Below, we give a summary of its definition and of its most important properties.

The model is defined starting from the algebraic data of a unitary fusion category $\mathcal{C}$. For simplicity, we limit ourselves to multiplicity-free self-dual categories. The generalization to generic unitary fusion categories is straightforward but quite tedious. Since we will work with the Fibonacci category later (which is self-dual), we will not be needing the general case. The algebraic data of such an object consists of:

1. **String types**: A set of all possible string types $\{\mathbf{1}, i_2, \ldots, i_N\}$. The label $\mathbf{1}$ is referred to as the *vacuum label*, and represents the absence of a string on a particular edge.

2. **Branching rules**: The set of all triplets $\{i, j, k\}$ that are allowed to meet at a vertex (also known as *fusion rules*). We introduce the symbol δ_{ijk} defined by the branching rules as

$$\delta_{ijk} = \begin{cases} 1, & \text{if the triplet } \{i, j, k\} \text{ is allowed,} \\ 0, & \text{otherwise.} \end{cases} \tag{3.1}$$

For every label i, there is unique dual label i^* satisfying $\delta_{ii^*\mathbf{1}} = 1$, and $(i^*)^* = i$. Note that we are considering self-dual categories[1], which satisfy $i^* = i$ for every string type i.

3. **Numerical data**: For each string type i, a real constant d_i, called the *quantum dimension*, satisfying

$$d_i d_j = \sum_k \delta_{ijk} d_k, \quad d_1 = 1, \quad \text{and} \quad d_{i^*} = d_i, \tag{3.2}$$

and a six-index symbol F^{ijm}_{kln}, which is a complex constant dependent on 6 string types i, j, k, l, m, n. These quantities are required to satisfy the following consistency

---

[1]For generic unitary fusion categories, one must pick an orientation for every edge. An edge with label i^* is equivalent to an edge with label i and the opposite orientation.

conditions:

$$\text{physicality}: \quad F^{ijm}_{kln}\delta_{ijm}\delta_{klm^*} = F^{ijm}_{kln}\delta_{iln}\delta_{jkn^*} \tag{3.3}$$

$$\text{pentagon equation}: \quad \sum_{n=1}^{N} F^{mlq}_{kpn}F^{jip^*}_{mns}F^{jsn}_{lkr} = F^{jip^*}_{q^*kr}F^{r^*iq^*}_{mls} \tag{3.4}$$

$$\text{unitarity}: \quad \left(F^{ijm}_{kln}\right)^* = F^{lin}_{jkm^*} \tag{3.5}$$

$$\text{tetrahedral symmetry}: \quad F^{ijm}_{kln} = F^{jim}_{lkn^*} = F^{lkm^*}_{jin} = F^{imj}_{k^*nl}\frac{v_m v_n}{v_j v_l} \tag{3.6}$$

$$\text{normalization}: \quad F^{ii^*1}_{j^*jk} = \frac{v_k}{v_i v_j}\,\delta_{ijk} \tag{3.7}$$

where $v_i = \sqrt{d_i}$.

Note that for self-dual categories, the F-symbols are real-valued.

We associate the string types to the elements of an orthonormal basis of the qudit Hilbert space $\mathbb{C}^N$ at each edge. The extended Levin-Wen Hamiltonian is then defined as:

$$H_\Lambda = -\sum_v Q_v - \sum_p B_p\,, \tag{3.8}$$

where v and p label the vertices and plaquettes of the trivalent (tailed) lattice Λ, and $\{Q_v, B_p\}$ are a set of commuting projectors whose support is shown in Fig. 3.2.

For every vertex v, the three-body projector Q_v imposes the branching rules:

$$Q_v|i\overset{j}{\underset{k}{\diagup}}\rangle = \delta_{ijk}|i\overset{j}{\underset{k}{\diagup}}\rangle\,. \tag{3.9}$$

The subspace $\mathcal{H}_{\text{s.n.}}$ of states that satisfy all vertex projectors is known as the *string-net subspace*. States in $\mathcal{H}_{\text{s.n.}}$ can be understood as superpositions of string-nets, which are defined as string configurations that obey the branching rules.

We will work with the tailed honeycomb lattice shown in Fig. 3.2, for which the plaquette projector B_p is a 16-body operator. On a generic trivalent tailed lattice, is defined as

$$B_p = \frac{1}{\mathcal{D}^2}\sum_s d_s O^s_p\,, \tag{3.10}$$

where $\mathcal{D} = \sqrt{\sum_i d_i^2}$, and

$$O^s_p \left|\begin{smallmatrix} m_r & j_r & & m_{r-1} \\ j_{r+1} & \diagdown & \ddots & \\ j_1 & x & \vdots & \\ m_1 & j_2 & j_3 & m_3 \\ & & m_2 & \end{smallmatrix}\right\rangle = \delta_{x,\mathbf{1}}\sum_{k_1,\dots,k_{r+1}}\delta_{k_1,k_{r+1}}\left(\prod_{\nu=1}^{r}F^{m_\nu j_\nu j_{\nu+1}}_{sk_{\nu+1}k_\nu}\right)\left|\begin{smallmatrix} m_r & k_r & & m_{r-1} \\ k_{r+1} & \mathbf{1} & \ddots & \\ k_1 & & \vdots & \\ m_1 & k_2 & k_3 & m_3 \\ & & m_2 & \end{smallmatrix}\right\rangle\,. \tag{3.11}$$

The error correction scheme and numerical simulations described in Chapters 4, 5 and 7 were designed specifically for the Fibonacci input category ($\mathcal{C} = \text{FIB}$), which contains only two string types, $\mathbf{1}$ and τ. Hence the model we consider in the remainder of this dissertation is a system of qubits. We choose to relate the string types to the standard computational basis states: $\mathbf{1} \to |0\rangle$ and $\tau \to |1\rangle$. The only nontrivial fusion rule is $\tau \times \tau = \mathbf{1} + \tau$, which leads to the following branching rules:

$$\delta_{ijk} = \begin{cases} 1, & \text{if } (ijk) \in \{\mathbf{111},\, \tau\tau\mathbf{1},\, \mathbf{1}\tau\tau,\, \tau\mathbf{1}\tau,\, \tau\tau\tau\},\\ 0, & \text{otherwise}. \end{cases} \tag{3.12}$$

The quantum dimensions are

$$d_{\mathbf{1}} = 1 \,, \qquad d_\tau = \phi \,, \tag{3.13}$$

where $\phi = \frac{1+\sqrt{5}}{2}$ is the golden ratio. The only nontrivial F-matrix is

$$[F_{\tau\tau}^{\tau\tau}] = \begin{pmatrix} \phi^{-1} & \phi^{-\frac{1}{2}} \\ \phi^{-\frac{1}{2}} & -\phi^{-1} \end{pmatrix} . \tag{3.14}$$

For all other combinations of indices, F_{kln}^{ijm} is either 1 or 0, depending on whether or not the corresponding indices in Eq. (3.3) satisfy the branching rules.

The ground space of Hamiltonian (3.8) has a degeneracy that depends on the genus of the surface on which the model is defined. On a torus, and with the Fibonacci input category, the code space is four-dimensional, which enables one to encode the state of two logical qubits.

3.2   The fattened lattice picture

It is convenient to think about the string-net Hilbert space as the lattice realization of the ribbon graph Hilbert space on a punctured surface (see Chapter 2 for a more details). For our current purpose, it is sufficient to state that this is the space of formal linear combinations of labeled trivalent graphs which satisfy the branching rules Eq. (3.1), modulo continuous deformations and the following relations:

$$\overset{j}{\underset{}{\bigcirc}}{}^{i} = \delta_{j1} d_i \,, \tag{3.15}$$

$$\begin{array}{c}\overset{i}{} \overset{l}{} \\ m \\ \overset{.}{j} \quad \overset{.}{k}\end{array} = \sum_n F_{kln}^{ijm} \begin{array}{c}\overset{i}{} \overset{l}{} \\ n \\ \overset{.}{j} \quad \overset{.}{k}\end{array} . \tag{3.16}$$

The second relation is known as an F-move or 2-2 Pachner move.

These *ribbon graphs* are defined on a compact, orientable, surface Δ containing one puncture for every plaquette in the lattice. Each boundary component has a unique marked boundary point, and ribbons are only allowed to end on these marked boundary points. We can relate ribbon graphs on the surface Δ to string-nets using the "fattened lattice" picture, which represents an embedding of the lattice Λ in the surface Δ as shown in Fig. 3.3a. Whenever ribbons have a more complicated shape that can't be smoothly deformed to the shape of the embedded lattice, one can first deform them using 2-2 Pachner moves, and 1-3 Pachner moves. The latter are defined as

$$\begin{array}{c}\overset{i}{} \\ \nu \overset{}{\bigcirc} \mu \\ \overset{.}{j} \quad \overset{.}{\lambda} \quad \overset{.}{k}\end{array} = v_\lambda v_\mu v_\nu G_{\lambda\mu\nu}^{ijk} \begin{array}{c}\overset{i}{} \\ \\ \overset{.}{j} \qquad \overset{.}{k}\end{array} , \tag{3.17}$$

where

$$G_{\lambda\mu\nu}^{ijk} = \frac{1}{v_i v_\lambda} F_{k\mu i}^{\nu j\lambda} = \frac{1}{v_\nu v_k} F_{\lambda\mu\nu}^{ijk} \,. \tag{3.18}$$

The action of O_p^s, defined in Eq. (3.11), can be represented in the fattened lattice picture as the inclusion of a loop with label s around the puncture in plaquette p. Hence, the action of the plaquette projector B_p can be represented on the fattened lattice as follows:

$$B_p : \quad \text{⬡} \quad \mapsto \quad \frac{1}{\mathcal{D}} \, \text{⬡} \,, \tag{3.19}$$

(a)   (b)
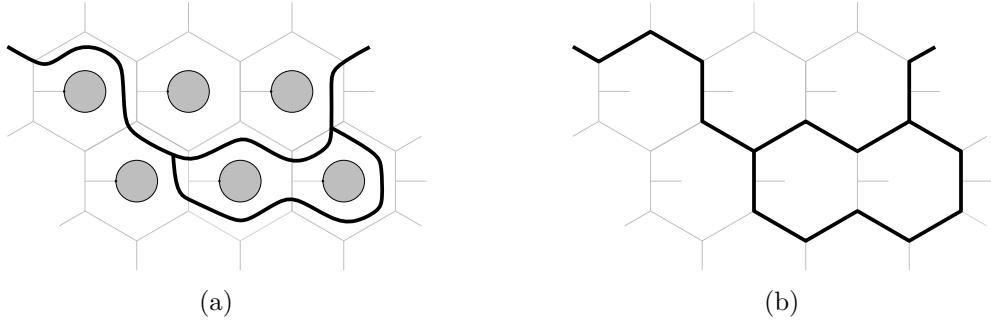
Figure 3.3: (a) A ribbon graph on the fattened lattice. (b) The corresponding string-net configuration on the lattice. The grey edges correspond to qudits in the $|0\rangle$ state, while the state of the black edges is given by the string type of the corresponding piece of the ribbon graph in (a).

where

$$
\vdots \quad = \frac{1}{\mathcal{D}} \sum_i d_i \ \Big|_i \ .
\tag{3.20}
$$

The dashed loop is referred to as a *vacuum loop*. Note that we have omitted the tail edge on the right hand side of Eq. (3.19). After resolving the loop into the lattice using a sequence of F-moves, a trivial tail edge should be included. Keep in mind that $B_p = 0$ whenever there is a (nontrivial) ribbon ending in the puncture p, which corresponds to a nontrivial tail edge.

Ground states (up to a normalization factor) then correspond to ribbon graph configurations without any ribbons ending in punctures, and where a vacuum loop is added around every puncture as shown in Fig. 3.4. A short calculation shows that ribbons can be "pulled across" vacuum loops without changing the corresponding string-net state, meaning that one can obtain the same ground state by applying the $\prod_p B_p$ to different string-net states.
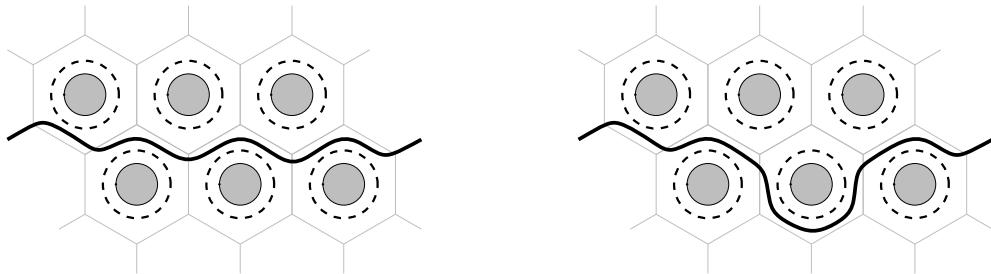
Figure 3.4: Two ribbon graphs on the fattened lattice representing the same string-net ground state.

3.3 Code deformation using Pachner moves

The Pachner moves introduced above can be used to relate string-net states defined on different lattice geometries. In particular, when transforming the lattice Λ to Λ', Eqs. (3.16) and (3.17) give the appropriate map between the corresponding code spaces $\mathcal{H}_\Lambda$ and $\mathcal{H}_{\Lambda'}$, defined as the ground spaces of Hamiltonians H_Λ and $H_{\Lambda'}$, respectively. For instance, applying the unitary F-move (see Fig. 4.3) to the qudits on certain edges of a ground
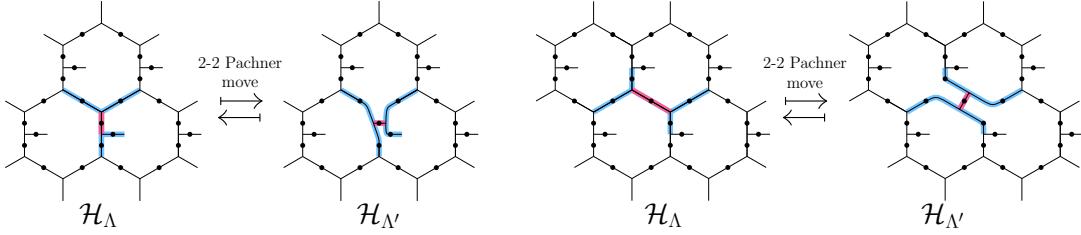
Figure 3.5: Lattice deformations by 2-2 Pachner moves on different edges. The affected edge is highlighted in pink, the other edges contained in the F-matrix is are highlighted in blue.
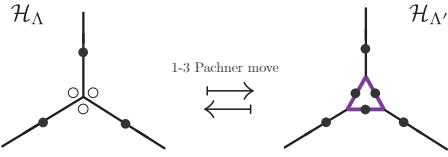
Figure 3.6: Lattice deformation by 1-3 Pachner move. The white dots on the left represent ancilla qudits. The fine-
graining process (left to right) entangles the three ancilla qudits into the code space. The coarse-graining process (right to left) disentangles the three central qudits out of the code space.

state, transforms this state to a ground state of the string-net Hamiltonian defined on a new lattice obtained by recoupling these edges in the original lattice. The recoupling of lattice edges by a 2-2 Pachner move is shown in Fig. 3.5. The lattice deformation corresponding to a 1-3 Pachner move is shown in Fig. 3.6, where one adds a triangular loop on the original vertex. The 1-3 Pachner move can be thought as a fine/coarse-graining process. In Fig. 3.6 one entangles three ancilla qudits (white dots) from left to right and add them into the code space, which effectively fine-grain the lattice. The inverse process from right to left disentangles the qudits in the center out of the code space, which effectively coarse-grains the lattice. Both 2-2 and 1-3 Pachner moves can be implemented via unitary circuits as will be shown in Sec. 4.2.

3.4   Anyonic excitations

Localized excitations in the string-net model exhibit anyonic statistics, described by the quantum double $\mathcal{DC}$ of the input category $\mathcal{C}$ [50]. It is important to note that the categorical double of a unitary fusion category is always braided. Hence, the input category $\mathcal{C}$ does not need to include any braiding structure for the excitations to have well defined anyonic statistics. When the input category $\mathcal{C}$ is modular (implying it is braided), such as for $\mathcal{C} = $ FIB, the doubled category has a special structure $\mathcal{DC} \cong \mathcal{C} \otimes \bar{\mathcal{C}}$, and its string types can be labeled by pairs $a_+\overline{a_-}$ with $a_+, a_- \in \mathcal{C}$ (see Chapter 2 for more details). For notational simplicity, we will drop the bar notation for labels in $\bar{\mathcal{C}}$, and we will indicate the string-types of the doubled category with bold labels: $\boldsymbol{a} = a_+a_- \in \mathcal{DC}$.

The motivation for introducing the tail qudit in every plaquette is that they allow us to define the action of an operator algebra known as Ocneanu's tube algebra [67] in TQFT (see Sec. 2.5), at the level of the lattice qudits. The central idempotents of the tube algebra form projectors onto the different superselection sectors of the theory. By defining their action on an individual plaquette, one obtains a set of projectors corresponding to
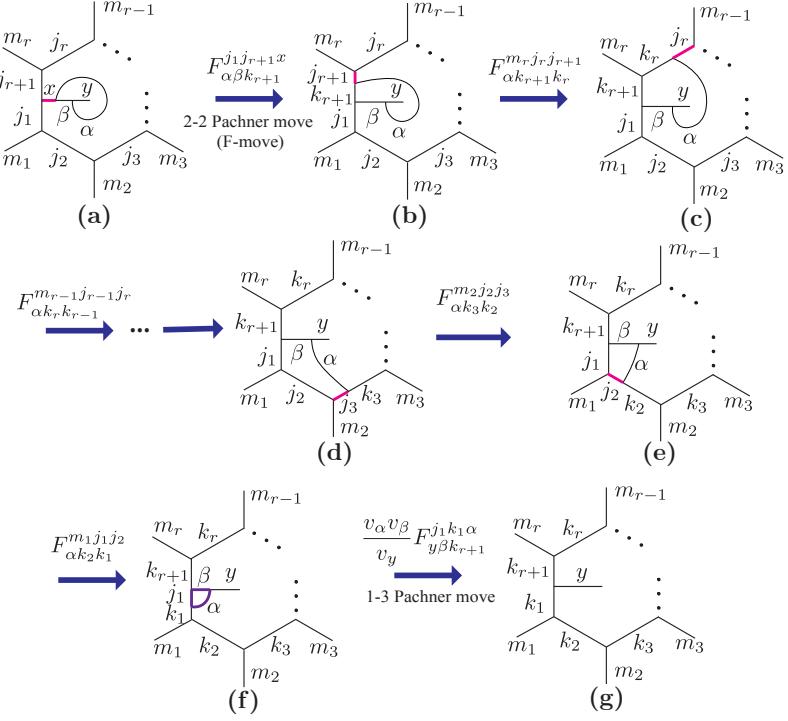
Figure 3.7: Derivation of the expression for the tube operator.

the different possible values of the doubled anyonic charge contained in that plaquette . The generators of this algebra act on an individual plaquette as

$$
O_{xy\alpha\beta} \left| \begin{matrix} m_r & j_r & m_{r-1} \\ j_{r+1} & x' & \vdots \\ j_1 & & m_3 \\ m_1 & j_2 & j_3 \\ & m_2 & \end{matrix} \right\rangle = \delta_{x,x'} \left| \begin{matrix} m_r & j_r & m_{r-1} \\ j_{r+1} & x & y \\ j_1 & \beta & m_3 \\ m_1 & j_2 & j_3 \\ & m_2 & \end{matrix} \right\rangle \tag{3.21}
$$

$$
= \delta_{x,x'} \frac{v_\alpha v_\beta}{v_y} \sum_{k_1,\ldots,k_{r+1}} F^{j_1 j_{r+1} x}_{\alpha\beta k_{r+1}} \left(\prod_{\nu=1}^{r} F^{m_\nu j_\nu j_{\nu+1}}_{\alpha k_{\nu+1} k_\nu} \right) F^{k_{r+1} k_1 y}_{\alpha j_1 \beta} \left| \begin{matrix} m_r & k_r & m_{r-1} \\ k_{r+1} & y & \vdots \\ k_1 & & m_3 \\ m_1 & k_2 & k_3 \\ & m_2 & \end{matrix} \right\rangle .
$$

This corresponds to gluing the "tube"

$$
\alpha \overset{y}{\underset{\beta}{\bigcirc}}_{x} \tag{3.22}
$$

onto the tail edge and resolving it into the lattice using a sequence of 2-2 Pachner moves followed by a 1-3 Pachner move, as shown in Fig. 3.7.

For the Fibonacci input category, the doubled category (DFIB) contains the labels

$\{\mathbf{11}, \mathbf{1}\tau, \tau\mathbf{1}, \tau\tau\}$, which label the different corresponding projectors

$$\mathcal{P}^{\mathbf{11}} = \frac{1}{\mathcal{D}^2} \left(O_{\mathbf{1111}} + \phi O_{\mathbf{11}\tau\tau} \right), \tag{3.23}$$

$$\mathcal{P}^{\mathbf{1}\tau} = \frac{1}{\mathcal{D}^2} \left(O_{\tau\tau\mathbf{1}\tau} + \mathrm{e}^{4\pi\mathrm{i}/5} O_{\tau\tau\tau\mathbf{1}} + \sqrt{\phi}\,\mathrm{e}^{-3\pi\mathrm{i}/5} O_{\tau\tau\tau\tau} \right), \tag{3.24}$$

$$\mathcal{P}^{\tau\mathbf{1}} = \frac{1}{\mathcal{D}^2} \left(O_{\tau\tau\mathbf{1}\tau} + \mathrm{e}^{-4\pi\mathrm{i}/5} O_{\tau\tau\tau\mathbf{1}} + \sqrt{\phi}\,\mathrm{e}^{3\pi\mathrm{i}/5} O_{\tau\tau\tau\tau} \right), \tag{3.25}$$

$$\mathcal{P}^{\tau\tau} = \frac{1}{\mathcal{D}^2} \left(\phi^2 O_{\mathbf{1111}} - \phi O_{\mathbf{11}\tau\tau} + \phi O_{\tau\tau\mathbf{1}\tau} + \phi O_{\tau\tau\tau\mathbf{1}} + \frac{1}{\sqrt{\phi}} O_{\tau\tau\tau\tau} \right). \tag{3.26}$$

Note that $B_p = \mathcal{P}^{\mathbf{11}}$ i.e., the ground space is precisely the anyonic vacuum. The last entry decomposes into two simple idempotents: $\mathcal{P}^{\tau\tau} = \mathcal{P}_{\mathbf{1}}^{\tau\tau} + \mathcal{P}_{\tau}^{\tau\tau}$, with

$$\mathcal{P}_{\mathbf{1}}^{\tau\tau} = \frac{1}{\mathcal{D}^2} (\phi^2 O_{\mathbf{1111}} - \phi O_{\mathbf{11}\tau\tau}), \tag{3.27}$$

$$\mathcal{P}_{\tau}^{\tau\tau} = \frac{1}{\mathcal{D}^2} \left(\phi O_{\tau\tau\mathbf{1}\tau} + \phi O_{\tau\tau\tau\mathbf{1}} + \frac{1}{\sqrt{\phi}} O_{\tau\tau\tau\tau} \right). \tag{3.28}$$

This decomposition of the central $\tau\tau$ idempotent into two simple idempotents should be interpreted as the fact that a $\tau\tau$ anyon charge in a plaquette does not fix the state of its tail qubit. The corresponding projector has a block-diagonal form with the two blocks corresponding to a $|0\rangle$ or a $|1\rangle$ state for the tail qubit. We label these two cases as $\tau\tau_{\mathbf{1}}$ and $\tau\tau_\tau$, respectively. Both are in the $\tau\tau$ anyon sector, and their respective $+1$ eigenspaces are related as follows:

$$\mathcal{P}_{\mathbf{1}\tau}^{\tau\tau}\mathcal{P}_{\mathbf{1}}^{\tau\tau} = \mathcal{P}_{\tau}^{\tau\tau}, \qquad \mathcal{P}_{\tau\mathbf{1}}^{\tau\tau}\mathcal{P}_{\tau}^{\tau\tau} = \mathcal{P}_{\mathbf{1}}^{\tau\tau}, \tag{3.29}$$

where

$$\mathcal{P}_{\mathbf{1}\tau}^{\tau\tau} = \mathrm{e}^{-3\pi\mathrm{i}/10} \frac{\phi}{\mathcal{D}} O_{\mathbf{1}\tau\tau\tau} \qquad \text{and} \qquad \mathcal{P}_{\tau\mathbf{1}}^{\tau\tau} = \mathrm{e}^{3\pi\mathrm{i}/10} \frac{\sqrt{\phi}}{\mathcal{D}} O_{\tau\mathbf{1}\tau\tau} \tag{3.30}$$

are nilpotent operators.

3.5 The anyonic fusion basis

An important property of the extended Levin-Wen model is the fact that one can construct a basis of the string-net subspace $\mathcal{H}_{\text{s.n.}}$, whose elements are labeled by fusion states of $|P|$ anyons (of the doubled category $\mathcal{DC}$), where $|P|$ is the number of plaquettes. This basis is called the *anyonic fusion basis*. Below, we will give the expressions for these bases in terms of ribbon configurations in the fattened lattice picture, for modular input categories. More details on their origin in the ribbon graph Hilbert space were given in Sec. 2.4.

For simplicity we first show anyonic fusion basis states on a sphere. Since we are considering the fusion space of anyonic excitations in plaquettes, the corresponding fusion diagram are labeled by the string types of the doubled category $\mathcal{DC}$ (indicated by bold labels):

$$|\vec{a}, \vec{b}\rangle = \qquad . \tag{3.31}$$

As mentioned above, the anyon label of a plaquette alone does not always fix the state of the tail qudit. Hence, to fully specify the state, we must also fix the tail labels

$$\vec{\ell} = [\ell_1, \ell_2, \ldots, \ell_{|P|}],$$

where $\vec{\ell}$ must be consistent with the anyon labels of all plaquettes. For $\mathcal{C} = \text{FIB}$, the allowed combinations of plaquette anyon (DFIB) labels $\boldsymbol{a} = a_+ a_-$ and tail labels ℓ are $(a_+ a_-)_\ell \in \{\boldsymbol{11_1}, \boldsymbol{1\tau_\tau}, \boldsymbol{\tau 1_\tau}, \boldsymbol{\tau\tau_1}, \boldsymbol{\tau\tau_\tau}\}$, which are simply all combinations satisfying $\delta_{a_+ a_- \ell} = 1$.

Throughout the remainder of this work, we will often adopt a slight abuse of notation by also using a bold label to indicate the joined labels of the plaquette anyon and tail labels: $\boldsymbol{a} = (a_+ a_-)_\ell$. It will always be clear from the context whether or not a bold label includes the tail label. In particular, only leaf labels can contain a tail label. Internal branch labels of a doubled anyonic fusion tree never include them, since they do not correspond to plaquettes.

An anyonic fusion basis is determined by fixing the branching structure of the corresponding fusion tree and its embedding in the fattened lattice. The basis states are then labeled as $|\vec{\ell}, \vec{a}, \vec{b}\rangle$, where $\vec{\ell}$ are the tail labels, $\vec{a}$ are the leaf labels (corresponding to the anyon charge of each plaquette), and $\vec{b}$ are the internal branch labels. Before embedding them into the fattened lattice, the corresponding ribbon configurations are

$$|\vec{\ell}, \vec{a}, \vec{b}\rangle = \qquad\qquad\qquad\qquad\qquad\qquad (3.32)$$

$$= \sum_{\vec{\alpha},\vec{\beta},\vec{k}} X^{\vec{\ell},\vec{a},\vec{b}}_{\vec{\alpha},\vec{\beta},\vec{k},\vec{l}} \qquad\qquad\qquad , \qquad (3.33)$$

where the coefficients $X^{\vec{\ell},\vec{a},\vec{b}}_{\vec{\alpha},\vec{\beta},\vec{k},\vec{l}}$ are found by resolving the crossings in Eq. 3.32 using

$$\underset{i}{\overset{}{\diagup}}\kern-0.3em\underset{j}{\overset{}{\diagdown}} = \sum_k \frac{v_k}{v_i v_j} R^{ij}_k \quad \underset{i \quad j}{\overset{j \quad i}{k}} , \qquad (3.34)$$

followed by a sequence of F-moves and 1-3 Pachner moves. The object R^{ij}_k above is known as the R-matrix of the input category $\mathcal{C}$, and defines its braiding properties. It must satisfy

certain consistency equations, which are listed in Sec. 2.1. For the Fibonacci category, the only nonzero entries are

$$R_{\mathbf{1}}^{\tau\tau} = e^{\frac{4\pi i}{5}}, \qquad R_{\tau}^{\tau\tau} = e^{\frac{-3\pi i}{5}}, \qquad R_a^{\mathbf{1}a} = R_a^{a\mathbf{1}} = 1, \tag{3.35}$$

where $a \in \{\mathbf{1}, \tau\}$. The necessary calculations to obtain $X_{\vec{\alpha},\vec{\beta},\vec{k},\vec{l}}^{\vec{\ell},\vec{a},\vec{b}}$ were performed explicitly in Sec. 2.4.

After picking some embedding in the fattened lattice, we find

$$|\vec{\ell}, \vec{a}, \vec{b}\rangle = \quad\text{}\tag{3.36}$$

$$= \sum_{\vec{\alpha},\vec{\beta},\vec{k},\vec{l}} X_{\vec{\alpha},\vec{\beta},\vec{k},\vec{l}}^{\vec{\ell},\vec{a},\vec{b}} \quad\text{}, \tag{3.37}$$

where we chose not to draw the leaves with vacuum labels explicitly, but included vacuum loops in the corresponding plaquettes instead. The fattened lattice state above corresponds to the case where only 4 plaquettes carry a nontrivial anyonic charge, other cases are analogous. The final expression for the anyonic fusion basis states (as a state of qudits) can then be found by resolving these ribbon graph configurations into the lattice.

Different anyonic fusion bases (that is, bases corresponding to trees with different branching structures, or different embeddings in the fattened lattice), can be related using F-moves, braid moves and Dehn twists defined by the categorical data of $\mathcal{DC}$:

$$\text{} = \sum_f F_{cdf}^{abe} \quad\text{}, \tag{3.38}$$

$$\text{} = R_c^{ab} \quad\text{}, \qquad \text{} = \left(R_c^{ba}\right)^* \quad\text{}. \tag{3.39}$$

$$\text{} = \theta_{\mathbf{a}} \quad\text{}, \qquad \text{} = (\theta_{\mathbf{a}})^* \quad\text{}, \tag{3.40}$$

Due to the particular structure of the doubled category, $\mathcal{DC} \simeq \mathcal{C} \otimes \bar{\mathcal{C}}$, its numerical data can be deduced from that of the input category $\mathcal{C}$. In particular, one has

$$F_{cdf}^{abe} = F_{c_+ d_+ f_+}^{a_+ b_+ e_+} F_{c_- d_- f_-}^{a_- b_- e_-}, \tag{3.41}$$

$$R_c^{ab} = R_{c_+}^{a_+ b_+} (R_{c_-}^{b_- a_-})^*, \tag{3.42}$$

$$\theta_{\mathbf{a}} = \theta_{a_+} (\theta_{a_-})^*. \tag{3.43}$$
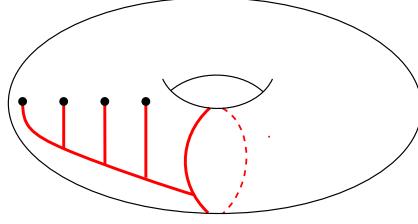
Figure 3.8: Fusion diagram of a system of anyons defined on a torus. Note the line wrapping around the torus.

We discussed how these are obtained for modular input categories in more detail in Sec. 2.4.

The construction of anyonic fusion basis states on a torus is similar. An important difference is that fusion states of anyons on a torus requires us to specify a *handle label* (see Sec. 2.4.2 for more details), which determines how the state transforms when an anyon is moved along a non-contractible loop [74]. An example of such a fusion state is shown in Fig. 3.8. Anyonic fusion basis states are then labeled as $|\vec{\ell}, \vec{a}, \vec{b}, c\rangle$, where $\vec{\ell}$, $\vec{a}$ and $\vec{b}$ are again the tail, leaf and internal branch labels, respectively, and c is the handle label. The corresponding ribbon configurations are:

$$|\vec{\ell}, \vec{a}, \vec{b}, c\rangle = \qquad \qquad$$

$$= \qquad \qquad , \qquad (3.44)$$

where the gray box represents the periodic boundary conditions of a torus. Note that the handle label must satisfy $\delta_{b_{n-1}cc}$ or, equivalently, $\delta_{b_{n-1}^+ c^+ c^+}$ and $\delta_{b_{n-1}^- c^- c^-}$, for the corresponding ribbon graph to obey the branching rules. The crossings on the right-hand side of Eq. (3.44) must again be resolved using Eq. 3.34, which will lead to superposition of ribbon configurations similar to Eq. (3.33). These ribbons must then be embedded in the fattened lattice like in Eq. (3.37), and resolved into the lattice using F-moves.

Ground states of the model correspond to (linear combinations of) configurations in which all plaquettes carry a trivial charge ($a_i = \mathbb{1}$, $b_j = \mathbb{1}$, $\forall i, j$). On a torus, the degenerate ground space is spanned by the states $|\vec{\mathbb{1}}, \vec{\mathbb{1}\mathbb{1}}, \vec{\mathbb{1}\mathbb{1}}, c\rangle$, where $\vec{\mathbb{1}}$ and $\vec{\mathbb{1}\mathbb{1}}$ represent arrays containing only trivial entries. One can show that these states are indeed orthonormal. Hence, when storing the state of two logical qubits (for $\mathcal{C} = \text{FIB}$), this information is encoded in the handle label. The operations that affect the handle, are precisely those

in which anyons interact along a non-contractible path such as the process depicted in Fig. 3.9. As long as no such operations are performed, the encoded information is preserved. Local qubit errors will create pairs, triplets or quadruplets of nontrivial anyons in neighboring plaquettes. The initial ground state can then be recovered by fusing these nontrivial anyons pairwise until none are left, without creating any non-contractible loops in the process.

Clearly, all anyonic fusion basis states correspond to exceedingly complicated superpositions of qubit states. It would seem that this makes them highly impractical for any actual computation. Fortunately, however, one can formulate a tensor network representation for such states, which enables us to use them for practical applications (in particular, see Sec. 7.5). This tensor network representation is derived in Chapter 6.

(a)   (b)
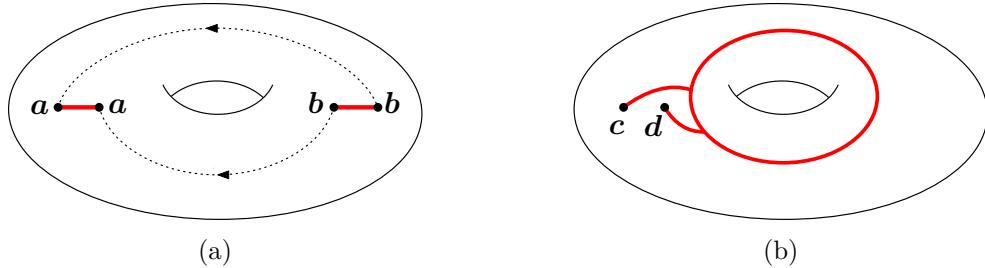
Figure 3.9: (a) Topologically nontrivial process which results in a logical error: two pairs of anyons $(\boldsymbol{a}, \boldsymbol{a})$ and $(\boldsymbol{b}, \boldsymbol{b})$ are created by local qubit errors. The anyons are then fused along the dotted black paths with outcomes $\boldsymbol{c}$ and $\boldsymbol{d}$. (b) The fusion diagram of the resulting state winds around the torus.

4 | Error correction scheme

We now present the circuits to measure and fix arbitrary local errors. From here on, we will work exclusively with the Fibonacci input category, hence we are working with a system of qubits on a lattice. Our overall error correction procedure is composed of two major steps:

1. Measure all the vertex operators Q_v in the extended Levin-Wen model Eq. (3.8), and apply a correction which fixes the vertex errors through unitaries U_V conditioned by a measurement projection P_V. This measurement and correction processes projects the many-body state onto the string-net subspace $\mathcal{H}_{\text{s.n.}}$.

2. After projecting to the string-net subspace $\mathcal{H}_{\text{s.n.}}$, we apply additional measurement circuits to measure the simple idempotents of the tube algebra [Eqs. (3.23), (3.24), (3.25), (3.27), and (3.28)] and extract the error syndromes, i.e., the anyon charges and tail labels of all plaquettes. Based on these syndromes, we use our decoders to identify the error location (up to equivalent classes) and apply the corresponding recovery maps to project the state back to the code (ground) space $\mathcal{H}_\Lambda$. We note that the code space is a subspace of the string-net subspace: $\mathcal{H}_\Lambda \subset \mathcal{H}_{\text{s.n.}}$.

In the following chapter, we discuss all measurement and recovery operators required to implement these steps. In particular, we discuss the vertex measurement and correction processes in Sec. 4.1, the anyon charge measurements in Sec. 4.2, and the recovery operations in Sec. 4.3.

(b)

(a)
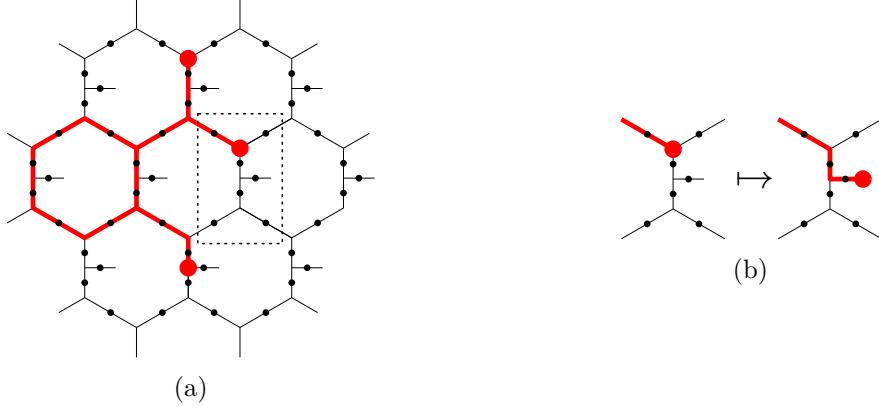
Figure 4.1: (a) A configuration that violates the branching rules by having string-endings in 3 vertices, indicated the red dots. (b) Graphical illustration of a vertex correction operator, acting on the segment inside the dotted box in (a). Note that the resulting configuration does not violate any vertex conditions.

4.1   Vertex measurements and correction

Certain types of errors, such as a single bit-flip error σ_x^e or a coherent error generated by the Pauli-X operator, i.e., $e^{i\theta\sigma_x^e}$, can cause the violation of the vertex projector Q_v of an vertex v adjacent to edge e, and hence induce a vertex error. As shown in Fig. 4.1(a), the vertex error violates the branching rules on the corresponding vertices and corresponds to a broken string ending at the vertex in the string-net configurations of the system wave function. Note that a generic error such as $e^{i\theta\sigma_x^e}$ will put the system wave function in a superposition of violating and not violating the vertex condition at adjacent vertex v. When we measure the vertex operator Q_v, we project to either of the two situations. To resolve this vertex error and project the system back to the string-net subspace $\mathcal{H}_{\text{s.n.}}$, we apply a unitary to pull the broken string into the tail qubit and hence trap the string in the central puncture of the plaquette, as illustrated in Fig. 4.1(b).

Previously, the circuit for measuring the vertex errors has been discovered in Ref. [57]. Here we adopt this circuit to measure the top, middle and bottom vertices, with three ancilla qubits (white dots) denoted by 't', 'm' and 'b' respectively as shown in Fig. 4.2(a-c). The circuit for measuring the top and bottom are symmetric, so we only show the top one in Fig. 4.2(a). For the middle vertex 'm', we also apply a measurement of the tail qubit 4 in the end of the circuit. To respect the usual convention of quantum circuits, we represent the unoccupied edge as $|0\rangle$, which corresponds to the vacuum string label $\mathbf{1}$, and the occupied edge as $|1\rangle$ corresponding to the string label τ. The ancilla qubits are all initialized in the state $|0\rangle$.

Conditioned by the measurement results of the three vertex operators and the tail qubit 4, we apply a correction U_V to fix the vertex error as one out of 14 possible unitaries:

$$U_{m,1}: \quad \vdash \; \mapsto \; \vdash \qquad\qquad U_{m,\tau}: \quad \vdash \; \mapsto \; \vdash$$

$$\vdash \; \mapsto \; \vdash$$

$U_{t,1}:$ $U_{t,\tau}:$

$U_{tm,1}:$ $U_{tm,\tau}:$

$U_{tb,1}:$ $U_{tb,\tau}:$

$U_{tmb,1}:$ $U_{tmb,\tau}:$
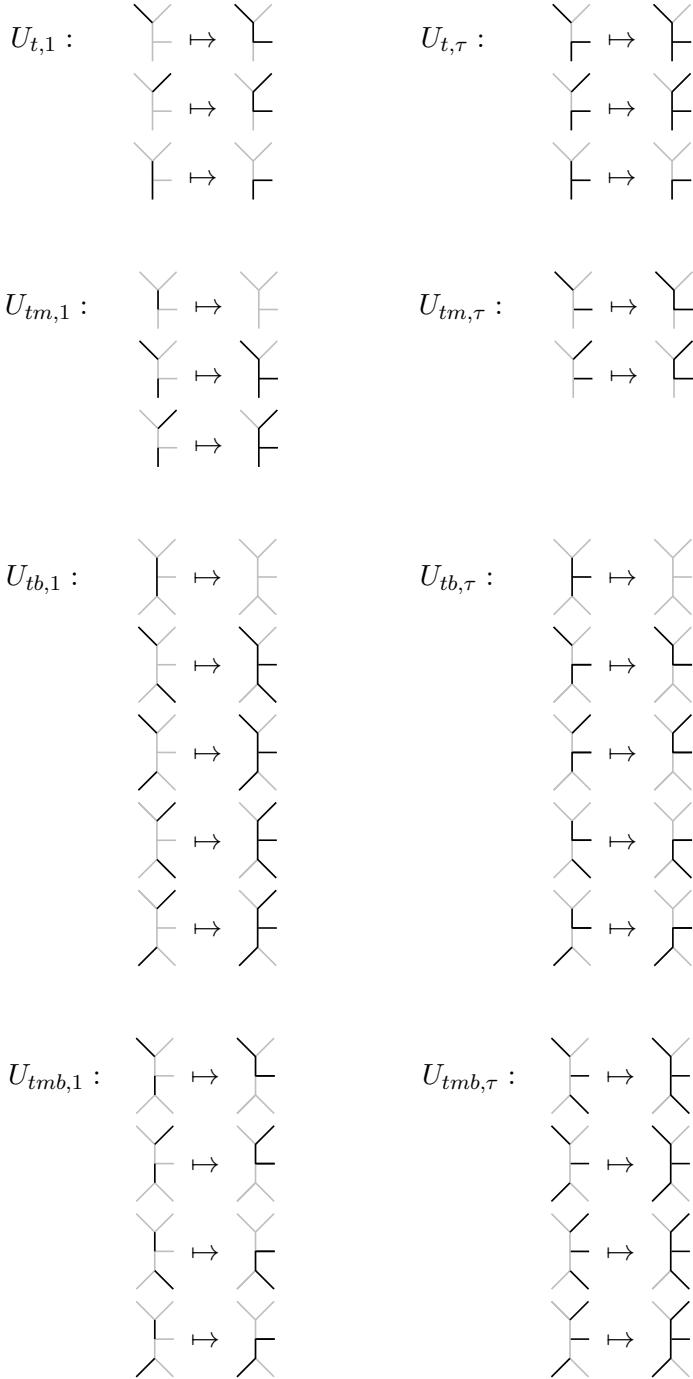
Note that we have omitted the mappings which are mirror symmetric ($t \leftrightarrow b$) to the listed ones.

The corresponding quantum circuits of the above unitaries are listed in Fig. 4.2(d-l). For gates which do not have overlap in qubit support, we can parallelize them in a single time step, as indicated by the dashed boxes. As we can see, most of the unitary circuits have depth 1 or 2, while only one of them, $U_{tb,\tau}$ in Fig. 4.2(i), has depth 4. The overall measurement and correction circuit is summarized in Fig. 4.2(m) where we have parallelized the measurement of the three vertex operators into a depth-5 circuit (in terms of the unitary gates). When taking into account the readout of the ancilla qubits before
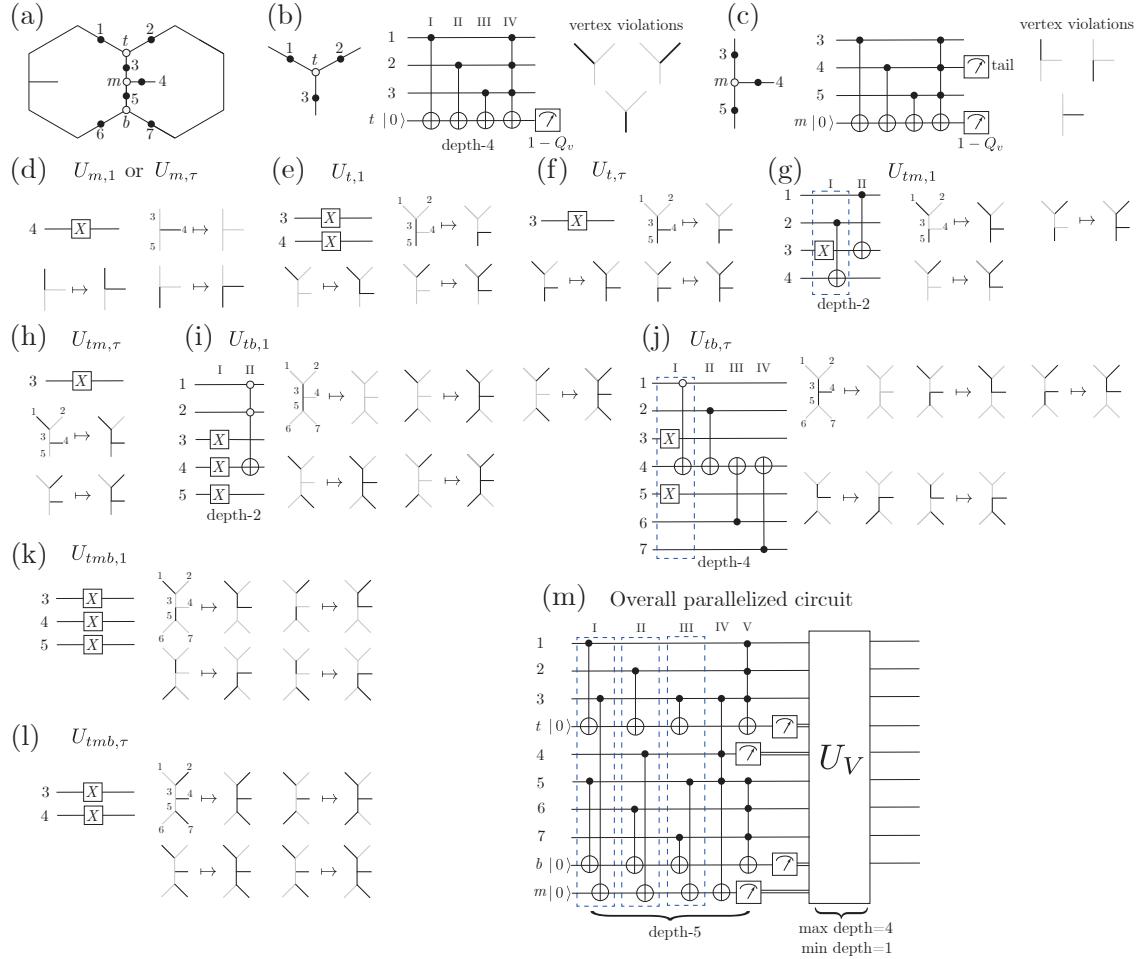
Figure 4.2: Measurement and correction circuit for the vertex errors. (a) The qubit labeling. The black dots represent data qubits, while the white dots represent ancilla qubits for measurements. (b,c) The measurement circuit for the top and middle vertex projectors. The circuit for measuring the bottom vertex can be inferred by symmetry. (d-l) The circuits for pulling an open string into the tail edge when certain vertices are violated. (m) The overall circuit for measuring and correcting vertex errors. The first part of the circuit measures the three vertex projectors. The second part is the unitary U_V conditioned on the measurement results which is summarized in (d-l).

applying U_V, the depth is 6. Note that we have neglected the step of state preparation of the ancilla qubits in the beginning, because in the situation of repetitive syndrome measurements, the ancilla qubits can always be prepared during the application of the correction unitary U_V. Overall, the depth of the measurement circuits ranges from 5 to 9, or from 6 to 10 when taking into account the measurement step.

Note that a different scheme of fixing vertex errors has been previously proposed in Refs. [64] and [65], which also uses tail qubits and hence has a similar spirit.

4.2 Anyon charge measurements

After measuring the vertex operators and applying the corresponding corrections on the extended string-net code, we have transformed the many-body state to the string-net subspace $\mathcal{H}_{s.n.}$, where it can be described in terms of anyonic fusion states. The local qubit errors, including both the Pauli-X and Z types, now result in the creation of anyonic excitations inside $\mathcal{H}_{s.n.}$.

As explained in Sec. 3.4, one can measure the anyonic charge of a plaquette using the central idempotents of the tube algebra. To fully characterize an excitation, we must also measure its tail label. A joint measurement of the anyonic charge and the tail label is achieved by measuring the *irreducible* idempotents of the tube algebra, listed in Eqs. (3.23-3.25), (3.27), and (3.28). Measuring these irreducible idempotents is done in 3 steps:

1. Grow a tube inside the plaquette by introducing ancilla qubits and performing the appropriate quantum circuit, as shown in Fig. 4.5.

2. Measure the tube qubits in the appropriate basis.

3. Either trace out the tube qubits immediately, or first resolve the tube back into the lattice before tracing out the ancillas.

As a basic ingredient, the quantum circuit to implement the F-move (2-2 Pachner move) operation F^{abe}_{cdf} in the Fibonacci Turaev-Viro code is shown in Fig. 4.3. This circuit was first proposed in Ref. [57]. The F-move operation can be viewed as a controlled unitary operation, where the external legs a, b, c, d are control qubits determining the resulting unitary F^{ab}_{cd}, with the matrix elements being $[F^{ab}_{cd}]_{ef}$. For the Fibonacci Turaev-Viro code, the F-matrix is given in Eq. (3.14). The circuit inside the red dashed box, composed of a 5-qubit Toffoli gate in between two single-qubit rotations, applies the conditional unitary corresponding to the F-matrix $F^{\tau\tau}_{\tau\tau}$, where $R_y(\pm\theta) = e^{\pm i\theta\sigma_y/2}$ are single-qubit rotations about the y-axis with angle $\theta = \tan^{-1}(\phi^{-\frac{1}{2}})$. Note that this conditional unitary is only activated if the control qubits a, b, c and d are all in the $|1\rangle$ state corresponding to the string label τ. All the other conditional unitaries are implemented by the rest of the quantum circuit.

Based on the circuit for 2-2 Pachner move (F-move), one can also implement the 1-3 Pachner move with unitary circuit, as shown in Fig. 4.4. The protocol consists of the following steps: (1) Initialize three ancilla qubits (white dots) in state $|0\rangle$. (2) Apply a CNOT gate which entangles the data qubit labeled j to the new ancilla qubit on the same edge, CNOT: $|j\rangle|0\rangle \mapsto |j\rangle|j\rangle$, as shown in (a) and (b). (3) Apply a modular-$\mathcal{S}$ gate on one ancilla to create a tadpole diagram, as shown in (b). The modular–$\mathcal{S}$ does the following transformation: $\mathcal{S} : |0\rangle \mapsto \sum_\lambda \frac{d_\lambda}{D}|\lambda\rangle$. For the Fibonacci Turaev-Viro code, the modular $\mathcal{S}$-matrix is

$$\mathcal{S} = \frac{1}{\sqrt{2+\phi}}\begin{pmatrix} 1 & \phi \\ \phi & -1 \end{pmatrix}. \tag{4.1}$$
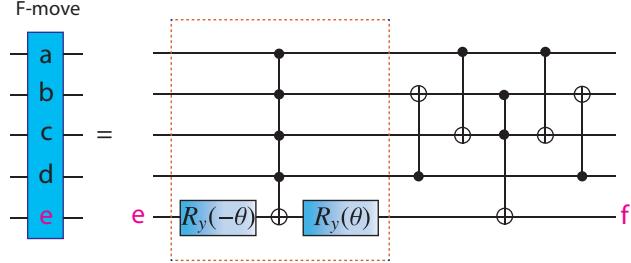
Figure 4.3: Quantum circuit to implement the F-move (2-2 Pachner move) operation F_{cdf}^{abe} for the Fibonacci Turaev-Viro code.
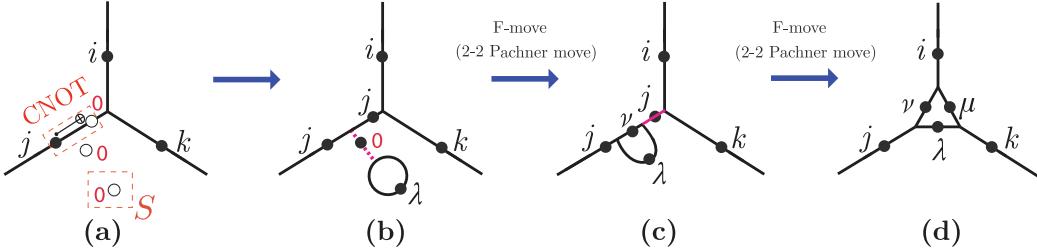
Figure 4.4: Quantum circuit to implement the 1-3 Pachner move for the Fibonacci Turaev-Viro code.

(4) Apply an F-move to absorb the tadpole onto the edge, as shown in (b) and (c). (5) Apply another F-move to sweep edge λ to attach the right leg (with label k), as shown in (c) and (d). From left to right, the circuit effectively fine-grain the lattice by entangling the three ancilla qubits into the code space. One can also reverse the circuit (from right to left) which corresponds to a coarse-graining process disentangling three qubits out of the code space.

The "growing" of a tube onto the tailed lattice can then be implemented by a quantum circuit, as shown in Figs. 4.5 and 4.6. We start with the tailed lattice in Fig. 4.5(a) with data qubits (black dots) residing on every edge. We then introduce three ancilla qubits (white dots) in Fig. 4.5(b) initialized at $|0\rangle$. From panel (b) to (e), we apply a series of operations to achieve a 1-3 Pachner move to add a triangle loop below the tail: (1) Apply a CNOT gate which entangles the data qubit on the tail to the new ancilla qubit on the tail, CNOT: $|y\rangle |0\rangle \mapsto |y\rangle |y\rangle$, as shown in (b) and (c). (2) Apply a modular-$\mathcal{S}$ gate on one ancilla to create a tadpole diagram, as shown in (b), i.e., $\mathcal{S} : |0\rangle \mapsto \sum_\alpha \frac{d_\alpha}{D} |\alpha\rangle$. (3) Apply an F-move to absorb the tadpole onto the tail, as shown in (c) and (d). (4) Apply another F-move to sweep edge α to attach the left edge, as shown in (d) and (e). Afterwards, we apply a sequence of F-moves to sweep edge α around the whole plaquette, after which it ends up in the upper side of the tail. In this way, we have grown a tube. Note that as a result, the qubits on the plaquette and tail edges (with labels k_i and y in Fig. 4.5(a)) get rotated one position counterclockwise. The details of the quantum gates in this circuit are shown in Fig. 4.6. As we can see, in total 9 F-moves have been applied.

After growing the tube, the anyon charge can be inferred by measuring the four qubits on the tube. In order to find the appropriate basis for this measurement, we first note that the growing procedure can be thought of as creating a vacuum bubble (in Fig. 4.5 (c)), stretching it out along the boundary of the plaquette, and finally resolving only half of it into the lattice. The remaining half then constitutes the tube in Fig. 4.5 (j). In terms of
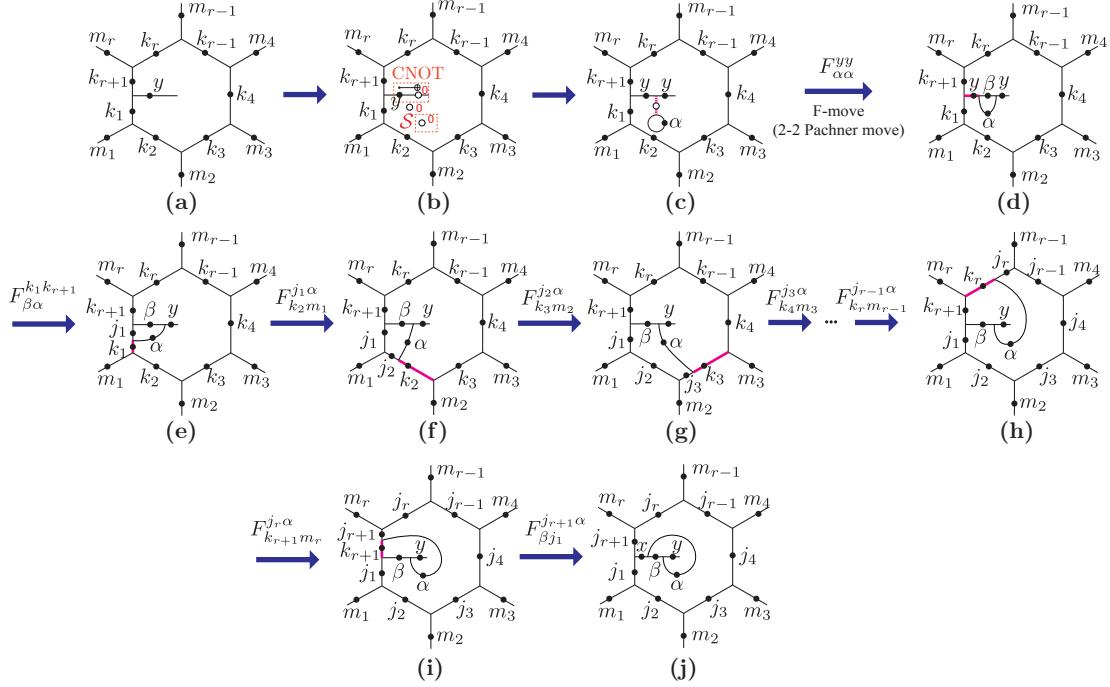
Figure 4.5: Protocol and circuit of growing a tube onto a puncture in a plaquette on a tailed lattice via a sequence of local gates and Pachner moves.

ribbon diagrams the stretched out vacuum bubble inside the plaquette can be written as

$$
\sum_\alpha \frac{d_\alpha}{\mathcal{D}} \quad = \sum_\alpha \frac{d_\alpha}{\mathcal{D}} \quad = \sum_{\alpha,\beta} \frac{d_\alpha}{\mathcal{D}} \frac{v_\beta}{v_\alpha v_y}
$$

$$
= \sum_{\alpha,\beta,x} \frac{1}{\mathcal{D}} \frac{v_x}{v_y} \qquad . \qquad (4.2)
$$

The sequence of F-moves appearing in the grow circuit corresponds exactly to resolving only the outer tube into the lattice[1]. If we denote the initial state of the lattice qubits as $|\Psi_0\rangle = \sum_y \varepsilon_y |\phi_y\rangle \otimes |y\rangle$ and the ancillas are initially in the $|000\rangle$ state, then the action of the grow circuit is

$$
|\Psi_0\rangle \otimes |000\rangle \mapsto \sum_y \varepsilon_y \sum_{\alpha,\beta,x} \frac{1}{\mathcal{D}} \frac{v_x}{v_y} \tilde{O}_{yx\alpha\beta} \left(|\phi_y\rangle \otimes |y\rangle \right) \otimes |y\alpha\beta\rangle , \qquad (4.3)
$$

where we used the following abbreviation for the tube state vectors

$$
\left| \; \alpha \quad y \atop \beta \quad x \right\rangle \equiv |x\rangle \otimes |y\alpha\beta\rangle ,
$$

---

[1] Note that resolving both the inner and the outer tube into the lattice would yield a trivial operation, since they constitute the vacuum bubble together.
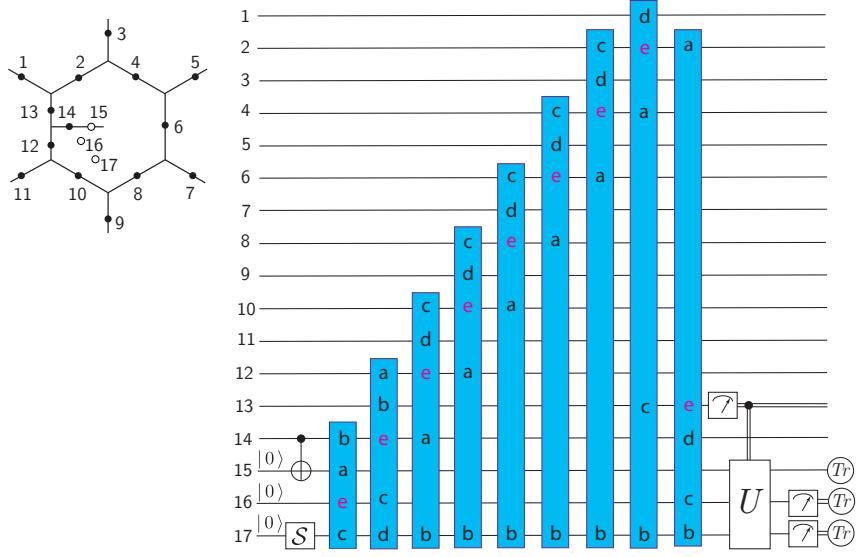
Figure 4.6: The complete quantum circuit for the joint measurement of the anyon charge and tail label of a plaquette. Note that after the grow circuit, qubit 13 corresponds to the tail edge.

and where

$$\tilde{O}_{yx\alpha\beta} \equiv \sum_{\gamma} F_{\alpha y \gamma}^{\alpha x \beta} \, O_{yx\alpha\gamma} \tag{4.4}$$

is the operator which corresponds to resolving a outer tube[2] into the lattice. If a measurement projects the tube qubits onto the state

$$|\psi\rangle = \sum_{\alpha,\beta} A_{\alpha\beta} \, |x\rangle \otimes |y\alpha\beta\rangle \;, \tag{4.5}$$

then the full state gets projected onto

$$|\Phi\rangle \otimes |\psi\rangle = \frac{1}{\mathcal{N}} \left(\sum_{\alpha,\beta} \frac{1}{\mathcal{D}} \frac{v_x}{v_y} \left(A_{\alpha\beta}\right)^* \left(\mathbb{1} \otimes \langle x|\right) \tilde{O}_{yx\alpha\beta}(|\phi_y\rangle \otimes |y\rangle) \right) \otimes |\psi\rangle \;, \tag{4.6}$$

where $\mathcal{N}$ is a normalization factor. We choose the following basis[3] for the measurement of the tube qubits:

$$|\psi_{\mathbf{11}}\rangle = \frac{1}{\mathcal{D}} \left| ^1 \bigcirc{}^{\,1}_{\,1}{}_{1} \right\rangle + \frac{\phi}{\mathcal{D}} \left| ^\tau \bigcirc{}^{\,1}_{\,\tau}{}_{1} \right\rangle \tag{4.7}$$

$$= |0\rangle \otimes \frac{1}{\mathcal{D}} \Big(|000\rangle + \phi \, |011\rangle \Big) \equiv |0\rangle \otimes |\tilde{\psi}_{\mathbf{11}}\rangle \;,$$

---

[2]Note that it has a different shape than our convention Eq. (3.22), hence the F-matrix in Eq. (4.4)

[3]There are seven ways to label a tube according to the Fibonacci fusion rules. Hence, the 7 vectors below span the string-net subspace for the 4 qubits on the tube.

$$|\psi_{\mathbf{1}\tau}\rangle = \frac{1}{\mathcal{D}}\left|\;\right\rangle + \frac{e^{4\pi i/5}}{\mathcal{D}}\left|\;\right\rangle + \sqrt{\phi}\frac{e^{-3\pi i/5}}{\mathcal{D}}\left|\;\right\rangle \tag{4.8}$$

$$= |1\rangle \otimes \frac{1}{\mathcal{D}}\Big(|101\rangle + e^{4\pi i/5}|110\rangle + \sqrt{\phi}e^{-3\pi i/5}|111\rangle\Big)$$

$$\equiv |1\rangle \otimes |\tilde{\psi}_{\mathbf{1}\tau}\rangle\,,$$

$$|\psi_{\tau\mathbf{1}}\rangle = \frac{1}{\mathcal{D}}\left|\;\right\rangle + \frac{e^{-4\pi i/5}}{\mathcal{D}}\left|\;\right\rangle + \sqrt{\phi}\frac{e^{3\pi i/5}}{\mathcal{D}}\left|\;\right\rangle \tag{4.9}$$

$$= |1\rangle \otimes \frac{1}{\mathcal{D}}\Big(|101\rangle + e^{-4\pi i/5}|110\rangle + \sqrt{\phi}e^{3\pi i/5}|111\rangle\Big)$$

$$\equiv |1\rangle \otimes |\tilde{\psi}_{\tau\mathbf{1}}\rangle\,,$$

$$|\psi_{\tau\tau,\mathbf{1}}\rangle = \frac{\phi}{\mathcal{D}}\left|\;\right\rangle - \frac{1}{\mathcal{D}}\left|\;\right\rangle \tag{4.10}$$

$$= |0\rangle \otimes \frac{1}{\mathcal{D}}\Big(\phi|000\rangle - |011\rangle\Big) \equiv |0\rangle \otimes |\tilde{\psi}_{\tau\tau,\mathbf{1}}\rangle\,,$$

$$|\psi_{\tau\tau,\tau}\rangle = \frac{\sqrt{\phi}}{\mathcal{D}}\left|\;\right\rangle + \frac{\sqrt{\phi}}{\mathcal{D}}\left|\;\right\rangle + \frac{1}{\phi\mathcal{D}}\left|\;\right\rangle \tag{4.11}$$

$$= |1\rangle \otimes \frac{1}{\mathcal{D}}\Big(\sqrt{\phi}|101\rangle + \sqrt{\phi}|110\rangle + \frac{1}{\phi}|111\rangle\Big)$$

$$\equiv |1\rangle \otimes |\tilde{\psi}_{\tau\tau,\tau}\rangle \otimes |1\rangle\,,$$

$$|\psi_{\tau\tau,\mathbf{1},\tau}\rangle = \left|\;\right\rangle = |1\rangle \otimes |011\rangle \equiv |1\rangle \otimes |\tilde{\psi}_{\tau\tau,\mathbf{1},\tau}\rangle\,, \tag{4.12}$$

$$|\psi_{\tau\tau,\tau,\mathbf{1}}\rangle = \left|\;\right\rangle = |0\rangle \otimes |111\rangle \equiv |0\rangle \otimes |\tilde{\psi}_{\tau\tau,\tau,\mathbf{1}}\rangle\,. \tag{4.13}$$

Note that the coefficients appearing in the different states are proportional to those in the corresponding irreducible idempotents in Eqs. (3.23)-(3.25), (3.27) and (3.28), or nilpotents in Eq. (3.30) respectively. In fact, these states are precisely the anyonic fusion basis states on Σ_2, as described in Eq. (2.51).

The measurement of the tube qubits in Fig. 4.5(j) is done in three steps:

1. Measure the tail qubit (label x).

2. Apply one of the following unitaries[4] conditioned on the measurement of the tail qubit:

---

[4] The operators below must of course be completed to true unitary operators. The missing terms were left out to improve readability.

(a) if the tail qubit is in state $|0\rangle$ (string label $\mathbf{1}$):

$$U_{\mathbf{1}} = |0\rangle \left(|00\rangle \langle \tilde{\psi}_{\mathbf{11}}| + |11\rangle \langle \tilde{\psi}_{\tau\tau,\mathbf{1}}| + |10\rangle \langle \tilde{\psi}_{\tau\tau,\tau,\mathbf{1}}| \right),$$

(b) if the tail qubit is in state $|1\rangle$ (string label τ):

$$U_{\tau} = |0\rangle \left(|00\rangle \langle \tilde{\psi}_{\tau\tau,\tau}| + |11\rangle \langle \tilde{\psi}_{\tau\tau,\mathbf{1},\tau}| + |01\rangle \langle \tilde{\psi}_{\mathbf{1}\tau}| + |10\rangle \langle \tilde{\psi}_{\tau\mathbf{1}}| \right).$$

3. Measure the two qubits (16 and 17 in Fig. 4.6) in the Z-basis.

Once the tube qubits have been measured in the basis Eqs. (4.7)-(4.13) using the procedure above, we can trace out the three ancilla qubits (15, 16 and 17 in Fig. 4.6, constituting the inner three edges of the tube in Fig. 4.5(j)) to return to the initial tailed lattice layout with a single tail qubit in each plaquette (i.e., the configuration in Fig. 4.5(a)). This results in the following POVM:

$$\left\{ \mathcal{P}^{\mathbf{11}}, \ \mathcal{P}^{\mathbf{1}\tau}, \ \mathcal{P}^{\tau\mathbf{1}}, \ \frac{1}{\phi^2}\mathcal{P}^{\tau\tau}_{\mathbf{1}}, \ \frac{1}{\phi}\mathcal{P}^{\tau\tau}_{\tau}, \ \frac{1}{\phi}\mathcal{P}^{\tau\tau}_{\mathbf{1}}, \ \frac{1}{\phi^2}\mathcal{P}^{\tau\tau}_{\tau} \right\}. \tag{4.14}$$

Note that both a $\tau\tau_{\mathbf{1}}$ and a $\tau\tau_{\tau}$ excitation corresponds to two different measurement outcomes. However, within each of these pairs, the post-measurement states are not identical. For instance, a $\tau\tau_{\mathbf{1}}$ excitation can result in measurement outcomes $|\psi_{\tau\tau,\mathbf{1}}\rangle$ and $|\psi_{\tau\tau,\mathbf{1},\tau}\rangle$. Obtaining outcome $|\psi_{\tau\tau,\mathbf{1}}\rangle$, effectively applies the $\mathcal{P}^{\tau\tau}_{\mathbf{1}}$ idempotent, meaning the post-measurement state will have trivial tail label $\mathbf{1}$. On the other hand, the outcome $|\psi_{\tau\tau,\mathbf{1},\tau}\rangle$ corresponds to the application of the $\mathcal{P}^{\tau\tau}_{\mathbf{1}\tau}$ nilpotent. Since $\mathcal{P}^{\tau\tau}_{\mathbf{1}\tau} = \mathcal{P}^{\tau\tau}_{\tau}\mathcal{P}^{\tau\tau}_{\mathbf{1}\tau}\mathcal{P}^{\tau\tau}_{\mathbf{1}}$, this means the plaquette initially contained a $\tau\tau_{\mathbf{1}}$ excitation, which gets transformed to a $\tau\tau_{\tau}$ excitation in the post-measurement state. The situation for a $\tau\tau_{\tau}$ excitation is analogous. Hence these measurements do not preserve the tail label in case of a $\tau\tau$ anyon, and a subsequent measurement might yield a different outcome. It is important to note that this only affects the tail label *within one anyon sector*, the anyon label itself cannot be altered by subsequent measurements.

In case one prefers to preserve the tail label of excitations in subsequent measurements[5], an additional step is required before tracing out the three ancillas. This step consists of resolving the tube into the lattice by applying the gates of the grow circuit in reverse, as indicated in Fig. 4.7. For measurement outcome $|\psi\rangle$ defined Eq. (4.5), the resolving process is equivalent to the following transformation on the post-measurement state in Eq. (4.6):

$$|\Phi\rangle \otimes |\psi\rangle \mapsto \sum_{\alpha,\beta} A_{\alpha\beta} O_{xy\alpha\beta} \left(|\Phi\rangle \otimes |x\rangle \right) \otimes |000\rangle. \tag{4.15}$$

This guarantees that the initial tail label y will indeed be recovered. For instance, when obtaining the $|\psi_{\tau\tau,\mathbf{1}\tau}\rangle$ measurement outcome, the resolving process result in the application of the $\mathcal{P}^{\tau\tau}_{\tau\mathbf{1}}$ nilpotent on top of the $\mathcal{P}^{\tau\tau}_{\mathbf{1}\tau}$ nilpotent applied by the measurement, resulting in the combined action $\mathcal{P}^{\tau\tau}_{\mathbf{1}} = \mathcal{P}^{\tau\tau}_{\tau\mathbf{1}}\mathcal{P}^{\tau\tau}_{\mathbf{1}\tau}$, meaning that the tail label is now preserved.

The same result can be achieved by using repeated measurements with the circuit in Fig. 4.6. In case one measures a $\tau\tau$ excitation but finds that the tail label gets flipped (corresponding to the tube states in Eqs. (4.12) and (4.13), and the last two entries in

---

[5]Possibly, this could improve the performance of certain decoders. Furthermore, we will assume this for the numerical simulations discussed in Chapter 7.

the POVM Eq. (4.14)), each subsequent measurement[6] has a fixed probability of flipping the tail label back to its initial value (as determined by the first measurement). For a $\tau\tau_1$ excitation, it follows from Eq. (4.14) that the probability that n measurements are required before the post-measurement state has a trivial tail label is $\phi^{-(n+1)}$. Hence, on average, ϕ^2 measurements are required to ensure that the tail label is preserved for a $\tau\tau_1$ excitation. Likewise, for a $\tau\tau_\tau$ excitation the probability of needing n measurements to recover the initial tail label is $\phi^{-(2n-1)}$, resulting in an average of ϕ measurements. Whether one should choose for the longer measurement circuit depicted in Fig. 4.7 or for repeated measurements with the shorter measurement circuit depicted in Fig. 4.6 depends on what types of excitations are more likely to appear.
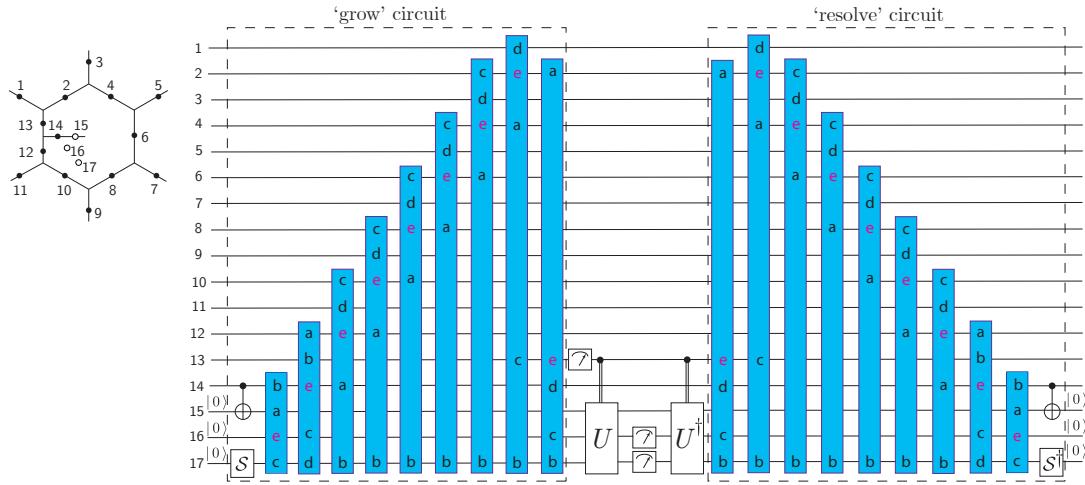
Figure 4.7: Alternative quantum circuit for the joint measurement of the anyon charge and tail label of a plaquette, which does preserve the tail label.

The procedure described above determines the anyon charge of a single plaquette. By repeating it for all plaquettes, we obtain the complete error syndrome (in the form of the anyonic content of every plaquette), which must then be passed to a decoding algorithm to determine the appropriate recovery operation to be performed (see Chapter 5).

---

[6]Provided that no errors happen on qubits in or adjacent to the plaquette between these repeated measurements.

4.3 Recovery operations

After extracting the error syndrome as described above, the decoding algorithm will suggest a sequence of actions to take to fuse pairs of anyonic excitations along specific paths. There are two types of fundamental revovery operations: *Fuse*, and *Exchange.* Here we will introduce the quantum circuits that implement these recovery operations. The key components of these circuits include the 2-2 Pachner moves and the 1-3 Pachner moves as introduced in Chapter 3 and the corresponding unitary circuits introduced in Sec. 4.2.

4.3.1 Fuse

We start by defining *fuse* as a recovery operation that fuses anyons in neighboring plaquettes. The fuse protocol is shown in Fig. 4.8 for the case where an anyon $\boldsymbol{a}_2$ inside a plaquette is fused with the anyon $\boldsymbol{a}_1$ inside its left neighboring plaquette. Other fusing directions will have a very similar process and we omit showing all of them. The application of this protocol to a particular example of a ribbon graph state is illustrated in Fig. 4.9 with further parallelization of the steps in the protocols shown in Fig. 4.8.

Before detailing the quantum circuit to fuse a pair of neighboring anyons, we consider what such an operation means on the level of the ribbon graphs introduced in Sec. 3. We consider two plaquettes p_1 and p_2, and wish to fuse the anyons contained within them. For this purpose, we pick an anyonic fusion basis which explicitly shows the total charge $\boldsymbol{b} = b^+ b^-$ of the anyons $\boldsymbol{a}_1 = (a_1^+ a_1^-)_{\ell_1}$ and $\boldsymbol{a}_2 = (a_2^+ a_2^-)_{\ell_2}$ contained in the two plaquettes:

$$|\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}}\rangle = \quad\equiv\quad , \tag{4.16}$$

where we have denoted all other branch labels on the fusion tree collectively by $\vec{\boldsymbol{c}}$, and have omitted showing all other leaf labels $\{\boldsymbol{a}_3, \boldsymbol{a}_4, \ldots\}$. We will refer to these other leaf labels collectively as $\vec{\boldsymbol{a}}_{\text{ot}}$, writing $\vec{\boldsymbol{a}} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \vec{\boldsymbol{a}}_{\text{ot}}]$. For simplicity, we only show the corresponding ribbon graphs, and not their embedding in the fattened lattice as described in Sec. 3.5.

After the syndrome measurement detailed in Sec. 4.2, the system is in definite charge eigenstate for all plaquettes, which means that the leaf labels $\vec{\boldsymbol{a}}$ are fixed in the state superposition, while the internal labels $\{\boldsymbol{b}, \vec{\boldsymbol{c}}\}$ are not. In general we then have

$$|\Psi_0\rangle = \sum_{\boldsymbol{b},\vec{\boldsymbol{c}}} \alpha_{\boldsymbol{b},\vec{\boldsymbol{c}}} |\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}}\rangle . \tag{4.17}$$

The fusion basis state $|\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}}\rangle$ appearing in this decomposition, can be rewritten as follows:

$$|\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}}\rangle = \frac{1}{\mathcal{D}} \quad = \frac{1}{\mathcal{D}} \sum_{\ell=1,\tau} F_{b^+ b^- \ell}^{b^- b^+ 1} \quad , \tag{4.18}$$

where we have used the property that vacuum loops can be "doubled" (see Eq. 2.46 in Sec. 2.4), and have applied an F-move on the double ribbon b^+b^- of corresponding to the total charge of plaquettes p_1 and p_2. The fusion of the anyons in these plaquettes corresponds replacing the upper part of the diagram in Eq. 4.18 by a single puncture as follows

$$\tag{4.19}$$
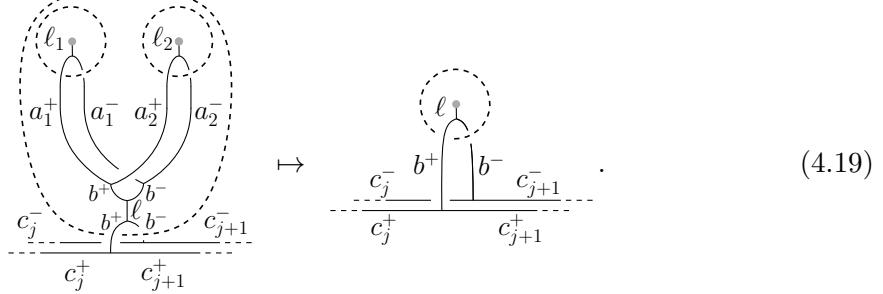
The fusion can then be represented by the following map on anyonic fusion basis states:

$$\text{fuse}: \quad |\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{[\boldsymbol{a}_1,\boldsymbol{a}_2,\vec{\boldsymbol{a}}_{\text{ot}}]}\rangle \mapsto \sum_{\ell=\boldsymbol{1},\tau} F_{b^+b^-\ell}^{b^-b^+\boldsymbol{1}} |\psi_{\vec{\boldsymbol{c}}}^{[\boldsymbol{b}_\ell,\vec{\boldsymbol{a}}_{\text{ot}}]}\rangle \;, \tag{4.20}$$

where[7]

$$|\psi_{\vec{\boldsymbol{c}}}^{[\boldsymbol{b}_\ell,\vec{\boldsymbol{a}}_{\text{ot}}]}\rangle = \quad$$ $$\quad . \tag{4.21}$$

The post-measurement state $|\Psi_0\rangle$ then gets mapped to

$$|\Psi_0\rangle \mapsto |\Psi_1\rangle = \sum_{\boldsymbol{b},\vec{\boldsymbol{c}}} \alpha_{\boldsymbol{b},\vec{\boldsymbol{c}}} \sum_{\ell=\boldsymbol{1},\tau} F_{b^+b^-\ell}^{b^-b^+\boldsymbol{1}} |\psi_{\vec{\boldsymbol{c}}}^{[\boldsymbol{b}_\ell,\vec{\boldsymbol{a}}_{\text{ot}}]}\rangle \;. \tag{4.22}$$

Measuring the tail label and the anyonic charge of plaquette p_1 after fusing the two anyons, will yield outcomes $\boldsymbol{b}_\ell$ with the probabilities

$$p(\boldsymbol{b}_\ell) = \sum_{\vec{\boldsymbol{c}}} \sum_{\ell=\boldsymbol{1},\tau} \left| \alpha_{\boldsymbol{b},\vec{\boldsymbol{c}}} F_{b^+b^-\ell}^{b^-b^+\boldsymbol{1}} \right|^2 \;. \tag{4.23}$$

Note that the probabilities for the outcomes of a tail measurement, depend only on the total charge $\boldsymbol{b}$. Furthermore, in any measurement scheme where the anyonic charge of a plaquette is measured independently of its tail label, these measurements will commute.

At the end of the charge measurement cycle, we normally trace out the tube qubits or resolve the tube into the lattice. However, before fusing a pair of anyons, we can keep the tube on the left plaquette (instead of growing it again) in the beginning of the fusion protocol, as shown in Fig. 4.8(b). The purpose is to carry out the subsequent F-moves for an edge to climb around the tube without encountering any broken string as in the case with a single tail qubit on the left plaquette.

The central idea of the protocol is to merge the two plaquettes and hence the corresponding plaquette excitations, while also combining the tail qubits such that the vertex excitations also get merged. Overall, this process fuses the two anyon charges inside the two plaquettes. The protocol starts at the end of measurement cycle t. One first applies

---

[7]To be precise, we should have written $|\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{[\boldsymbol{b}_\ell,\boldsymbol{11}_1,\vec{\boldsymbol{a}}_{\text{ot}}]}\rangle$, since plaquette p_2 now has a trivial anyonic charge, and the total charge of plaquettes p_1 amd p_2 remains unchanged.
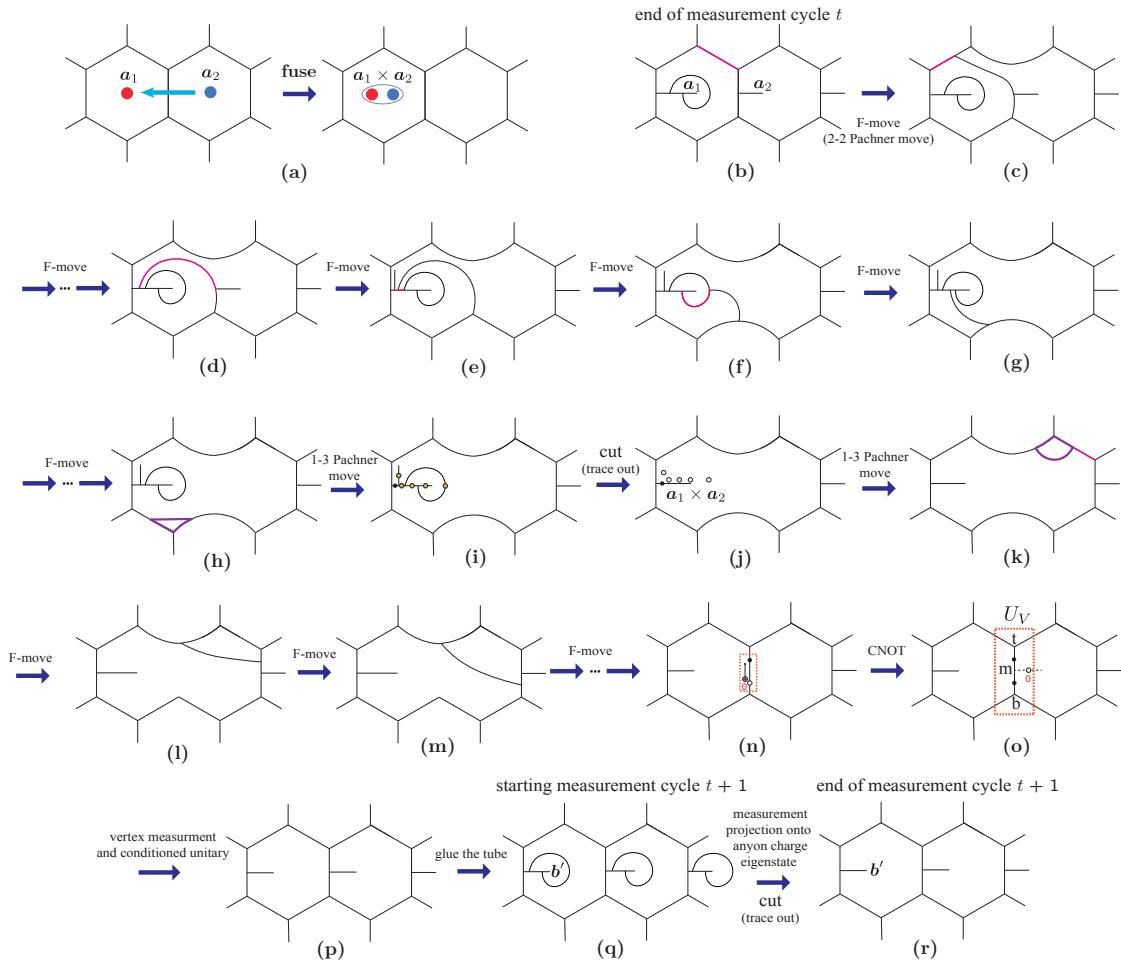
Figure 4.8: (a) The basic idea of the *fuse* protocol: fuse anyons $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$ by moving $\boldsymbol{a}_2$ on the right plaquette to the left plaquette and then fuse them into a single anyon with charge $\boldsymbol{a}_1 \times \boldsymbol{a}_2$. (b) Starting the fuse protocol in the end of measurement cycle t. The tube on the left plaquette is preserved for the convenience of the fuse protocol. (c) Use an F-move to sweep the middle edge towards the left plaquette. (d) Sweep the central edge onto the left tail. (e) Use an F-move to shuttle the right tail onto the left tail. (f-h) Keep sweeping the middle edge until it reaches the bottom vertex of the left plaquette. (i) Apply a 1-3 Pachner move to shrink the triangular bubble on the bottom vertex. (j) Cut the tubes and extra tail by tracing out all the qubits except the one on the root. Now the two plaquettes have been merged into one and the two anyons are fused into one with charge $\boldsymbol{a}_1 \times \boldsymbol{a}_2$, which is in general in a superposition state. (k) Apply a 1-3 Pachner move to grow a triangular bubble on the top vertex of the right plaquette. (l-n) Use F-moves to keep sweeping the new edge towards the bottom edge and hence grow a new plaquette on the right. Apply a CNOT from the qubit residing on the middle edge to another ancilla qubit initialized at $|0\rangle$, which splits the edge into two. (o, p) Prepare an ancilla qubit at $|0\rangle$ on the new tail. Measure the vertices t, m and b, and apply the unitary U_V conditioned on the measurement result to pull the broken string in and fix all the vertex errors. We have now rebuilt the tailed lattice in (p). (q,r) In the next measurement cycle $t+1$, start by growing and measuring the tube to project the fused anyon charge $\boldsymbol{a}_1 \times \boldsymbol{a}_2$ to a definite anyon charge $\boldsymbol{b}'$, and then resolve all the tubes back to the tailed lattice.
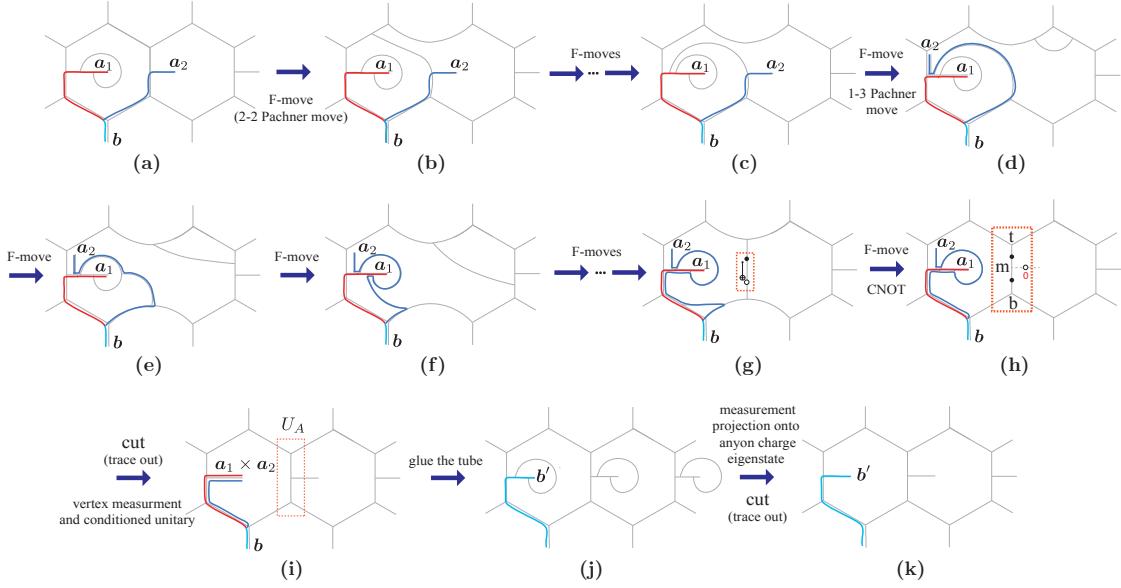
Figure 4.9: Illustration of the *fuse* protocol with an example of a ribbon graph state with two anyons a_1 and a_2 and their total branch charge b. Note that we have further parallelized the fuse protocol in Fig. 4.8 by simultaneously doing the plaquette merging and growing a new plaquette on the right.

an *F*-move (2-2 Pachner move) that sweeps the central edge between the two plaquettes towards the left plaquette, as shown in Fig. 4.8(b). After several *F*-moves, the central edge now climbs onto the tail of the left plaquette as shown in Fig. 4.8(d). Now we apply an *F*-move to move the tail on the right plaquette onto the left tail in Fig. 4.8(e). In Fig. 4.8(e), we further sweep the central edge onto the tube. After several steps of further moves, the central edge now forms a triangular bubble with the other two edges in the bottom of the left plaquette, as shown in Fig. 4.8(h). One now applies a 1-3 Pachner move to absorb the bubble into the vertex in Fig. 4.8(i), and now the left and right plaquettes are merged into a single large plaquette. The corresponding procedures with the example of the ribbon graph state is also illustrated in Fig. 4.9(a-h). Now we trace out all the qubits residing on the left tail and only a single tail qubit remains in the lattice, as shown in Fig. 4.8(j). We are now left with two ribbons going into the same tail and which correspond to the fused anyon charge $a_1 \times a_2$. Note that due to the non-Abelian nature, the fused anyon can be in a superposition of multiple anyon charge eigenstates, and hence may not have a definite charge yet. For example, if $a_1 = a_2 = \tau \mathbf{1}_\tau \equiv \tau \mathbf{1}$, then one has $a_1 \times a_2 = \mathbf{11} + \tau \mathbf{1}$.

Since the two initial plaquettes are now merged into a single one, we need to grow another plaquette to recover the original lattice, which is accomplished in Fig. 4.8(k-p). The growing is achieved by first applying a 1-3 Pachner move in Fig. 4.8(k) to grow a triangular bubble on the top vertex, followed by a sequence of *F*-moves that sweep the newly created edge to form the original middle edge dividing the two plaquettes, illustrated in Fig. 4.8(n). Now we need to regrow the tail on that middle edge. We first apply a CNOT from the qubit (black dot) residing on the edge on an ancilla qubit (white dot) initialized in the state $|0\rangle$ in order to split the edge in two, as shown in Fig. 4.8(n). We further prepare another ancilla qubit (white dot) in the state $|0\rangle$ corresponding to the tail qubit (dashed line) in Fig. 4.8(o). We then measure the vertex projectors Q_v on the three vertices t, m and b and apply the corresponding unitary U_V to pull the potentially broken string into

60
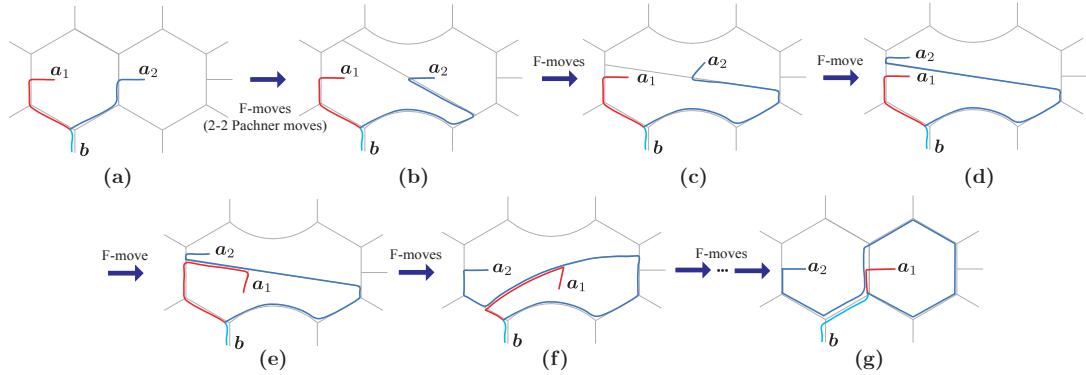
Figure 4.10: Illustration of the counter-clockwise exchange protocol with an example of ribbon graph state. (a-f) A sequence of *F*-moves gradually rotates the two plaquettes and the tails accordingly. (g) In the end of the protocol, the position of the two anyons have been exchanged with the ribbons attached to them being braided.

the tail based on the measurement result V, using the measurement and unitary circuits previously shown in Fig. 4.2 in Sec. 4.1. We note that if there is no error introduced in the recovery process, the vertices t, m and b should not have any error and the strings on them should not be broken. However, in reality, additional errors will be introduced during the recovery, and one hence needs to apply the vertex measurements and unitary U_V to pull the string in. Although we have introduced the plaquette growing process separately, we note it can actually be parallelized with the merging of the two original plaquettes, as illustrated in Fig. 4.9(d-i).

After performing the fusion protocol described here, one obtains a state where the second plaquette has a vacuum charge ($B_p = 1$), while the first plaquette contains the total charge of the two excitations that where fused. Note that the first plaquette is not in a definite charge state, instead, it contains a superposition of all possible values of the total charge of the two fused excitations. Likewise, its tail label is also in a superposition of **1** and τ. In principle, no more actions are required, since the excitations have been fused as expected. However, in the numerical simulation described in Chapter 7, we will assume that after every fusion process, the fusion outcome is projected to some definite anyon label and tail label. This can simply be achieved by including the charge measurement circuit of Sec. 4.2 in the first plaquette at the end of the fusion procedure.

4.3.2 Exchange and move

The second type of recovery operation is a counterclockwise/clockwise *exchange* (braiding) of anyons in neighboring plaquettes. The need for this operation arises when one wants to transport an anyon along a path (which might include plaquettes containing nontrivial excitations). In Fig. 4.10, we show the counterclockwise exchange protocol along with the example of the same ribbon graph state shown above in Fig. 4.9. The central idea is to rotate the central edge dividing the two plaquette such that the two plaquettes undergo a counterclockwise exchange. One also needs to shuttle the tail accordingly. The final ribbon graph state after the exchange operation is shown in Fig. 4.9(g), and a counterclockwise exchange/braid can be clearly seen on the ribbon graph. Note that, when applied to excitations with a definite anyon and tail labels, these labels are preserved by the exchange procedure.
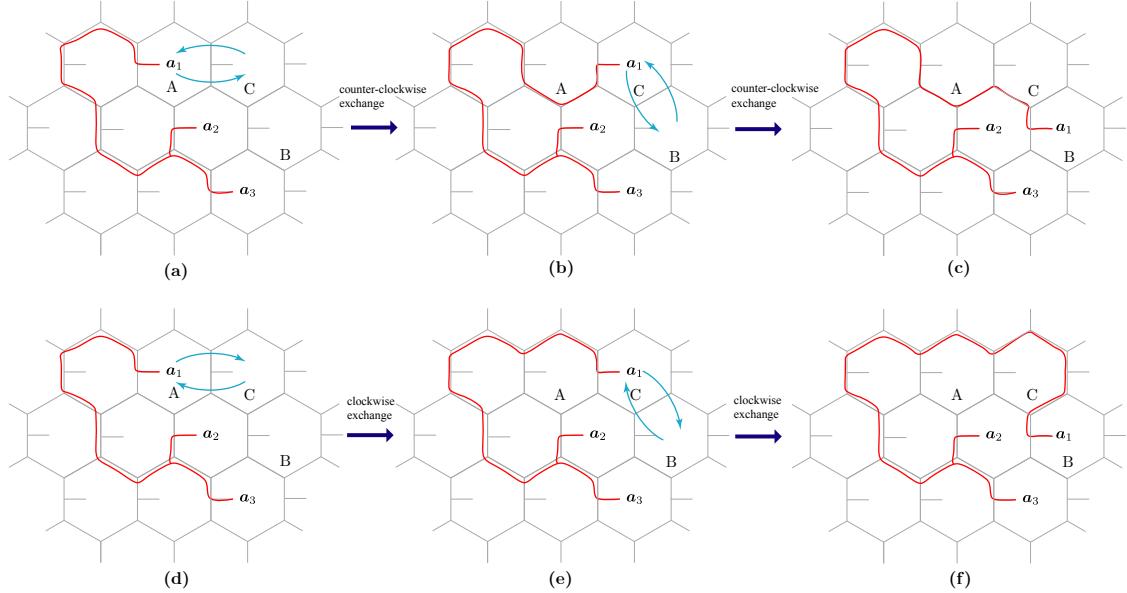
Figure 4.11: Two equivalent processes to move anyon $\boldsymbol{a}_1$ from plaquette A to plaquette B through the path A→C→B. (a-c) Implement the move through two steps of counter-clockwise exchange between plaquettes A and C, and then plaquettes C and B. The anyon reaches plaquette B in (C). (d-f) Implement the move through two steps of clockwise exchange of neighboring plaquettes. Note that the ribbon graph states (b) and (e) are equivalent because the exchanges happen with vacuum plaquettes and since ribbons can be pulled across those (see Eq. (2.45) in Sec. 2.4, and also Fig. 3.4 in Sec. 3.2). In the same way, the state in (c) is equivalent to the one in (f). However, if any plaquette on the movement path contains a nontrivial anyon, the resulting states are not equivalent, since they correspond to different sequences of braid-moves on the fusion state.

We define *move* as a process of shuttling anyon $\boldsymbol{a}$ from plaquette A to plaquette B along some specified path, as illustrated in Fig. 4.11 with a path $A \rightarrow C \rightarrow B$. This is achieved through a sequence of clockwise or counterclockwise exchanges with all plaquettes along this path. Note that if any other anyons are encountered on the path, the results of the move procedures with clockwise and counterclockwise exchanges are not identical, as they differ by a braid in the anyonic fusion space.

In case all plaquettes along the path have a vacuum charge, one could also implement the movement procure as a sequence of fusions, since fusion with a **11** anyon is trivial. However, note that this does not guarantee that the tail label of the excitation that is being moved, is preserved. In particular, when fusing any $\tau\tau$ anyon ($\tau\tau_{\mathbf{1}}$ or $\tau\tau_{\tau}$) with a neighboring trivial (vacuum) anyon, the probabilities for the measurement outcomes of the tail qubit after the fusion are $1/\phi^2$ and $1/\phi$ for the outcomes $|0\rangle$ and $|1\rangle$, respectively, independent of the tail label of the excitation prior to the fusion. This is due to the fact that *only* the doubled anyon label of an excitation is topologically protected, its tail label is not and can thus be changed by local interactions if one is not careful. Furthermore, the exchange procedure is considerably simpler than the fusion protocol.

5 | Decoding algorithms

After the error syndromes (anyon charges) on all the plaquettes are measured using the circuits discussed in the previous chapter, this syndrome information is passed to a decoder, which in turn outputs the recovery operations required to correct the errors.

In this chapter we discuss two types of decoding algorithms. The first type is the clustering decoder which has been previously applied to decode a phenomenological model of Fibonacci anyons [55]. This decoder is based on a hierarchical clustering algorithm [53] and shares a similar strategy to the hard-decision renormalization-group decoder [76]. The clustering decoder does not use the detailed syndrome information corresponding to the anyon type. Instead, it just uses the limited syndrome information of the presence of absence of anyon, i.e., whether the anyon charge is nontrivial or trivial (in the doubled vacuum sector **11**).

The second type is a fusion-aware iterative minimum-weight perfect matching (MWPM) decoder which modifies the standard MWPM algorithm and incorporates the detailed syndrome information corresponding to anyon type into the decoding strategy. As a comparison, we also show the results of a '*blind*' iterative MWPM decoder which does not use the detailed syndrome information of anyon type but only the presence or absence of anyon. Due to the use of the detailed syndrome information, the logical error rate of the fusion-aware iterative MWPM decoder is lower than the other one.
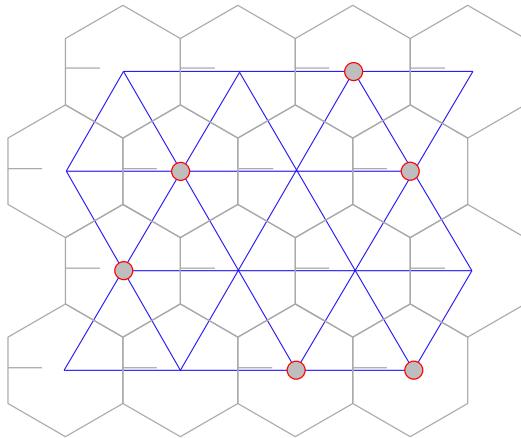
Figure 5.1: The decoding graph of the extended string-net code. By connecting the center of the plaquettes of the tailed lattice, we obtain a triangular lattice (blue) as the decoding graph. The anyons charge (grey circles) on the plaquettes of the tailed lattice serve as the syndromes in the error correction scheme and are located on the vertices of the triangular decoding graph.

We will indicate the syndrome using a decoding graph, shown in Fig. 5.1. This is a triangular lattice which is the dual of the original trivalent graph on which the extended string-net code is defined. Anyon charges located in the plaquettes of the trivalent graph, correspond to syndromes on vertices of this triangular decoding graph.

5.1 Clustering decoder

The spirit of the clustering decoder is based on the charge conservation of anyons. As will be discussed in Sec. 7.4, local noise generates certain pairs or more generally clusters of anyons. However, since these anyon clusters are generated from the vacuum sector $\mathbf{11}$, i.e., the ground space of the extended string-net code, the total charge of these anyon clusters should still be trivial ($\mathbf{11}$) due to charge conservation. Therefore, if we fuse all the anyons within this particular cluster by moving them towards the same plaquette, we will get a total trivial vacuum charge $\mathbf{11}$, i.e., all the anyons are annihilated back to the vacuum sector. In this case, we have applied a correct recovery operation to correct all the errors.

However, the decoder does not know which cluster each anyons were generated from based on the syndrome information, i.e., the anyon charge. Therefore, the decoder may not always apply a correct recovery operation to fuse all the anyons originated from the same cluster. Instead, the decoder may fuse anyons from different clusters, and effectively joins the two clusters. Due to the total charge conservation and the following fusion rule: $\mathbf{11} \times \mathbf{11} = \mathbf{11}$, we know the total anyon charge of the two clusters is still zero. If we fuse all the anyons in these two clusters together, all the charges will still be annihilated into the vacuum $\mathbf{11}$. The same argument applies to merging multiple clusters. Now as long as the size of the joined cluster is much smaller than the system size, all the errors can be corrected by merging them into the vacuum $\mathbf{11}$. On the other hand, if the joined anyon cluster forms a non-contractible (homologically nontrivial) region, i.e., either wrapping around a cycle of a torus or more generally a high-genus surface in the context of a closed manifold, or connecting two or more gapped boundaries in the context of an open manifold, the recovery operation with the merging procedure may still annihilate all the anyons but

end up applying a nontrivial logical operator along certain homologically nontrivial cycle
[1] which will cause a logical error and the decoder fails. It is also possible that such a non-contractible anyon cluster will have a nonzero total charge, therefore there is some residual anyon after the merging procedure which cannot be annihilated. In both cases, we will claim a failure of the clustering decoder. Therefore, in order to apply a successful recovery operation, we need to make sure that the individual clusters are annihilated before growing to a size comparable to the system size, i.e., with a linear dimension comparable to the code distance.

The clustering decoding algorithm for the Fibonacci Turaev-Viro code defined on a torus is summarized below by the pseudo-code and will be explained in detail with a concrete example:

Algorithm (clustering decoder)

 # Measure the anyon charge of each plaquette on the tailed trivalent lattice; store the nontrivial anyon charge in a list "anyon_charge"
```
anyon_charge = get_syndrome(state);
```

 # Initialize a cluster for each plaquette with a nontrivial charge
```
clusters = Cluster(anyon_charge);
```

 # Join any connected clusters
```
join(clusters, 1);
```

 # While there is more than one nontrivial charge
```
while size(anyon_charge)>1
```

    # Fuse all anyons within each cluster and measure the resulting charge
```
    for cluster in clusters
        fuse_anyons(cluster);
```

    # Discard any empty cluster with trivial vacuum charge **11***;*
```
    clusters = non_vac(clusters);
```

    # Grow each cluster by a unit length on the triangular decoding graph
```
    grow(clusters, 1);
```

    # Join any overlapping clusters
```
    join(clusters, 0);
```

end

 # If the list of nontrivial anyon charge is empty, i.e., with no remaining anyon
```
if anyon_charge == []
```
    # The decoder declares success
```
    return Success
```
 # If there is a single nontrivial remaining charge
```
else
```
    # The decoder claims failure
```
    return Failure
```

---

[1]In the case of open manifold with gapped boundaries, the logical operator corresponds to nontrivial relative homology cycle.

As described above, for a given state of the entire system, we first measure the tube operator within each plaquette of the tailed lattice using the circuit in Figs. 4.5 and 4.6 and obtain the corresponding anyon charge as the syndromes on the decoding graph. This process is defined as `get_syndrome()` in the decoding algorithm, with the `state` as the input and `anyon_charge` as the output. Now we initialize a cluster on each non-zero anyon charge, defined as `clusters=Cluster(anyon_charge)` in the above algorithm. An example for the decoding graph on a torus with the initial `clusters` (blue shadow) is shown below:
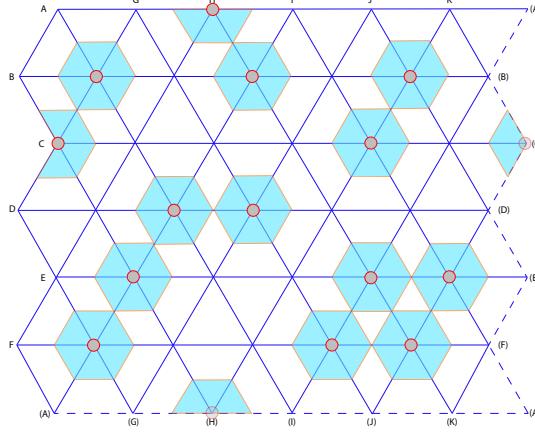
In the next step, we call `join(clusters, 1)` to join neighboring clusters which are separated by only 1 lattice spacing. Within each cluster we randomly choose a root anyon, and *move* all the other anyons to the root and finally fuse all of them into a single anyon. The *move operation* is the recovery operation introduced in Sec. 4.3.2 and Fig. 4.11. We note that the order of moving and detailed path do not affect the resulting state after fusing all the anyons within each order, and therefore we could choose an arbitrary order. The only requirement is that the chosen paths must be inside each cluster. For simplicity, we choose the shortest paths (with shortest graph distance) towards the root anyons from all the remaining anyons (indicated by the red arrows in the figure below), and start moving the anyons in an order with an increasing graph distance between the remaining and root anyons in our numerical simulation.
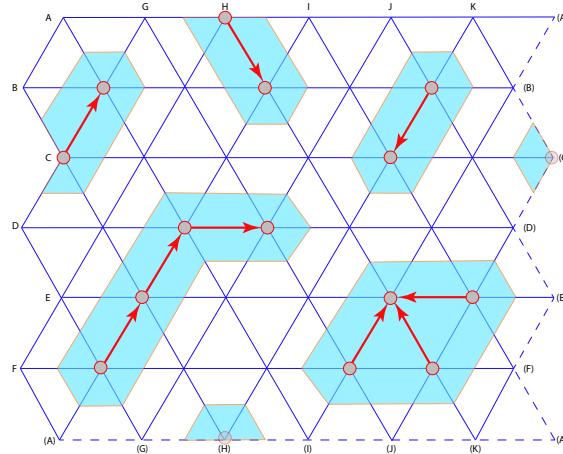
During the classical simulation of the error correction process, which we will detail in Chapter 7, we will actively check whether any anyon path l forms a non-contractible

(homologically nontrivial) cycle, which will give rise to a logical error as previously illustrated in Fig. 3.9 in Chapter 3. The anyon path here is the sum of the error path and the recovery path: $l = l_e + l_r$. We note that, although the decoder itself does not have access to such information, the classical simulation does. We can hence claim `failure` of the decoder in our Monte Carlo simulation, if such non-contractible cycle occurs.

If the decoder does not fail, we continue to measure all the charges of the root anyons. If the measured charge in a particular cluster is zero, i.e., in the vacuum sector **11**, we discard the corresponding cluster. The list of clusters hence gets updated via *clusters=non_vac(clusters)*.
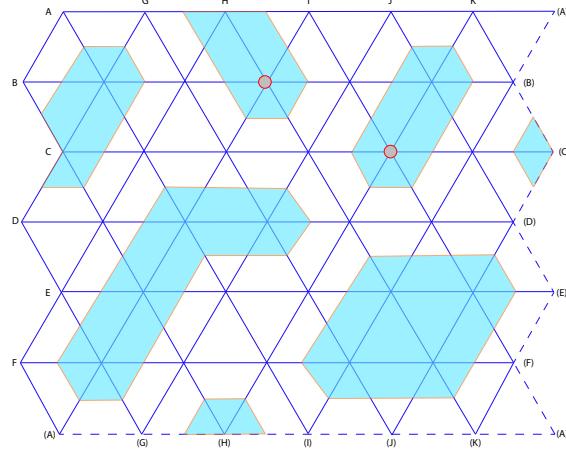
In the next step, we need to call `grow(clusters, 1)` to grow each remaining cluster by one unit of lattice spacing in all possible directions (six directions for each vertex within the cluster), as indicated by the blue arrows in the figure below. As shown in the figure, the remaining two clusters (blue) grow to the larger clusters (green and orange):
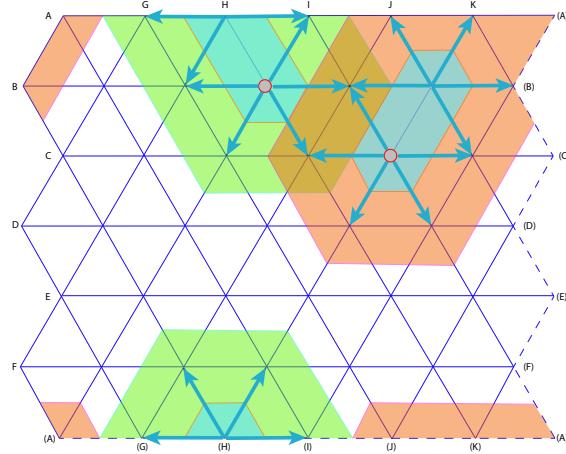
Next, we call `join(clusters, 0)` to join overlapping clusters, i.e., any two clusters sharing a common set of vertices will be joined into a single one and this process is done iteratively until there are no overlapping clusters. In the current example, the green and red clusters in the above figure are joined into a single bigger cluster (blue) in the following figure:
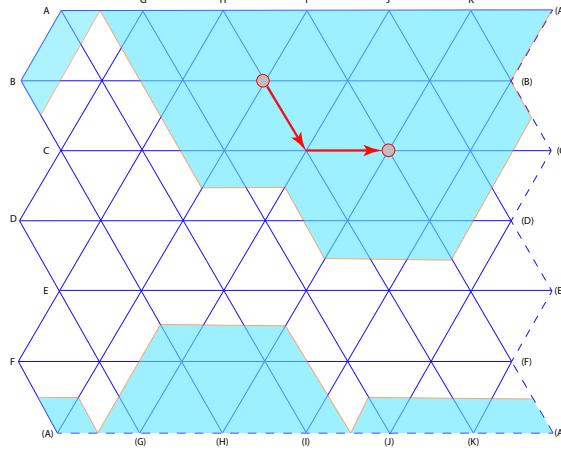
After the merging of clusters, we repeat the above process, i.e., fuse all the anyons within each cluster (indicated by the red arrows in the above figure) and discard the empty cluster with trivial total charge **11** (vacuum), and then further grow and join the remaining clusters. This iteration is stopped when we have zero or one remaining nontrivial anyon charge.

After the end of the above iteration, we are at the final stage of our decoding algorithm. If there is no any remaining nontrivial anyon charge, i.e., `anyon_charge == []`, we then have successfully corrected all the errors, and the decoder declares `success`. In the other situation with a single nontrivial remaining anyon, we end up with a logical error, and will claim `failure` of the decoder.

5.2 Fusion-aware iterative matching decoder

The fusion-aware minimum-weight perfect matching (MWPM) decoder applies matching on the anyons with a preferred order of matching certain types of anyons according to the underlying structure in the anyon pair generation under the Pauli noise, which will be discussed in detail in Sec. 7.4.

As we have stated before, the non-Abelian Fibonacci fusion rule $\tau \times \tau = \mathbf{1} + \tau$ implies that a typical fusion of anyons in DFIB has probabilistic outcome into one of several fusion channels, e.g., $\tau\mathbf{1} \times \tau\tau = \mathbf{1}\tau + \tau\tau$. This is in stark contract to Abelian anyon models, where fusion processes are always deterministic. This difference can be seen using a graphical representation of the fusion rules (which we will call a *fusion graph* from now on) as shown in Fig. 5.2(a,c,d). Since standard minimum-weight perfect matching (MWPM) decoder of the Abelian surface code is based on deterministically fusing pairs of Abelian anyons into the vacuum sector, this type of decoder will not work properly when applied to the Fibonacci string-net code.

In Ref. [51], a MWPM decoder has been applied to the non-Abelian phenomenological Ising-anyon model, corresponding to the Ising category $\{\mathbf{1}, \sigma, \psi\}$. The corresponding fusion rules are:

$$\psi \times \psi = \mathbf{1}, \quad \psi \times \sigma = \sigma, \quad \sigma \times \sigma = \mathbf{1} + \psi, \tag{5.1}$$

as illustrated in the fusion graph Fig. 5.2(b). The basic strategy there is to apply MWPM to pair up and fuse a particular type of anyons: the σ-anyons. As indicated by the above fusion rule, the matched pair of anyons either fuse into the vacuum or the ψ-anyon. When new ψ-anyons are generated from fusion, one further applies MWPM to pair and fuse all the pre-existing and newly generated ψ-anyons. Afterwards, one can clean up all the anyons and hence correct all the errors if the decoder succeeds.
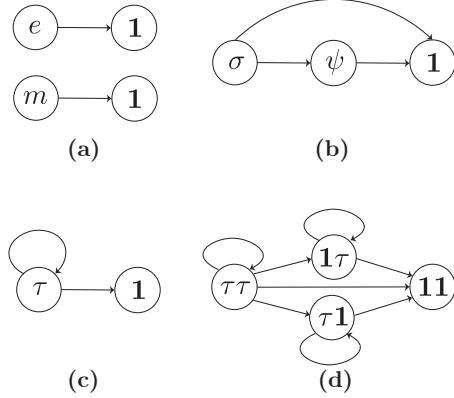
Figure 5.2: A graphic representation of the fusion rules (fusion graph) of (a) Toric code; (b) Ising category; (c) Fibonacci category (FIB); (d) doubled Fibonacci category (DFIB) corresponding to the Fibonacci Turaev-Viro code. The fusion graph is a directed graph with the directed edge pointing from pair of incoming anyons represented by one vertex to the possible fusion outcome represented by another vertex. (a) and (b) have non-cyclic fusion structure, while (c) and (d) are cyclic.

In the case of Fibonacci Turaev-Viro code, we can also adopt a similar strategy. We first apply MWPM to pair up and fuse all the existing anyons, which will either fuse to vacuum or a certain type of anyons. We then apply MWPM again to match the newly-generated anyons. We iterate above process until all the anyons are annihilated into the vacuum and then declare `success` of the decoder if no logical error occurs. If there is any residual anyon or a logical error corresponding to any anyon-path forming a non-contractible cycle, we claim `failure` of the decoder. One can further introduce a particular order for merging different types of anyons, similar to the above case in the phenomenological Ising-anyon model in Ref. [51].

However, this type of MWPM decoder does not have a threshold. The underlying reason is related to the particular fusion structure of the Fibonacci and doubled-Fibonacci categories. As shown in Fig. 5.2, the Ising category has a *non-cyclic* fusion structure [54] such that the fusion graph does not have any cycle (an edge pointing from and to the same vertex, i.e., the same anyon type). In contrast, both the Fibonacci category (FIB) and the doubled Fibonacci category (DFIB) have a *cyclic* structure, i.e., containing at least a cycle in the fusion graph, which complicates the fusion process. As we can see, the Ising fusion rule naturally implies a pair creation process: a pair of anyons, either two σ-anyons or two ψ-anyons are generated from the vacuum **1**. Applying the matching can immediately annihilate these pairs into the vacuum. If one of the σ-anyon might further split into a σ-anyon and a ψ-anyon, we get a triplet (σ, σ, ψ) which still contains a pair of σ. When applying the above MWPM algorithm, two σ's will be paired up and must generate a ψ-particle if the anyons still stay close by. A further matching with the other ψ can annihilate the pair of ψ's back to the vacuum **1**.

As we can see, such pair creation mechanism naturally fits the above MWPM algorithm. However, the Fibonacci category (FIB) and its doubled version (DFIB) does not have such simple pair creation structure. In FIB, two τ-anyons can fuse into τ as well due to the cyclic structure, which can further fuse with another τ into the vacuum **1**. This means that not only a pair of τ can be generated from the vacuum **1**, but also a triplet (τ, τ, τ). In fact, due to the cyclic structure, any number of τ can be created from the vacuum. The absence of a pair generation structure is in contrast to the situation in

the Ising category where only pairs of σ can be generated from the vacuum. Similarly, in DFIB, two $\tau\mathbf{1}$ can fuse into $\tau\mathbf{1}$, which means a triplet $(\tau\mathbf{1}, \tau\mathbf{1}, \tau\mathbf{1})$ or more $\tau\mathbf{1}$ can be generated from the vacuum $\mathbf{11}$. Similar argument applies to triplets of $\mathbf{1}\tau$ and $\tau\tau$. In the example of Pauli noise, we will see in Sec. 7.4 that a single Pauli-X or Pauli-Y error can create 2, 3 or 4 anyons, without a clear pair creation mechanism like the σ-anyons in the Ising category.

Now we can give a proof that the above MWPM decoder does not have a threshold for the Fibonacci Turaev-Viro code (DFIB). Consider a counterexample shown in Fig. 5.3, where two triplets of $\tau\mathbf{1}$ anyons are created by $O(1)$ errors from the vacuum and spatially separated by $O(d)$, i.e., the order of the code distance. Note that since these triplets were created separately, the total fusion space factorizes into a tensor product structure of the fusion spaces of the two separate triplets, as is indicated by the red curve diagrams (see Sec. 7.6) in Fig. 5.3(a). According to the algorithm stated above, we will first apply MWPM to pair up all the present anyons[2]. As indicated by the highlighted edges in Fig. 5.3(b), two of the anyons within each triplet are matched, while one anyon within one cluster is matched with the one in another triplet. We then fuse the matched anyon pair by transporting one anyon (randomly chosen) along the highlighted path to fuse with the other anyon. Since the two anyons involved in the second fusion process do not have a definite total charge, both $\mathbf{11}$ and $\tau\mathbf{1}$ are possible outcomes. The probabilities of these events are $1/\phi^2$ and $1/\phi$, respectively, as follows from Eq. 4.23 in Chapter 4. Both outcomes will lead to the same situation eventually: if the outcome is $\tau\mathbf{1}$ that anyon will be located near one of the other $\tau\mathbf{1}$ anyons (obtained from the two other fusion processes), and will then be matched with it by the MWPM decoder (and that fusion will yield a $\tau\mathbf{1}$ anyon). In case the outcome is $\mathbf{11}$, one ends up with the configuration in Fig. 5.3(c). Since anyons from the two triplets were fused, the two remaining anyons are now part of the same fusion tree, as indicated by the their joint curve diagram. We then apply MWPM to pair up the two remaining anyons. A short path is preferred by the matching algorithm as shown by the highlighted edge. When fusing the anyons along this short path, the anyon path forms a non-contractible (homologically nontrivial) cycle, which equivalently applies a chain of errors of length $O(d)$ and hence induces a logical error.

In summary, when applying the above MWPM algorithm, even $O(1)$ errors may not be corrected, meaning the effective code distance in this decoding scheme is only $O(1)$ instead of $O(d)$. Therefore, the logical error rate is not exponentially suppressed as a function of code distance d, and the decoder does not have a threshold. As one can easily see, such an issue also exists in the case of single copy of the Fibonacci category (FIB). Nevertheless, we note that such problem is not present in the case of the triplet (σ, σ, ψ) in Ising anyon models mentioned above. Using the MWPM algorithm for the Ising-anyon model, one will first apply matching only to the σ-anyons within each triplet, which fuse into a ψ-particle, and then apply matching to the ψ-anyons, which fuse the two remaining ψ-anyons within each cluster. The non-cyclic feature of Ising fusion rule avoids the above issue.

Due to this issue, we need to modify the MWPM algorithm for FIB and DFIB. As we have seen, the main issue is that sometimes the matching algorithm pairs up anyons which are far apart and do not belong to the same anyon cluster generated from the vacuum. Therefore, we propose an iterative MWPM algorithm. In each iteration, we apply the MWPM algorithm, but only fuse the matched anyon pairs which are within a certain fusing radius r (starting at one lattice spacing in the first iteration). When fusing

---

[2]If the total number of anyons is odd, then there will be a single anyon which is not paired up with any other anyon.
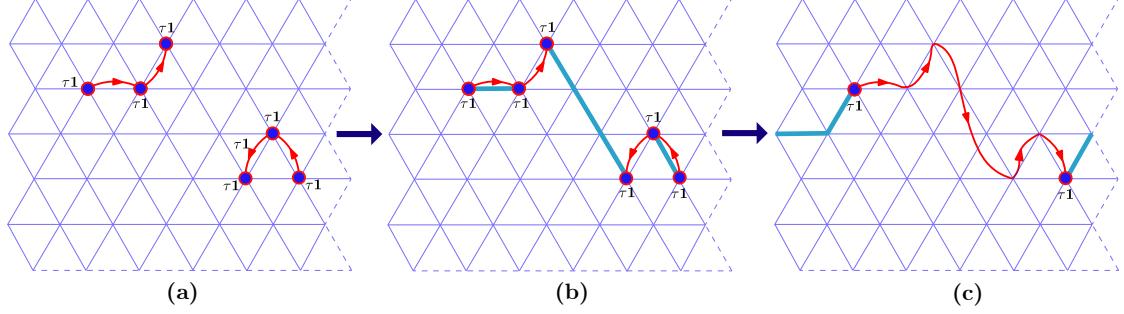
Figure 5.3: Illustration of the absence of threshold when applying the usual MWPM algorithm to the Fibonacci Turaev-Viro code. (a) Two triplets of $\mathbf{1}\tau$ anyons are created from the vacuum by $O(1)$ errors and are spatially separated apart at the order of code distance $O(d)$. The underlying fusion structure is indicated by the curve diagrams. (b) Pairs chosen by the MWPM decoder (indicated by highlighted blue edges). For each pair of the matched anyons, one of them (randomly chosen) will be transported along the highlighted path to fuse with the other. (c) Result of the fusion processes suggested by the MWPM decoder. The MWPM algorithm will now select the shortest path (highlighted in blue), which will create a non-contractible cycle when the fusion is performed.
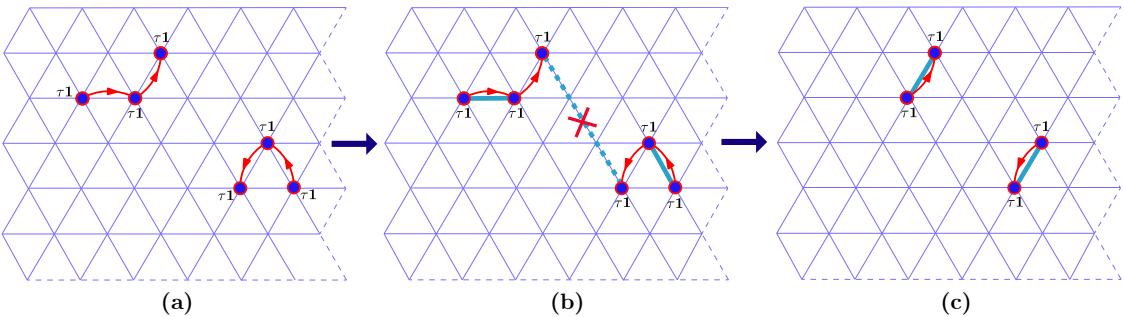
Figure 5.4: Resolve the issue in the previous example in Fig. 5.3 by applying the iterative MWPM algorithm. In panel (b), one applies MWPM algorithm to match three pairs of anyons. The matched pair involving anyons from different triplet clusters has a graph distance 3, exceeding the fusing radius $r = 1$ in the first iteration. Therefore, the fusion of this matched pair is rejected. In panel (c), the two previously matched pairs were fused and there are four remaining anyons. Apply another MWPM will match two pairs of anyons within their own cluster and hence successfully correct the errors.

a certain anyon pair, we may generate some new anyons. Therefore, we match and fuse anyons until all possible anyon pairs within the fusing radius r have been fused and begin the next iteration by increasing the fusing radius by one lattice spacing, i.e., $r \rightarrow r+1$. We keep doing this until all the anyons are annihilated. During this process, if any anyon path forms a non-contractible cycle, we claim `failure`. If there is a single remaining anyon in the end of the algorithm, we also claim `failure`. Otherwise, we declare `success`. As we will see in the next section, the numerical simulation suggests that such an iterative MWPM algorithm does actually have a threshold.

The above iterative MWPM algorithm is summarized below:

Algorithm (iterative MWPM)

 # Measure the anyon charge of each plaquette on the tailed trivalent lattice; store the nontrivial anyon charge in a list "anyon_charge"
```
anyon_charge = get_syndrome(state);
```

 # Initialize the fusing distance at one lattice spacing
```
r=1;
```

 # While there is more than one anyon left
```
while size(anyon_charge) > 1
```

> *# Apply the matching algorithm; only record anyon pairs with maximal distance r and the corresponding path connecting them*
> ```
> [pairs, paths]=MWPM(anyon_charge, r);
> ```
>
> *# If no anyon pair within the fusing radius is matched*
> ```
> if pairs == []
> ```
>   *# Increase the fusion radius by one lattice spacing*
>   ```
>     r = r+1;
>   ```
>
> ```
> else
> ```
> *# Fuse the anyons along the paths given by MWPM; update the list "anyon_charge" with the fused anyon charge; drop the vacuum charge **11** from the list "anyon_charge"*
> ```
>     anyon_charge = fuse_anyons(pairs,
>     paths);
> ```

```
end
```

 # If there is no anyon excitations left
```
if anyon_charge == []
```
    # The decoder declares success
    ```
    return Success
    ```
 # If there is a single anyon left
```
else
```
    ```
    return Failure
    ```

As we can see, the iterative MWPM algorithm shares a similar hierarchical structure with the clustering algorithm where the cluster is grown by one lattice space in each iteration. When we apply this algorithm to the previous triplet example, we see that the previous issue is resolved, as illustrated in Fig. 5.4. In the first iteration, we have fusing
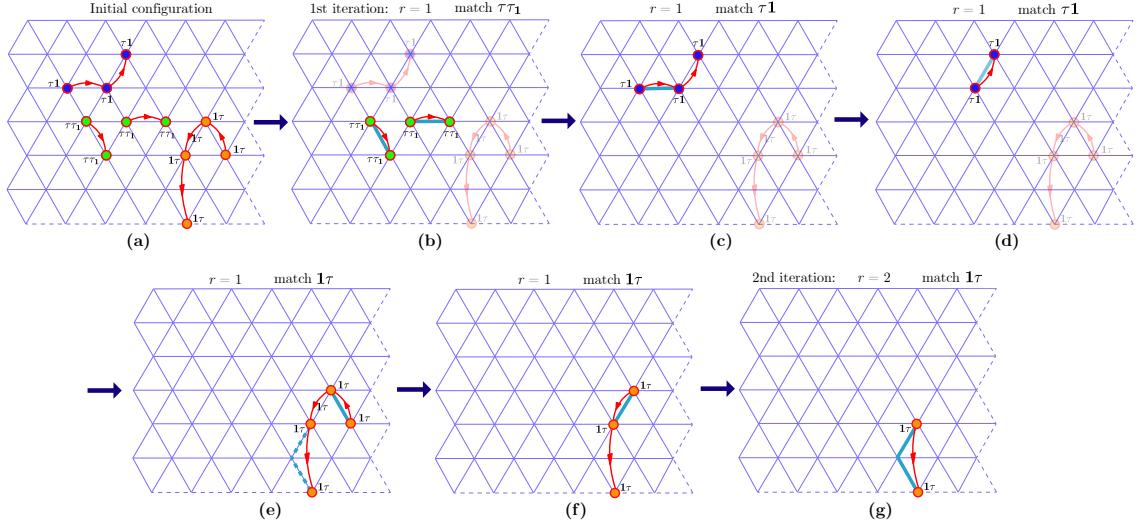
Figure 5.5: Illustration of the fusion-aware iterative MWPM decoder with an example of initial anyon configuration. (a) Four anyons clusters are generated from the vacuum sector. The anyon charges are shown on the nodes and branches of the curve diagram (see the branch label on the cluster with four 1τ anyons). The two $\tau\tau_{\mathbf{1}}$ clusters are created by σ_z-errors. (b) Match and fuse $\tau\tau_{\mathbf{1}}$ anyons only with fusing distance $r = 1$. (c, d) Match and fuse $\tau\mathbf{1}$ anyons only with fusing distance $r = 1$. (e) Match and fuse $\mathbf{1}\tau$ anyons only with fusing distance $r = 1$. The matched path (dashed lines) between the lower two anyons has length 2 and hence exceeds the fusing distance $r = 1$. The fusion between these two anyons is hence rejected. (f) Continue to match and fuse $\mathbf{1}\tau$ anyons with fusing distance $r = 2$. (g) In the second iteration, the decoder increases the fusing distance to $r = 2$, and hence is able to match and fuse the remaining pair of $\mathbf{1}\tau$ anyons.

radius $r = 1$. So when we apply matching, the fusion of the matched pair with a long separation (three lattice spacing) is rejected by the decoder since the separation exceeds the fusing radius. When we do the matching again, these anyons get matched and fused with their neighboring $\tau\mathbf{1}$ anyons generated from the last fusion. Therefore, all the fusion still occur within each triplet cluster and all the anyons are fused back into the vacuum.

In the above iterative MWPM algorithm, we have not used any detailed syndrome information about the anyon types, similar to the case in the clustering algorithm. We call such an algorithm *'blind'*. On the other hand, there are definitely certain features and patterns in the anyon creation process. Therefore, we expect that an iterative MWPM decoder which is fusion-aware, i.e., has access to the detailed information of anyon type, should have a better performance. In particular, in the context of depolarizing noise, we will show in Sec 7.4 that a single σ_z error generates a pair of anyons with charge $\tau\tau_{\mathbf{1}}$ in neighboring plaquettes. This is a distinct feature of σ_z error when the system is exposed to all three types of noise ($\sigma_x, \sigma_y, \sigma_z$). In order to exploit such error feature, we require the decoder to have the following behavior: when the fusion-aware MWPM decoder sees a pair of $\tau\tau_{\mathbf{1}}$ anyons, it will prefer to match and fuse them first, rather than to pair any of them with other types of anyons. This choice will tend to clean up the anyon clusters generated by σ_z errors first, and avoids to fuse anyons belonging to different anyon clusters which will join these two clusters.

We hence propose the following modification to obtain the fusion-aware iterative MWPM decoder. In each iteration, we start matching and fusing specific type of anyons in the following order $[\tau\tau_{\mathbf{1}}, \tau\mathbf{1}, \mathbf{1}\tau, \tau\tau_{\tau}, \tau\tau_{\mathbf{1}}, \tau\mathbf{1}, \mathbf{1}\tau]$. We only start fusing the next type

73

once the current type of anyon pairs within the fusion radius have all been fused. After we exhausted all the types, we match and fuse the remaining anyons 'blindly' within the fusing radius r, i.e., ignoring the detailed anyon charge type. In this case, one is allowed to match and fuse anyons with different anyon types. We then increase the fusing radius by one lattice spacing ($r \rightarrow r + 1$) and continue to the next iteration.

Now we explain the reason of choosing the above order of fusion. According to the fusion graph in Fig. 5.2(d), one can choose the fusion order from left to right such that the anyon types generated in the fusion process are never on the left of the current type on the fusion graph. In that case, we can choose either the fusion order $[\tau\tau, \tau\mathbf{1}, \mathbf{1}\tau]$ or $[\tau\tau, \mathbf{1}\tau, \tau\mathbf{1}]$. Note that here we do not distinguish $\tau\tau_{\mathbf{1}}$ and $\tau\tau_\tau$ and they make no difference in the fusion graph. However, due to the structure of σ_z-noise as mentioned above, we prefer to match and fuse $\tau\tau_{\mathbf{1}}$ first. On the other hand, $\tau\tau_\tau$-anyons are less likely to be produced, so we fuse them after $\tau\mathbf{1}$ and $\mathbf{1}\tau$. Then when we start fusing $\tau\tau_\tau$-anyons, we are again on the very left of the fusion graph and will again produce all types of anyons ($\tau\tau_{\mathbf{1}}$, $\tau\mathbf{1}$, $\mathbf{1}\tau$, and $\tau\tau_\tau$). Therefore, after exhausting fusion of $\tau\tau_\tau$, we still need to start fusing $\tau\tau_{\mathbf{1}}$, $\tau\mathbf{1}$ and $\mathbf{1}\tau$ in turn.

We illustrate this algorithm with a specific example in Fig. 5.5. As we can see that the initial configuration in Fig. 5.5(a) has several anyon clusters generated from the vacuum sector by local noise, and each cluster has a total vacuum charge $\mathbf{11}$. In particular, two clusters of $\tau\tau_{\mathbf{1}}$ anyons (green) are created by σ_z errors. The anyon clusters are all connected together, which poses significant challenge for the decoder. Now the decoder starts with fusing radius $r = 1$ and only perform matching on the $\tau\tau_{\mathbf{1}}$ anyons as shown in Fig. 5.5(b). As we see, the two pairs of $\tau\tau_{\mathbf{1}}$ are immediately matched and fused into the vacuum sector, and we are left with two separate anyon clusters with $\tau\mathbf{1}$ (blue) and $\mathbf{1}\tau$ (orange) anyons. Due to the significant separation, decoding becomes much easier now. The decoder continues to match and fuse only the $\tau\mathbf{1}$ anyons, which takes two steps to annihilate all the $\tau\mathbf{1}$ anyons, as shown in Fig. 5.5(c, d). The decoder now switches to match the $\mathbf{1}\tau$ anyons. It performs two steps of matching and fusion with fusing radius $r = 1$ in (e,f), and has exhausted the matching of all possible anyon pairs with a separation of one lattice spacing. Now the decoder increases the fusing distance to $r = 2$, and matches the remaining pair of $\mathbf{1}\tau$ anyons separated with two lattice spacing in (g), which are then fused into the vacuum. We can conclude that in this case, the fusion awareness makes the decoding much easier than the blind MWPM decoder which could prefer to join neighboring anyon clusters into a larger cluster and hence increases the chance of failure.

As we will see in the numerical results in the next section, this fusion-aware iterative MWPM decoder does give an overall improvement in logical fidelity over the 'blind' iterative MWPM decoder.

We note that Ref. [51] has also adopted such fusion-aware strategies to the clustering decoder in the context of Ising anyon model, i.e., clustering different types of anyons separately in a certain chosen order. We have also adopted this strategy to the Fibonacci Turaev-Viro code (DFIB). However, we do not observe any advantage compared to the clustering decoder described in Sec. 5.1 which does not use the detailed syndrome information of anyon types. This is potentially related to the sophisticated cyclic fusion structure of Fibonacci anyon as opposed to the non-cyclic fusion structure of Ising anyons.

Part III

Threshold simulation

6 | Tensor network representations of anyonic fusion states

It is well known that string-net ground states allow for a remarkably simple PEPS (projected entangled pair state) representation, constructed from the algebraic data of the input category [77, 78]. Below, we will illustrate how such a tensor network representation is obtained, and then expand on these known results by constructing an explicit PEPS representation of any anyonic fusion basis state on a tailed lattice.

6.1   A tensor network representation for the string-net ground states

We start by constructing the PEPS representation of the ground state

$$\mathcal{N} \prod_p B_p \left(\otimes_e |\mathbf{1}\rangle_e \right) , \tag{6.1}$$

where $\mathcal{N}$ is a normalization factor. Graphically, (a patch of) this state can be represented as
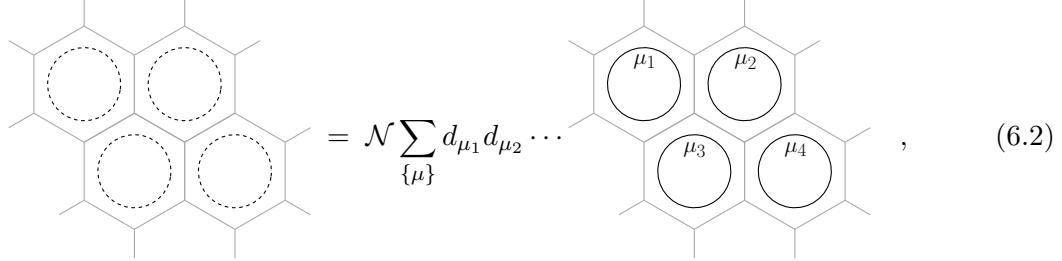
$$, \tag{6.2}$$

where qudits in the $|\mathbf{1}\rangle$ state are represented by the gray edges. To find the state from this graphical notation, one has to resolve the loops appearing in Eq. (6.2) into the lattice. This is done in two steps. First we fuse the neighboring loops along each edge using Eq. (2.33):

Then, the resulting configuration at every vertex is resolved into the lattice using Eq. (2.52):
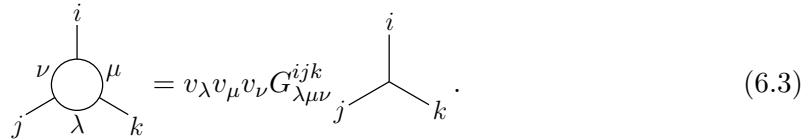
$$\tag{6.3}$$

 The PEPS tensor is obtained by splitting the factor in Eq. (6.3) evenly between the two adjacent vertices. The result is
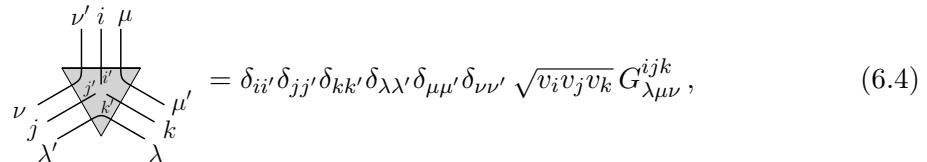
$$= \delta_{ii'} \delta_{jj'} \delta_{kk'} \delta_{\lambda\lambda'} \delta_{\mu\mu'} \delta_{\nu\nu'} \sqrt{v_i v_j v_k}\, G^{ijk}_{\lambda\mu\nu} , \tag{6.4}$$

where i', j' and k' represent the physical degrees of freedom associated to the qudits on the 3 edges. Note that the physical degrees of freedom are doubled, since each of them is appearing at the two vertices connected to a given edge. The diagonal structure of Eq. (6.4), ensures that the values of the two physical indices representing the same qudit are always equal. We impose the convention that for every closed loop with label μ on the virtual level, a factor d_μ is implied. This convention automatically takes care of the

factors d_{μ_i} appearing in Eq. (6.2), which can then be represented as

$$. \tag{6.5}$$
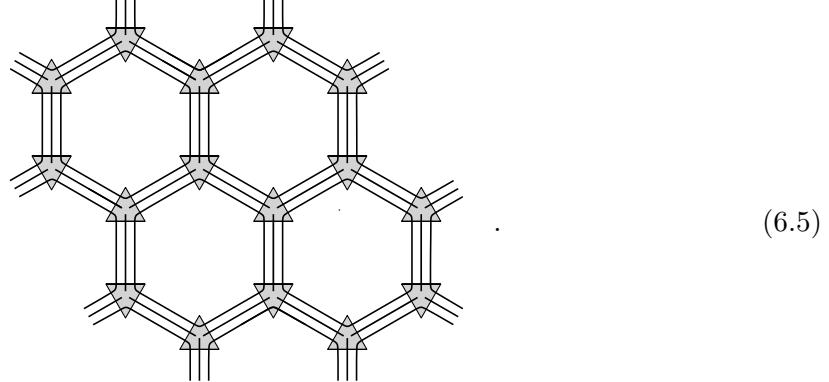
To simplify the notation, we will follow the convention that a pair of legs crossing through a tensor represents a Kronecker delta between the indices on these legs, indicating that the tensor is diagonal in this pair of indices.

The tensor network state constructed above corresponds specifically to the ground state Eq. (6.1). It can, however, be modified to represent any string-net ground state. To do this, first note that we can obtain any ground state by acting with the total projector $B = \prod_p B_p$ on some state $|\phi\rangle \in \mathcal{H}_{\text{s.n.}}$ corresponding to a configurations of strings on the lattice. The tensor network state for $B|\phi\rangle$, can then be obtained by modifying our original construction, to account for additional strings running between the vacuum loops in Eq. (6.2). Locally, such a string-configuration can take two forms: a single string, or two strings fusing to a third one.

The first configuration looks as follows

$$\sum_{\{\mu\}} d_{\mu_1} d_{\mu_2} d_{\mu_3} d_{\mu_4}$$

$$= \sum_{\{\mu\}} \sum_{\mu_1', \mu_2', \mu_4'} d_{\mu_3} \frac{v_{\mu_1} v_{\mu_1'}}{v_s} \frac{v_{\mu_2} v_{\mu_2'}}{v_s} \frac{v_{\mu_4} v_{\mu_4'}}{v_s}$$

$$, \tag{6.6}$$

where we have pulled the string s into the different loops along its path using Eq. (2.33). Using F-moves, we can again pull ribbons from neighboring plaquettes into each edges:

$$= \sum_k \sqrt{v_s v_\mu v_{\mu'}} \sqrt{\frac{v_k}{v_\mu v_\nu}} \sqrt{\frac{v_k}{v_{\mu'} v_{\nu'}}} \sqrt{v_s v_\nu v_{\nu'}} G_{\nu'\nu k}^{\mu\mu's}$$

$$. \tag{6.7}$$

The first two factors on the right hand side can be recognized as the symmetrized contribution to each vertex from Eq. (6.3). The first factor on the right hand side of Eq. (6.7) will therefore contribute to the vertex to the left of the edge, while the second factor will contribute to the right vertex, ensuring that we can use the PEPS tensor Eq. (6.4) on both vertices. The third and fourth factors on the right hand side will contribute to the upper and lower plaquette of the edge in Eq. (6.7), respectively. When combined with the factors in Eq. (6.6), these contributions result in a total factor of the form $\frac{v_\mu v_{\mu'}}{v_s} \cdot v_s v_\mu v_{\mu'} = d_\mu d_{\mu'}$ for each plaquette separately (instead of a single quantum dimension factor like in Eq. (6.2)). If we now place the crossing tensor
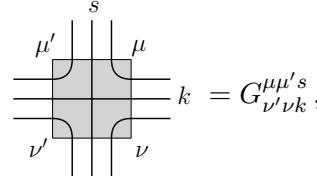
$$\vcenter{\hbox{}} \quad k \;\; = G^{\mu\mu's}_{\nu'\nu k}, \tag{6.8}$$

in the PEPS at every crossing of the ribbon with label s with a lattice edge, this gives the correct superposition of qudit states. Note that the quantum dimension factors in each plaquette are again taken care of by the convention for closed loops at the virtual level. Eq. (6.6) can then be represented with the following tensor network state:
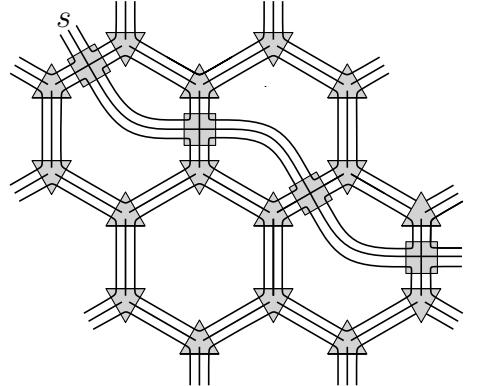
$$\vcenter{\hbox{}} \quad , \tag{6.9}$$

where the additional loops in the plaquette correspond to the second summation on the right hand side of Eq. (6.6). Note that in the tensor network representation above, one should ensure that the string-label is fixed to the correct string-label s. In case one were to use the tensors above to represent a closed string, an additional Kronecker-delta tensor must be included to avoid summing over all string-labels.

The tensor network representation of a ground state obtained from a string-configuration containing fusing strings, is derived in a similar fashion. We again start by pulling the

strings onto the various loops using Eqs. (2.33) and (2.52):

$$\sum_{\{\mu\}} d_{\mu_1} d_{\mu_2} \quad = \sum_{\mu_1',\mu_2',\mu_2''} \frac{v_{\mu_1} v_{\mu_1'}}{v_s} \frac{v_{\mu_2'} v_{\mu_2''}}{v_t v_u}$$

$$= \sum_{\mu_1',\mu_2',\mu_2''} \frac{v_{\mu_1} v_{\mu_1'}}{v_s} v_{\mu_2} v_{\mu_2'} v_{\mu_2''} G^{s\mu_2''\mu_2'}_{\mu_2 ut} \quad .$$

(6.10)

The edge crossings can again be resolved as in Eq. (6.7), giving exactly the same situation as before for the upper plaquette. For the lower plaquette, the combined contribution from of the last factor in Eq. (6.7) for the three edge crossings combines with the right hand side of Eq. (6.10) to a total factor of the form

$$\sqrt{v_s v_{\mu'} v_{\mu''}} \sqrt{v_t v_\mu v_{\mu''}} \sqrt{v_u v_\mu v_{\mu'}} v_\mu v_{\mu'} v_{\mu''} G^{s\mu''\mu'}_{\mu ut} = d_\mu d_{\mu'} d_{\mu''} \sqrt{v_s v_t v_u} G^{s\mu''\mu'}_{\mu ut} \, . \tag{6.11}$$

We can now define the tensor

$$= \sqrt{v_s v_t v_u} G^{s\mu''\mu'}_{\mu ut}$$

$$= \sqrt{v_s v_t v_u} G^{stu}_{\mu\mu''\mu'} \, , \tag{6.12}$$

to represent the fusion of strings on the virtual level. This tensor, along with the closed loop convention at the virtual level, takes care of all remaining factors in Eq. (6.11), and gives the correct superposition of qudit states. A ground state obtained from a initial configuration containing fusing strings can then be represented as:

(6.13)

where one must again be cautious to fix the string-labels to the correct values when "closing off" this tensor network on the virtual level.

The tensors in Eqs. (6.4), (6.8) and (6.12) can be used to construct the tensor network representation of any ground state of the Levin-Wen model. Ground states of the extend Levin-Wen model can be constructed by adding the following tensor on ever vertical edge:
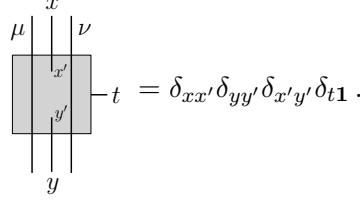
$$\text{(figure)} \quad = \delta_{xx'}\delta_{yy'}\delta_{x'y'}\delta_{t\mathbf{1}} \,. \tag{6.14}$$

This tensor simply includes a trivial tail qudit and replaces the qudit on the vertical edge by two identical ones. Since its action is trivial, we deem it unnecessary to draw it explicitly. In the tensor network diagrams below, its pretense is implied in any plaquette where the tail qudit is not specified explicitly.

6.2 Tensor network representations for anyonic fusion basis states

The construction above can be extended to a tensor network representation of any anyonic fusion basis state. An important ingredient that is missing is the tensor network representation of the following ribbon configuration:

$$\sum_{\mu} d_{\mu} \;\text{(figure)}\; = \sum_{\mu}\sum_{\mu'} \frac{v_{\mu}v_{\mu'}}{v_k} \;\text{(figure)}\; . \tag{6.15}$$
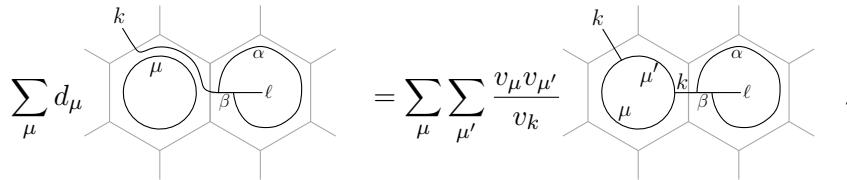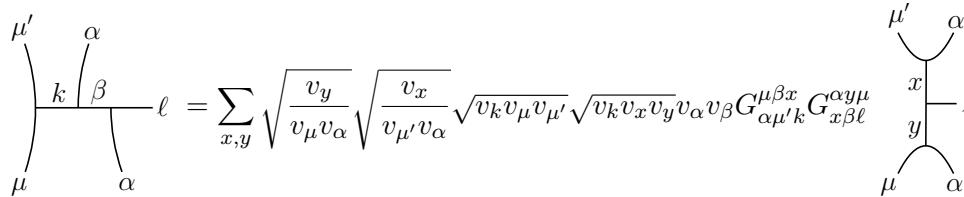
$$\text{(figure)}\; \ell = \sum_{x,y} \sqrt{\frac{v_y}{v_{\mu}v_{\alpha}}} \sqrt{\frac{v_x}{v_{\mu'}v_{\alpha}}} \sqrt{v_k v_{\mu} v_{\mu'}} \sqrt{v_k v_x v_y} v_{\alpha} v_{\beta} G^{\mu\beta x}_{\alpha\mu'k} G^{\alpha y\mu}_{x\beta\ell} \;\text{(figure)}\; \ell \,, \tag{6.16}$$

where the steps in the reduction are completely analogous to those in Eq. (6.7). The first two factors on the right hand side will contribute to the bottom and top vertices, respectively, ensuring that we can use the vertex tensor Eq. (6.5) at both vertices. The third factor can be recognized from Eq. (6.7) as the factor contributing to the left plaquette. We then absorb the remaining factors into the definition of the string-end tensor
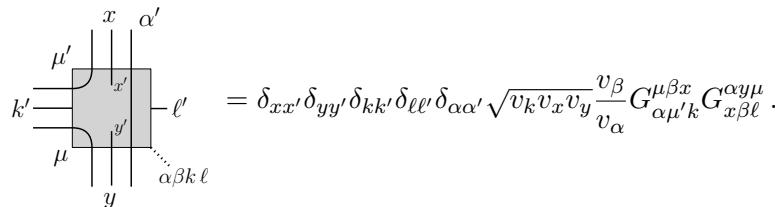
$$\text{(figure)}\; \ell' = \delta_{xx'}\delta_{yy'}\delta_{kk'}\delta_{\ell\ell'}\delta_{\alpha\alpha'} \sqrt{v_k v_x v_y} \frac{v_{\beta}}{v_{\alpha}} G^{\mu\beta x}_{\alpha\mu'k} G^{\alpha y\mu}_{x\beta\ell} \,. \tag{6.17}$$

This definition differs from the remaining factors on the right hand side of Eq. (6.16) by $\frac{1}{d_{\alpha}}$, in order to counter the factor d_{α} that arises from the closed loop convention on the virtual level. This tensor has three physical indices, x', y' and ℓ', representing the physical state of the two qudits on the vertical edge and the connected tail qudit. The dotted leg

with index $\alpha\beta k\ell$ is included to avoid summing over different values of the string labels, as we are representing a state in which the labels α, β k and ℓ are fixed.

We now have all the ingredients we need to construct a PEPS realization of an arbitrary anyonic fusion basis state on the tailed lattice. All that remains is to construct PEPS tensors representing the doubled leaf and pants segments appearing in Eq. (2.55) by combining the string end and fusion tensors with the decompositions of the doubled ribbons to two-dimensional ribbon configurations. The factors $\frac{v_k}{v_a v_b}$ in Eq. (2.55) that result from the reduction of the doubled ribbons in the interior branches of the fusion basis state are split evenly into both sides of the resulting single ribbon. Each doubled leaf segment therefore receives an extra factor $\sqrt{\frac{v_k}{v_a v_b}}$, which, when combined with the factors on the right hand side of Eq. (2.51) yields the definition of the excitation tensor for a doubled anyon with a leaf label $\boldsymbol{a} = a\bar{b}_\ell$:

$$
\text{(diagram)} \ = \ \frac{\sqrt{v_a v_b}}{\mathcal{D}} \sum_{\alpha\beta k} \sqrt{v_k} v_\alpha v_\beta \sum_{\gamma,\delta} d_\gamma d_\delta R_\gamma^{a\alpha} R_\delta^{\alpha b} G_{\alpha ab}^{k\delta\gamma} G_{b\alpha\ell}^{\beta a\delta} G_{a\gamma\delta}^{k\beta\alpha} \quad \text{(diagram)} \quad ,
$$

$$\tag{6.18}$$

or, equivalently,

$$
\text{(diagram)} \ = \ \delta_{xx'}\delta_{yy'}\delta_{\ell\ell'} \frac{\sqrt{v_a v_b}}{\mathcal{D}} \sqrt{v_x v_y} d_k \sum_\beta d_\beta \sum_{\gamma,\delta} d_\gamma d_\delta R_\gamma^{a\alpha} R_\delta^{\alpha b} G_{\alpha ab}^{k\delta\gamma} G_{b\alpha\ell}^{\beta a\delta} G_{a\gamma\delta}^{k\beta\alpha} G_{\alpha\mu' k}^{\mu\beta x} G_{x\beta\ell}^{\alpha y\mu}
$$

$$\tag{6.19}$$

For the tensor representing the root of the fusion tree, a factor $\left(R_{\ell_5}^{b_5 a_5} \right)^*$ must be included, in accordance with Eq. (2.55). We will not include it implicitly, but it is implied whenever a tensor is the root of the fusion tree.

In a similar way, the doubled pants segment on the left hand side of Eq. (2.56) gets an additional factor $\sqrt{\frac{v_i v_j v_k}{v_a v_b v_c v_e v_f}}$ which, when combined with the reduction on the right hand side, leads to a doubled fusion tensor for three doubled Fibonacci anyons $\boldsymbol{a} = a\bar{b}$, $\boldsymbol{b} = c\bar{d}$ and $\boldsymbol{c} = e\bar{f}$ of the form

$$
\text{(diagram)} \ = \ \sqrt{v_i v_j v_k v_a v_b v_c v_d v_e v_f} \sum_{\gamma,\delta} d_\gamma d_\delta R_\gamma^{ad} G_{def}^{kb\delta} G_{dae}^{c\delta\gamma} G_{\gamma dc}^{j\delta a} G_{\delta ba}^{ijk} \quad \text{(diagram)} \quad .
$$

$$\tag{6.20}$$

A PEPS representation of an arbitrary anyonic fusion basis state of the form (3.31)

can then be constructed by applying the following correspondence rules:

$$
\tag{6.21}
$$

For example, consider the anyonic fusion state

$$
\tag{6.22}
$$
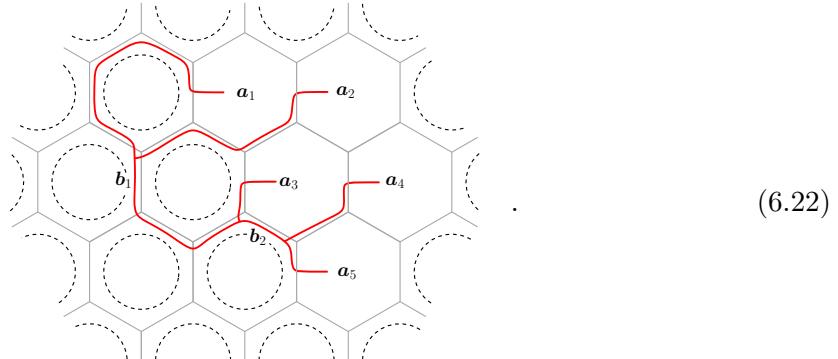
For this state, the rules in Eq. (6.21), result in the PEPS depicted in Fig. 6.1.
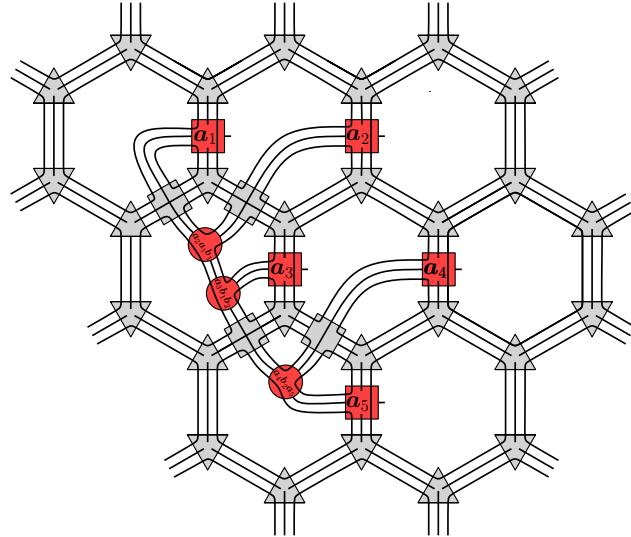
Figure 6.1: PEPS representation of the anyonic fusion basis state in Eq. (6.22).

The PEPS representation for anyonic fusion basis states we have just derived provides a powerful tool for numerical calculations with the relevant states in the physical subspace. It was obtained here through the explicit reduction of the anyonic fusion basis states defined in Sec. 3.5. Even though the graphical calculus on these three-dimensional ribbon configurations shows that they indeed behave as fusion states of doubled Fibonacci anyons

under twists and braiding, it is not intuitively clear how this behavior translates once such a configuration has been reduced to a superposition of qudit states. It is therefore useful to explicitly verify that our PEPS ansatz itself has the desired behavior under relevant operations.

6.3 Square PEPS tensors

In practice, it is often much more convenient to work with a square geometry for the PEPS. This can be achieved by contracting the tensors within each segment. In order to simplify the resulting square tensors, we combine each group of triple indices to a single index. Note that, since the definition of the tensor (6.4) requires that each set of tripled indices satisfies the Fibonacci fusion rules in order to yield a nonzero value, these regrouped virtual indices only need to have dimension 5 (corresponding to the 5 combinations abc satisfying $\delta_{abc} = 1 : \{\mathbf{111}, \mathbf{1}\tau\tau, \tau\mathbf{1}\tau, \tau\tau\mathbf{1}, \tau\tau\tau\}$). A similar trick can be used when grouping the physical indices, since these come in three sets of 3 labels that must satisfy the fusion condition in each vertex (furthermore, these sets share indices that where doubled by the tensor network construction). The resulting square tensors will be colored gray and red for segments of which the tails end inside plaquettes containing trivial and nontrivial DFIB charges, respectively:

$$(6.23)$$

With this simplified notation, the fusion state depicted in Fig. 6.1 can be written as

$$(6.24)$$

where we rotated the lattice counterclockwise by 60 before obtaining the squared lattice. In this simplified notation, we assume that the crossing tensors defined in Eq. (6.8) are inserted at every edge crossing of a virtual string, and assume that the strings are fused using the appropriate doubled fusion tensors Eq. (6.20). Note that, when using the simplified tensors (without the triple indices), one must be careful to correctly enforce the closed loop convention adopted when defining the PEPS tensors on the honeycomb lattice.

7 | Classical simulation of the extended string-net code

Besides formulating an error correction scheme for the extended Fibonacci string-net code, the main achievement of this work consists of obtaining error correction thresholds for this code with a microscopic noise model. This was achieved through Monte Carlo simulations, in which the quantum state of the system is updated to reflect the application of noise, measurement, and recovery operations until either the initial state is recovered successfully or a logical error occurs. Due to the complicated nature of the extended string-net code, and the fact that our simulations require the ability to simulate the dynamics of non-Abelian anyons, this is a very nontrivial task. Below we discuss the various technical details of these simulations. Note that the numerical simulations presented here are independent of the specific circuits used to realize the projective measurements and unitary transformations that are required during the error correction process.

The structure of this chapter is as follows: We start by going over some comments on the classical simulation of non-Abelian QEC, made in Refs. [51, 52, 55] in Sects. 7.1 and 7.2. We then introduce a microscopic noise model of Pauli errors in Sec. 7.3, and study the effect of individual qubit errors on anyonic fusion states in Secs. 7.4 and 7.5. Sec. 7.6 describes the framework of curve diagrams [79], which is used to efficiently characterize anyonic fusion basis bases during the simulation. Finally, in Sec. 7.7 we provide a detailed outline of all steps performed for a single Monte Carlo sample.

7.1 Simulating non-Abelian quantum error correction

The defining difference between Abelian and non-Abelian anyon models is the fact that the fusion outcome of a pair of anyons is no longer uniquely determined for non-Abelian models. This fact makes simulating noise, syndrome measurement and error correction processes for a system exhibiting non-Abelian anyonic excitations considerably more complicated than for Abelian models such as the surface code.

After anyonic excitations have been created through the application of noise, their locations and anyon labels can be determined through a syndrome measurement. If we think of the state in terms of fusion trees of anyons, the syndrome measurement only projects onto fixed values for the leaf labels. For non-Abelian anyons, the internal (branch) labels of the fusion tree may still be in a superposition of different configurations. One of the implications is that braiding processes occurring during the recovery phase do not necessarily result in a global phase. Hence, different paths used to physically approach a pair of anyons in order to fuse them might give rise to different fusion outcomes (more precisely, to different probability distributions for the measurement outcome of the total charge of the pair). When simulating the error-correction process, we must therefore be very careful in specifying and keeping track of the precise paths followed by the various anyons.

Another implication is that, for non-Abelian models, the decoding process must happen in an iterative fashion. Based on the initial positions and charges of the anyons, a recovery step that consists of a number of fusion processes is suggested. As the result of this recovery cannot be predicted, all fusion processes must be performed in the given order, after which the fusion outcomes must be measured. These measured outcomes then give a new error syndrome that serves as the basis for suggesting a next recovery step, consisting of a new series of fusion processes. This cycle continues until all anyons are fused away and decoding is successful, or until some anyon is wound along a nontrivial cycle during a recovery step resulting in a decoding failure. Error correction thus proceeds as a dialogue between syndrome extraction and decoder, where the new syndrome resulting from a given recovery step is used to determine the next recovery step. This is in stark contrast to Abelian models, where a single syndrome measurement provides all the necessary information to determine all fusion processes that must be caried out in order to return the system to the code space.

7.2 Classical simulability

The ability to reliably simulate the general dynamics of Fibonacci anyons implies the ability to simulate (and hence perform) universal quantum computation. It is therefore highly unlikely that such simulations are feasible on a classical computer. However, typical noise and recovery processes such as those that we will simulate in the remainder of this work exhibit structure that can be exploited to classically simulate them in regimes where we expect successful error correction to be possible.

Individual local error operations either create a distinct connected group of anyons with vacuum total charge, or extend such an existing group (see Sec.7.4). These groups can be understood as anyons that have interacted at some point during their lifetime. Since each disconnected group has a trivial total charge, braiding between separate groups is trivial. Hence, the total fusion space factorizes into a tensor product of fusion spaces of individual connected groups, and we are only required to simulate anyon dynamics within each of these groups separately.

With regards to the creation of connected groups of anyons, the noise process behaves as a kind of percolation process. Hence, below the percolation threshold, one expects that the size of the largest connected group scales as $O(\log(L))$ (with variance $O(1)$), where L is the linear system size [80]. As this is a probabilistic statement, there will be instances where the largest connected group has a size larger than $O(\log(L)$, but the probability of such events is suppressed exponentially with the system size L. This logarithmic scaling of the largest cluster size $s = O(\log(L))$ counters the exponential scaling of the dimension of the fusion space $d = O(\exp(s))$ for individual connected groups. Therefore, the fusion spaces of individual connect groups will have dimension $\dim = O(\text{poly}(L))$, meaning that the dynamics within connected groups can be simulated efficiently.

These arguments on the classical simulability of topological quantum error correction with a universal anyon model where first made in Ref. [55]. The behavior of connected clusters of anyons created in the phenomenological model studied there corresponds exactly to the bond percolation model for which the logarithmic scaling mentioned above was verified numerically [80]. To ensure that this still holds for the microscopic model studied here, we explicitly verified the logarithmic scaling of the average size of the largest connected group of anyons after subjecting all qubits to a depolarizing noise model, using Monte Carlo simulations. The results of these simulations are presented and discussed in App. B.

During the iterative decoding procedure, anyons are fused over increasing length scales, thereby potentially joining together connected groups of anyons through this interaction. As long as large connected groups are sparsely distributed (which is the case on average below the percolation threshold) this should not pose a problem, as the dimension of the fusion space is automatically reduced once the fusion process is actually performed, since this then reduces the number of anyons that must be simulated. We can conclude that we expect efficient simulation of the dynamics relevant to error correction to be possible in the regime where the combined action of noise and recovery does not percolate.

If noise is strong enough, connected groups of anyons will percolate and the fusion space will no longer factorize into small disconnected parts. In this case classical simulation of braiding and fusion will become intractable. However, this is precisely the regime where we expect regular transport of anyons along nontrivial cycles during recovery, leading to failed error correction. Therefore, the error correction threshold itself will lie below this regime and estimating its value through classical simulations should be possible.

7.3 Noise model

Our goal is to stimulate the dynamics of a quantum-computing architecture of qubits, which directly implements our error-correcting code. We model noise in this system as individual (either depolarizing or dephasing) Pauli errors acting on random qubits, while the system is constantly being monitored[1]. We treat the occurrence of Pauli errors on individual qubits as independent Poisson processes with a fixed rate p, which characterizes the noise strength. The total number of qubit errors, denoted by T, then follows a Poisson distribution with mean $T_0 = |E|p$, where $|E|$ is the total number of edges. (For a tailed honeycomb lattice with periodic boundary conditions and a total of L^2 plaquettes, the total number of edges (and hence qubits) is $|E| = 5L^2$, resulting in $T_0 = 5L^2p$.) This fixed-rate sampling noise model is similar to those used for the simulation of non-Abelian quantum

---

[1]Note that under these assumptions, it is not appropriate to characterize the noise in terms of an i.i.d. noise strength per qubit, since the non-Abelian nature of our code implies that the combined action of individual error and measurement operators do not commute for consecutive errors. We discuss the relation with i.i.d. noise in App. C

error correction with phenomenological models such as in Refs. [51, 55]. For simplicity, we assume that all measurements are perfect and are carried out on timescales which are negligible compared to the average time between individual qubit errors [$\sim 1/(|E|p)$].

We simulate the dynamics of the system for T time steps, each of which corresponds to the occurrence of a single Pauli qubit error, and where T is drawn from a Poisson distribution with mean T_0. Each individual time step consists of the following:

1. An edge e of the lattice is chosen at random and a Pauli operator σ_i is picked according to relative probabilities $\{\gamma_x, \gamma_y, \gamma_z\}$. We specifically use depolarizing noise ($\gamma_x = \gamma_y = \gamma_z = 1/3$), dephasing noise ($\gamma_z = 1$), and bit-flip noise ($\gamma_x = 1$). The operator σ_i is then applied on the qubit corresponding to edge e.

2. All vertex stabilizers Q_v, and tail qubits are measured (in the Z-basis).

3. The appropriate unitary operator U_V, conditioned on the measurement outcome V from the measurements above, is applied to fix any violated vertices.

4. The anyon charge in each plaquette is measured.

The unitary operators used to locally correct violated vertices were introduced in Chapter 4, their purpose is to move the system back to the string-net subspace $\mathcal{H}_{\text{s.n.}}$. Inside this subspace states can be described in terms of anyonic fusion states, and plaquette anyon charges are well defined. Note that the measurement at the end of each time step means that we will never encounter any superpositions of different anyonic charges in individual plaquettes. (In terms of fusion states, this fixes all leaf labels.)

After the T error operations have been applied, and if no logical error was induced by the noise process, the syndrome is fed to an iterative (classical) decoding algorithm, which returns a list of anyons to be fused and paths to be followed when doing so. We assume that all recovery operations and additional syndrome measurements are perfect and instantaneous, meaning no additional errors happen during the recovery process.

While the connection between percolation and logical errors in topological codes is not exact [81] (i.e., not all percolation events cause logical errors), all logical errors are the result of events where a pair of anyons are fused along a non-contractible loop. Our Monte Carlo simulations (detailed below in Sec. 7.7) classify all such percolation events as failures, and therefore slightly overestimate the logical failure rates. This provides a heuristic argument for the validity of our noise model. The immediate collapse of superpositions in the anyon charge of individual plaquettes does not inhibit the occurrence of percolation processes. We therefore do not expect our assumption of constant syndrome measurement to significantly affect the obtained decoder failure rates. Furthermore, in the low noise strength limit, our noise model approximates an i.i.d. noise model, as the random qubits affected by Pauli errors are unlikely to be in each others vicinity. The existence of an error correction threshold for our fixed-rate sampling noise model therefore implies the existence of a threshold for an i.i.d. noise model as well.

7.4 Pauli-noise in the anyonic fusion basis

We will now investigate the effect of the application of noise and subsequent measurement and vertex recovery operations performed in each of the T time steps of the simulation as described above. Because of the syndrome measurement performed at the end of every time step, the quantum state of the system between these steps can always be decomposed

as a superposition of anyonic fusion basis states, which all share the same set of leaf labels. It is therefore sufficient to understand the action of these different operations on an initial state

$$|\Psi_0\rangle = \sum_t \alpha_t |\psi_t\rangle \ , \tag{7.1}$$

where the $|\psi_t\rangle$ represent anyonic fusion basis states (6.22), and the states $\{|\psi_t\rangle\,|\alpha_t \neq 0\}$ all share the same set of leaf labels.

7.4.1 Pauli-Z errors

We begin our analysis by studying the case where we act with a σ_z operator on the qubit residing at edge e. One can easily see that for any edge e, σ_z^e commutes with all vertex operators Q_v defined in (3.9), since both operators are diagonal in the same basis. Hence, a σ_z error does not take the system out of the string-net subspace $\mathcal{H}_{\text{s.n.}}$, and the resulting state can again be decomposed in the anyonic fusion basis:

$$|\Psi_1\rangle = \sigma_z^e |\Psi_0\rangle = \sum_t \alpha_t \sigma_z^e |\psi_t\rangle \tag{7.2}$$

$$= \sum_{t'} |\psi_{t'}\rangle \left(\sum_t \alpha_t \langle \psi_{t'}|\sigma_z^e|\psi_t\rangle \right), \tag{7.3}$$

where we have used that $\sum_{t'} |\psi_{t'}\rangle \langle \psi_{t'}|$ acts as the resolution of the identity within $\mathcal{H}_{\text{s.n.}}$. In particular, if we start from a superposition of basis states $|\psi_t\rangle$ that all have the same specific handle label (see Sec. 3.5), the states $|\psi_{t'}\rangle$ in the resulting superposition will also have that same label, unless a logical error has occurred through the interaction of the thermal anyons due to the noise operator. Since the simulation of error correction is immediately aborted in the event of such a logical error, it is sufficient to only consider basis states that all have the same handle label as the initial state.

When the edge e is a tail edge, acting with σ_z^e results in a global (± 1) factor, which has no physical consequences. On a general edge e different from a tail edge, the action of σ_z^e commutes with the irreducible idempotents of the tube algebra (Eq. (3.23) (3.24), (3.25), (3.27) and (3.28)) acting on any plaquette, except for the two that contain the edge e. This means that a σ_z^e error on a general edge can modify the anyon charge of at most two plaquettes. Furthermore, the total charge of these two plaquettes cannot be changed by the action of σ_z^e, since this is a collective property of the pair that is insensitive to the local operator σ_e^z.

With these insights in mind, we pick an anyonic fusion basis of the following form:

$$|\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}}\rangle = \quad \underset{\boldsymbol{c}_j \qquad \boldsymbol{c}_{j+1}}{\overset{\boldsymbol{a}_1 \qquad \boldsymbol{a}_2}{\boldsymbol{b}}} \quad , \tag{7.4}$$

where $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$ are the charges of the two affected plaquettes, $\boldsymbol{b}$ denotes their total charge, and $\vec{\boldsymbol{c}}$ collectively denotes all other leaf, branch and handle labels. We can then rewrite the initial state (7.1) as

$$|\Psi_0\rangle = \sum_{\boldsymbol{b},\vec{\boldsymbol{c}}} \alpha_{\boldsymbol{b},\vec{\boldsymbol{c}}} |\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}}\rangle \ . \tag{7.5}$$

Note that we dropped the summation over $\vec{\boldsymbol{a}}$, which is allowed because the initial state contains no superposition in plaquette charges. Since the labels $\boldsymbol{b}$ and $\vec{\boldsymbol{c}}$ are unaffected by

σ_z^e, the matrix elements appearing on the right hand side of (7.2) are block diagonal in these labels, and the expression for the state $|\Psi_1\rangle = \sigma_z |\Psi_0\rangle$ can be reduced to

$$|\Psi_1\rangle = \sum_{\boldsymbol{b},\vec{\boldsymbol{c}}} \sum_{\vec{\boldsymbol{a}}'} |\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}'}\rangle \left(\alpha_{\boldsymbol{b},\vec{\boldsymbol{c}}} \langle \psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}'} | \sigma_z^e | \psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}} \rangle \right). \tag{7.6}$$

The probability of finding outcome $\vec{\boldsymbol{a}}' = (\boldsymbol{a}_1', \boldsymbol{a}_2')$ when performing a syndrome measurement in the two affected plaquettes is then given by

$$p(\vec{\boldsymbol{a}}') = \sum_{\boldsymbol{b},\vec{\boldsymbol{c}}} \left| \alpha_{\boldsymbol{b},\vec{\boldsymbol{c}}} \langle \psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}'} | \sigma_z^e | \psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}} \rangle \right|^2. \tag{7.7}$$

After performing this measurement, the state of the system collapses to

$$|\Psi_2\rangle = \frac{1}{\sqrt{p(\vec{\boldsymbol{a}}')}} \sum_{\boldsymbol{b},\vec{\boldsymbol{c}}} |\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}'}\rangle \left(\alpha_{\boldsymbol{b},\vec{\boldsymbol{c}}} \langle \psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}'} | \sigma_z^e | \psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}} \rangle \right), \tag{7.8}$$

which is again a superposition of basis states with fixed leaf labels.

Note that the matrix elements on the right hand side of (7.7) and (7.8) are independent of the unaffected labels $\vec{\boldsymbol{c}}$. Hence, we may replace then with matrix elements of the form $\langle \psi_{\boldsymbol{b}}^{\vec{\boldsymbol{a}}'} | \sigma_z^e | \psi_{\boldsymbol{b}}^{\vec{\boldsymbol{a}}} \rangle$, where $|\psi_{\boldsymbol{b}}^{\vec{\boldsymbol{a}}}\rangle$ are states that only contain nontrivial anyon labels for the two affected plaquettes and for their total charge. Also note that since the total charge of the affected plaquettes is a collective property, the precise location of the third plaquette which contains this total charge does not affect the value of the matrix elements, and furthermore, the tail label of that plaquette does not affect the matrix elements.

When calculating the matrix elements, we must pick some convention for the basis elements $|\psi_{\boldsymbol{b}}^{\vec{\boldsymbol{a}}}\rangle$ (i.e., for how the fusion tree (7.4) is embedded on the lattice), for each of the possible orientations of e. Since σ_z^e has no physical effect when e is a tail edge, we only need to consider four orientations for e. Our choice of basis convention for the σ_e^z matrix elements is given in Fig. 7.1.

(a) Orientation 1   (b) Orientation 2   (c) Orientation 3   (d) Orientation 4
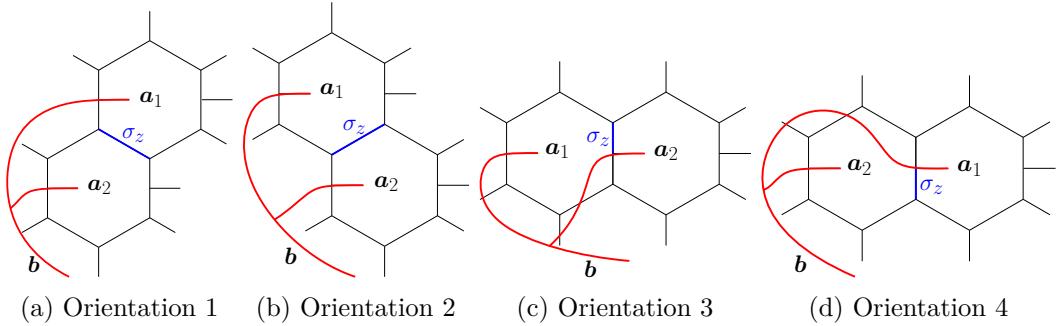
Figure 7.1: Basis convention for the affected plaquette charges for all nontrivial orientations of the edge e (highlighted in blue) in the case of a σ_z error.

7.4.2 Pauli-X and Y errors

The case of a σ_x^e error is similar, but comes with some additional complications. Since a σ_x^e operator does not commute with the vertex operators Q_v associated to the vertices bounding e, the state

$$|\Psi_1\rangle = \sum_t \alpha_t \sigma_x^e |\psi_t\rangle. \tag{7.9}$$

does not necessarily belong to the sting-net subspace $\mathcal{H}_{\text{s.n.}}$ and hence cannot be expressed as a superposition of anyonic fusion basis states. In particular, upon measuring the vertex stabilizers for the vertices v_1 and v_2 that bound e, one might find that the ribbon graph branching rules are violated in either of these vertices. Each violated vertex will belong to a set of 7 connected qubits on the lattice that we will call a *segment*:
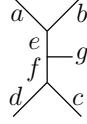
The three vertices in each segment will be denoted as t, m and b, corresponding to the top, middle and bottom vertex, respectively. For each segment with some violated vertices, we also measure the label of the tail qubit (corresponding to edge g in the diagram above). Depending on these measurement outcomes V, a unitary operator U_V is applied to the segment in order to take it back to the $+1$ eigenstate of the three vertex operators Q_t, Q_m and Q_b associated to the segment. The definition and corresponding circuits for the unitaries conditioned on all possible measurement outcomes is given in Fig. 4.2 in Chapter 4. Intuitively, one can think of the action of these operators as pulling string ends into the tail edge of a neighboring plaquette as depicted in Fig. 4.1b.

The probability $p(V)$ of a combined outcome V for the vertex and tail qubit measurements in the relevant segments is given by

$$
\begin{aligned}
p(V) &= \langle \Psi_1 | P_V | \Psi_1 \rangle \\
&= \sum_{t',t} \bar{\alpha}_{t'} \alpha_t \langle \psi_{t'} | \sigma_x^e P_V \sigma_x^e | \psi_t \rangle ,
\end{aligned}
\tag{7.10}
$$

where P_V is the projector onto the total measurement outcome V. For example, for the case of a σ_x^e acting on the edge e bounded by vertices v_1 and v_2 where only v_1 is violated and the tail qubit q_1 of the segment containing v_1 has label τ, the associated projector P_V is given by

$$
P_V = (1 - Q_{v_1})(Q_{v_2})(|\tau\rangle_{q_1} \langle\tau|_{q_1}).
\tag{7.11}
$$

After performing the vertex and tail qubit measurements with outcome V and applying the appropriate unitary operator U_V to bring the system back to the string-net subspace, the state is given by

$$
\begin{aligned}
|\Psi_2\rangle &= \frac{1}{\sqrt{p(V)}} \sum_t \alpha_t U_V P_V \sigma_x^e |\psi_t\rangle \\
&= \frac{1}{\sqrt{p(V)}} \sum_{t'} |\psi_{t'}\rangle \left(\sum_t \alpha_t \langle \psi_{t'} | U_V P_V \sigma_x^e | \psi_t \rangle \right) ,
\end{aligned}
\tag{7.12}
$$

where we have again inserted the resolution of the identity $\sum_{t'} |\psi_{t'}\rangle \langle \psi_{t'}|$ in $\mathcal{H}_{\text{s.n.}}$ on the right hand side in order to explicitly express the state as a superposition of anyonic fusion basis states.

As in the case of a σ_e^z error, the expressions (7.10) and (7.12) can be simplified by considering which plaquette charges are actually affected by the combined action of the operators U_V, P_V and σ_x^e. Since a σ_x operator acting on an edge e of the tailed lattice results in at most two violated vertices, the combined action of U_V, P_V and σ_x^e involves at most two segments. These operators therefore commute with any tube operators acting on plaquettes that have no edges in common with these segments. This means that only the charges associated to the plaquettes in the immediate neighborhood of the error can be affected. The number of affected plaquettes depends on the orientation of the edge e.

As before, we pick our anyonic fusion basis to reflect this fact. In case of 4 affected plaquettes, the basis states have the form
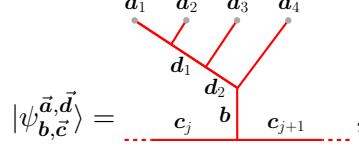
$$
|\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}},\vec{\boldsymbol{d}}}\rangle = \qquad , \tag{7.13}
$$

where, again, $\vec{\boldsymbol{a}}$ denotes the charges of the affected plaquettes, $\boldsymbol{b}$ denotes their total charge, and $\vec{\boldsymbol{c}}$ collectively denotes all other unaffected leaf, branch and handle labels. Note that we now need additional labels $\vec{\boldsymbol{d}}$ to denote the affected internal branch labels.

With this choice of basis, the sum over t in the initial state (7.1) is split into a sum over the unaffected labels $\vec{\boldsymbol{c}}$, the total charge $\boldsymbol{b}$ and the branch labels $\vec{\boldsymbol{d}}$ of the affected part:

$$
|\Psi_0\rangle = \sum_{\boldsymbol{b},\vec{\boldsymbol{c}},\vec{\boldsymbol{d}}} \alpha_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{d}}} |\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}},\vec{\boldsymbol{d}}}\rangle \tag{7.14}
$$

As all matrix elements are block diagonal in the unaffected labels, the expression for the vertex measurement outcome probability (7.10) is reduced to

$$
p(V) = \sum_{\boldsymbol{b},\vec{\boldsymbol{c}},\vec{\boldsymbol{d}}} \sum_{\vec{\boldsymbol{d}}'} \bar{\alpha}_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{d}}'} \alpha_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{d}}} \langle \psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}},\vec{\boldsymbol{d}}'} | \sigma_x^e P_V \sigma_x^e |\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}},\vec{\boldsymbol{d}}}\rangle . \tag{7.15}
$$

The sum over t' in the resulting state (7.12) can again be split into a sum over the unaffected labels $\vec{\boldsymbol{c}}'$, the affected leaf labels $\vec{\boldsymbol{a}}'$ and the branch labels $\vec{\boldsymbol{d}}'$ of the affected part, reducing the expression to

$$
|\Psi_2\rangle = \frac{1}{\sqrt{p(V)}} \sum_{\vec{\boldsymbol{a}}',\vec{\boldsymbol{d}}'} \sum_{\boldsymbol{b},\vec{\boldsymbol{c}}} |\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}',\vec{\boldsymbol{d}}'}\rangle \left(\sum_{\vec{\boldsymbol{d}}} \alpha_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{d}}} \langle \psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}',\vec{\boldsymbol{d}}'} | U_V P_V \sigma_x^e |\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}},\vec{\boldsymbol{d}}}\rangle \right) , \tag{7.16}
$$

since all matrix elements are again block diagonal in the unaffected labels. Measurement of the affected plaquette charges will then yield charges $\vec{\boldsymbol{a}}'$ with a probability $p(\vec{\boldsymbol{a}}')$ given by

$$
p(\vec{\boldsymbol{a}}') = \frac{1}{p(V)} \sum_{\vec{\boldsymbol{d}}'} \sum_{\boldsymbol{b},\vec{\boldsymbol{c}}} \left| \sum_{\vec{\boldsymbol{d}}} \alpha_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{d}}} \langle \psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}',\vec{\boldsymbol{d}}'} | U_V P_V \sigma_x^e |\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}},\vec{\boldsymbol{d}}}\rangle \right|^2 . \tag{7.17}
$$

The resulting state after this charge measurement is

$$
|\Psi_3\rangle = \frac{1}{\sqrt{p(\vec{\boldsymbol{a}}')p(V)}} \sum_{\vec{\boldsymbol{d}}'} \sum_{\boldsymbol{b},\vec{\boldsymbol{c}}} |\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}',\vec{\boldsymbol{d}}'}\rangle \left(\sum_{\vec{\boldsymbol{d}}} \alpha_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{d}}} \langle \psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}}',\vec{\boldsymbol{d}}'} | U_V P_V \sigma_x^e |\psi_{\boldsymbol{b},\vec{\boldsymbol{c}}}^{\vec{\boldsymbol{a}},\vec{\boldsymbol{d}}}\rangle \right) . \tag{7.18}
$$

Once again, the matrix elements on the right hand side of (7.15), (7.17) and (7.18) are independent of the unaffected labels $\vec{\boldsymbol{c}}$, so we only require matrix elements of the form $\langle \psi_{\boldsymbol{b}}^{\vec{\boldsymbol{a}}',\vec{\boldsymbol{d}}'} | O |\psi_{\boldsymbol{b}}^{\vec{\boldsymbol{a}},\vec{\boldsymbol{d}}}\rangle$ for fusion basis states involving up to four plaquettes and their total charge.

Our choice of basis convention for the charges of the affected plaquettes for all possible orientations of e is given in Fig. 7.2. In the case of a σ_x error all five possible orientations of e are nontrivial. By considering the potentially violated vertices and the definition of the associated unitaries given in Sec. 4.1 for each orientation, it can easily be verified that the depicted plaquettes are indeed the only affected ones and the combined action of the error, measurements and unitaries commutes with all other tube operators.

The case of a σ_y error is entirely analogous to that of a σ_x. The same operators P_V and U_V appear, and we use the same basis convention as the one depicted in Fig. 7.2.

In summary, all that is required to capture the effect of Pauli-noise on the state of the system are the following matrix elements:

$$\langle\psi_{\vec{b}}^{\vec{a},\vec{d'}}|\sigma_x^e P_V \sigma_x^e|\psi_{\vec{b}}^{\vec{a},\vec{d}}\rangle,\tag{7.19}$$

$$\langle\psi_{\vec{b}}^{\vec{a'},\vec{d'}}|U_V P_V \sigma_x^e|\psi_{\vec{b}}^{\vec{a},\vec{d}}\rangle,\tag{7.20}$$

$$\langle\psi_{\vec{b}}^{\vec{a},\vec{d'}}|\sigma_y^e P_V \sigma_y^e|\psi_{\vec{b}}^{\vec{a},\vec{d}}\rangle,\tag{7.21}$$

$$\langle\psi_{\vec{b}}^{\vec{a'},\vec{d'}}|U_V P_V \sigma_y^e|\psi_{\vec{b}}^{\vec{a},\vec{d}}\rangle,\tag{7.22}$$

$$\langle\psi_{\vec{b}}^{\vec{a'}}|\sigma_z^e|\psi_{\vec{b}}^{\vec{a}}\rangle.\tag{7.23}$$

These matrix elements must be calculated for all possible orientations of the edge e in Figs. 7.1 and 7.2, and for all possible combined outcomes V of the vertex and tail qubit measurements in the case of a σ_x or σ_y error.

(a) Orientation 1 (b) Orientation 2 (c) Orientation 3

(d) Orientation 4 (e) Orientation 5
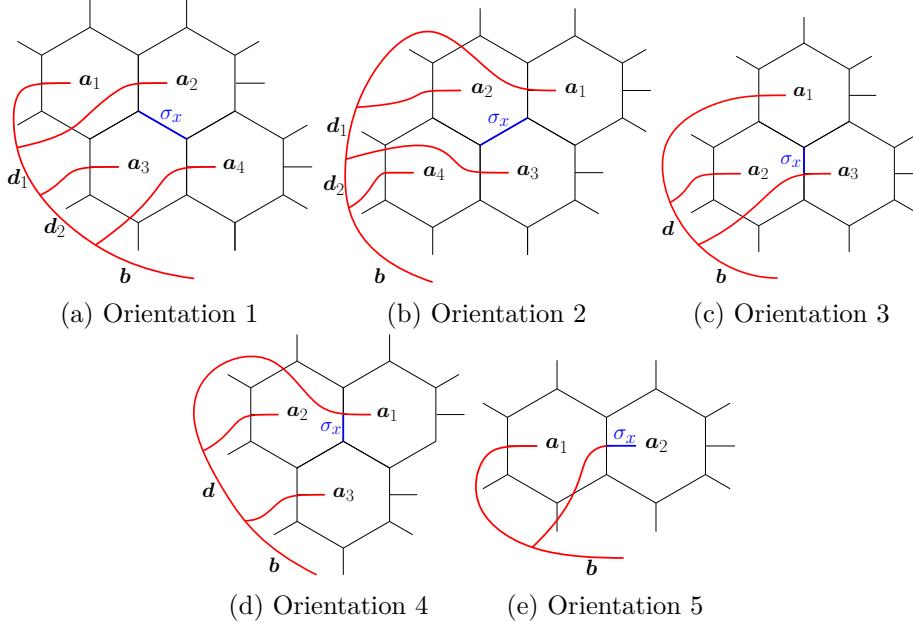
Figure 7.2: Basis convention for the affected plaquette charges for the different orientations of e (highlighted in blue) in the case of a σ_x or a σ_y error.

7.5 Computing the relevant matrix elements

At first sight, calculating the matrix elements listed in Eqs. (7.19)-(7.23) seems like an intractable job, due to the highly entangled nature of the anyonic fusion basis states on the tailed lattice. Fortunately, as detailed in Chapter 6, this complex entanglement structure can be captured using tensor network representations for anyonic fusion basis states. By using some key insights, together with tensor network techniques, we can harness the computational power of tensor networks to compute said matrix elements. Below, we will illustrate this procedure for the matrix element (7.19), where e is an edge with orientation

1, as depicted in Fig. 7.2a. All other matrix elements can be computed in an analogous manner.

For simplicity, we will work with square PEPS tensors (see Sec. 6.3), which each correspond to one segment of the lattice as shown in the following diagram:
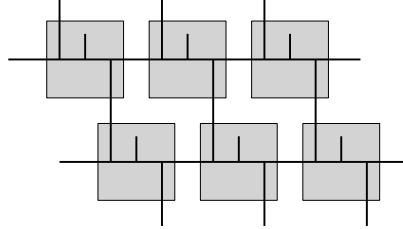
where we have rotated the lattice by 60 counterclockwise. The PEPS tensors will be colored gray and red for segments of which the tails end inside plaquettes containing trivial and nontrivial DFIB charges, respectively. With this notation (and after a counterclockwise rotation by 60), the anyonic fusion state represented in Fig. 7.2a looks as follows:

$$|\psi_{\boldsymbol{b}}^{\vec{\boldsymbol{a}},\vec{\boldsymbol{d}}}\rangle = \cdots \qquad \cdots . \tag{7.24}$$
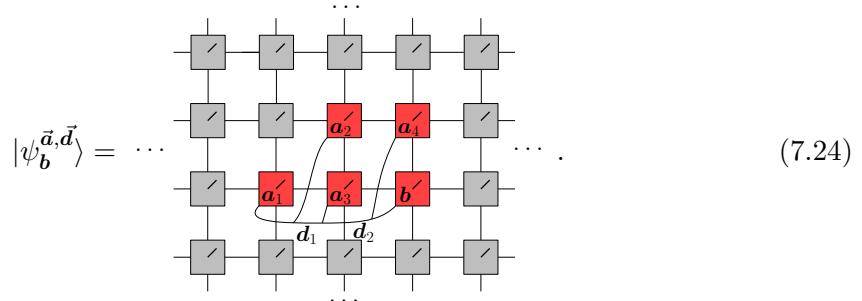
The crossings and branchings in this diagram imply the presence of certain crossing and fusion tensors. Details of these are given in Chapter 6. The total charge $\boldsymbol{b}$ of the four leaf charges in Fig. 7.2a was assigned to a neighboring segment. As was mentioned before, since the total charge of the group of anyons is a collective property, the precise location of the excitation tensor encoding this total charge does not affect the computation of the matrix elements itself. Hence, we choose to place it next to the other charges for convenience.

The matrix elements can then be computed by applying the appropriate operators on the physical indices, and then contracting the result with the conjugate PEPS corresponding to the bra vector $\langle\psi_{\vec{\boldsymbol{b}}}^{\vec{\boldsymbol{a}}',\vec{\boldsymbol{d}}'}|$. In this specific case, the operator P_V that projects out the combined vertex and tail qubit measurement outcome V can be decomposed into two operators P_A and P_B that act on the segments of leaf charges $\boldsymbol{a}_2$ and $\boldsymbol{a}_4$, and project out the measurement outcomes for the vertices and tail edge in these segments, respectively. The matrix element is then given by the contraction

$$\langle\psi_{\boldsymbol{b}}^{\vec{\boldsymbol{a}}',\vec{\boldsymbol{d}}'}|\sigma_x^e P_V \sigma_x^e|\psi_{\boldsymbol{b}}^{\vec{\boldsymbol{a}},\vec{\boldsymbol{d}}}\rangle = \cdots \qquad \cdots , \tag{7.25}$$
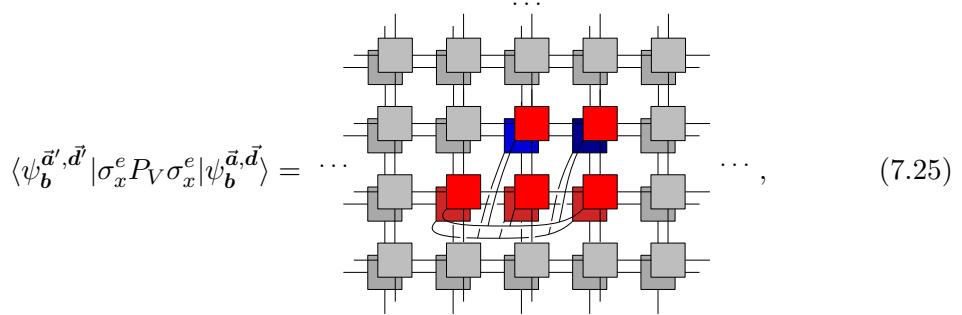
where

 $\quad$ and $\quad$ $\quad . \tag{7.26}$

In order to avoid too much clutter, we have omitted the different labels in the graphical notation. The labels $\vec{a}', \boldsymbol{b}, \vec{d'}$ and $\vec{a}, \boldsymbol{b}, \vec{d}$ are always implied for the top and bottom layer, respectively. Note that the σ_x^e operator was applied on two different tensors. This is nothing more than a side-effect of the fact that the physical degrees of freedom are doubled in the PEPS representation.

The result of the tensor contraction in (7.25) should be independent of the size of the system, and must be entirely determined by the nontrivial anyonic charges of the colored segment tensors. We may therefore assume the system to be infinite, where all tensors except the colored ones depicted in (7.25) correspond to segments with a trivial charge. This infinite contraction can then be simplified by determining the top fixed point MPS of the double layer transfer matrix [82]

$$\tag{7.27}$$

After finding a similar bottom fixed point MPS, Eq. (7.25) can be reduced to

$$\langle \psi_{\boldsymbol{b}}^{\vec{a}',\vec{d'}} | \sigma_x^e P_V \sigma_x^e | \psi_{\boldsymbol{b}}^{\vec{a},\vec{d}} \rangle = \cdots \qquad \cdots . \tag{7.28}$$

In the same way a left fixed point for this expression can be determined,

$$= \quad , \tag{7.29}$$

and similarly for the right fixed point. This finally results in the finite contraction

$$\langle \psi_{\boldsymbol{b}}^{\vec{a}',\vec{d'}} | \sigma_x^e P_V \sigma_x^e | \psi_{\boldsymbol{b}}^{\vec{a},\vec{d}} \rangle = \quad , \tag{7.30}$$

which can be computed in the regular way. In using (7.27) and (7.29) we have implicitly assumed that the PEPS tensors were rescaled such that the leading eigenvalue of the relevant transfer matrices in these expressions is equal to 1. Because of these rescalings the PEPS for the fusion basis states will not have a unit norm, and the final expression (7.30) should therefore be divided by $\sqrt{\langle \psi_{\vec{b}}^{\vec{a},\vec{d}} | \psi_{\vec{b}}^{\vec{a},\vec{d}} \rangle \langle \psi_{\vec{b}}^{\vec{a}',\vec{d'}} | \psi_{\boldsymbol{b}}^{\vec{a}',\vec{d'}} \rangle}$ in order to obtain a matrix element that is properly normalized.

7.6 Storing and manipulating anyonic fusion basis states

As explained in Sec. 3.5, states in $\mathcal{H}_{\text{s.n.}}$ can be expressed as linear combinations of any-onic fusion basis states. The anyonic fusion basis itself is determined by picking a pants decomposition of the surface and choosing a basis for each handle if the genus is nonzero. To keep track of the quantum state $|\Psi\rangle$ of the system, one could in principle pick a basis $\{|\psi_i\rangle\}$ for the entire string-net subspace and then update all coefficients $\langle\psi_i|\Psi\rangle$ throughout the different steps in the simulation. Such a naive approach is doomed to fail however as implementing F-moves and braiding in the exponentially large Hilbert space $\mathcal{H}_{\text{s.n.}}$ quickly becomes intractable as the system-size grows. Instead, we need to select a basis that reflects the factorization of the fusion space discussed in Sec. 7.2. Hence, we require the ability to dynamically introduce a basis for each of the connected groups of anyons separately, in a way that takes into account the specific structure of the relevant noise processes.

For each connected group of anyons, we can pick out a linear ordering by drawing a directed curve connecting all the anyons (which we locate at the center of their corresponding plaquette) within the group. Directed curves, corresponding to the same linear ordering and only differing by continuous deformations that keep the anyon positions fixed, form a set of equivalence classes that we shall call *curves* or *curve diagrams*. Each configuration of (non-intersecting) curves on the surface corresponds to an equivalence class of anyonic fusion bases that are related up to local Dehn twists in individual plaquettes.

A rigorous definition of curve diagrams, using the language of modular functors, can be found in Ref. [79]. For our purposes however, the following simplified construction will suffice. For a given curve containing n anyons, the corresponding fusion basis (up to local Dehn twists) is found as follows: start by drawing a fusion tree connecting only the first and last anyons, in a way that is homologically equivalent to the curve. Next, following the orientation provided by the curve, add anyons $2, 3, \ldots, n-1$ to this fusion tree one by one, by adding a leaf to the tree on the left hand side. This is best illustrated with an example:

$$\tag{7.31}$$
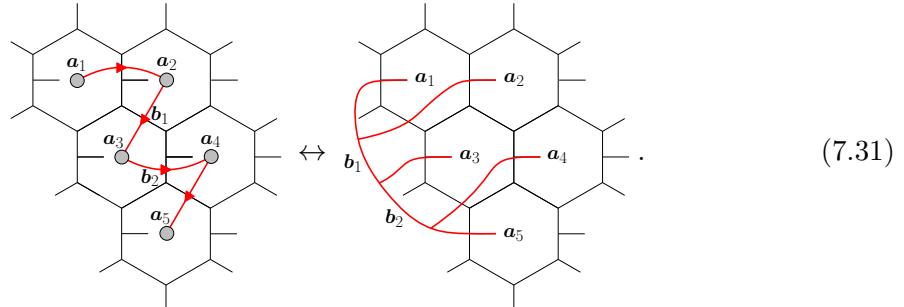
A different basis for the same fusion space corresponding to a different curve diagram would be given by

$$\tag{7.32}$$
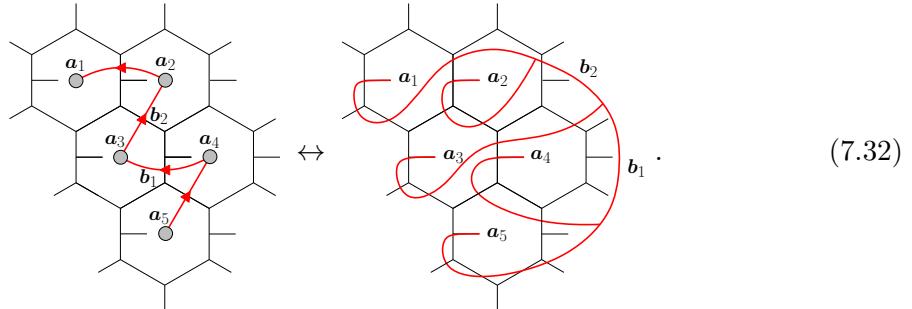
Note that the way in which the leaf ribbons wrap around the plaquettes containing

anyons is not uniquely determined by this construction. It is precisely this freedom that translates to local Dehn twists in individual plaquettes. While the construction could be modified in order to remove this freedom, there is no need for us to bother with such details. Our simulations are set up such that we never need to store any superposition in the anyon labels of individual plaquettes. As such, local Dehn twists give rise to global phases, which have no physical relevance.

The relation between curve diagrams and anyonic fusion bases detailed above, does not specify the handle labels. However, the nature of the simulation does not require the ability to store these values. The handle label (or superposition of such labels), can only be modified by processes in which a pair of anyons on the same curve interact along a path that is *not* homologically equivalent to the piece of curve between them. Since such events are precisely those for which the simulation declares a failure, we do not need the ability to update the handle label. Futhermore, since the total charge of all created excitations is trivial, the value of the handle label has no influence on the outcome of any physical process. Hence, all we need to store is the fusion state of anyonic excitations created *on top of* some initial ground state.

7.6.1 Data-structure

In order to simulate the noise and error correction processes, one needs to be able to efficiently store and update the basis in which the quantum state of the system is expressed. The curve diagrams, as defined above, provide a convenient way of doing so. An efficient method for storing the curve diagrams describing the current basis was introduced in Ref. [55]. We slightly modify this construction in order for it to be better suited for the system at hand.

We will store the configuration of curves by storing the shape of the curves running through each individual plaquette separately. Since the presence of the tail edges is irrelevant in this context, we will represent the curve diagram layout on hexagonal *tiles* (each of which corresponds to a plaquette). The total configuration of the curves can be obtained by piecing all tiles together.

In order to represent the curves of the connected groups of anyons on the lattice, we assign to each connected curve a unique integer label. An example of three connected curves on a 4×4 lattice is depicted in Fig. 7.3a.

Since they represent the layout of the curve diagrams on the lattice, each tile may contain at most one anyon, and we require that the different curves intersect the edges of tiles transversely.

In order to save the configuration of curves inside a tile we first assign a unique integer label to each *piece of curve* inside the tile. By a *piece of curve*, we mean a segment of a curve diagram connecting either two edges of the tile, or an edge and an anyon. Note that every piece of curve has an orientation. The edges of a tile are numbered clockwise from 1 to 6 as depicted in Fig. 7.4a. For each edge, we record the label and orientation (+1 for an incoming line, -1 for an outgoing line) of each piece of curve intersecting it, in the order in which they are encountered when going clockwise around the tile. In addition, when an anyon is present inside the tile, we also store which pieces of curve are connected to it. Finally, for every piece of curve inside the tile, we store the curve label, along with its position inside that curve. If a piece of curve is positioned between the nth and $n+1$th anyons on a curve (following the orientation of the curve), its position label is n.

An example of such a tile is depicted in Fig. 7.4b. The configuration inside this tile
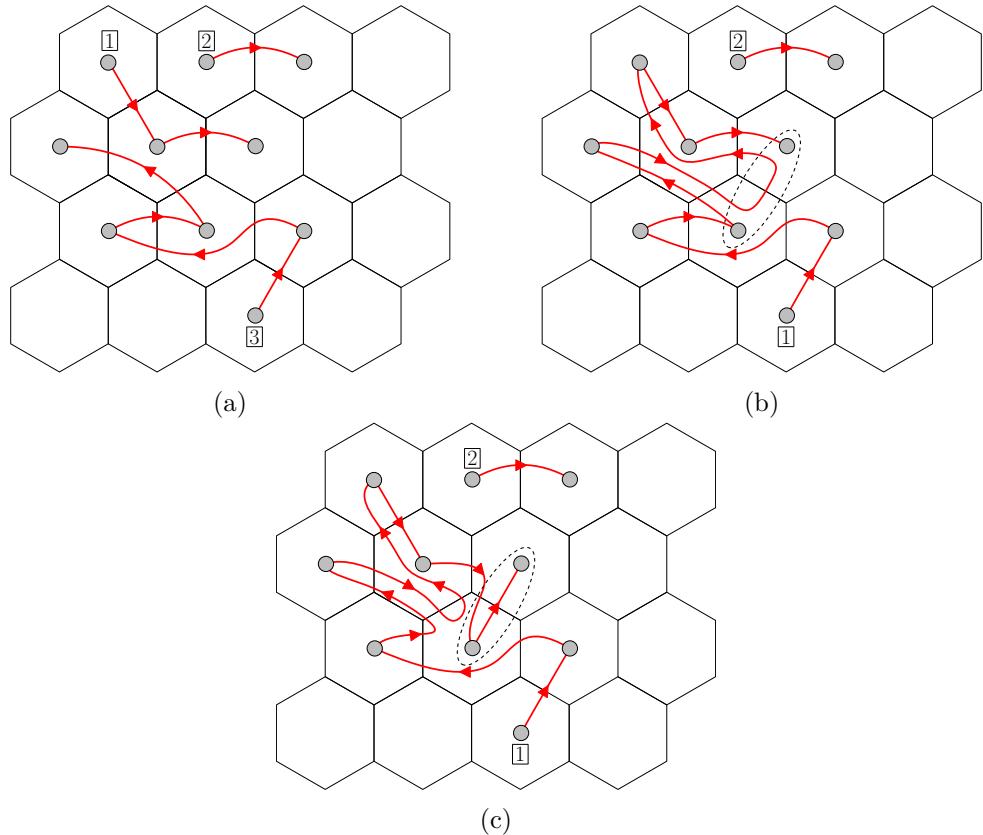
Figure 7.3: (a): Example of a configuration of three connected curves on a 4×4 periodic hexagonal lattice. Each curve is assigned a unique integer label, depicted here in the square boxes at the start of each curve. (b): Result of a merge procedure in the case the two anyons in the dashed ellipse interact. (c): Result of repeated use of refactoring moves to make the the anyons in the dashed ellipse appear sequentially in the same curve.

would be stored as follows:

$$
\begin{bmatrix} \end{bmatrix} \begin{bmatrix} 1 & +1 \end{bmatrix} \begin{bmatrix} \end{bmatrix} \begin{bmatrix} \end{bmatrix} \begin{bmatrix} 1 & -1 \\ 2 & +1 \end{bmatrix} \begin{bmatrix} 3 & -1 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix} \begin{bmatrix} 1 & 3 & 2 \\ 2 & 3 & 3 \\ 3 & 3 & 4 \end{bmatrix}. \tag{7.33}
$$

Here, the first six arrays correspond to the labels and orientations of lines crossing each of the six tile edges, respectively. The next to last array indicates that lines 2 and 3 are connected to the anyon in the tile. The last array indicates the curve and position along the curve of each piece of curve. In this case, all lines belong to curve 3, and the lines with labels 1, 2, 3 appear in the curve after the second, third and fourth anyon along the curve, respectively.

7.6.2 Merge

If neighboring anyons on the lattice that lie on different connected curves interact at some point, their corresponding curves must be merged in order to compute the effect of the interaction. In terms of fusion trees, one must think of this procedure as connecting two separate fusion diagrams with a line carrying the trivial label.

The curves are merged by connecting the end of one curve to the start of the other,
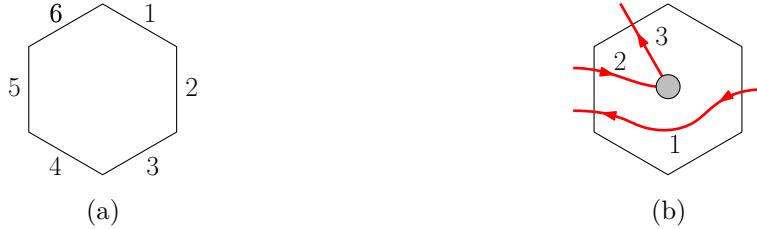
Figure 7.4: (a): Labeling of the edges of a hexagonal tile in a clockwise direction. (b): Example of a configuration of lines running through tile 10 in Fig. 7.3a (when counting from left to right and from top to bottom). Each line segment is assigned a unique integer label.

with the condition that the combined path of the resulting curve and the interaction[2] does *not* contain any homologically nontrivial loop. Other than this requirement, the way in which the curves are merged is arbitrary, one simply has to connect the end of one curve to the start of the other in some suitable way.

In our framework merging is implemented by extending the end of one curve parallel along the curve in its reverse direction until it reaches the tile of one of the interacting anyons. The extended curve is then crossed over to the tile of the other anyon, after which it follows the other curve in its reversed direction and is attached to the start of the latter. An example of this merging procedure is depicted in Fig. 7.3b.

Updating the state superposition in the case of a merge operation is trivial, as it simply amounts to taking the tensor product of the state superpositions associated to each curve.

7.6.3 Passive exchange

Calculating the outcome of both individual Pauli errors and fusion operations, requires expressing the state in the appropriate anyonic fusion basis. Basis transformations concerning individual curves are performed using passive exchanges or *swaps*, which are represented as follows for the clockwise and the counterclockwise case, respectively:

$$S^{ab} : \quad \mapsto \quad , \tag{7.34}$$

$$\left(S^{ab}\right)^{-1} : \quad \mapsto \quad . \tag{7.35}$$

It is important to stress that this does *not* represent an active braid move in which two anyons are exchanged. This is a basis transformation: the quantum state of the system remains unchanged, and the coefficients of the state superposition must be updated to express this state in a new basis.

The right transformation on the state vector components is found by considering the relation between the anyonic fusion bases corresponding to the left and right hand side of

---

[2]By the path of an interaction we mean the shortest path connecting any pair of plaquettes affected by it that initially belong to two different curves.

these equations. For a clockwise swap, represented in Eq, (7.34), this relation is

$$\raisebox{-1em}{(diagram)} = \sum_{b_2'} B_{a_{j+1}b_3 b_2'}^{b_1 a_j b_2} \raisebox{-1em}{(diagram)}, \tag{7.36}$$

where

$$B_{a_{j+1}b_3 b_2'}^{b_1 a_j b_2} = \sum_{c} F_{b_3 b_1 b_2'}^{a_j a_{j+1} c} R_c^{a_j a_{j+1}} F_{a_{j+1} b_3 c}^{b_1 a_j b_2}. \tag{7.37}$$

Analogously, for the counter clockwise case, one finds

$$\raisebox{-1em}{(diagram)} = \sum_{b_2'} \left(B_{a_{j+1} b_3 b_2'}^{b_1 a_j b_2} \right)^* \raisebox{-1em}{(diagram)}. \tag{7.38}$$

7.6.4 The paperclip algorithm

In order to determine the outcome of an interaction on a set of anyons, be it some local noise operator or the fusion of a pair, we must always transform to a basis where these anyons appear sequentially in the linear ordering determined by their curve diagram. Below, we outline how such a basis transformation can be performed using a sequence of swaps for the case where only 2 anyons are involved. The general case for n anyons can then be deduced iteratively.

In general, any two curve diagrams f and f', containing the same anyons, and such that the combined path of f and f' does not contain any homologically nontrivial loop, can be related to each other by a sequence of swaps. The algorithm for determining this sequence, was introduced in Ref. [79] and dubbed the *refactoring algorithm*. We will describe a different but entirely equivalent formulation of this algorithm, called the *paperclip algorithm* which is more convenient for our purpose. This alternative formulation was introduced in Ref. [55], and determines the sequence of swaps corresponding to a basis transformation where we "move" an anyon (or rather, its position on the curve) to the next piece of curve encountered when moving along the boundary of its tile in a clockwise fashion, as shown in the following diagram:

$$\tag{7.39}$$
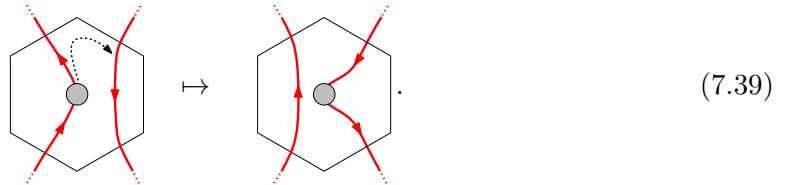
We will call such transformations *refactoring moves*. Schematically, their action on the curve diagram can be represented as follows:

$$\tag{7.40}$$

The initial position and destination of a refactoring move divide a curve into three disjoint segments which we name *tail* (T), *body* (B) and *head* (H), respectively (following the orientation of the curve). Their content is defined as follows:
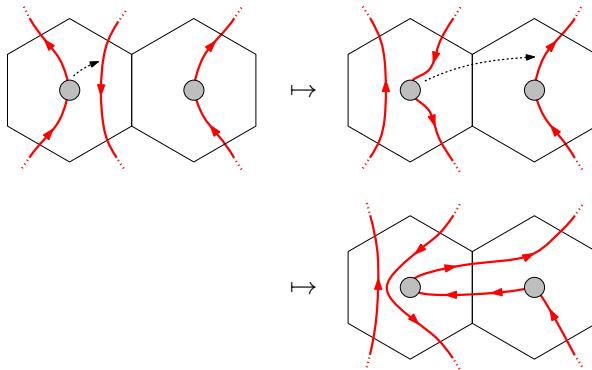
Figure 7.5: Repeated use of refactoring moves to make a pair of anyons in neighboring tiles appear subsequently on the same curve.
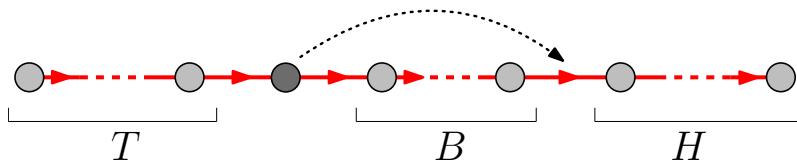
where the dotted line represents the refactoring move. Note that we do not include the anyon which is being moved (indicated in dark above) in any of these segments.

As interacting anyons are always neighbors on the lattice, repeated use of such moves can be used to obtaining a curve diagram where these anyons appear subsequently on the same curve. An example of how this is done this is given in Fig. 7.5, where we assumed that all lines in the diagram belong to the same curve.

If any lines lie between the two interacting anyons but do not belong to either of their curves, one of the following two actions must be performed. In case the interacting anyons do not belong to the same curve initially, one can attempt to pull such lines through one of these curves to get it out of the way. Since each curve corresponds to a connected group of anyons with trivial total charge, such a deformation does not affect the state vector and is therefore permitted. Whenever this is not possible (i.e., when that does not "remove" the obstructing line), the corresponding curve diagram is essentially trapped between the others, and one is forced to merge it with those containing the interacting anyons, before proceeding with the clockwise "moves". The latter is of course not desirable, since it increases the size of the associated fusion space, but there are situations where this is unavoidable. Of course, in case the pair of anyons are members of different curves initially, these must be merged in the process.

The sequence of swaps corresponding to the refactoring moves described above can be determined with the corresponding *turn number*. This number is found by counting the number of right hand 60 turns made by the curve between the initial position of the anyon and its destination (that is, the next piece of curve that intersects the tile boundary). Starting at the anyon's initial position, every 60 right hand turn contributes $+1$, while every left hand turn contributes[3] -1. In order to ensure that the turn number is independent of where the "destination piece of curve" enters the tile, the total value must then be decreased by 1 for every additional edge of the tile boundary we have to move to (in a clockwise fashion) before encountering it (0 if it appears directly after the initial line, -1 if it appears on the next edge, ...).

---

[3]Note that we always count the number of right hand turns while following the curve from the start towards the destination of the refactoring move, independently of whether or not the refactoring is along or against the orientation of the curve itself

Depending on whether the refactoring move is with or against the orientation of the curve, and depending on whether the piece of curve associated with the initial position of the refactoring move has an incoming or outgoing orientation, it is always isotopic to one of 4 different "paperclips", each of which is associated to a specific turn number. For example, when the anyon is transported along the curve, and the initial piece of curve has an incoming orientation, the possible turn numbers and corresponding "paperclips" are as follows:
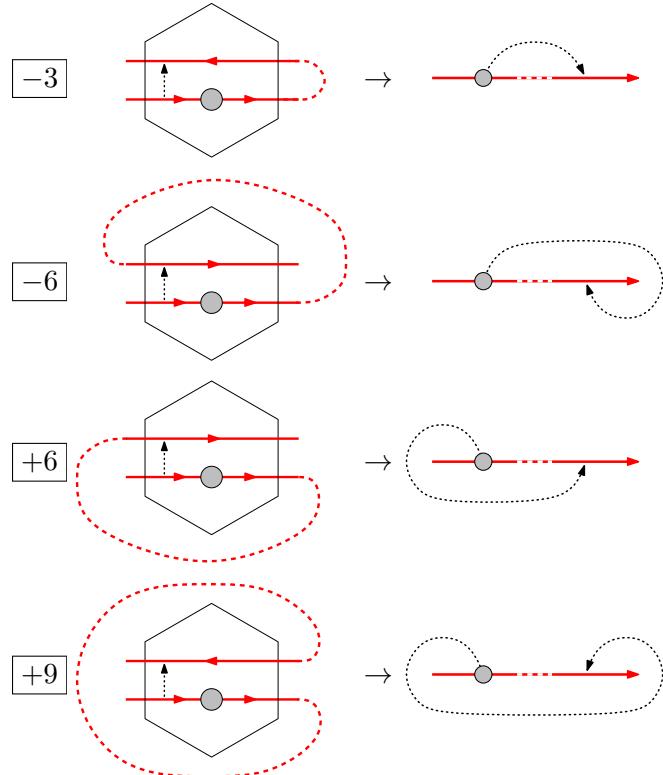
The appropriate sequences of swaps for these four different situations are

$$
\begin{aligned}
-3: & \quad S^{-1}[B]\,, \\
-6: & \quad S^{-1}[B]\,S^{-1}[H]\,S^{-1}[H^r]\,, \\
+6: & \quad S[T^r]\,S[T]\,S[B]\,, \\
+9: & \quad S[T^r]\,S[T]\,S[B]\,S[H]\,S[H^r]\,,
\end{aligned}
$$

where the notation $S[H]$ stands for sequentially swapping the anyon with all entires in H in a clockwise fashion. H^r indicates the sequence of anyons in H, in reversed order. The paperclip configurations for the other cases are similar. All of them are listed, together with the corresponding sequences of swaps, in App. D.

7.6.5 Fusion

The fundamental operation during recovery is a pairwise fusion process, where one excitation is moved along a specific path until it neighbors another anyon, followed by the fusion of the pair. The physical implementation of these operations is discussed in Sec. 4.3. Here we discuss how they are implemented on the level of curve diagrams during the simulation.

Before fusing neighboring anyons, one must ensure that they appear sequentially on the same curve diagram, which is by achieved using the paperclip algorithm described

above. To simulate the fusion process, we must first isolate the two anyons from the rest of the fusion tree, which is done using an *F*-move:

$$
\begin{array}{c}
\boldsymbol{a}_j \quad \boldsymbol{a}_{j+1} \\
\vdots\!\!\begin{array}{c}|\;\;\;|\end{array}\!\!\vdots \\
\boldsymbol{b}_1 \quad \boldsymbol{b}_2 \quad \boldsymbol{b}_3
\end{array}
= \sum_{c} F^{\boldsymbol{b}_1 \boldsymbol{a}_j \boldsymbol{b}_2}_{\boldsymbol{a}_{j+1}\boldsymbol{b}_3 \boldsymbol{c}}
\begin{array}{c}
\boldsymbol{a}_j \quad \boldsymbol{a}_{j+1} \\
\boldsymbol{c} \\
\boldsymbol{b}_1 \quad \boldsymbol{b}_3
\end{array}
. \tag{7.41}
$$

The result of the fusion of anyons $\boldsymbol{a}_j$ and $\boldsymbol{a}_{j+1}$ is then picked from the possible values of their total charge $\boldsymbol{c}$ using the probability distribution dictated by the coefficients of the state superposition and the state vector is then projected to the selected outcome. Note that this probability distribution only concerns the resulting anyon charge of the plaquette. In case the outcome $\boldsymbol{c} = \tau\tau$ was selected, the tail label is picked from the probability distribution $\{p(\mathbf{1}) = \frac{1}{\phi^2},\, p(\tau) = \frac{1}{\phi}\}$, which follows from Eq. (4.18).

On the level of the curve diagrams, we must then remove one of the anyons as depicted Fig. 7.6. Note that we are required to choose a "target plaquette" in which the fusion outcome is placed.
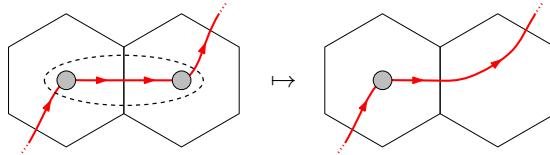
Figure 7.6: Update of the curve diagram following the fusion of the anyons in the dashed ellipse.

7.6.6   Move and exchange

The moving procedure required to bring a pair of anyons to neighboring tiles can be broken down into a sequence of basic steps where an anyon is moved to a neighboring plaquette. If the neighboring plaquette does not contain an anyon, the curve is updated by moving the excitation to the neighboring tile while continuously deforming the curves in the corresponding tiles, as depicted in Fig. 7.7a. If the neighboring plaquette contains an anyon, the two anyons are exchanged in a clockwise fashion while deforming the curves in the corresponding tiles accordingly, as depicted in Fig. 7.7b. Note that the choice for a clockwise exchange over a counterclockwise one is arbitrary. A different choice would result in different probabilities for the fusion outcomes after the moving procedure, as remarked in Sec. 7.1.

Both operations do not affect the coefficients appearing in the state vector. However, they *do* modify the state by changing the corresponding basis elements. Such a transformation can be expressed as

$$
|\Psi\rangle = \sum_{i} \alpha_i |\psi_i\rangle \mapsto |\Psi'\rangle = \sum_{i} \alpha_i |\psi_i'\rangle \,, \tag{7.42}
$$

where the basis state $|\psi_i'\rangle$ is obtained from the state $|\psi_i\rangle$ by changing the embedding of the corresponding fusion tree on the surface as dictated by action on the curve diagram in Fig. 7.7.

7.7   Outline of the simulation

With all the groundwork completed, we are now ready to sketch the outline of the entire error correction threshold simulation, performed with the Fibonacci input category on

(a)

(b)
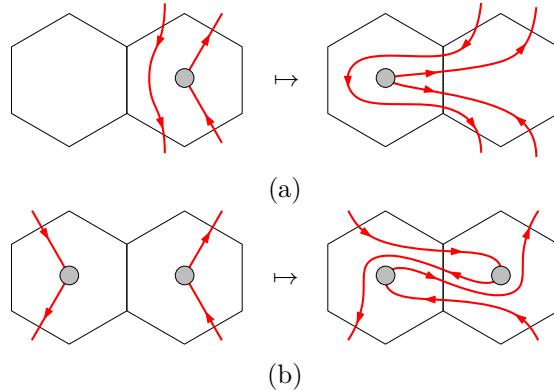
Figure 7.7: Example of the effect of a move (a) and clockwise exchange (b) operation on the curves.

a tailed hexagonal lattice with periodic boundary conditions in both directions (giving the lattice the topology of a torus). The simulation consists of a fixed number of Monte Carlo samples, each of which simulates the application of noise and recovery processes to an initial ground state. The quantum state of the system is tracked throughout these processes until either a topologically nontrivial process occurs, in which case *failure* is declared, or all anyonic excitations have been removed (without any logical errors), in which case *success* is declared.

For a given a system size and noise strength, the logical failure rate P_L is then found by the ratio of failures compared to the total number of Monte Carlo steps. Below, we describe in detail all the important aspects of a single Monte Carlo step, with system size $L \times L$ and a noise strength characterized by p.

7.7.1 Cutoff parameters

In addition to logical errors, failure is also declared whenever any of two cutoff parameters are exceeded. The first of these is the maximal allowed tree size $\mathtt{N_{max}}$. As discussed in Sec. 7.2, the size of connected groups of anyons can grow very large in some rare occasions. Since the size of the associated fusion space grows exponentially, such situations are extremely costly, both in terms of memory usage and in computation time. Hence we fix some cutoff size $\mathtt{N_{max}}$, and the simulation is aborted, and a failure is declared, whenever the number of anyons on an individual curve exceeds this value. Note that a similar cutoff was used in Ref. [55].

The second cutoff parameter that we introduce is $\mathtt{V_{max}}$, which is the maximal number of nonzero coefficients we allow in the vector associated to any individual curve. The motivation behind this cutoff rule is as follows. Since the state vectors appearing during the simulation are generally very sparse, these are stored as sparse arrays. This enables us the keep the value of $\mathtt{N_{max}}$ higher than one would naively expect (as no memory is allocated to all zero entries, which form the vast majority of the exponentially large state vector). To avoid the extreme time and memory cost of the rare cases where any state vector contains a very large number of nonzero entries, such cases are aborted and a failure is declared.

One must choose these cutoff parameters to be as high as possible, in order to minimize the amount of triggered cutoffs, while still keeping the memory and time cost of the simulations reasonable. Of course, any finite values of these parameters will negatively

affect the observed logical failure rates, once we leave the regime in which events with very large connected groups are sufficiently rare. However we argue that their influence can only *lower* the obtained threshold, meaning our results will provide a valid lower bound on the actual error correction threshold, independent of the values of $\mathtt{N_{max}}$ and $\mathtt{V_{max}}$. For a fixed value of p, larger system sizes (on average) result in higher values for the size of the largest connected group of anyons (see App. B). Hence, it is clear that the cutoffs will be triggered more often for larger system sizes. Likewise, for a fixed system size L, larger values of p will also result in more triggered cutoffs. When displaying the logical failure rate as a function of p, the intersection of the curves corresponding to different system sizes, will be shifted left compared to its true value (had the cutoff parameters been infinite), meaning that our obtained error correction threshold is indeed a valid lower bound to its unknown true value.

7.7.2   Noise phase

The system is initialized in a ground state of the code Hamiltonian Eq. (3.8), corresponding to the anyonic vacuum with some specific handle labels. However, as explained in Sec. 7.6, there is no need to store these handle labels, since they do not affect any relevant processes. Processes in which the handle labels do affect the outcome are precisely those that result in a logical error. Because the Monte Carlo simulation will automatically declare a failure in those cases, such processes must never be simulated on the level of state-evolution.

The first half of the simulation consists of sequentially applying T noise processes, described in Sec. 7.3, to this initial ground state, where T is drawn from a Poisson distribution with mean $p\,5L^2$. For each of these T steps, an edge e is chosen at random, and a Pauli operator σ_i is selected according to the relative probabilities $\{\gamma_x, \gamma_y, \gamma_z\}$. Before the matrix elements computed in Sec. 7.5 can be used to determine the state after the application of error σ_i^e, one must first rewrite the state vector in the appropriate basis.

Given the orientation of e and the type of error σ_i, the affected plaquettes can be read off from Figs. 7.1 or 7.2. If none of them contain a nontrivial charge initially, we simply create a new curve diagram with the appropriate shape and with a corresponding trivial fusion state (containing only vacuum charges). If only some of these plaquettes contain a nontrivial charge, we can add vacuum charges to the fusion state of one of the nontrivial charges and modify ("grow") the curve accordingly such that all affected plaquettes are included in the same cure.

The desired basis is the one where the affected anyons appear sequentially along the same curve, in the order depicted in Figs. 7.1 or 7.2. This is obtained by applying a series of refactoring moves as described in Sec. 7.6.4, and performing the corresponding sequences of swaps and merge operations on the affected state vectors. If any of the required refactoring moves is topologically impossible, for instance the one depicted in Fig. 7.8, this indicates that a logical error would be caused, as the joint path of the curve diagram and the interaction path form a non-contractible loop. Whenever this happens the simulation declares *failure* and aborts the current Monte Carlo step.

If all refactoring moves above were legal, a sequence of F-moves is performed to isolate the affected anyons from the rest of the fusion tree. This transforms the standard fusion tree shape to the one in Eq. (7.4), Eq. (7.13), or a similar shape in the case of three affected plaquettes. In terms of ribbons in the fattened lattice, the basis then locally looks like the ones depicted in Fig. 7.1 or Fig. 7.2, depending on the type of error. Note that such a basis can no longer be represented using curve diagrams. This is not an issue, since we will return to a standard basis before the full machinery of curve diagrams is required again.
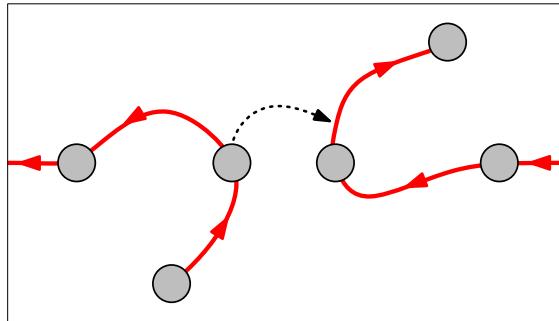
Figure 7.8: An illegal refactoring move on a torus, which will cause the simulation to declare *failure* and abort.

For the case of a σ_z error, the coefficients in the state vector and the matrix elements Eq. (7.23) are used to calculate the probabilities Eq. (7.7) of the different possible charge measurement outcomes. A result is picked according to this probability distribution, after which the state vector is updated using Eq. (7.8).

In the case of a σ_x error, the matrix elements (7.19) are used together with the coefficients in the state vector to compute the probabilities Eq. (7.15) for the outcome of the vertex and tail measurements. A result is then sampled from this distribution. Next, the matrix elements Eq. (7.20) are used to compute the probabilities Eq. (7.17) for the various outcomes of the charge measurement (that is performed after the appropriate unitary vertex correction was applied). We again pick a result according these probabilities, and update the state vector using Eq. (7.18). In case of a σ_y error, we proceed analogously to the σ_x case, using the matrix elements Eq. (7.21) or Eq. (7.22).

After the state vector has been updated to reflect the collective effect of the noise and the measurements (with the specific outcomes that we picked at random above), we conclude the current "noise step" by transforming the fusion basis back to the one that corresponds to the curve diagram we ended up with earlier (where all affected anyons appear sequentially). This is done by a sequence of F-moves that reverts the transformation performed by the first series of F-moves.

7.7.3 Recovery phase

After completing the process above T times, the error syndrome is given by the locations and charges of all thermal anyons on the lattice. The decoding algorithm is then used to determine an appropriate recovery step. Such a recovery step can be broken down into a sequence of pairwise fusion processes of anyonic excitations. In each such a pairwise fusion process, one member of the pair is moved along a specific path on the lattice until it neighbors the other. The moving procedure consists of basic operations where the anyon is moved to a neighboring tile and the curves are continuously deformed accordingly, as described in Sec. 7.6.6. This basic moving step is repeated until the two anyons reside in neighboring tiles. A sequence of refactoring moves (and possibly merges) is then performed to obtain a basis in which they are direct neighbors on the same curve, and the corresponding sequences of swap and merge operations are applied to the affected state vectors. As during the noise process, any illegal refactoring moves cause the simulation to abort the current Monte Carlo step and report a decoding failure. Note that this happens precisely when the intended fusion would create a non-contractible loop in terms of the curve diagrams, which in turn corresponds to a logical error.

If necessary, an *F*-move is applied to transform to a fusion tree shape where the anyons are fused directly. Their resulting charge is then projected according to the probabilities dictated by the coefficients in the state superposition, and it is placed in one of the two neighboring plaquettes while the state vector and the curve diagram are updated accordingly.

This basic pairwise fusion process is repeated until the current recovery step is completed, at which point the resulting error syndrome is used to determine the next recovery step. This dialogue is iterated until either all anyonic excitations fused away and decoding is successful, or a logical error occurs during some fusion process and a decoding failure is declared.

Note that throughout the entire Monte Carlo step, the system is constantly monitored for violations of the cutoff parameters. If at any point an individual curve contains more than $\mathtt{N_{max}}$ anyons, or a state vector contains more than $\mathtt{V_{max}}$ nonzero elements, the current Monte Carlo step is automatically reported as a failure.

8 | Numerical results

The Monte Carlo simulations described in Chapter 7 were performed for the three differ-
ent decoders described in Chapter 5. Individual Pauli errors were picked using relative
probabilities corresponding to the following noise models:

- depolarizing noise: $\gamma_x = \gamma_y = \gamma_z = \frac{1}{3}$,

- dephasing noise: $\gamma_x = \gamma_y = 0$, $\gamma_z = 1$,

- bit-flip noise: $\gamma_x = 1$, $\gamma_y = \gamma_z = 0$.

Simulations were performed for a wide range of physical error rates. For depolarizing
noise and pure bit-flip noise, we considered the linear system sizes $L = 10, 12, 14, 16, 18$.
For dephasing noise the values $L = 12, 14, 16, 18, 20, 22$ were used.

The logical failure rate for each (p, L)-pair was computed by averaging over 10^5 Monte
Carlo samples. For the lowest error rates $p = 0.01$ (or $p = 0.02$ in case of dephasing noise),
10^6 Monte Carlo samples were used in order to improve the accuracy of our results. As
visible in the results below, this is ample to guarantee sufficiently small (95%) confidence
intervals for the average logical failure rates.

All simulations were done with the following values for the cutoff parameters:

$$\mathtt{N_{max}} = 27,$$
$$\mathtt{V_{max}} = 2.5 \cdot 10^7.$$

For depolarizing noise and bit-flip noise with $L = 18$ the ratio of aborted Monte Carlo
samples near the observed thresholds is shown in the table below. The corresponding ratio
of aborted failures, is indicated between parentheses.

	Clustering	Fusion-aware MWPM	Blind MWPM
Depolarizing noise	6.1 % (17.4%)	0.6 % (2.1%)	0.8 % (2.9 %)
Bit-flip noise	4.6 % (15.1 %)	0.3 % (1.3 %)	0.4 % (1.7 %)

We found that the ratio of aborted Monte Carlo samples drops rapidly below the threshold.
For instance, at $p = 0.04$ less than 1.1% of Monte Carlo samples (7.4% of reported failures)
were aborted for depolarizing noise with $L = 18$.

For dephasing noise, the rate of aborted iterations is negligible for all system sizes and
noise strengths we studied.

Determining the threshold

The error correction threshold is given by the critical value p_c below which the logical failure rate P_L is exponentially suppressed in terms of the system size L. For large system sizes, the threshold manifests itself as the physical error rate for which the logical failure rates of all system sizes coincide. Hence, a rough estimate of the threshold can be obtained by plotting P_L in function of p for different system sizes and finding the error rate at which the various curves intersect.

A more accurate estimate for the error correction threshold can be obtained using the critical exponent method of Ref. [83]. This method was introduced in the context of the toric code, where an exact mapping to a statistical model (the 2-dimensional random-bond Ising model(RBIM)) is known [27, 83]. The RBIM is known to undergo a phase transition from an ordered to a disordered phase as the parameter corresponding to the physical error rate increases. This implies a phase transition in the logical failure rate of the toric code. Wang et al. demonstrated that in the regime $L \gg |p - p_c|^{-\nu}$, where ν is the critical exponent for the correlation length in the RBIM, the logical failure rate P_L depends only on the dimensionless ratio $L(p - p_c)^{\nu}$.

While the statistical model corresponding to the Fibonacci Turaev-Viro code is not known, it is expected that a similar scale invariant behavior occurs near the threshold here as well. Specifically, for sufficiently large system sizes, we define the rescaled variable

$$x = (p - p_c)L^{1/\nu}\,, \tag{8.1}$$

where ν is some critical exponent, such that the logical failure rate P_L as a function of x is explicitly scale invariant. We can find the correct values for p_c and ν by fitting the values of P_L to the quadratic ansatz

$$P_L(x) = A + Bx + Cx^2\,, \tag{8.2}$$

originating from a truncated Taylor expansion in the neighborhood of $x = 0$ ($p = p_c$). We perform this fit explicitly with the data obtained for the clustering decoder with depolarizing noise and dephasing noise.

8.1 Clustering decoder

The logical failure rate P_L of the clustering decoder in function of the noise strength p, are shown in Fig. 8.1a, Fig. 8.1b and Fig. 8.1c for depolarizing, dephasing noise and bit-flip noise, respectively. These results clearly manifest threshold behavior. For pure bit-flip noise, the threshold can be estimated from the corresponding plot as $p_c \approx 0.0375 \pm 0.0025$. A more precise estimation, based on the finite-size ansatz discussed above, was made for depolarizing noise and dephasing noise.

For depolarizing noise, the above finite-size scaling ansatz was fitted to the logical failure rates for p ranging from 0.045 to 0.05 in increments of 0.00125. The following values were found using a non-linear least squares fit:

$$p_c = 0.0470 \pm 0.0011\,,$$
$$\nu = 1.62 \pm 0.33\,.$$

For dephasing noise, the ansatz was fitted to the logical failure rates obtained for p ranging from 0.07 to 0.075 in increments of 0.00125. With this data, we found

$$p_c = 0.0732 \pm 0.0006\,, \tag{8.3}$$
$$\nu = 1.17 \pm 0.08\,. \tag{8.4}$$

The confidence intervals were estimated using the jackknife resampling method. In both cases the obtained threshold is compatible with the rough estimate based on the crossing of the curves in Fig. 8.1. The logical failure rates in terms of rescaled error rate x defined in Eq. (8.1) are shown in Figs. 8.2a and 8.2b for depolarizing noise and dephasing noise, respectively. One can see that the obtained parameters do indeed result in a clear "collapse" of the data, as predicted by the finite-size scaling hypothesis.

(a)

(b)

(c)
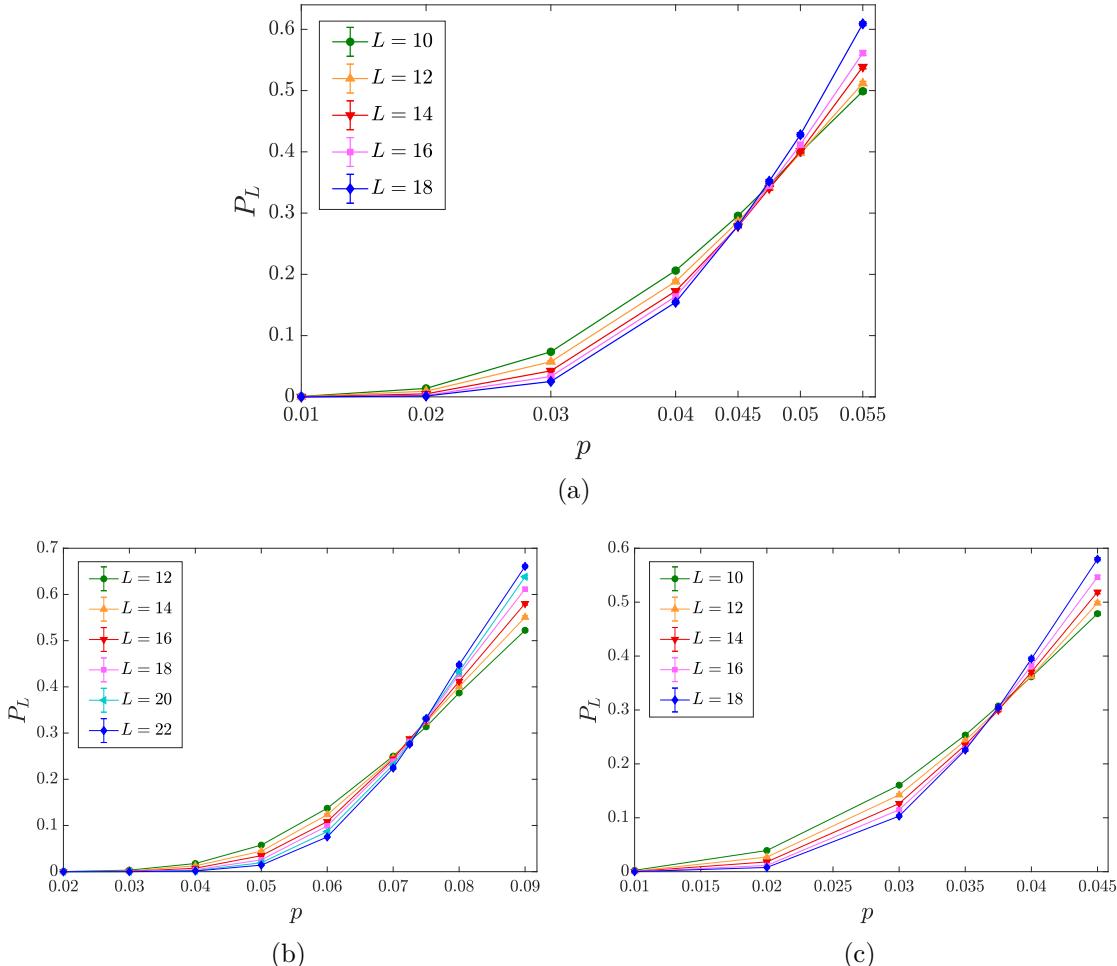
Figure 8.1: Logical failure rate P_L as a function of the physical error rate p for the clustering decoder with (a) depolarizing noise, (b) pure dephasing noise, and (c) pure bit-flip noise.

It is no surprise that the highest threshold is found for dephasing noise: no more than two anyonic excitations can be created by a single σ_z error, while up to four anyons can be created by a single σ_x or σ_y error. Hence, the average number of new anyonic excitations created in each time step is the lowest for dephasing noise and the highest for pure bit-flip noise, with the value for depolarizing noise lying somewhere in between these two extremes. The discrepancy between the thresholds for dephasing and bit-flip noise indicates that for biased noise, there is a preferred choice for the computational basis of the physical qubits.
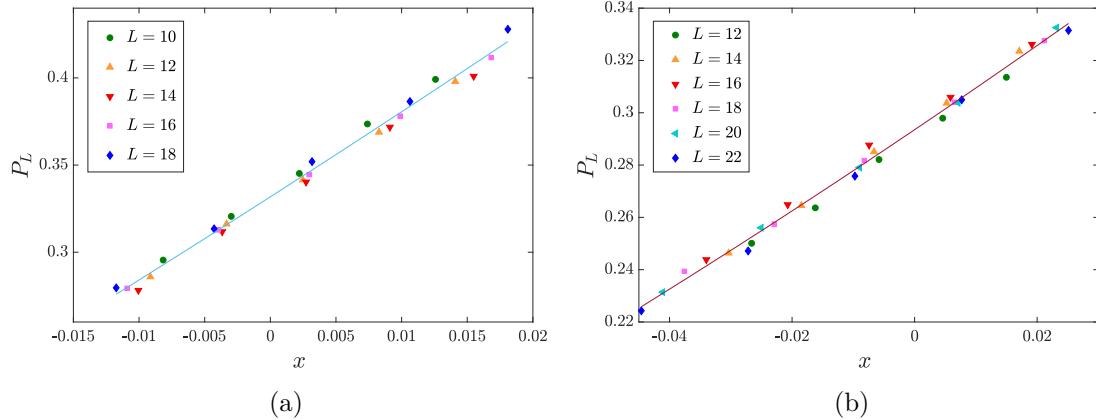
Figure 8.2: Logical failure rate P_L as a function of the rescaled error rate $x = (p - p_c)L^{(1/\nu)}$ for (a) depolarizing noise and (b) dephasing noise. The solid line represents the best fit of the model $P_L = A + Bx + Cx^2$.

8.2 Iterative matching decoders

We consider two types of iterative MWPM decoders: a fusion-aware one and a blind one (see Chapter 5). The performance of these decoders under both types of noise are shown in Fig. 8.4.

Note that the threshold obtained with these two types of decoders are very close. Under depolarizing noise, both decoders exhibit a threshold around $p_c \approx 0.0300 \pm 0.0025$. Under dephasing noise and bit-flip noise, respectively, we find $p_c \approx 0.0600 \pm 0.0025$ and $p_c \approx 0.0250 \pm 0.0025$ for both decoders. However, when closely comparing the results, as in Fig. 8.3, one finds a slight overall advantage for the fusion-aware decoder in the sense that its failure rates are lower than those obtained with the blind decoder.

On the other hand, we see that both matching decoders have worse performance and lower thresholds compared to the clustering decoder which has not used the detailed syndrome information of anyon types. This is related to the fact that clustering decoder seems to be more natural for the Fibonacci code than the matching decoder. It remains an open question whether the optimal decoder for the Fibonacci Turaev-Viro code is fusion-aware.

8.3 Discussion

The results above are expressed in terms of the average qubit error rate p in the fixed-rate sampling noise model described in Sec. 7.3. It is therefore not straightforward to compare these results to those of Abelian models such as the surface code, which are typically expressed in terms of an independent and identically distributed binomial noise strength $p_{\text{i.i.d.}}$. However, for Abelian codes, both noise models are equally valid and their respective noise strengths can be compared using the relation between the i.i.d. noise strength $p_{\text{i.i.d.}}$ and the error rate p of a fixed-rate sampling noise model derived in App. C.

Using this relation, one finds that the thresholds obtained for the clustering decoder under depolarizing ($p_c \approx 0.047$) and dephasing ($p_c \approx 0.073$) noise correspond to i.i.d noise strengths of 4.6% and 7.0%, respectively. Remarkably, despite the complexity of the extended string-net code and the fact that it is not known whether or not the clustering decoder is optimal for this this code, these values compare very favorably with the optimal
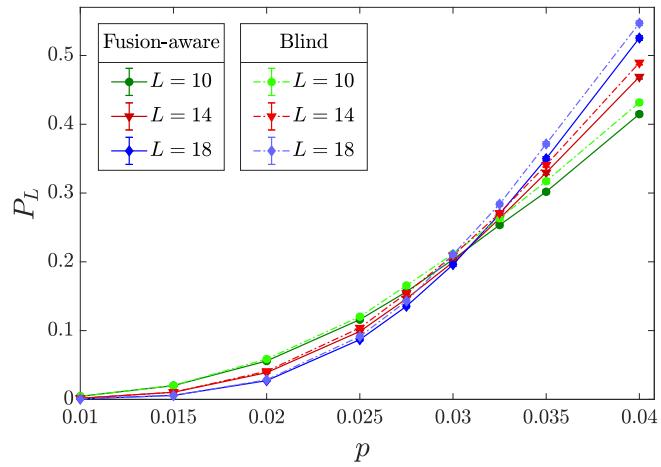
Figure 8.3: Comparison between the logical failure rates for the fusion-aware and the blind iterative MWPM decoders under depolarizing noise.

thresholds for the surface code[1] (under the assumption of perfect measurements), which are 18.9% for depolarizing noise [85] and around 10% for dephasing noise [27, 83].

---

[1]The results mentioned here were in fact obtained for the toric code (i.e., with periodic boundary conditions). It is likely that the true thresholds for surface codes are slightly lower [84].
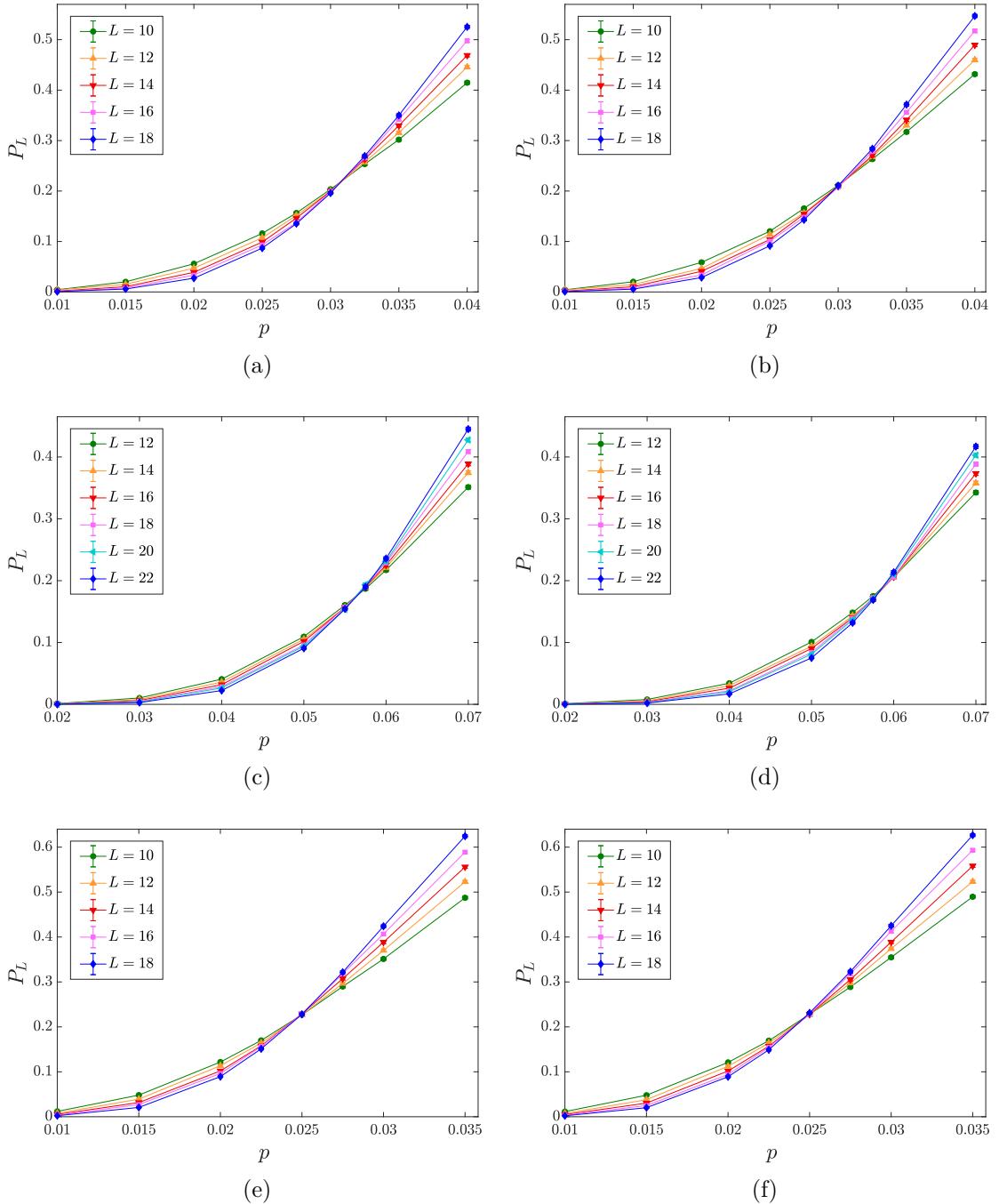
Figure 8.4: (a,c,e) Logical failure rate P_L as a function of the physical error rate p for the fusion-aware iterative MWPM decoder with (a) depolarizing noise, (c) pure dephasing noise, and (e) pure bit-flip noise.
(b,d,e) Logical failure rate P_L as a function of the physical error rate p for the blind iterative MWPM decoder with (b) depolarizing noise, (d) pure dephasing noise, and (e) pure bit-flip noise.

9 | Conclusion and outlook

In order to estimate the non-Abelian error threshold, we have combined concepts and techniques from three seemingly distant fields: quantum error correction, topological quantum field theory, and tensor networks. In particular, we have developed a complete error correction scheme and decoding protocols for the Fibonacci Turaev-Viro code, which supports a universal logical gate set via braiding or Dehn twists. Making use of the framework of tensor networks and tube algebra, we were able to estimate the code-capacity error correction threshold using a clustering decoder and a fusion-aware iterative matching decoder. The threshold of 4.7% obtained for the clustering decoder is comparable to the code-capacity error threshold of the Abelian surface code, which is around 10% [27, 59, 83].

A natural future extension of the work presented here will be the adaptation of our error correction protocols to take into account measurement errors, and to determine the error threshold of the code in the presence of both measurement and circuit-level noise. In this setting of full fault-tolerant error correction, our measurement scheme which allows to distinguish the charges of different anyonic excitations may prove useful, since this information could help in identifying measurement errors when performing repeated syndrome measurements through a consistency check. Thus, while it is still an open question what the optimal decoder for the Fibonacci Turaev-Viro code is and whether it would make use of the detailed charge information in the error syndrome, it is likely that this charge information would yield a notable advantage in the presence of measurement errors. More generally, the tensor-network representation used in the current work can also be further used to simulate coherent noise in these non-Abelian codes as well as in the usual surface code.

Another direction to explore is the application of our techniques to different models. A first interesting route would be to investigate other types of Turaev-Viro codes, such as the Ising Turaev-Viro code, which has doubled Ising anyons as excitations. While not universal for quantum computation by itself, this code would have a simpler non-cyclic fusion rule structure which could lead to a higher threshold, especially in the presence of measurement noise. In this context, our measurement scheme to extract the specific anyon charges would be particularly useful, since it was shown in Ref. [51] that fusion-aware decoders can yield a significant advantage for Ising-type anyons. A second important direction here will be to investigate non-Abelian codes with a simpler structure, such as lower-weight syndrome operator and lower-depth measurement circuits. The Levin-Wen string-net models are sophisticated in the sense that their plaquette syndrome operator has weight 16. On other hand, Kitaev's non-Abelian quantum double models [25] can have a weight-4 syndrome operator, and could possibly be analyzed using an adaptation of the techniques presented in this work. It would therefore be interesting to further explore the error threshold of these alternative models which could be more practical in terms of an experimental implementation.

A different avenue of further research would be the investigation of planar string-net codes with suitable gapped boundary conditions, where information is then encoded

in the fusion state of a number of well separated anyons rather than in the ground state degeneracy associated to a closed manifold with a nontrivial topology (high-genus surface). Alternatively, the logical information can also be encoded in the boundary degeneracy of an open manifold corresponding to a planar geometry in analogy with the Abelian surface code with e and m boundaries. This matter is of significant interest, since a planar geometry is highly attractive regarding experimental realization. The classification of these gapped boundaries has been performed in the language of (bi)module category theory [86], and recent progress has been made in capturing this formalism in terms of tensor network representations [87]. This latter strategy is not applicable to the Fibonacci Turaev-Viro code, since this model only admits a single type of gapped boundary [88, 89]. Nevertheless, the investigation of these concepts in the context of error correction using different models would be of great interest.

An additional advantage of non-Abelian codes appears when they are placed on a hyperbolic surface, which admits both constant-rate encoding ($O(1)$ space overhead) and parallel universal logical gates via constant-depth circuits [71]. A promising direction is to explore non-Abelian codes in higher spatial dimension or on an expander graph. Along this direction, the ultimate goal is to explore and achieve the fundamental limit of space-time overhead. This is because in higher dimension such as 4D, one can obtain a self-correcting quantum memory. In that case, local errors created when implementing a logical gate with a constant depth circuit [69, 70, 71] can be corrected locally in O(1) time. Eventually, this direction could evolve into a flourishing interface between quantum information and quantum topology.

An important aspect in terms of experimental implementation of the Fibonacci Turaev-Viro code is the realization of multi-controlled-Z (multi-qubit Toffoli) gates. Therefore, it would be interesting to further explore hardware-efficient implementations of multi-qubit gates, instead of always decomposing these into a longer sequence of two-qubit gates. Such multi-qubit gates are widely studied in Rydberg-atom and ion-trap systems, and it would be useful to further develop these gates in superconducting qubit systems as well.

Finally, besides the application to quantum error correction, the scheme developed in this dissertation also paves the way for quantum simulation of topological quantum field theory on a near-term quantum computer. In particular, the measurement and correction schemes will be a crucial ingredient for the state preparation of the TQFT wave functions.

Appendices

A | Consistency of the tensor network representation

From the PMPO description of topological order [68] it is known that an anyon ansatz in PEPS must satisfy certain consistency conditions. It was shown through numerical calculations that our PEPS representation of the anyonic fusion basis states indeed satisfies all properties required of a tensor network description of anyonic excitations, providing an important consistency check for our framework. In particular, the braiding and fusion behavior of the DFIB excitation tensors was explicitly verified. Below we showcase some of these consistency checks, which were performed numerically with the Fibonacci input category, but should hold in general for any modular ribbon category satisfying Eqs. (2.26), (2.27) and (2.28). It is worth noting that these consistency conditions are in fact implied by the construction of the tensors. The numerical checks merely confirm that no mistakes where made in the derivation.

We start by noting that the representation of ribbons on the virtual level as depicted in Eq. (6.9), can be understood as a MPO operator acting on the virtual level of a PEPS. The MPO tensor itself is given by what we have previously called the crossing tenor,
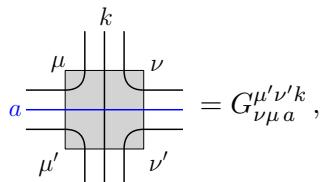
$$
\begin{array}{c}

\end{array}
= G^{\mu'\nu'k}_{\nu\mu a}\,,
\tag{A.1}
$$

where we have rotated the definition Eq. (6.8) over 90 degrees and we now denote the index that encodes the ribbon label a in blue. To match the convention of the MPO tensors in Ref. [68], the tripled lines should be grouped as done in Sec. 7.5. We refer to the label a of the virtual ribbon as the *block label* of the MPO tensor. For notational convenience we will often assume that the tripled lines are grouped, in which case we only explicitly write the block label for the grouped index. It should be emphasized however that we still maintain the closed loop convention introduced before, even when we depict indices as being grouped.

From the pentagon equation for the input category it follows that the MPOs can be moved freely through the groundstate PEPS vertex tensors, which is referred to as the *pulling through* property. This corresponds to the freedom to continuously deform the ribbons is Sec. 2.2, and the fact that the string-net ground state inherits this property.

Adding the single block MPOs with a weight $w_a = d_a/\mathcal{D}^2$ for each block a and closing the resulting operator into a loop results in a projector MPO (PMPO) that acts trivially on any region of the PEPS with vacuum total charge. This PMPO can be tought of as the virtual representation of a vacuum loop (divided by the total quantum dimension).

The fusion tensors Eq. (6.12) that represent ribbon fusion on the virtual level can be

interpreted as tensors that represent the fusion of MPOs of different blocks:

$$(\text{A.2})$$

Using these fusion tensors we can build a virtual representation of the tube algebra introduced in Sec. 2.5, which is then generated by elements of the form

$$(\text{A.3})$$

It was shown in Ref. [68] that these objects form a $\mathcal{C}^*$-algebra whose central idempotents correspond to the topological superselection sectors of the theory, which corresponds to the results presented in Sec. 2.5 that were we was obtained using the graphical calculus of the ribbon graph Hilbert space. In particular, we can derive the expression for the idempotents and nilpotents of the $\mathcal{C}^*$-algebra by simply realizing Eq. (2.69) on the virtual level. The operators of interest are the tube operators of the form $\mathcal{P}_{kl}^{a\bar{b}}$, that represent both the simple idempotents and nilpotents of the tube algebra. Just as in Sec. 2.5 these can be decomposed into a superposition of the basis elements Eq. (2.66) as

$$\mathcal{P}_{kl}^{a\bar{b}} = \sum_{\alpha\beta} P_{\alpha\beta k}^{(abl)} O_{kl\alpha\beta} \tag{A.4}$$

where the coefficients $P_{\alpha\beta k}^{(abl)}$ are given by

$$P_{\alpha\beta k}^{(abl)} = \frac{1}{\mathcal{D}^2} \frac{d_a d_b}{v_k} v_\alpha v_\beta \sum_{\gamma,\delta} d_\gamma d_\delta R_\gamma^{a\alpha} R_\delta^{\alpha b} G_{\alpha ab}^{k\delta\gamma} G_{b\alpha l}^{\beta a\delta} G_{a\gamma\delta}^{k\beta\alpha} . \tag{A.5}$$

On the virtual level, these operators can be represented by the tensor

$$(\text{A.6})$$

giving rise to an MPO of the form

$$\tag{A.7}$$
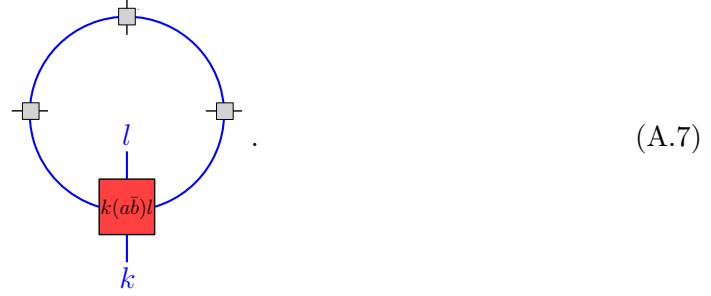
As the central idempotents $\mathcal{P}^{a\bar{b}}$ of the tube algebra can be constructed as $\mathcal{P}^{a\bar{b}} = \sum_l \mathcal{P}^{a\bar{b}}_{ll}$, the tensors Eq. (A.6) can be combined to give the square tensors representing the central idempotents of the $\mathcal{C}^*$-algebra

$$\tag{A.8}$$

giving a representation for the central idempotents on the virtual level.

Using these explicit expressions for the $\mathcal{P}^{a\bar{b}}_{kl}$ on the virtual level, all properties required of the anyon ansatz in our model can be explicitly verified. As a first check, the stacking behavior of the idempotents and nilpotents on the virtual level was verified by explicitly computing $\mathcal{P}^{a\bar{b}}_{kl}\mathcal{P}^{a'\bar{b}'}_{k'l'}$, giving the expected results:

$$\tag{A.9}$$

This identity can be seen as the virtual representation of Eq. (2.68).

Next it was explicitly verified that the excitation tensors Eq. (6.18) behave correctly under the virtual action of the tube algebra idempotent and nilpotent MPO operators

Eq. (A.6):

$$(A.10)$$

$$(A.11)$$

In these expressions the excitation tensors (6.18) were rotated by 90 degrees counterclockwise. For the specific case of DFIB excitations, (A.11) shows that the $\tau\bar\tau_1$ and $\tau\bar\tau_\tau$ excitations have *virtual* support in both the $\mathcal{P}_{11}^{\tau\bar\tau}$ and $\mathcal{P}_{\tau\tau}^{\tau\bar\tau}$ simple idempotents. It should be stressed that, even though we represent the diagrams in single line notation, the closed loop convention for the PEPS representation of string-net states must be used in the actual computations of all contractions.

We conclude this section with the topological properties (fusion rules, braiding properties and topological spin) of our anyon ansatz. To simplify the expressions we again denote a DFIB charge and FIB tail label with a single bold label, $\boldsymbol{a} = a_+\overline{a_-}_\ell$, or $\boldsymbol{a} = a_+\overline{a_-}$ depending on the context.

It was verified that fusion of two excitations tensors using the doubled fusion tensor (6.20) only has virtual support in the correct MPO central idempotent,

$$(A.12)$$

The braiding properties of the doubled anyons, can be translated to the following

relation on the virtual level:

$$= R_c^{ab} \qquad \text{(A.13)}$$

As for all other relations in this section, this was explicitly verified for the Fibonacci input category. This confirms that our ansatz does indeed possess the correct DFIB braiding behavior.

Finally, the topological spin of anyonic excitations, emerges on the virtual level as follows

$$= \theta_a \qquad , \qquad \text{(A.14)}$$

where $\theta_a = \theta_{a^+}(\theta_{a^-})^*$ according to Eq. (2.41). Alternatively, this can be expressed on the level of the central idempotents as

$$= \theta_a \qquad . \qquad \text{(A.15)}$$

Both relations where verified for the Fibonacci input category, which ensures that the PEPS ansatz was indeed derived correctly.

B | Scaling of largest connected group of anyons

Below, we present the results concerning the scaling of the average size of the largest group of connected anyons created by the application of depolarizing noise. These result represent a "worst case scenario" for the noise model (see Sec. 7.3) used in our threshold simulations, in the sense that they correspond to the improbable case where charge measurements of the plaquettes affected by noise operators always yield a nontrivial outcome.

The average size S of the largest connected group of anyons after noise application as a function of the linear system size L, was determined by simulating the (fixed-rate sampling) noise process for a total of 10^4 Monte Carlo samples with depolarizing noise. The results for a range of noise strengths p are shown in Fig. B.1.
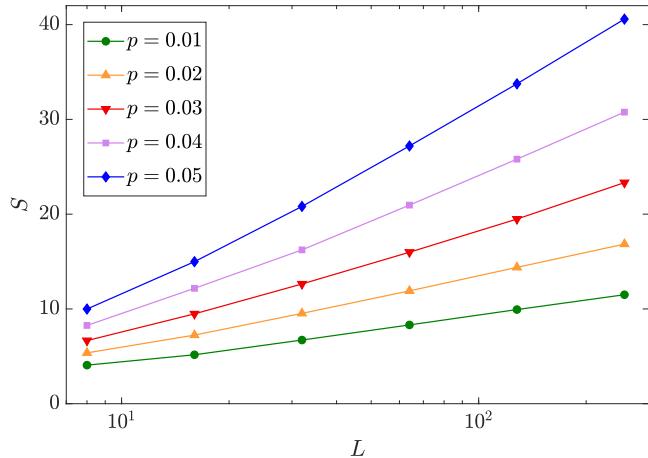
Figure B.1: The worst case scenario scaling of the average size S of the largest connected group of anyons with linear system size L, for the depolarizing noise model.

As expected, we find that S scales logarithmically with the system size L. This confirms that classical simulation of the system subjected to depolarizing noise, is indeed possible for noise strengths up to at least $p = 0.05$.

C | Relating fixed-rate sampling and i.i.d. noise

The results above were presented in terms of the average number of errors per qubit p, which is the parameter characterizing the noise strength in our fixed-rate sampling noise model. This is a natural measure in the context of this work, as the non-Abelian nature of our quantum memory and the fact that it undergoes constant syndrome measurements mean that noise processes cannot be formulated in terms of an independent and identically distributed noise model. However, in order to compare our results to known error threshold results for Abelian codes, we require a way of relating p to the i.i.d. noise strength $p_{\text{i.i.d.}}$ used in such models. To this end, we illustrate the relation between the Poisson and i.i.d. binomial noise models in Abelian stabilizer codes, for which both these noise models are equally valid, following the reasoning presented in [51].

We again consider our Poisson noise model in which a total of T error operations is executed, where T is drawn from a Poisson distribution with mean $T_0 = |E|p$ and $|E|$ is the total number of edges, but now apply it in the context of an Abelian stabilizer code. For each individual error process, an edge e of the lattice is chosen at random and a Pauli operator σ_i is applied according to relative probabilities $\{\gamma_x, \gamma_y, \gamma_z\}$. If we think of the global fixed-rate sampling noise model as arising from a superposition of fixed-rate sampling noise processes on each edge of the lattice individually, then we may assume that the latter can be approximately characterized as Poisson distributed events themselves. According to this reasoning, the noise model would then be captured by a superposition of three different Poisson distributions at the level of each individual edge, with means $\gamma_x p$, $\gamma_y p$ and $\gamma_z p$ for σ_x, σ_y and σ_z errors, respectively.

Since all Pauli errors acting on a given edge either commute or anticommute, we can compute the i.i.d probability for a net σ_x error on an individual edge, $p_{\text{i.i.d.}}^x$, by adding the probabilities of all Poisson error processes where the number of σ_x errors acting on the edge is odd and the number of σ_y and σ_z errors acting on the edge are both even, or where the number of σ_x errors acting on the edge is even and the number of σ_y and σ_z errors acting on the edge are both odd. We therefore get:

$$
\begin{aligned}
p_{\text{i.i.d.}}^x &= \left(\sum_{k=0}^{+\infty} \mathrm{e}^{-\gamma_x p} \frac{(\gamma_x p)^{(2k+1)}}{(2k+1)!} \right) \left(\sum_{k=0}^{+\infty} \mathrm{e}^{-\gamma_y p} \frac{(\gamma_y p)^{(2k)}}{(2k)!} \right) \left(\sum_{k=0}^{+\infty} \mathrm{e}^{-\gamma_z p} \frac{(\gamma_z p)^{(2k)}}{(2k)!} \right) \\
&\quad + \left(\sum_{k=0}^{+\infty} \mathrm{e}^{-\gamma_x p} \frac{(\gamma_x p)^{(2k)}}{(2k)!} \right) \left(\sum_{k=0}^{+\infty} \mathrm{e}^{-\gamma_y p} \frac{(\gamma_y p)^{(2k+1)}}{(2k+1)!} \right) \left(\sum_{k=0}^{+\infty} \mathrm{e}^{-\gamma_z p} \frac{(\gamma_z p)^{(2k+1)}}{(2k+1)!} \right) \\
&= \frac{\mathrm{e}^{-p}}{4} \left(\mathrm{e}^p + \mathrm{e}^{(\gamma_x - \gamma_y - \gamma_z)p} - \mathrm{e}^{(-\gamma_x + \gamma_y - \gamma_z)p} - \mathrm{e}^{(-\gamma_x - \gamma_y + \gamma_z)p} \right).
\end{aligned}
$$

(C.1)

Similar expressions can be derived in an analogous fashion for $p_{\text{i.i.d.}}^y$, $p_{\text{i.i.d.}}^z$ and $p_{\text{i.i.d.}}^{\text{none}}$, which denote the i.i.d. error probability for a net σ_y error, net σ_z error or no net error at all on

an individual edge, respectively:

$$p^y_{\text{i.i.d.}} = \frac{\mathrm{e}^{-p}}{4}\left(\mathrm{e}^p + \mathrm{e}^{(-\gamma_x+\gamma_y-\gamma_z)p} - \mathrm{e}^{(\gamma_x-\gamma_y-\gamma_z)p} - \mathrm{e}^{(-\gamma_x-\gamma_y+\gamma_z)p}\right) \tag{C.2}$$

$$p^z_{\text{i.i.d.}} = \frac{\mathrm{e}^{-p}}{4}\left(\mathrm{e}^p + \mathrm{e}^{(-\gamma_x-\gamma_y+\gamma_z)p} - \mathrm{e}^{(\gamma_x-\gamma_y-\gamma_z)p} - \mathrm{e}^{(-\gamma_x+\gamma_y-\gamma_z)p}\right) \tag{C.3}$$

$$p^{\text{none}}_{\text{i.i.d.}} = \frac{\mathrm{e}^{-p}}{4}\left(\mathrm{e}^p + \mathrm{e}^{(\gamma_x-\gamma_y-\gamma_z)p} + \mathrm{e}^{(-\gamma_x+\gamma_y-\gamma_z)p} + \mathrm{e}^{(-\gamma_x-\gamma_y+\gamma_z)p}\right) \ . \tag{C.4}$$

Hence, for $p \ll 1$ we have $p^x_{\text{i.i.d.}} \approx \gamma_x p$, $p^y_{\text{i.i.d.}} \approx \gamma_y p$, $p^z_{\text{i.i.d.}} \approx \gamma_z p$ and $p^{\text{none}}_{\text{i.i.d.}} \approx 1 - p$.

D | Paperclip configurations

The paperclip algorithm (see Sec. 7.6.4) works by exploiting the fact that refactoring moves along the boundary of an individual tile, can be classified into 16 different paperclip configurations, and that these entirely determine the corresponding sequence of swaps. These 16 configurations can be split into two groups corresponding to refactoring moves along or against the orientation of the curve diagram. Each of these groups can itself be split into two groups, depending on whether the initial piece of curve containing the anyon has an incoming or outgoing orientation. Within each of these groups, the different configurations can be distinguished using their respective turn numbers, as defined in Sec. 7.6.4

Below, we list all possible paperclip configurations, and the corresponding sequence of swaps for each of them. The sequences are expressed in terms of the *head* (H), *body* (B) and *tail*(T) portions of a cruve diagram. For a refactoring move indicated by the dotted arrow, these are defined as follows:
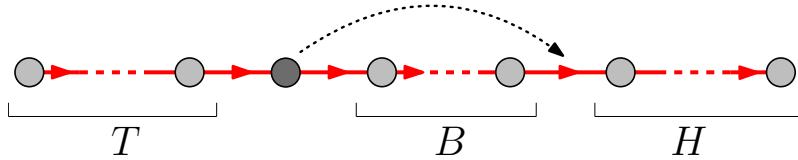
We use the following notation to indicate the sequences: $S[H]$ stands for sequentially swapping the anyon with all entires in H in a clockwise fashion, and $S^{-1}[H]$ stands for an analogous sequence with counterclockwise swaps. H^r indicates the sequence of anyons in H, in reversed order.

D.1 Refactoring along the curve

-3 $\rightarrow$: $S^{-1}[B]$

-6 $\rightarrow$: $S^{-1}[B]\,S^{-1}[H]\,S^{-1}[H^r]$

$+6$ $\rightarrow$: $S[T^r]\,S[T]\,S[B]$

$+9$ $\rightarrow$: $S[T^r]\,S[T]\,S[B]\,S[H]\,S[H^r]$

$+3$ $\rightarrow$: $S[B]$

$+6$ $\rightarrow$: $S[B]\,S[H]\,S[H^r]$

-6 $\rightarrow$: $S^{-1}[T^r]\,S^{-1}[T]\,S^{-1}[B]$

-9 $\rightarrow$: $S^{-1}[T^r]\,S^{-1}[T]\,S^{-1}[B]\,S^{-1}[H]$
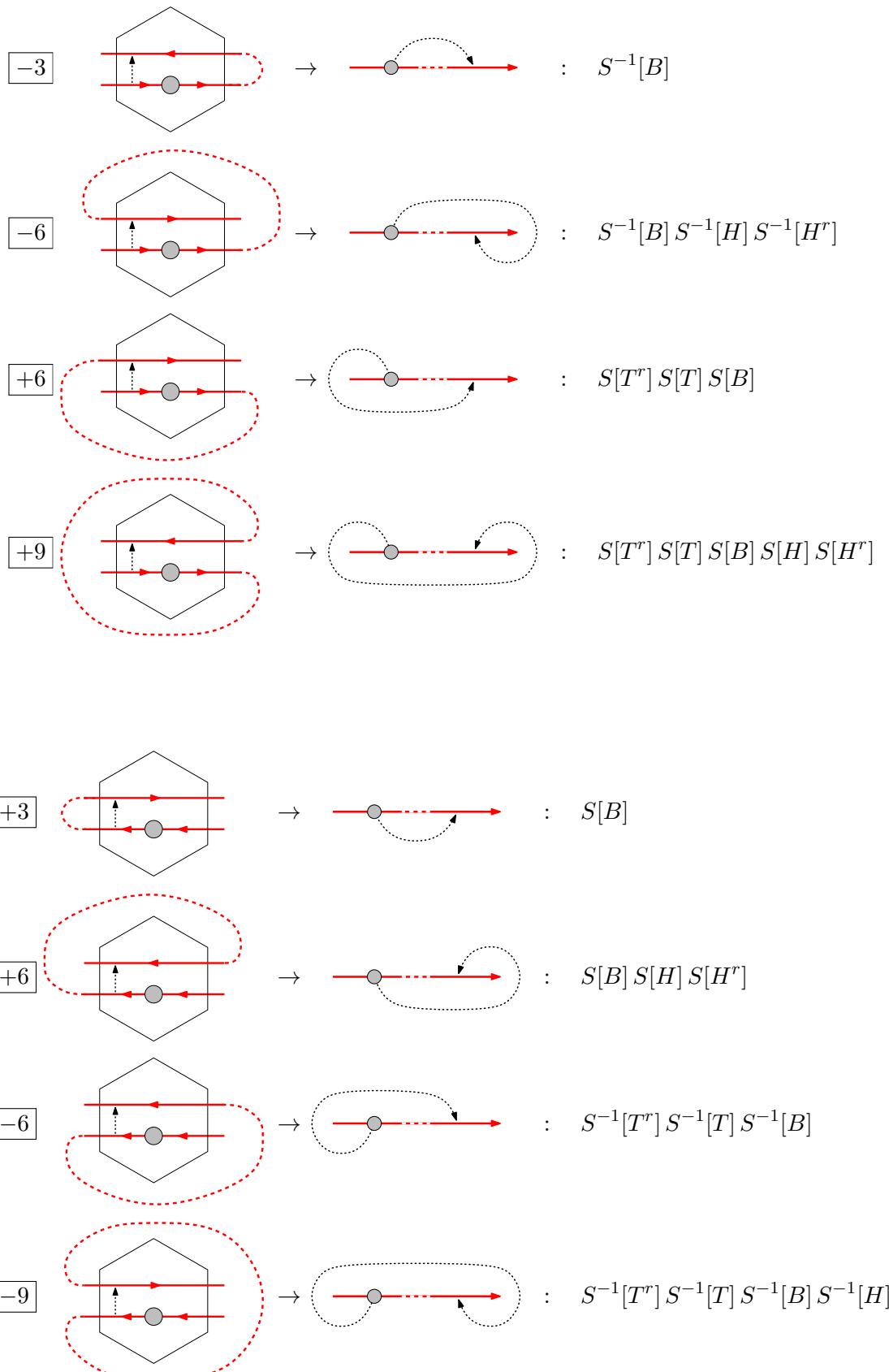
D.2 Refactoring against the curve

$+3$ $\rightarrow$ $:$ $S[B^r]$

$+6$ $\rightarrow$ $:$ $S^{-1}[H]\,S^{-1}[H^r]\,S^{-1}[B]$

-6 $\rightarrow$ $:$ $S[B]\,S[T^r]\,S[T]$

-9 $\rightarrow$ $:$ $S^{-1}[H]\,S^{-1}[H^r]\,S^{-1}[B^r]\,S^{-1}[T^r]\,S^{-1}[T]$

-3 $\rightarrow$ $:$ $S^{-1}[B^r]$

-6 $\rightarrow$ $:$ $S[H]\,S[H^r]\,S[B]$

$+6$ $\rightarrow$ $:$ $S^{-1}[B]\,S^{-1}[T^r]\,S^{-1}[T]$

$+9$ $\rightarrow$ $:$ $S[H]\,S[H^r]\,S[B^r]\,S[T^r]\,S[T]$
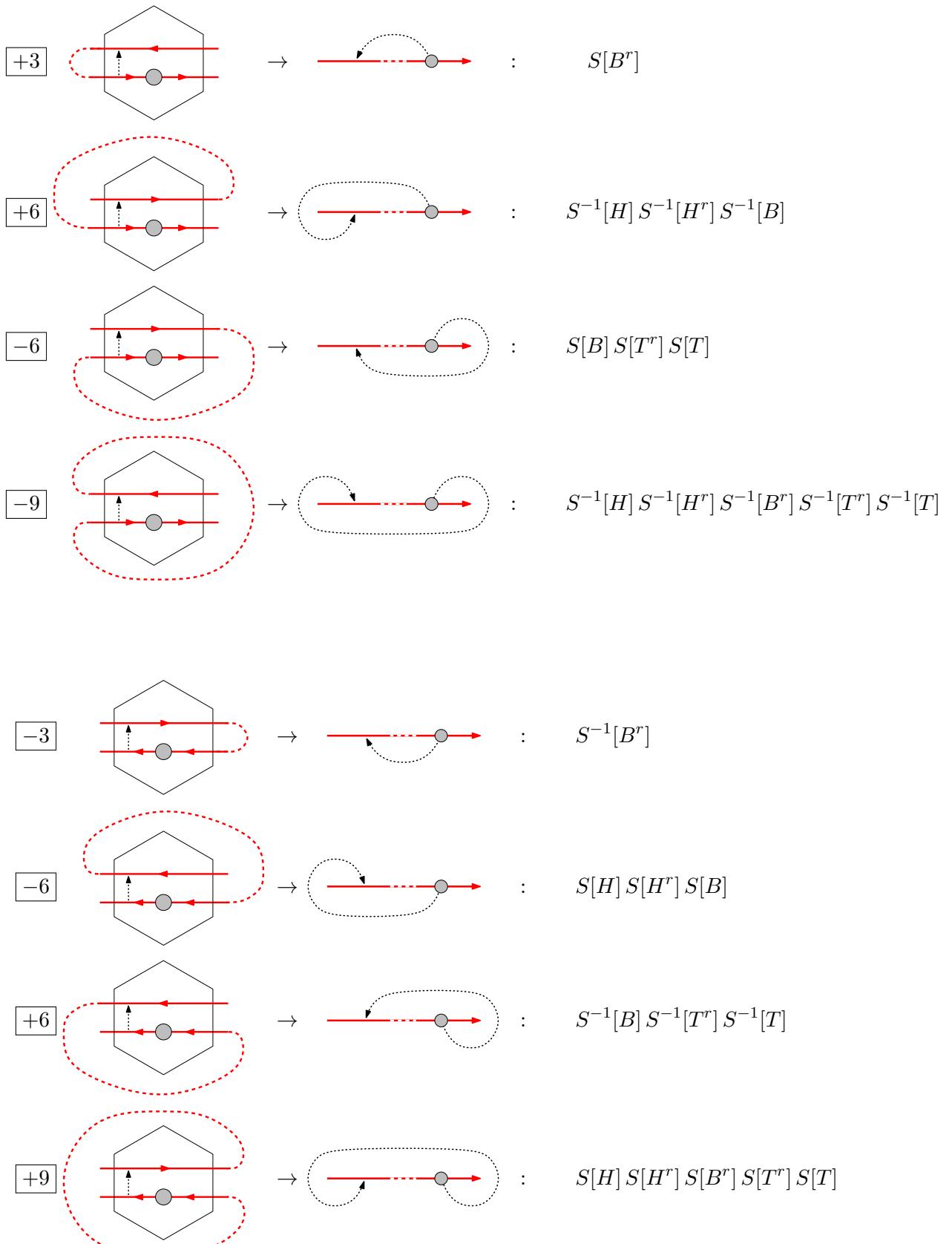
E | Phase transitions in string-net models

In this dissertation, we have considered string-net models as error correcting codes which can be realized by actively measuring the projectors in the Hamiltonian and correcting any excitations. However, the Levin-Wen model was originally conceived to study topological order, since they are conjectured to realize all non-chiral doubled topological phases. Hence, this model is particularly interesting for the study of topological phase transitions. Since such phase transitions are not descibed by Landau's theory of symmetry breaking, alternative methods had to be developed to study them. A promising appoach is to use a novel class of variational tensor network states.

Below, we include a paper in which non-Abelian topological phase transitions are studied for a Fibonacci string-net Hamiltonian to which a string-tension term is added. The phase transitions were found by using a variational PEPS ansatz, obtainded by modifying the tensor network representation of string-net ground states constructed in Sec. 6.1.

Tensor-network approach to phase transitions in string-net models

Alexis Schotte,[1,*] Jose Carrasco,[2,3] Bram Vanhecke,[1] Laurens Vanderstraeten,[1] Jutho Haegeman,[1]
Frank Verstraete,[1] and Julien Vidal [4,†]

[1]*Department of Physics and Astronomy, Ghent University, Krijgslaan 281, 9000 Gent, Belgium*
[2]*Departamento de Física Teórica, Universidad Complutense de Madrid, 28040 Madrid, Spain*
[3]*Institute for Theoretical Physics, University of Innsbruck, Technikerstrasse 21A, 6020 Innsbruck, Austria*
[4]*Sorbonne Université, CNRS, Laboratoire de Physique Théorique de la Matière Condensée, LPTMC, F-75005 Paris, France*

We use a recently proposed class of tensor-network states to study phase transitions in string-net models. These states encode the genuine features of the string-net condensate such as, e.g., a nontrivial perimeter law for Wilson loops expectation values, and a natural order parameter detecting the breakdown of the topological phase. In the presence of a string tension, a quantum phase transition occurs between the topological phase and a trivial phase. We benchmark our approach for $\mathbb{Z}_2$ string nets and capture the second-order phase transition which is well known from the exact mapping onto the transverse-field Ising model. More interestingly, for Fibonacci string nets, we obtain first-order transitions in contrast with previous studies but in qualitative agreement with mean-field results.

I. INTRODUCTION

Since its discovery in the late 1980s, topological order aroused much interest in physics. The long-range entanglement structure as well as the exotic quasiparticle excitations associated with this order may prove essential in attempts to achieve scalable fault-tolerant quantum computers or quantum memories [1]. As such, it is of paramount importance to understand how perturbations generate dynamics and interactions between the anyonic excitations and induce a breakdown of topological phases.

One of the most famous models hosting topological order was proposed by Levin and Wen in 2005 [2]. The string-net Hamiltonian allows to describe all doubled achiral topological phases. Thus, it has been the starting point for many studies concerning phase transitions [3–14]. Nevertheless, in the absence of local order parameter, the nature of these transitions remains an open question in many cases since one cannot use Landau's theory of symmetry breaking.

Among all alternative methods developed to study these topological phase transitions, a particularly versatile framework for constructing variational states is provided by tensor networks. In two dimensions, the projected entangled-pair states (PEPSs) [15] are known to describe the string-net ground states [16,17] and directly encode the topological properties in the virtual symmetries of the local PEPS tensor [18–20]. This feature has been exploited to detect possible topological phase transitions and to identify the associated anyon-condensation mechanism [13] in Abelian [21–24] and non-Abelian [25,26] cases at the level of wave functions.

However, variational PEPS calculations for concrete models have so far been restricted to $\mathbb{Z}_N$ toric codes [27–29] whose excitations are Abelian anyons. Recently, a family of PEPS based on perturbative expansions has been introduced to describe different ground states across a phase transition [30].

For a given Hamiltonian that exhibits a phase transition, the procedure to build these "perturbative PEPSs" can be summarized as follows: (i) we start from a wave function that describes the phase transition at the mean-field level; (ii) we apply tensor-network operators which implement the perturbative expansions in an extensive way to wave functions on both sides of the transition; (iii) we promote the *ad hoc* coefficients of these expansions to variational parameters. In two dimensions, these perturbatively exact variational states are still PEPSs and the tensor-network machinery [31,32] can be used to perform an efficient variational optimization. In Ref. [30], this method has been notably applied to the $\mathbb{Z}_2$ toric code perturbed with a string tension [33,34] for which the virtual symmetry of the local PEPS tensor emerges as an order parameter.

In this work, we go one step beyond and implement a variational PEPS to study phase transitions in both Abelian ($\mathbb{Z}_2$) and non-Abelian (Fibonacci) string-net models on the the honeycomb lattice. For the $\mathbb{Z}_2$ case, we capture the second-order quantum phase transition known from the mapping onto the transverse-field Ising model on the triangular lattice [6]. In the Fibonacci case, we only find first-order phase transitions, in contrast to Ref. [7] but in agreement with the mean-field results [35].

II. STRING-NET MODELS

We consider the two-dimensional string-net model introduced by Levin and Wen [2] in the presence of a tension

---

*alexis.schotte@ugent.be
†vidal@lptmc.jussieu.fr

term. For simplicity, we focus on the simplest case where the microscopic degrees of freedom, defined on the links of a honeycomb lattice, can only be in two different states, 0 and 1 (when possible, we omit the ket notation to describe states). The Hilbert space $\mathcal{H}$ is defined as the set of configurations obeying the branching rules that stem from the fusion rules of the theory considered [2].

In the present work, we discuss two different theories, $\mathbb{Z}_2$ and Fibonacci, whose fusion rules are given by

$$\mathbb{Z}_2: \quad 0 \times a = a \times 0 = a, 1 \times 1 = 0, \quad (1)$$

$$\text{Fibonacci}: \quad 0 \times a = a \times 0 = a, 1 \times 1 = 0 + 1, \quad (2)$$

for $a = 0, 1$. As underlined in Ref. [2], there are actually two different theories obeying $\mathbb{Z}_2$ fusion rules that give rise to either a doubled $\mathbb{Z}_2$ (D$\mathbb{Z}_2$) or a doubled semion (Dsem) topological phase. For the string tension considered thereafter, phase diagrams are the same for both theories.

At each vertex of the honeycomb lattice, the fusion rules must be satisfied [2], i.e., if two links are in states a and b the third link must be in a state $c \in a \times b$. Following Ref. [36], one can compute the dimension of the Hilbert space. For any trivalent graph with N_v vertices, one then gets

$$\mathbb{Z}_2: \quad \dim \mathcal{H} = 2^{N_v/2+1}, \quad (3)$$

$$\text{Fibonacci}: \quad \dim \mathcal{H} = (1 + \varphi^2)^{N_v/2} + (1 + \varphi^{-2})^{N_v/2}, \quad (4)$$

where $\varphi = \frac{1+\sqrt{5}}{2}$ is the golden ratio.

In order to analyze the breakdown of the topological phase originating from the string-net model, we consider the following Hamiltonian:

$$H = -J_p \sum_p B_p - J_l \sum_l L_l. \quad (5)$$

The first term corresponds to the usual string-net Hamiltonian introduced by Levin and Wen in Ref. [2]. Operators B_p's are mutually commuting projectors that "measure" the flux in the plaquette p. The action of B_p on a given link configuration depends on the theory under consideration through its F-symbols [see Eq. (C1) in Ref. [2] for details]. The operator B_p only modifies the six inner links of the plaquette p but its action depends (diagonally) on the six outer links [2]. For $J_p > 0$ and $J_l = 0$, all ground states are flux free and hence characterized by $B_p = 1$ for all p, up to a topology-dependent degeneracy.

The second term is also a sum of mutually commuting projectors. Operators L_l's are diagonal in the canonical (link) basis and act as $L_l|a\rangle_l = \delta_{a,0}|a\rangle_l$, where $|a\rangle_l$ denotes the state of the link l. For $J_l > 0$, this second term favors configurations with links in the state 0 and penalizes strings of links in the state 1, hence the name string tension.

For $J_l > 0$ and $J_p = 0$, the ground state is unique (trivial phase) and given by the product state $|0\rangle = \otimes_l |0\rangle_l$ for *both* $\mathbb{Z}_2$ and Fibonacci fusion rules. For $J_l < 0$ and $J_p = 0$, the ground-state manifold depends on the fusion rules. Indeed, for Fibonacci fusion rules, the product state $|1\rangle = \otimes_l |1\rangle_l$ is allowed and is the unique ground state. By contrast, for $\mathbb{Z}_2$ fusion rules, this state is not allowed (since $1 \times 1 = 0$) and the

ground space is spanned by all allowed states with N_v links in the state 1 and $\frac{1}{2} N_v$ links in the state 0.

III. METHODOLOGY

The goal of this work is to analyze phase transitions from the topological phase existing for $J_p > 0$ in the small $|J_l/J_p|$ limit to the trivial phases found in the large $|J_l/J_p|$ limit. To this aim, let us set $J_p = \cos \theta$ and $J_l = \sin \theta$ and consider first the region where $\theta \in [0, \pi/2]$. Following the variational tensor-network approach introduced in Ref. [30], we consider the state

$$|\alpha, \beta\rangle = \mathcal{N} \exp\left(\beta \sum_l L_l\right) \prod_p (\mathbb{1} + \alpha Z_p)|0\rangle, \quad (6)$$

where $Z_p = 2B_p - \mathbb{1}$, α and β are variational parameters, and $\mathcal{N}$ is a normalization factor. According to Ref. [30], a better description of the trivial phase would be obtained by adding an extra term $\exp(-\gamma \sum_p B_p)$. However, it considerably increases the complexity of the PEPS so that we do not consider it in the following.

For $\beta = 0$, the state $|\alpha, 0\rangle$ describes the phase transition at the mean-field level [35] and the states $|1, 0\rangle$ and $|0, 0\rangle$ are the exact ground states for $\theta = 0$ and $\theta = \pi/2$, respectively. Furthermore, the first-order contribution in α to $|\alpha, \beta\rangle$ around $(\alpha, \beta) = (0, 0)$ corresponds to the first-order perturbative correction to the exact ground state around $\theta = \pi/2$. Likewise, the first-order contribution in β to $|\alpha, \beta\rangle$ around $(\alpha, \beta) = (1, 0)$ corresponds to the first-order perturbative correction to the exact ground state around $\theta = 0$ [30].

The state $|\alpha, \beta\rangle$ can be interpreted as a PEPS, whose bond dimension depends on the theory considered. The variational energy per plaquette

$$e_0(\alpha, \beta) = \frac{1}{N_p} \frac{\langle \alpha, \beta | H | \alpha, \beta \rangle}{\langle \alpha, \beta | \alpha, \beta \rangle}, \quad (7)$$

can be efficiently computed using the VUMPS algorithm [31] for contracting two-dimensional tensor networks in the thermodynamic limit. Note that the previous approach reproduces the linear perturbative corrections up to second order both near $\theta = 0$ and $\theta = \pi/2$. As explained in Ref. [30], additional tensor-network operators can be added in order to reproduce higher-order perturbative corrections.

The PEPS framework allows for a natural characterization of the topological nature of the variational ground state. Indeed, the topological properties of a PEPS are related to the virtual symmetries of the local PEPS tensor [18]. In the Fibonacci theory, this virtual symmetry is described by a matrix product operator (MPO) [19,20]. Since, as shown in the Appendix, the state $|\alpha, \beta\rangle$ exhibits such a virtual MPO symmetry only when $\alpha = 1$, this parameter can be naturally interpreted as an order parameter [30] to detect the transition between the topological phase ($\alpha = 1$) and the trivial one ($\alpha < 1$). Indeed, at $\alpha = 1$, the expectation value of a Wilson loop operator changes from a trivial perimeter law for $\beta = 0$ to a nontrivial perimeter law for $\beta > 0$, still indicating deconfinement of the anyonic excitations, so that the state remains in the topological phase. For $\alpha < 1$, the Wilson loop expectation value satisfies a nontrivial area law, indicating that anyons are confined and the state is in the trivial phase.

Yet, even in the presence of the virtual symmetry ($\alpha = 1$), the parameter β can drive the state into a trivial phase by a spontaneous breaking of this symmetry, resulting in an area law for the Wilson loop. This process was shown to occur at $\beta = \frac{1}{2} \ln(1 + \sqrt{2})$ for the $\mathbb{Z}_2$ case [37]) and at $\beta \simeq 0.168$ for the Fibonacci case [25]. For the problem at hand, we checked that there is no spontaneous symmetry breaking so that α can be used as a bona-fide order parameter.

IV. RESULTS FOR THE $\mathbb{Z}_2$ THEORY

Let us first discuss the simplest theory and consider $\mathbb{Z}_2$ (Abelian) fusion rules. As discussed in Ref. [6], the Hamiltonian (5) for any theory obeying $\mathbb{Z}_N$ fusion rules can be exactly mapped onto the N-states Potts model in a magnetic field on the dual lattice. Thus, for $N = 2$, H is equivalent to the transverse-field Ising model on a triangular lattice where J_p and J_l are the strength of the magnetic field and of the spin-spin coupling, respectively. As a result, the sign of J_p is irrelevant for this problem and we assume $J_p > 0$ in the following.

In the antiferromagnetic case ($J_l < 0$), the Ising model on a triangular lattice is highly frustrated. So, clearly, the *Ansatz* $|\alpha, \beta\rangle$ is not adapted to that situation since the ground space has an extensive degeneracy for $J_p = 0$. In this region $\theta \in [-\pi/2, 0]$, a critical point in the universality class of the three-dimensional classical XY model was found for $\theta \simeq \arctan(-1/65) \simeq -0.545$ [38].

Here, we rather aim at benchmarking our *Ansatz* with the phase transition in the region $\theta \in [0, \pi/2]$ corresponding to ferromagnetic interactions ($J_l > 0$). The phase diagram in this region has been studied by high-order series expansion and a second-order transition occurs at $\theta_c \simeq \arctan(0.2097) \simeq 0.207$ [39], the critical point belonging to the universality class of the three-dimensional classical Ising model. Our results obtained from the variational *Ansatz* (6) are summarized in Fig. 1 (top panel). As already discussed in Ref. [35], for $\beta = 0$ (mean-field approximation), one obtains a continuous transition at $\theta = \arctan(1/6) \simeq 0.165$ which is qualitatively correct but about 20% off from θ_c. Remarkably, by adding β as a second variational parameter, the transition remains continuous (no jump of the order parameter α) and shifts to $\theta \simeq 0.198$ which is only 4% off from θ_c. Given that our variational *Ansatz* has only short-ranged correlations [except for $\alpha = 1$ and $\beta = \frac{1}{2} \ln(1 + \sqrt{2})$, which is not a variational optimum for any value of θ], and that the exact correlations decay algebraically near θ_c, these results can be considered as unexpectedly good. This can also be seen by comparing the variational ground-state energy with the numerical results obtained from exact diagonalization on a 25-plaquettes system with periodic boundary conditions, as shown in Fig. 1 (bottom panel).

Note, finally, that for the $\mathbb{Z}_2$ theory, our *Ansatz* satisfies $|\alpha, \beta\rangle \sim |\alpha^{-1}, \beta\rangle$, such that the expansion of the energy density $e_0(\alpha, \beta)$ around $\alpha = 1 + \delta\alpha$ can only contain even terms in $\delta\alpha$. As a consequence, the way α deviates from 1 around the transition point is as $|\delta\alpha| = (\theta - \theta_c)^{1/2}$, both for the mean-field *Ansatz* (with fixed $\beta = 0$) and for the *Ansatz* where β is also optimized.
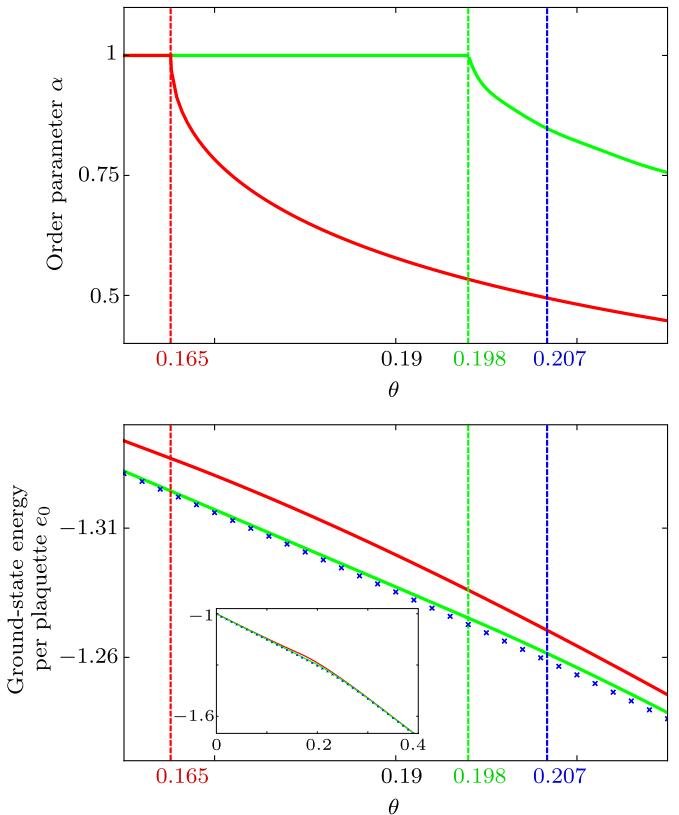

FIG. 1. Variational results for the $\mathbb{Z}_2$ theory obtained for $\beta = 0$ (red) and $\beta \neq 0$ (green). Top: order parameter α indicating a continuous transition from a topological phase ($\alpha = 1$) to a trivial phase ($\alpha < 1$). Bottom: ground-state energy per plaquette e_0 compared with exact diagonalization data (blue crosses) (see inset for a broader range). Dashed lines give the position of the transition point obtained from series expansions [39] (blue) and from the order parameter behavior [red [35] and green (this work)].

V. RESULTS FOR THE FIBONACCI THEORY

The phase diagram of the Hamiltonian (5) for the Fibonacci theory has already been computed by combining exact diagonalization results with high-order series expansions for the ground-state and gap energies [7]. The doubled Fibonacci (DFib) topological phase has been found to extend from $\theta_2 \simeq -0.63$ to $\theta_1 \simeq 0.255$ identifying θ_1 and θ_2 as critical points. Our variational results in this case are displayed in Figs. 2 and 3 for $\theta \in [0, \pi/2]$ and $\theta \in [0, -\pi/2]$, respectively.

In the region $\theta \in [0, \pi/2]$, the mean-field approach [35] corresponding to $\beta = 0$ indicates a first-order transition for $\theta = \arctan\left(\frac{1+\varphi}{6+3\varphi}\right) \simeq 0.237$. This result is qualitatively different from the one proposed in Ref. [7] although the position of the transition point is only 7% off from θ_1. Since (i) there is no prior reason to believe that the mean-field result is exact [35] and (ii) higher-order series expansions need to be extrapolated to provide reliable information about the nature of the transition, it is interesting to see what the *Ansatz* (6) can bring to the understanding of the transition. As can be seen in Fig. 2, by adding β as variational parameter, one still obtains a first-order transition characterized by a jump of the order parameter, but the transition point is shifted to $\theta \simeq 0.254$,
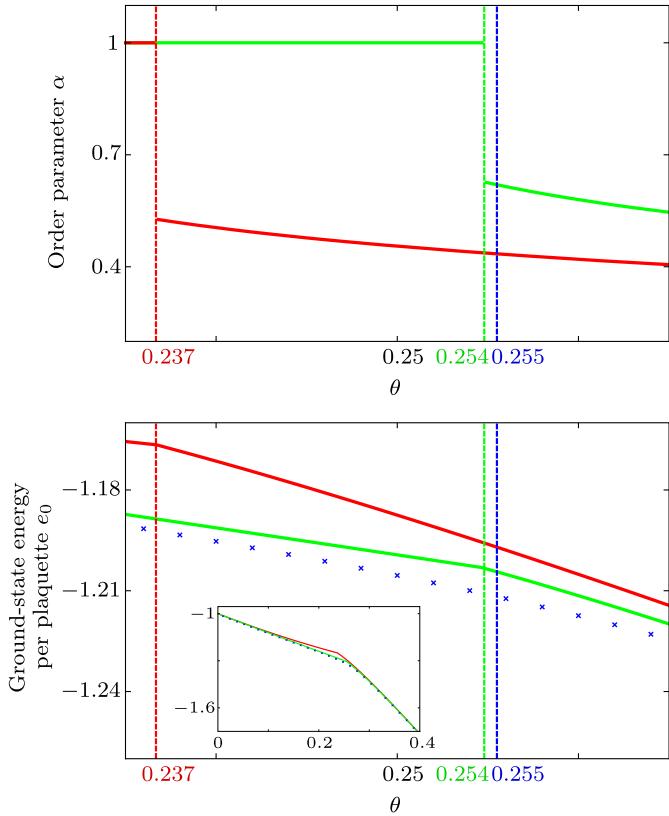
FIG. 2. Variational results for the Fibonacci theory (same conventions as in Fig. 1). Blue dashed lines indicate the position of the transition point computed from series expansions [7].

FIG. 3. Variational results for the Fibonacci theory (same conventions as in Fig. 1). Blue dashed lines indicate the position of the transition point computed from series expansions [7] which coincides with the one obtained with the *Ansatz* (8), i.e., $\theta \simeq -0.630$.

which is less than 1% off from θ_1. This leads us to conclude that in the region $\theta \in [0, \pi/2]$, there is a unique transition point located near $\theta \simeq 0.255$ (in agreement with Ref. [7]) corresponding to a first-order transition with a small gap at the transition point (weakly first order). Note that the proximity of the transition points obtained by the two approaches could suggest that the flux-flux correlation length is finite, which is a favorable case for a PEPS description of the ground state and justifies the relevance of our *Ansatz*.

In the region $\theta \in [-\pi/2, 0]$, we investigate the phase transition by slightly modifying the *Ansatz*. Indeed, the state defined in Eq. (6) is designed for interpolating between $|1, 0\rangle$ and $|0, 0\rangle$, that are the exact ground states at $\theta = 0$ and $\theta = \pi/2$, respectively. However, for $\theta = -\pi/2$, the ground state of the Hamiltonian (5) is unique (topologically trivial phase) and given by $|1\rangle = \otimes_l |1\rangle_l$. Consequently, to study the parameter range $\theta \in [-\pi/2, 0]$, we consider a variational *Ansatz*

$$|\alpha, \beta\rangle_- = \mathcal{N} \exp\left(\beta \sum_l L_l\right) \prod_p (\mathbb{1} + \alpha Z_p)|1\rangle, \qquad (8)$$

where, for simplicity, we kept the same notations as in Eq. (6). The PEPS tensor encoding this state is described in the Appendix. It is important to note that the "reference" states $|0\rangle$ and $|1\rangle$ are very different. Indeed, for $\beta = 0$, a key property of the *Ansatz* (6) is the factorization property that reads

$$\langle \alpha, 0| \prod_{p=1}^n B_p |\alpha, 0\rangle = \langle \alpha, 0| B_p |\alpha, 0\rangle^n, \qquad (9)$$

for any set of n plaquettes. This identity does not hold for the state (8) with $\beta = 0$, which can no longer be interpreted as a mean-field *Ansatz*. Yet, by construction, it is perturbatively exact near $\theta = 0$ and it also matches the exact ground state at $\theta = -\pi/2$. As such, this *Ansatz* is a good candidate to capture the transition between the DFib phase and the trivial phase. As can be seen in Fig. 3, for $\beta = 0$, one obtains a continuous transition at $\theta \simeq -0.317$ which is very far from the value $\theta_2 \simeq -0.63$ obtained in Ref. [7]. Interestingly, when including β, we obtain a discontinuous transition located at $\theta \simeq -0.630$ which is in agreement with the extrapolated values of Ref. [7]. Thus, we face a situation similar to the previous case, where the present variational study is in quantitative agreement with the series expansions studies. We emphasize that the series expansions in this region have to be resummed so that error bars on θ_2 are larger than for θ_1. Regarding the nature of the transition, the same arguments as before favor the first-order scenario.

VI. CONCLUSIONS AND OUTLOOK

This work presents the first variational results based on tensor networks for a topological phase transition out of a non-Abelian topological phase. We have studied the transitions between the topological and trivial phases of the Levin-Wen Hamiltonian with string tension, both for $\mathbb{Z}_2$ and Fibonacci fusion rules, by means of a simple two-parameters variational *Ansatz* inspired by Ref. [30]. For the $\mathbb{Z}_2$ case, we recover

the well-known second-order transition as predicted from the mapping onto the transverse-field Ising model the triangular lattice. For the Fibonacci case, our results are in quantitative agreement with series expansions and exact diagonalizations [7]. However, we only find first-order transitions (as in the mean-field treatment [35]) whereas series expansions combined with exact diagonalizations rather plead in favor of second-order transitions [7]. This qualitative discrepancy is likely due to the extrapolation of the series expansion and finite-size effects in the exact diagonalizations, but we cannot exclude that the present variational approach is not sufficient to properly describe the transitions in this model. Going beyond would require more sophisticated *Ansätze* that can be systematically constructed by using the ideas developed in Ref. [30]. Using recently developed contraction methods for three-dimensional tensor networks [40], we stress that such an approach can also be applied in three-dimensional systems, as recently illustrated in Ref. [41].

APPENDIX: CONSTRUCTION OF THE PEPS TENSOR

In the following section, we elaborate on the construction of the PEPS tensors representing the *Ansatz* $|\alpha, \beta\rangle_+ = |\alpha, \beta\rangle$ and $|\alpha, \beta\rangle_-$. It is sufficient to construct the PEPS for the one-parameter *Ansätze* $|\alpha, 0\rangle_\pm$. The PEPS tensor for $|\alpha, \beta\rangle_\pm \propto \prod_l \exp(\beta L_l)|\alpha, 0\rangle_\pm$ is then readily found by applying the right operator on the physical level.

For simplicity, we will assume that quantum dimensions d_0 and d_1 are non-negative real numbers, and that the F-symbols are all real. Both models studied in this work satisfy these assumptions (see Ref. [2] for details about $\mathbb{Z}_2$ and Fibonacci theories). In the general case, where one relaxes these assumptions, the construction of the PEPS can be done in a similar fashion.

In order to simplify the calculations in the following section, we will work with the states

$$|\gamma\rangle_\pm = |\gamma D^2/(-2\gamma + 2 + \gamma D^2), 0\rangle_\pm. \tag{A1}$$

To go back to the *Ansatz* used in the main body of the paper, one can simply use

$$|\alpha, 0\rangle_\pm = |2\alpha/(2\alpha - \alpha D^2 + D^2)\rangle_\pm, \tag{A2}$$

where $D = \sqrt{d_0^2 + d_1^2}$ is total quantum dimension.

1. *Ansatz* for $\theta \in [0, \pi/2]$

The one-parameter *Ansatz* is given by

$$|\gamma\rangle = \mathcal{N} \prod_p (\mathbb{1} + \gamma d_1 O_1^p)|0\rangle, \tag{A3}$$

where $|0\rangle = \otimes_l |0\rangle_l$, and $\mathcal{N}$ is a normalization factor. The operator O_i^p corresponds to inserting a i loop inside the plaquette [2], and then resolving it into the lattice using F moves

$$\tag{A4}$$

and the rule

$$\bigcirc_i = d_i. \tag{A5}$$

Contraction of a loop cannot happen across a plaquette; we treat plaquettes as if they have a puncture in their center. Applying these rules gives the full form of O_i^p:

$$\tag{A6}$$

Setting $\tilde{d}_0 = 1$ and $\tilde{d}_1 = \gamma d_1$ one can write

$$|\gamma\rangle = \mathcal{N} \prod_p \left(\sum_{\mu_p} \tilde{d}_{\mu_p} O_{\mu_p}^p\right)|0\rangle. \tag{A7}$$

Using the graphical representation of the operators O_i^p, $|\gamma\rangle$ can be represented as

$$\tag{A8}$$

The gray lines above are initially in the $|0\rangle$ state. To find the state from this graphical notation, one has to resolve the loops appearing in Eq. (A8) into the lattice. This is done in two steps. First we fuse the neighboring loops along each edge, using an F move:

$$\tag{A9}$$

The k appearing in the sum will be the physical degree of freedom in every edge, once we are done resolving everything into the lattice. The second step consists of using the following equality in every vertex:
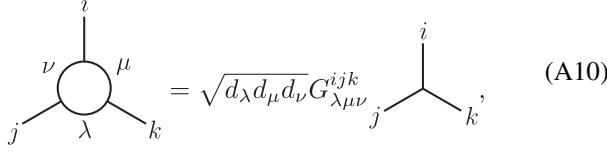
$$ \qquad (A10) $$

where

$$ G^{ijk}_{\lambda\mu\nu} = \frac{1}{\sqrt{d_i d_\lambda}} F^{\nu j\lambda}_{k\mu i} = \frac{1}{\sqrt{d_\nu d_k}} \left(F^{ijk}_{\lambda\mu\nu} \right)^*. \qquad (A11) $$

The PEPS tensor is obtained by splitting the factor $\sqrt{d_k/d_\mu d_\nu}$ in Eq. (A9) evenly between the two adjacent vertices, while splitting the factor $\tilde{d}_{\mu_p}$ in Eq. (A7) evenly between all six vertices appearing in plaquette p. The result is
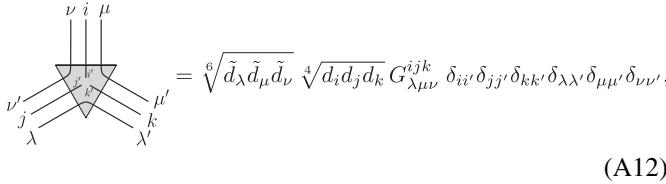
$$ \qquad (A12) $$

where i', j', and k' represent the physical degrees of freedom. Each virtual index is associated to two physical indices (appearing in the tensors at the vertices connected by that edge). The first three δ functions in Eq. (A12) guarantee that these two physical indices are always equal.

The PEPS tensor for the vertices with the inverse orientation is obtained by rotating Eq. (A12).

2. *Ansatz for $\theta \in [-\pi/2, 0]$*

For $\theta \in [-\pi/2, 0]$, the *Ansatz* is defined similarly:

$$ |\gamma\rangle_- = \mathcal{N} \prod_p \left(\mathbb{1} + \gamma d_1 O_1^p \right) |1\rangle, \qquad (A13) $$

$$ = \mathcal{N} \prod_p \left(\sum_i \tilde{d}_i O_i^p \right) |1\rangle. \qquad (A14) $$

Note that we are now acting on the $|1\rangle = \otimes_l |1\rangle_l$ product state, as opposed to the $|0\rangle$ state that we used for $\theta \in [0, \pi/2]$. We

can use the same graphical representation of this state:
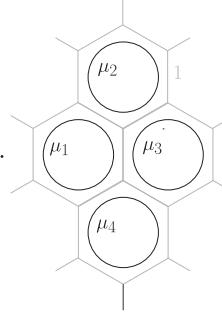
$$ \qquad (A15) $$

where the gray lines are now in the $|1\rangle$ state initially. As done for $\theta \in [0, \pi/2]$, we first fuse the loops along every edge:

$$ \qquad (A16) $$

The vertices then look like

To finish resolving everything into the lattice, these objects must be reduced to trivalent vertices. This is done by applying Eq. (A10) multiple times:

$$ \qquad (A17) $$

and analogously:
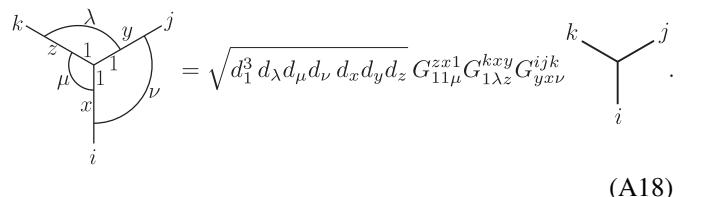
$$ \qquad (A18) $$

By splitting the factors appearing in Eq. (A16) equally between the two adjacent vertices, we obtain the following PEPS tensors:

$$= \sqrt[6]{\tilde{d}_\lambda \tilde{d}_\mu \tilde{d}_\nu} \sqrt[4]{d_i d_j d_k} \sqrt{d_x d_y d_z} \, d_1^{3/4} \, G_{11\nu}^{xy1} G_{\lambda 1y}^{xjz} G_{z\mu x}^{ijk} \, \delta_{ii'} \delta_{jj'} \delta_{kk'} \delta_{\lambda\lambda'} \delta_{\mu\mu'} \delta_{\nu\nu'}, \tag{A19}$$

$$= \sqrt[6]{\tilde{d}_\lambda \tilde{d}_\mu \tilde{d}_\nu} \sqrt[4]{d_i d_j d_k} \sqrt{d_x d_y d_z} \, d_1^{3/4} \, G_{11\mu}^{zx1} G_{1\lambda z}^{kxy} G_{yx\nu}^{ijk} \, \delta_{ii'} \delta_{jj'} \delta_{kk'} \delta_{\lambda\lambda'} \delta_{\mu\mu'} \delta_{\nu\nu'}. \tag{A20}$$

Note that there is one more virtual leg per side compared to Eq. (A12). This is due to the extra sum appearing in Eq. (A16).

Although, for $\gamma = 1$, the physical states given in Eqs. (A3) and (A13) are identical, the PEPS tensors representing these two states have very different properties. Indeed, the double-layer transfer matrix with PEPS tensors (A19) and (A20) appears to be critical with a central charge which is twice that of the three-state Potts model while the PEPS tensor given by Eq. (A12) does not share this property. This observation clearly deserves further investigation.

3. Reducing the bond dimensions

The PEPS tensor defined in Eq. (A12) has a bond dimension (both physical and virtual) of 2^3. However, the G-symbol present in its definition imposes certain rules which need to be met for the tensor to take a nonzero value. These rules can be exploited to rewrite this tensor as one with a lower bond dimension: for the $\mathbb{Z}_2$ theory (1), the bond dimension can be reduced to 4, while for the Fibonacci theory (2) it can be reduced to 5.

The structure of the PEPS tensor can also be exploited to reduce the bond dimension of the double-layer transfer matrix MPO tensors. Using the more efficient encoding of the tensor we just mentioned, the double-layer bond dimension already gets reduced from 64 to 16 for $\mathbb{Z}_2$ and to 25 for Fibonacci. The Kronecker delta functions appearing in the right-hand side of Eq. (A12) allow us to further reduce these to 8 and 13 respectively.

The same tricks can be applied to the tensors (A19) and (A20) (note that we only use this *Ansatz* for the Fibonacci theory). The virtual bond dimension can be reduced from 2^4 to 8, while the physical bond dimension can be reduced from 2^3 to 5. The bond dimension of double-layer MPO tensors can be reduced from 2^8 to 34.

As mentioned in the main body of this paper, the variational energy per plaquette (7) is calculated using the VUMPS algorithm. Due to memory constrains, the bond dimension of the boundary MPS has to be limited to 100 for the $\mathbb{Z}_2$ model, and for the Fibonacci model with $\theta \in [0, \pi/2]$. Due to the higher bond dimension of the double-layer MPO obtained from *Ansatz* (A19) and (A20), the bond dimension of the boundary MPS has to be limited to 60 for the Fibonacci model with $\theta \in [-\pi/2, 0]$.

[1] C. Nayak, S. H. Simon, A. Stern, M. Freedman, and S. DasSarma, Non-abelian anyons and topological quantum computation, Rev. Mod. Phys. **80**, 1083 (2008).

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

[3] C. Gils, S. Trebst, A. Kitaev, A. W. W. Ludwig, M. Troyer, and Z. Wang, Topology-driven quantum phase transitions in time-reversal-invariant anyonic quantum liquids, Nat. Phys. **5**, 834 (2009).

[4] C. Gils, Ashkin-Teller universality in a quantum double model of Ising anyons, J. Stat. Mech. (2009) P07019.

[5] E. Ardonne, J. Gukelberger, A. W. W. Ludwig, S. Trebst, and M. Troyer, Microscopic models of interacting Yang-Lee anyons, New J. Phys. **13**, 045006 (2011).

[6] F. J. Burnell, S. H. Simon, and J. K. Slingerland, Condensation of achiral simple currents in topological lattice models: Hamiltonian study of topological symmetry breaking, Phys. Rev. B **84**, 125434 (2011).

[7] M. D. Schulz, S. Dusuel, K. P. Schmidt, and J. Vidal, Topological Phase Transitions in the Golden String-Net Model, Phys. Rev. Lett. **110**, 147203 (2013).

[8] S. C. Morampudi, C. von Keyserlingk, and F. Pollmann, Numerical study of a transition between $\mathbb{Z}_2$ topologically-ordered phases, Phys. Rev. B **90**, 035117 (2014).

[9] M. D. Schulz, S. Dusuel, G. Misguich, K. P. Schmidt, and J. Vidal, Ising anyons with a string tension, Phys. Rev. B **89**, 201103(R) (2014).

[10] M. D. Schulz, S. Dusuel, and J. Vidal, Russian doll spectrum in a non-Abelian string-net ladder, Phys. Rev. B **91**, 155110 (2015).

[11] M. D. Schulz and F. J. Burnell, Frustrated topological symmetry breaking: geometrical frustration and anyon condensation, Phys. Rev. B **94**, 165110 (2016).

[12] M. D. Schulz, S. Dusuel, and J. Vidal, Bound states in string nets, Phys. Rev. B **94**, 205102 (2016).

[13] F. J. Burnell, Anyon condensation and its applications, Annu. Rev. Condens. Matter Phys. **9**, 307 (2018).

[14] J. Vidal, Ising versus $SU(2)_2$ string-net ladder, Phys. Rev. B **97**, 125152 (2018).

[15] F. Verstraete and J. I. Cirac, Renormalization algorithms for quantum-many body systems in two and higher dimensions, arXiv:cond-mat/0407066 .

[16] Z.-C. Gu, M. Levin, B. Swingle, and X.-G. Wen, Tensor-product representations for string-net condensed states, Phys. Rev. B **79**, 085118 (2009).

[17] X. Chen, Z.-C. Gu, and X.-G. Wen, Local unitary transformation, long-range quantum entanglement, wave function renormalization, and topological order, Phys. Rev. B **82**, 155138 (2010).

[18] N. Schuch, I. Cirac, and D. Pérez-García, PEPS as ground states: Degeneracy and topology, Ann. Phys. (NY) **325**, 2153 (2010).

[19] M. B. Şahinoğlu, D. J. Williamson, N. Bultinck, M. Mariën, J. Haegeman, N. Schuch, and F. Verstraete, Characterizing topological order with matrix product operators, arXiv:1409.2150.

[20] N. Bultinck, M. Mariën, D. J. Williamson, M. B. Şahinoğlu, J. Haegeman, and F. Verstraete, Anyons and matrix product operator algebras, Ann. Phys. (NY) **378**, 183 (2017).

[21] J. Haegeman, V. Zauner, N. Schuch, and F. Verstraete, Shadows of anyons and the entanglement structure of topological phases, Nat. Commun. **6**, 8284 (2015).

[22] J. Haegeman, K. VanAcoleyen, N. Schuch, J. I. Cirac, and F. Verstraete, Gauging Quantum States: From Global to Local Symmetries in Many-Body Systems, Phys. Rev. X **5**, 011024 (2015).

[23] S. K. Shukla, M. B. Şahinoğlu, F. Pollmann, and X. Chen, Boson condensation and instability in the tensor network representation of string-net states, Phys. Rev. B **98**, 125112 (2018).

[24] G-Y. Zhu and G.-M. Zhang, Gapless Coulomb State Emerging from a Self-Dual Topological Tensor-Network State, Phys. Rev. Lett. **122**, 176401 (2019).

[25] M. Mariën, J. Haegeman, P. Fendley, and F. Verstraete, Condensation-driven phase transitions in perturbed string nets, Phys. Rev. B **96**, 155127 (2017).

[26] W.-T. Xu and G.-M. Zhang, Tricritical point with fractional supersymmetry from a Fibonacci topological state, arXiv:1905.09960.

[27] Z.-C. Gu, M. Levin, and X.-G. Wen, Tensor-entanglement renormalization group approach as a unified method for symmetry breaking and topological phase transitions, Phys. Rev. B **78**, 205116 (2008).

[28] S. Dusuel, M. Kamfor, R. Orús, K. P. Schmidt, and J. Vidal, Robustness of a Perturbed Topological Phase, Phys. Rev. Lett. **106**, 107203 (2011).

[29] M. D. Schulz, S. Dusuel, R. Orús, J. Vidal, and K. P. Schmidt, Breakdown of a perturbed $\mathbb{Z}_N$ topological phase, New J. Phys. **14**, 025005 (2012).

[30] L. Vanderstraeten, M. Mariën, J. Haegeman, N. Schuch, J. Vidal, and F. Verstraete, Bridging Perturbative Expansions with Tensor Networks, Phys. Rev. Lett. **119**, 070401 (2017).

[31] M. T. Fishman, L. Vanderstraeten, V. Zauner-Stauber, J. Haegeman, and F. Verstraete, Faster methods for contracting infinite two-dimensional tensor networks, Phys. Rev. B **98**, 235148 (2018).

[32] L. Vanderstraeten, J. Haegeman, P. Corboz, and F. Verstraete, Gradient methods for variational optimization of projected entangled-pair states, Phys. Rev. B **94**, 155123 (2016).

[33] S. Trebst, P. Werner, M. Troyer, K. Shtengel, and C. Nayak, Breakdown of a Topological Phase: Quantum Phase Transition in a Loop Gas Model with Tension, Phys. Rev. Lett. **98**, 070602 (2007).

[34] A. Hamma and D. A. Lidar, Adiabatic Preparation of Topological Order, Phys. Rev. Lett. **100**, 030502 (2008).

[35] S. Dusuel and J. Vidal, Mean-field ansatz for topological phases with string tension, Phys. Rev. B **92**, 125150 (2015).

[36] S. H. Simon and P. Fendley, Exactly solvable lattice models with crossing symmetry, J. Phys. A **46**, 105002 (2013).

[37] C. Castelnovo and C. Chamon, Quantum topological phase transition at the microscopic level, Phys. Rev. B **77**, 054433 (2008).

[38] S. V. Isakov and R. Moessner, Interplay of quantum and thermal fluctuations in a frustrated magnet, Phys. Rev. B **68**, 104409 (2003).

[39] H.-X. He, C. J. Hamer, and J. Oitmaa, High-temperature series expansion for the (2+1)-dimensional Ising model, J. Phys. A **23**, 1775 (1990).

[40] L. Vanderstraeten, B. Vanhecke, and F. Verstraete, Residual entropies for three-dimensional frustrated spin systems with tensor networks, Phys. Rev. E **98**, 042145 (2018).

[41] D. A. Reiss and K. P. Schmidt, Quantum robustness and phase transitions of the 3D Toric Code in a field, Sci. Post. Phys. **6**, 078 (2019).

Bibliography

[1] Lev Davidovich Landau et al. *Statistical physics: theory of the condensed state*. Vol. 9. Butterworth-Heinemann, 1980.

[2] V Kalmeyer and RB Laughlin. "Equivalence of the resonating-valence-bond and fractional quantum Hall states". In: *Physical Review Letters* 59.18 (1987), p. 2095.

[3] Xiao-Gang Wen, Frank Wilczek, and Anthony Zee. "Chiral spin states and superconductivity". In: *Physical Review B* 39.16 (1989), p. 11413.

[4] Daniel C Tsui, Horst L Stormer, and Arthur C Gossard. "Two-dimensional magneto-transport in the extreme quantum limit". In: *Physical Review Letters* 48.22 (1982), p. 1559.

[5] Robert B Laughlin. "Anomalous quantum Hall effect: an incompressible quantum fluid with fractionally charged excitations". In: *Physical Review Letters* 50.18 (1983), p. 1395.

[6] Bei Zeng et al. *Quantum information meets quantum matter*. Springer.

[7] Xiao-Gang Wen. "Topological orders in rigid states". In: *International Journal of Modern Physics B* 4.02 (1990), pp. 239–271.

[8] Matthew B Hastings and Xiao-Gang Wen. "Quasiadiabatic continuation of quantum states: The stability of topological ground-state degeneracy and emergent gauge invariance". In: *Physical review b* 72.4 (2005), p. 045141.

[9] Sergey Bravyi, Matthew B Hastings, and Spyridon Michalakis. "Topological quantum order: stability under local perturbations". In: *Journal of mathematical physics* 51.9 (2010), p. 093512.

[10] Michael Victor Berry. "Quantal phase factors accompanying adiabatic changes". In: *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 392.1802 (1984), pp. 45–57.

[11] Barry Simon. "Holonomy, the quantum adiabatic theorem, and Berry's phase". In: *Physical Review Letters* 51.24 (1983), p. 2167.

[12] Frank Wilczek and Anthony Zee. "Appearance of gauge structure in simple dynamical systems". In: *Physical Review Letters* 52.24 (1984), p. 2111.

[13] Cumrun Vafa. "Toward classification of conformal theories". In: *Physics Letters B* 206.3 (1988), pp. 421–426.

[14] Frank Wilczek. "Quantum mechanics of fractional-spin particles". In: *Physical review letters* 49.14 (1982), p. 957.

[15] John Preskill. "Lecture notes for physics 229: Quantum information and computation". In: *California Institute of Technology* 16 (1998).

[16] Yu I Manin. *Vychislimoe i nevychislimoe (Computable and Noncomputable), Moscow: Sov.* 1980.

[17] Richard P Feynman. "Simulating physics with computers". In: *Int. J. Theor. Phys* 21.6/7 (1982).

[18] Michael A Nielsen and Isaac Chuang. *Quantum computation and quantum information.* 2002.

[19] Peter W Shor. "Scheme for reducing decoherence in quantum computer memory". In: *Physical review A* 52.4 (1995), R2493.

[20] Andrew M Steane. "Error correcting codes in quantum theory". In: *Physical Review Letters* 77.5 (1996), p. 793.

[21] A Robert Calderbank and Peter W Shor. "Good quantum error-correcting codes exist". In: *Physical Review A* 54.2 (1996), p. 1098.

[22] Peter W Shor. "Fault-tolerant quantum computation". In: *Proceedings of 37th Conference on Foundations of Computer Science.* IEEE. 1996, pp. 56–65.

[23] Emanuel Knill, Raymond Laflamme, and Wojciech H Zurek. "Resilient quantum computation: error models and thresholds". In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 454.1969 (1998), pp. 365–384.

[24] Dorit Aharonov and Michael Ben-Or. "Fault-tolerant quantum computation with constant error rate". In: *SIAM Journal on Computing* (2008).

[25] A Yu Kitaev. "Fault-tolerant quantum computation by anyons". In: *Annals of Physics* 303.1 (2003), pp. 2–30.

[26] Michael H Freedman, Michael Larsen, and Zhenghan Wang. "A modular functor which is universal for quantum computation". In: *Communications in Mathematical Physics* 227.3 (2002), pp. 605–622.

[27] Eric Dennis et al. "Topological quantum memory". In: *Journal of Mathematical Physics* 43.9 (2002), pp. 4452–55.

[28] Frank Verstraete, Valentin Murg, and J Ignacio Cirac. "Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems". In: *Advances in Physics* 57.2 (2008), pp. 143–224.

[29] Jacob C Bridgeman and Christopher T Chubb. "Hand-waving and interpretive dance: an introductory course on tensor networks". In: *Journal of Physics A: Mathematical and Theoretical* 50.22 (2017), p. 223001.

[30] Ignacio Cirac et al. *Matrix Product States and Projected Entangled Pair States: Concepts, Symmetries, and Theorems.* 2020. arXiv: `2011.12127` [`quant-ph`].

[31] Matthew B Hastings. "An area law for one-dimensional quantum systems". In: *Journal of Statistical Mechanics: Theory and Experiment* 2007.08 (2007), P08024.

[32] Mark Fannes, Bruno Nachtergaele, and Reinhard F Werner. "Finitely correlated states on quantum spin chains". In: *Communications in mathematical physics* 144.3 (1992), pp. 443–490.

[33] D. Pérez-García et al. "Matrix product state representations". In: *Quantum Inf. Comput.* 7 (2007), pp. 401–430.

[34] Frank Verstraete and J Ignacio Cirac. "Renormalization algorithms for quantum-many body systems in two and higher dimensions". In: *arXiv preprint cond-mat/0407066* (2004).

[35] Barbara M. Terhal. "Quantum Error Correction for Quantum Memories". In: *Reviews of Modern Physics* 87.2 (Apr. 2015), pp. 307–346. URL: http://arxiv.org/abs/1302.3428 (visited on 11/29/2020).

[36] Earl T. Campbell, Barbara M. Terhal, and Christophe Vuillot. "Roads towards fault-tolerant universal quantum computation". In: *Nature* 549.7671 (Sept. 2017), pp. 172–179. URL: http://arxiv.org/abs/1612.07330 (visited on 08/26/2020).

[37] S. B. Bravyi and A. Yu Kitaev. "Quantum codes on a lattice with boundary". In: *arXiv:quant-ph/9811052* (Nov. 1998). URL: http://arxiv.org/abs/quant-ph/9811052 (visited on 12/23/2019).

[38] Robert Raussendorf and Jim Harrington. "Fault-tolerant quantum computation with high threshold in two dimensions". In: *Physical review letters* 98.19 (2007), p. 190504.

[39] Austin G Fowler et al. "Surface codes: Towards practical large-scale quantum computation". In: *Physical Review A* 86.3 (Sept. 2012), p. 032324.

[40] Sergey Bravyi and Alexei Kitaev. "Universal quantum computation with ideal Clifford gates and noisy ancillas". In: *Phys. Rev. A* 71 (2 Feb. 2005), p. 022316.

[41] Sergey Bravyi and Robert König. "Classification of Topologically Protected Gates for Local Stabilizer Codes". In: *Phys. Rev. Lett.* 110.17 (Apr. 2013), pp. 170503–5.

[42] Hector Bombin. "Single-Shot Fault-Tolerant Quantum Error Correction". In: *Physical Review X* 5.3 (Sept. 2015), pp. 181–26.

[43] Michael Vasmer and Dan E. Browne. "Three-dimensional surface codes: Transversal gates and fault-tolerant architectures". In: *Physical Review A* 100.1 (July 2019), p. 012312. URL: http://arxiv.org/abs/1801.04255 (visited on 12/01/2020).

[44] Hector Bombin. "2D quantum computation with 3D topological codes". In: *arXiv preprint arXiv:1810.09571* (2018).

[45] Benjamin J Brown. "A fault-tolerant non-Clifford gate for the surface code in two dimensions". In: *Science advances* 6.21 (2020), eaay4929.

[46] Chetan Nayak et al. "Non-Abelian anyons and topological quantum computation". In: *Rev. Mod. Phys.* 80.3 (Sept. 2008), pp. 1083–1159.

[47] Daniel Eric Gottesman. "Stabilizer Codes and Quantum Error Correction". phd. California Institute of Technology, 1997. URL: https://resolver.caltech.edu/CaltechETD:etd-07162004-113028 (visited on 06/10/2020).

[48] Michael H. Freedman and Matthew B. Hastings. "Double Semions in Arbitrary Dimension". In: *Comm. Math. Phys.* 347.2 (2016), pp. 389–419.

[49] G Dauphinais et al. "Quantum error correction with the semion code". In: *New Journal of Physics* 21.5 (May 2019), p. 053035. URL: https://doi.org/10.1088%5C%2F1367-2630%5C%2Fab1ed8.

[50] Michael A Levin and Xiao-Gang Wen. "String-net condensation: A physical mechanism for topological phases". In: *Physical Review B* 71.4 (2005), p. 045110.

[51] Courtney G Brell et al. "Thermalization, error correction, and memory lifetime for ising anyon systems". In: *Physical Review X* 4.3 (2014), p. 031058.

[52] James R Wootton et al. "Error correction for non-Abelian topological quantum computation". In: *Physical review X* 4.1 (2014), p. 011051.

[53] James R Wootton and Adrian Hutter. "Active error correction for Abelian and non-Abelian anyons". In: *Physical Review A* 93.2 (2016), p. 022318.

[54]   Guillaume Dauphinais and David Poulin. "Fault-tolerant quantum error correction for non-abelian anyons". In: *Communications in Mathematical Physics* 355.2 (2017), pp. 519–560.

[55]   Simon Burton, Courtney G Brell, and Steven T Flammia. "Classical simulation of quantum error correction in a Fibonacci anyon code". In: *Physical Review A* 95.2 (2017), p. 022309.

[56]   Robert König, Greg Kuperberg, and Ben W Reichardt. "Quantum computation with Turaev–Viro codes". In: *Annals of Physics* 325.12 (2010), pp. 2707–2749.

[57]   NE Bonesteel and DP DiVincenzo. "Quantum circuits for measuring Levin-Wen operators". In: *Physical Review B* 86.16 (2012), p. 165113.

[58]   Sergey Bravyi, Martin Suchara, and Alexander Vargo. "Efficient algorithms for maximum likelihood decoding in the surface code". In: *Physical Review A* 90.3 (2014), p. 032326.

[59]   Theodore J Yoder and Isaac H Kim. "The surface code with a twist". In: *Quantum* 1 (2017), p. 2.

[60]   Edward Witten. "Quantum field theory and the Jones polynomial". In: *Communications in Mathematical Physics* 121.3 (1989), pp. 351–399.

[61]   M. Atiyah. In: *Publications Mathématiques de l'Institut des Hautes Scientifiques* 68 (1988), p. 175.

[62]   V.G. Turaev and O.Y. Viro. "State sum invariants of 3-manifolds and quantum 6j-symbols". In: *Topology* 31 (1992), pp. 865–902.

[63]   Michael Levin and Xiao-Gang Wen. "Detecting Topological Order in a Ground State Wave Function". In: *Physical review letters* 96.11 (Mar. 2006), p. 110405.

[64]   Weibo Feng. "Non-Abelian quantum error correction". PhD thesis. Florida State University, 2015.

[65]   Weibo Feng, N Bonesteel, and D DiVincenzo. In: *In preparation* (2020).

[66]   Yuting Hu, Nathan Geer, and Yong-Shi Wu. "Full dyon excitation spectrum in extended Levin-Wen models". In: *Physical Review B* 97.19 (2018), p. 195154.

[67]   Adrian Ocneanu et al. "Operator Algebras, Topology and Subgroups of Quantum Symmetry–Construction of Subgroups of Quantum Groups–". In: *Taniguchi Conference on Mathematics Nara'98*. Mathematical Society of Japan. 2001, pp. 235–263.

[68]   Nick Bultinck et al. "Anyons and matrix product operator algebras". In: *Annals of physics* 378 (2017), pp. 183–233.

[69]   Guanyu Zhu, Ali Lavasani, and Maissam Barkeshli. "Instantaneous braids and Dehn twists in topologically ordered states". In: *Phys. Rev. B* 102 (7 Aug. 2020), p. 075105. URL: https://link.aps.org/doi/10.1103/PhysRevB.102.075105.

[70]   Guanyu Zhu, Ali Lavasani, and Maissam Barkeshli. "Universal Logical Gates on Topologically Encoded Qubits via Constant-Depth Unitary Circuits". In: *Phys. Rev. Lett.* 125 (5 July 2020), p. 050502. URL: https://link.aps.org/doi/10.1103/PhysRevLett.125.050502.

[71]   Ali Lavasani, Guanyu Zhu, and Maissam Barkeshli. "Universal logical gates with constant overhead: instantaneous Dehn twists for hyperbolic quantum codes". In: *Quantum* 3 (Aug. 2019), p. 180.

[72] Zhenghan Wang. *Topological quantum computation.* 112. American Mathematical Soc., 2010.

[73] Pavel Etingof et al. *Tensor categories.* Vol. 205. American Mathematical Soc., 2016.

[74] Robert NC Pfeifer et al. "Translation invariance, topology, and protection of criticality in chains of interacting anyons". In: *Physical Review B* 86.15 (2012), p. 155111.

[75] Alexander Hahn and Ramona Wolf. "Generalized string-net model for unitary fusion categories without tetrahedral symmetry". In: *Physical Review B* 102.11 (2020), p. 115154.

[76] Sergey Bravyi and Jeongwan Haah. "Quantum Self-Correction in the 3D Cubic Code Model". In: *Phys. Rev. Lett.* 111.20 (Nov. 2013), pp. 200501–5.

[77] Zheng-Cheng Gu et al. "Tensor-product representations for string-net condensed states". In: *Physical Review B* 79.8 (2009), p. 085118.

[78] Oliver Buerschaper, Miguel Aguado, and Guifré Vidal. "Explicit tensor network representation for the ground states of string-net models". In: *Physical Review B* 79.8 (2009), p. 085119.

[79] Simon Burton. "A Short Guide to Anyons and Modular Functors". In: *arXiv preprint arXiv:1610.05384* (2016).

[80] Martin Z Bazant. "Largest cluster in subcritical percolation". In: *Physical Review E* 62.2 (2000), p. 1660.

[81] Matthew B Hastings, Grant H Watson, and Roger G Melko. "Self-correcting quantum memories beyond the percolation threshold". In: *Physical review letters* 112.7 (2014), p. 070501.

[82] Valentin Zauner-Stauber et al. "Variational optimization algorithms for uniform matrix product states". In: *Physical Review B* 97.4 (2018), p. 045145.

[83] Chenyang Wang, Jim Harrington, and John Preskill. "Confinement-Higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory". In: *Annals of Physics* 303.1 (2003), pp. 31–58.

[84] Austin G Fowler. "Accurate simulations of planar topological codes cannot use cyclic boundaries". In: *Physical Review A* 87.6 (2013), p. 062320.

[85] Héctor Bombin et al. "Strong resilience of topological codes to depolarization". In: *Physical Review X* 2.2 (2012), p. 021004.

[86] Alexei Kitaev and Liang Kong. "Models for gapped boundaries and domain walls". In: *Communications in Mathematical Physics* 313.2 (July 2012), pp. 351–373. URL: http://arxiv.org/abs/1104.5047 (visited on 02/29/2020).

[87] Laurens Lootens et al. "Matrix product operator symmetries and intertwiners in string-nets with domain walls". In: *arXiv:2008.11187 [hep-th, physics:math-ph, physics:quant-ph]* (Oct. 2020). URL: http://arxiv.org/abs/2008.11187 (visited on 11/29/2020).

[88] Victor Ostrik. "Module categories, weak Hopf algebras and modular invariants". In: *Transformation groups* 8.2 (2003), pp. 177–206.

[89] Thomas Booker and Alexei Davydov. "Commutative algebras in Fibonacci categories". In: *Journal of Algebra* 355.1 (2012), pp. 176–204.