Master's thesis

NTNU
Norwegian University of Science and Technology

Bjarte Matre

Fingerprinting of quantum ground state manifolds

Applying group- and representation theory to derive properties of ground states

Master's thesis in Physics
Supervisor: John Ove Fjærestad
May 2024

NTNU
Norwegian University of Science and Technology

Bjarte Matre

Fingerprinting of quantum ground state manifolds

Applying group- and representation theory to derive properties of ground states

Master's thesis in Physics
Supervisor: John Ove Fjærestad
May 2024

Norwegian University of Science and Technology

NTNU
Norwegian University of
Science and Technology

ABSTRACT

Determining the properties of ground states can be highly nontrivial. In this thesis, we investigate the ground states of the antiferromagnetic spin-$\frac{1}{2}$ chain (J-J') model numerically, and the Toric Code (TC) model and Doubled Semion (DS) model analytically. We employ group and representation theory to create a "fingerprint" of the behavior of these different systems. For the J-J' model, representation theory is used to visualize the behavior of the ground states, and group theoretical methods are applied to solve the system more efficiently. In the case of the TC and DS models, the method provides a group theoretical explanation for the degeneracies of the ground states, showing that they both transform like a four-dimensional irreducible representation of a set of symmetry operators isomorphic to "The inner holomorph of D_8" and "The unitriangular matrix group: $UT(3, \mathbb{Z}_4)$," respectively. Additionally, we analyze the 1D versions of these models, explaining the twofold degeneracy of their ground states in a similar group theoretical manner.

Utledning av egenskapene til grunntilstander kan være svært vanskelig. I denne oppgaven undersøker vi grunntilstandene til den antiferromagnetiske spin-$\frac{1}{2}$ modellen (J-J') numerisk, og Toric Code (TC) modellen og Doubled Semion (DS) modellen analytisk. Vi benytter gruppe- og representasjonsteori for å skape et "fingeravtrykk" av oppførselen til disse forskjellige systemene. For J-J' modellen brukes representasjonsteori til å visualisere grunntilstandene, og gruppeteoriske metoder anvendes for å løse systemet mer effektivt. For TC- og DS-modellene gir metoden en gruppeteoretisk forklaring på degenerasjonene av grunntilstandene, som viser at de begge transformeres som en firedimensjonal irredusibel representasjon av et sett med symmetrioperatorer isomorfe med "The inner holomorph of D_8" og "The unitriangular matrix group: $UT(3, \mathbb{Z}_4)$," henholdsvis. I tillegg analyserer vi 1D-versjoner av disse modellene, og forklarer den tofoldige degenerasjonen av grunntilstandene på en lignende gruppeteoretisk måte.

PREFACE

I would like to extend my deepest gratitude to my supervisor, John Ove Fjærestad, for his invaluable guidance and support both as a lecturer and as a supervisor for my thesis. His expertise and encouragement have been instrumental in the completion of this work.

I also wish to express my sincere thanks to all my professors and lecturers at NTNU for their dedication and the knowledge they have imparted throughout my studies. Their teaching has been crucial in shaping my understanding and passion for this field.

I am profoundly grateful to my family for their unwavering support and encouragement during my academic journey. Special thanks to my girlfriend, who has been my rock and the person I have relied on the most for support during the writing of this thesis. Her patience, understanding, and love have made this accomplishment possible.

Additionally, I would like to thank Huan Qiang Zhou at Chongqing University, Center for Modern Physics, for inviting me to Chongqing University in the weeks leading up to the finalization of my thesis. The experience was fantastic, and the opportunity to discuss my thesis with the researchers there undoubtedly improved the quality of my work.

Thank you all for your contributions to this important milestone in my life.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

- **Irrep** Irreducible representation
- **DS model** Doubled semion model
- **TC model** Toric code model
- **DSL model** Doubled semion ladder model
- **TCL model** Toric code ladder model
- **GSM** Ground state manifold
- **PGL** Projective linear group

CHAPTER

ONE

INTRODUCTION

Group theoretical methods are powerful tools for better understanding quantum systems. They can assist in obtaining exact solutions and, in computational physics, can drastically speed up calculations. In this project, we will investigate, in particular, a way of identifying certain quantum phases at low temperatures. This fingerprinting method can identify what type of order a system has based on how the lowest energy eigenstates, namely the ground state manifold, transform under the symmetry group of the Hamiltonian. The thesis will focus on both numerical and analytical applications of this method, and the sections in the thesis are divided accordingly.

In recent years, the field-theoretical study of generalized symmetries has emerged. Although we will not employ this formalism in detail in this paper, it bears mentioning as two of the systems we investigate exhibit generalized symmetries. Specifically, the string operators in the Toric Code and Doubled Semion models are examples of 1-form symmetries. These differ from traditional 0-form symmetries, such as total spin, translation, and magnetization, which act on point-like objects or are proportional to system size. In contrast, 1-form symmetries act on line-like objects, providing a richer structure of symmetry that can influence the topological order of a system [1].

1.1 Motivation

One of the strongest motivations for applying this method to numerical calculations is that it will let us interpret the complicated data one gets from numerical analysis and visualize the properties of the system. For example, the following arrays show the numerical data for the two ground states of the Majumdar-Gosh model. The entries in the arrays correspond to the coefficients needed to express the state as a linear combination of a certain set of states in the z-basis.

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.25,
-0.25, -0.25, 0.25, 0, 0, -0.25, 0.25, 0.25, -0.25, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, -0.25, 0.25, 0.25, -0.25, 0, 0, 0.25, -0.25, -0.25,
0.25, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0  ]

[ 0, 0, 0, 0, 0, 0, -0.25198, 0.25198, 0, 0.25198, -0.25198, 0, 0,
```

```
0, 0, 0, 0.25198, -0.25198, 0, -0.25198, 0.22048, 0.0315, 0.0315,
-0.0315, 0, 0, 0.0315, -0.0315, -0.0315, 0.0315, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0.0315, -0.0315, -0.0315, 0.0315, 0, 0, -0.0315, 0.0315,
0.0315, 0.22048, -0.25198, 0, -0.25198, 0.25198, 0, 0, 0, 0, 0,
-0.25198, 0.25198, 0, 0.25198, -0.25198, 0, 0, 0, 0, 0, 0]
```

At first glance, it is immensely difficult to deduce that the ground states of the Majumdar-Gosh model break translation symmetry by only looking at the data. With the fingerprinting method, however, we may use the numerical data for the ground states to determine that and how they break translation symmetry. For this particular model, there exist analytic solutions to the ground states which can be visualized nicely. This is not generally possible; however, with the fingerprinting method, one can associate the exact ground state with simpler pictures that will have the same fingerprint.

This fingerprinting method works by constructing a representation of the symmetry group of the system Hamiltonian using the ground states, or the states that will become degenerate ground states in the thermodynamic limit. This representation may be decomposed into so-called irreducible representations (irreps) of the symmetry group, where the types of irreps that appear tell us about the properties of the system. The fingerprinting method may also be used to explain the degeneracy of a ground state. When representation theory is applied to quantum mechanics, we say that a ground state n-fold degeneracy may be explained if we find an irreducible representation of the symmetry group of the system which acts on the set of ground states like that irreducible representation. So when the fingerprinting method tells us only a single irrep appears in the decomposition, we may conclude this irrep explains the degeneracy of the ground states in a group theoretical manner. We will apply this to the Toric Code model, the Doubled Semion model, and 1D versions of both of them in order to explain their ground state degeneracy from a group theoretical standpoint. Due to all these systems having ground states which may be derived analytically, we are able to derive the fingerprint analytically as well.

1.2 Project Description

The project is essentially split into two parts. The first part is the application of the fingerprinting method on numerically derived ground states, specifically from the 1D second nearest neighbour Heisenberg model (also called the J-J' model). This section also involves the numerical derivation of the space group and its irreducible representations, although some of this work is done analytically as well. The second part is a purely analytical symmetry analysis of four Toric-code-like models, as they have known exact eigenstates and symmetry operators. We will construct the representation for the group of symmetry operators in the ground state vector space, and it will produce an irreducible representation of the group which explains the degeneracy.

The methods chapter is split into two sections, one describing the method used for the numerical simulations and one for the analytic approach. The results are also presented in separate sections. As a visual aid, Fig. 1.2.1 shows how the different sections are connected, and the chapters are visualized with colors.

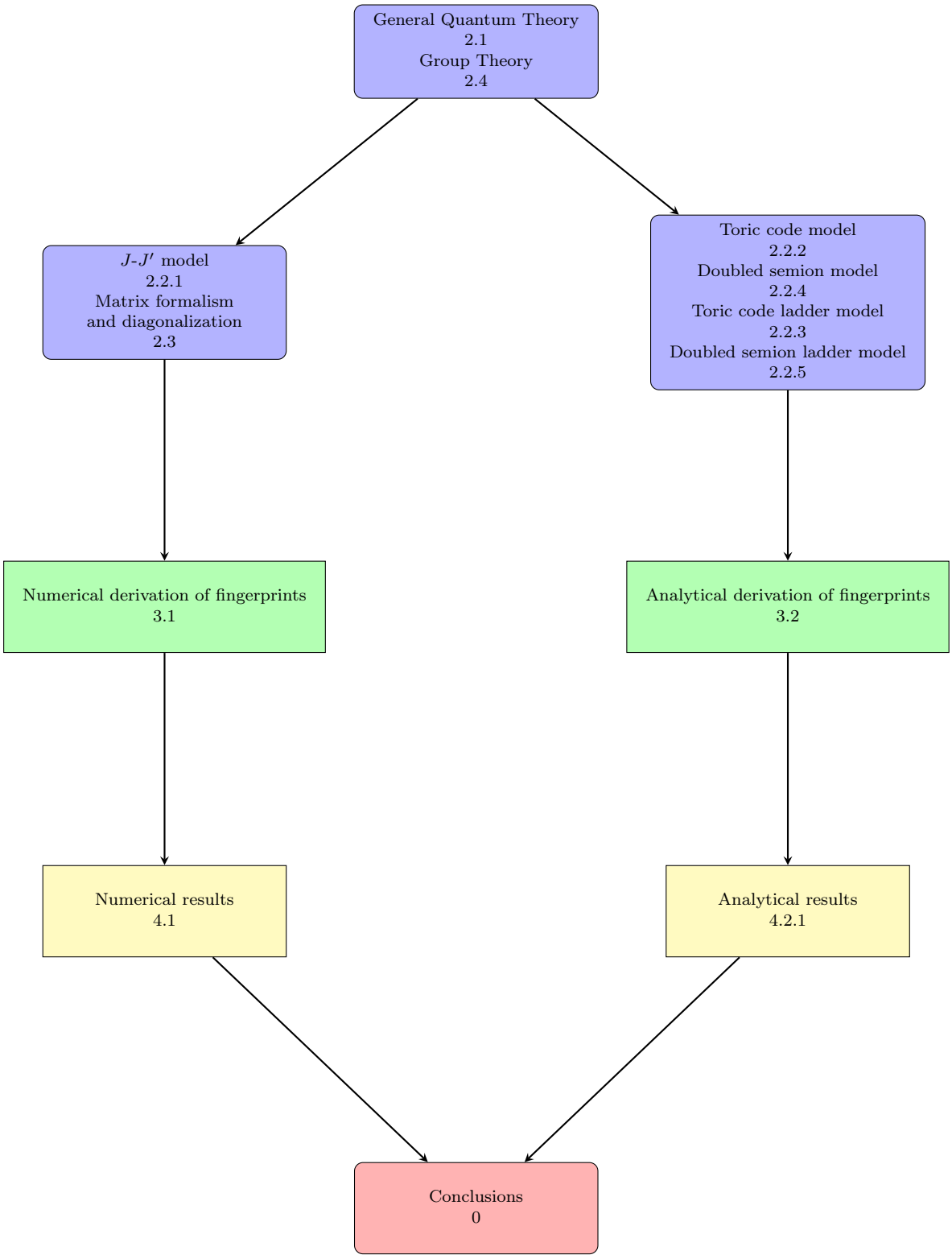


Figure 1.2.1: Flowchart showing the structure of the thesis. The Theory chapter is shown in blue, Methods in green, Results in yellow and conclusions in red

CHAPTER

TWO

THEORY

2.1 General Quantum Theory

In this section, we will go through some of the general quantum mechanics needed for the thesis.

2.1.1 States and Hilbert Space

In the general formulation of quantum mechanics, we describe the state of a system $|\psi\rangle$ as a vector in a complex linear vector space $\mathcal{H}$ called the Hilbert space. The dimension of the Hilbert space depends on the system we are considering. For instance, a spin-$\frac{1}{2}$ system with 1 spin will have a $2D$ Hilbert space.

In order to describe any state $|\phi\rangle$ in $\mathcal{H}$, we need a complete basis $\{|\psi_i\rangle\}$. It is very useful if, though it is not a requirement, if this basis is orthogonal. That is to say, that the inner product between any two basis elements is

$$\langle\psi_i\rangle\,\psi_j = \delta_{ij}, \tag{2.1}$$

where $\langle\cdot\rangle\,\cdot$ is the inner product. As a side note, we require all states in $\mathcal{H}$ to have a norm of 1. A basis being complete implies that the operator

$$\sum_k |\psi_k\rangle\,\langle\psi_k| = 1, \tag{2.2}$$

satisfies this relation, which is called the completeness relation. Using it, we may now write any state $|\phi\rangle$ in terms of the basis $\{|\psi_i\rangle\}$.

$$\begin{aligned} |\phi\rangle &= \sum_k |\psi_k\rangle\,\langle\psi_k|\,\phi\rangle \\ &= \sum_k c_k\,|\psi_k\rangle\,, \end{aligned} \tag{2.3}$$

where $c_k = \langle\psi_k|\,\phi\rangle$ are the coefficients in the linear combination. Note that for continuous systems, the sums would be replaced by integrals. In this thesis, however, we are only dealing with discrete systems, so these equations will hold as given [2].

2.1.2 Spin - $\frac{1}{2}$ Systems

Spin-$\frac{1}{2}$ systems are central to the study of quantum mechanics due to their elementary role in the quantum description of angular momentum and magnetic properties. Although there can be several Hamiltonians for a spin-$\frac{1}{2}$ system, the simplest basis is often one that exhibits rotational invariance or is subject to a magnetic field in a specific direction. In both these cases, the z-basis (eigenstates of σ^z) will be eigenstates of the system. In this thesis, natural units are used, where $\hbar = 1$.

The Hamiltonian for a single spin-$\frac{1}{2}$ particle in a magnetic field, which arises from the interaction of the magnetic moment with the field, can be expressed as:

$$\hat{H} = -\gamma \vec{B} \cdot \vec{S}$$

Here, $\vec{B}$ denotes the external magnetic field, and $\vec{S}$ represents the spin operators associated with the Pauli matrices:

$$S_x = \frac{1}{2}\sigma_x, \quad S_y = \frac{1}{2}\sigma_y, \quad S_z = \frac{1}{2}\sigma_z$$

Expanding to systems comprising multiple spin-$\frac{1}{2}$ particles, the Hilbert space is described as the tensor product of individual particle spaces, $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B$. The basis for the combined space consists of tensor products of the individual spin states, leading to a composite basis:

$$\{|+\rangle_A \otimes |+\rangle_B, |+\rangle_A \otimes |-\rangle_B, |-\rangle_A \otimes |+\rangle_B, |-\rangle_A \otimes |-\rangle_B\}$$

From this expression, we see that the dimension of the composite system is the product of the dimensions of $\mathcal{H}_A$ and $\mathcal{H}_B$. This is a general property when combining two Hilbert spaces. Therefore, each additional spin-$\frac{1}{2}$ added to the system doubles the dimension of the Hilbert space, implying that the size of the Hilbert space is exponential in the number of spins [3].

2.2 Quantum Models

2.2.1 J-J' Model

The J-J' model is a Heisenberg model with a second nearest neighbour term. We will investigate this model on a spin-$\frac{1}{2}$ chain. One of the earliest and seminal works discussing the spontaneous dimerization in such systems is by Haldane [**Haldane1982**]. This model has applications in quantum information processing, specifically in transporting a single q-bit down a spin chain using a quantum adiabatic process, which could potentially be used for sending data in a quantum computer made of artificial spins [4]. The main source for this section will be [3], which elaborates on the model in great detail. The Hamiltonian of the model may be written as

$$\hat{H} = J \sum_i \vec{S}_i \cdot \vec{S}_{i+1} + J' \sum_j \vec{S}_j \cdot \vec{S}_{j+2}, \tag{2.4}$$

where $\vec{S}_i = \frac{1}{2}(\sigma_i^x, \sigma_i^y, \sigma_i^z)$, and σ_i^α is a Pauli spin matrix acting on the i-th spin. The interactions in the model and the lattice are visualized in Fig. 2.2.1.

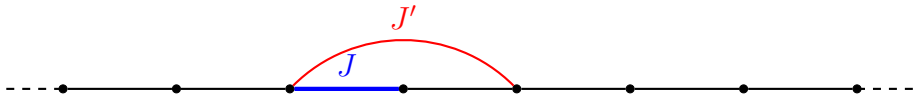


Figure 2.2.1: J-J' model interactions

We may rewrite this Hamiltonian as operators acting on the z-basis by replacing the σ^x and σ^y operators with the ladder operators

$$S_i^{\pm} = S_i^x \pm iS_i^y \tag{2.5}$$

Expanding equation 2.4 we get

$$\begin{aligned}\hat{H} = J\sum_i S_i^z S_{i+1}^z + J\sum_i (S_i^x S_{i+1}^y + S_i^y S_{i+1}^y) \\ + J'\sum_j S_j^z S_{j+2}^z + J'\sum_j (S_j^x S_{j+2}^x + S_j^y S_{j+2}^y)\end{aligned} \tag{2.6}$$

Inserting $S_i^x = \frac{1}{2}(S_i^+ + S_i^-)$, $S_i^y = \frac{1}{2i}(S_i^+ - S_i^-)$ and collecting the terms, we get

$$\begin{aligned}\hat{H} = J\sum_i S_i^z S_{i+1}^z + \frac{J}{2}\sum_i (S_i^+ S_{i+1}^- + S_i^- S_{i+1}^+) \\ + J'\sum_j S_j^z S_{j+2}^z + \frac{J'}{2}\sum_j (S_j^+ S_{j+2}^- + S_j^- S_{j+2}^+)\end{aligned} \tag{2.7}$$

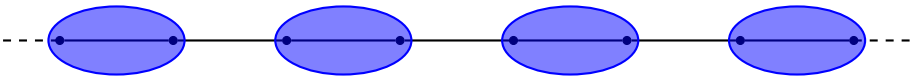
This expression for $\hat{H}$ might look more complicated than Eq. 2.4; however, the effect of this operator on a state in the z-basis is much simpler to measure. The z-basis elements are eigenstates of the terms in the first and third sums, so the effect of that operator is simple to calculate. The terms in the second sum have the effect of switching the i-th and $(i+1)$-th spin in a z-basis state if the spins are different, and give no contribution if the spins are the same.

The model has several symmetries. For one it has a space group generated by the translation by 1 and inversion $x \to -x$. Secondly the total magnetization operator $\hat{S}_{tot}^Z = \sum_i S_i^z$ and total spin operator $\hat{S}_{tot}^2 = \sum_i S_i^2$ commute with the Hamiltonian and yield good quantum numbers.

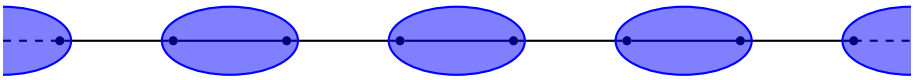
The J-J' model exhibits different properties depending on the ratio J'/J. For $J'/J < J_c$, the model is known to be in a quantum critical phase, while for $J'/J > J_c$, it transitions into a dimer phase (also called a Valence Bond Solid or VBS). The model has been investigated extensively using various methods. For instance, [5] used Lanczos and exact diagonalization techniques to get the numerical estimate for the phase transition, determining $J_c = 0.241167 \pm 0.000005$, while [6] employed field theory approaches. In this thesis, we will investigate the dimer phase in particular.

The point $J'/J = 1/2$ in phase space is called the Majumdar-Ghosh point. Majumdar and Ghosh first discovered this model [7], and here the ground states

may be derived analytically. The result is a pair of dimerized eigenstates which may be visualized nicely as [3]:



(a) $|\phi_1\rangle$



(b) $|\phi_2\rangle$

Figure 2.2.2: Visual representation of the states $|\phi_1\rangle$ and $|\phi_2\rangle$. The blue ovals represent a singlet between the two spins.

2.2.2 Toric Code Model

The Toric Code (TC) model is interesting in many aspects, particularly as a proposed error-correcting memory for quantum computation. It faces many challenges and remains an area of active research. There exist a number of candidates that could, in principle, be used; however, none so far are sufficiently good [8]. The TC model was originally proposed by Alexei Kitaev as an analytically solvable model demonstrating topological order and robustness to local perturbations [9]. We might be led to think that the analytical solutions implies the system is very simple, however it turns out the TC Hamiltonian possesses several interesting properties. In particular, we will investigate its topological properties. The main source for this section will be [3].

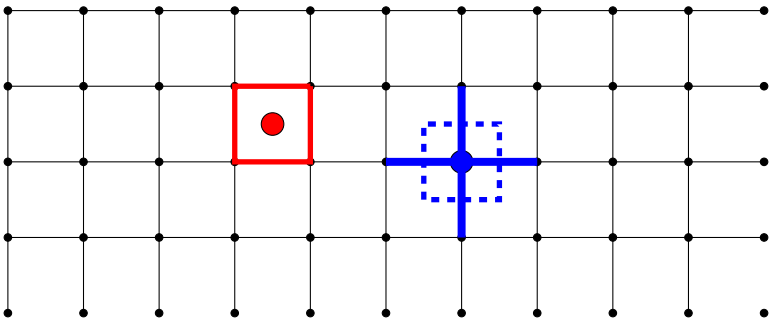


Figure 2.2.3: TC-system, A_S and B_p operators shown as a blue cross and a red square respectively. A_s may alternatively be visualized as a loop in the dual-lattice, shown in the figure with a dashed line around its center. The action of B_p may be thought of as creating or destroying a loop on the lattice, whilst A_s creates or destroys a loop on the dual lattice.

In this system the individual spins have spin $\frac{1}{2}$. Unlike most lattice models, they are not located on the lattice and are instead located in the middle of the edges. The Hamiltonian of this system written as

$$\mathcal{H}_{tc} = -\sum_s A_s - \sum_p B_p, \tag{2.8}$$

where the operators A_s are called star operators, while the operators B_p are called plaquette operators. The spins they affect are shown in Fig. 2.2.3, and they are defined as

$$A_s = \prod_{i \in s} \sigma_i^z, \qquad B_p = \prod_{i \in p} \sigma_i^x \tag{2.9}$$

The operators σ_i^x and σ_j^z have the property that they commute if $i \neq j$ and they anticommute if $i = j$. However, since a given A_s and B_p always share either zero or two spins, they will always commute. Thus, the simultaneous eigenstates of the set of all A_s and B_p are also eigenstates of the Hamiltonian. Note that the eigenvalues of A_s and B_p are $+1$ and -1 as they square to the identity operator. Furthermore, we may conclude that the ground states will satisfy the relation:

$$\begin{gathered} A_s \left|\psi\right\rangle = \left|\psi\right\rangle, \; B_p \left|\psi\right\rangle = \left|\psi\right\rangle \\ \forall A_s, \; \forall B_p, \; \forall \text{ground state } \left|\psi\right\rangle, \end{gathered} \tag{2.10}$$

as this minimizes the ground state energy. To derive the ground states, we can first start with the z-basis. Since the A_s operators are only comprised of σ^z operators, the z-basis will also be an eigenbasis for all the A_s operators. These states may be visualized by drawing the system as a grid, and representing vertices with spin up with a thin line and vertices with spin down as a thick line, a few examples of which are shown in Fig. 2.2.4. It turns out that if a state only has closed loops of spin down, it will always be a $A_s = 1 \; \forall s$ eigenstate. This is true as any closed loop will either not pass through a given lattice site, pass through once, or pass through it twice, which implies the loop will share either zero, two, or four spins with a given A_s operator. For example, if the loop in a loop state $\left|\text{loop}\right\rangle$ passes through a site s once, then

$$A_s \left|\text{loop}\right\rangle = -1 \cdot -1 \cdot 1 \cdot 1 \left|\text{loop}\right\rangle. \tag{2.11}$$

Thus, $\left|\text{loop}\right\rangle$ is a $+1$ eigenstate of this A_s operator. The same will be true for loops passing through a point twice. If we start with a given $\left|\text{loop}\right\rangle$, then we can project this state onto the $B_p = 1$ subspace by using a projection operator, yielding a ground state of the system $\left|\psi_0\right\rangle$. This is shown in the following equation

$$\begin{aligned} \left|\psi_0\right\rangle &= \prod_p P_p \left|\text{loop}\right\rangle \\ P_p &= \frac{1}{2}(B_p + 1) \end{aligned} \tag{2.12}$$

Any choice of a valid loop state will yield a ground state of the system. However, an interesting question is how they can be distinguished and how many there are. It turns out that the B_p operators can be used to create or destroy loops and to transform one loop into another. This is shown in Fig. 2.2.4, where one loop operator around the x-axis is deformed into another loop.

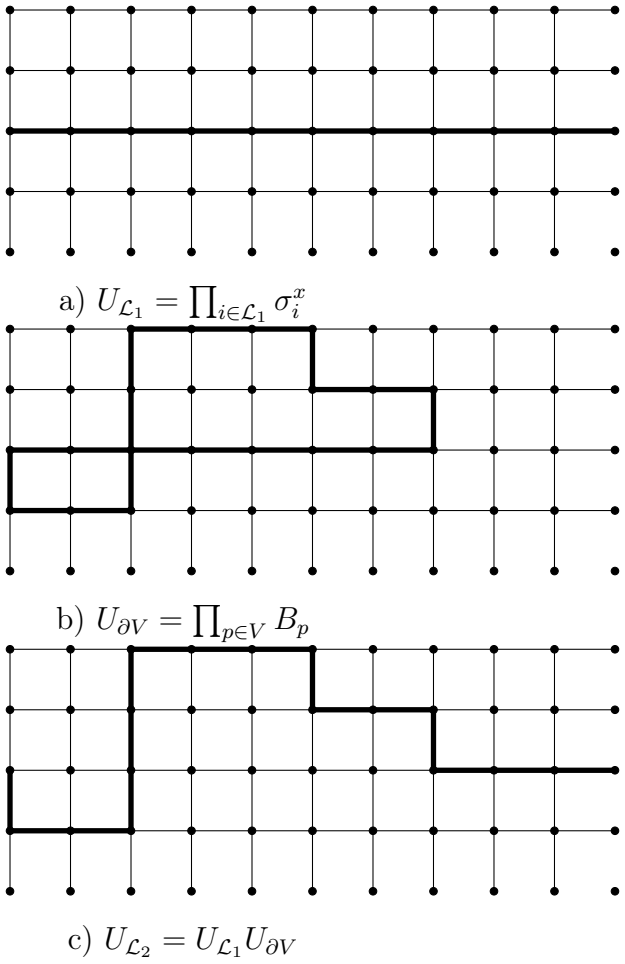


Figure 2.2.4: A product of B_p operators change the shape of $U_{\mathcal{L}_1}$ into $U_{\mathcal{L}_2}$

Certain loops can be entirely removed, by which we mean turned into the loop-less state, by use of B_p operators. However, interestingly, there exist loops that cannot be removed by use of B_p operators. These we will call non-contractible loops. For example, we can wind a loop around the x-axis once as shown in Fig. 2.2.4. This loop cannot be removed by use of B_p operators. If, however, we add another one of those loops, thereby winding it twice around the x-axis, the loop becomes contractible. The same is true for the y-axis. We therefore have two parity numbers which are 1 if there are no non-contractible loops in a given direction (which implies an even number of windings) or -1 if there is a non-contractible loop (which implies an odd number of windings). This yields four topologically distinct loops. The important part is that since these parity numbers are conserved by B_p, two loop states $|\text{loop}_1\rangle$ and $|\text{loop}_2\rangle$ with different parity numbers will be projected onto different ground states. This implies that we get four distinct ground states, which we can label by their parity numbers $\{|1,1\rangle, |-1,1\rangle, |1,-1\rangle, |-1,-1\rangle\}$. Furthermore, there cannot be any more ground states. This is true because only the loop states are legal starting states (as these are the $A_s = 1$ states), and any two loop states with the same loop-parity numbers will be projected onto the same ground state.

The non-contractible loops may be encapsulated as symmetry operators. These operators are given in [10]. We will use a slightly different notation which will be

easier to use for group theory analysis.

$$U_{\mathcal{L}} = \prod_{i \in \mathcal{L}} \sigma_i^x \tag{2.13}$$

For any $\mathcal{L}$, the corresponding $U_{\mathcal{L}}$ operator will be a symmetry operator of the TC Hamiltonian. This is true as $U_{\mathcal{L}}$ will commute with any B_p due to them only being comprised of σ_i^x operators, and it will commute with every A_s operator as $\forall s$, A_s and $U_{\mathcal{L}}$ will share an even number of spins. Only four choices of $\mathcal{L}$ will turn out to be relevant. I will later prove this, but for now, we define the four distinct choices as $\mathcal{L}$ being no loop, denoted by E, it being a single loop around the x-axis, denoted by U_x, it being a single loop around the y-axis, denoted by U_y, and it being a loop around both the x- and y-axes, denoted by U_xU_y. This yields the following set of symmetry operators, forming a group, which we will name the U-group: $\{E, U_x, U_y, U_xU_y\}$. The U-operators as chosen are shown in Fig. 2.2.5 as the bold lines.

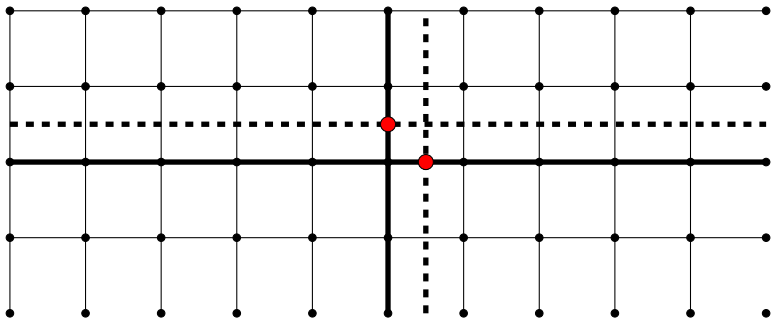


Figure 2.2.5: Visualization of the chosen U_x, U_y, V_x, V_y operators. U operators shown as bold, V operators shown as dashed. The shared spins are marked by a dot.

In addition to the U-group, we can also define a set of loops on the dual lattice, which are composed of σ_i^z operators instead. These are also given in [10], however, we will use the notation $V_{\mathcal{L}}$ instead, as shown in the following equation:

$$V_{\mathcal{L}} = \prod_{i \in \mathcal{L}} \sigma_i^z \tag{2.14}$$

These operators are symmetry operators for a similar reason that $U_{\mathcal{L}}$ are symmetry operators, that being they commute trivially with A_s and share an even number of spins with B_p. We can also form a group with these operators by choosing the loops "x" and "y" to be dual-lattice loops around the x and y axes, respectively. The chosen loops are also shown in Fig. 2.2.5 as dashed lines. We then get the group $\{E, V_x, V_y, V_xV_y\}$ which we will call the V-group. Both the U-group and V-group should be part of the symmetry group of the Hamiltonian; $Sym(\hat{H})$ (which we will formally define in 2.4), however, we cannot just trivially combine the two groups due to U_x and V_y sharing only a single spin. We can see this in Fig. 2.2.5, where the shared spin between the U and V operators is highlighted. This will result in anticommutation between certain group elements.

2.2.3 Toric Code Ladder Model

Suppose we removed the periodicity of the TC model in the y-axis and only kept the x-axis periodicity, transforming the model into a sort of ladder of either finite or infinite length. We will call this system the TCL model.

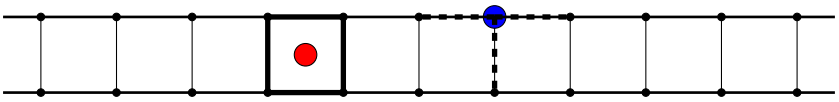


Figure 2.2.6: TCL-model, A_s operator shown as blue dot and dashed lines, B_p operator shown as red dot and full lines

The system would then look like Fig. 2.2.6, and it would affect the definition of the A_s operators. In this case, there would only be three spins per star (as seen in the figure), while the B_p operators would remain unchanged. The expression for the Hamiltonian will also remain unchanged. How the reduction of the periodicity affects the properties of the system will be investigated in later sections. Furthermore, we will investigate which U- and V-operators remain as symmetry operators.

2.2.4 Doubled Semion Model

The doubled semion model (DS model) is similar to the TC-model. It is defined on the honeycomb lattice and shares many similarities with the TC-model; however, the definition of B_p includes an additional phase factor (shown in equation 2.16), and the sign in front of the B_p term in the Hamiltonian is positive instead of negative. The equations are gathered from [10], although the notation has been adjusted to match the notation for the TC-model in [3].

The DS model was initially investigated by Levin and Wen, who introduced the concept in their work on string-net condensation as a physical mechanism for topological phases [11]. Their foundational study laid the groundwork for understanding the topological properties and phase transitions within such models.

Additionally, the experimental realization of topologically ordered states has been achieved in recent years. For instance, the ground state of the toric code Hamiltonian was prepared using an efficient quantum circuit on a superconducting quantum processor, demonstrating key aspects of topological quantum matter and quantum error correction [12].

$$\mathcal{H}_{ds} = -\sum_{s} A_s + \sum_{p} B_p \tag{2.15}$$

$$\begin{aligned} A_s &= \prod_{i \in s} \sigma_i^z \\ B_p &= (\prod_{i \in p} \sigma_i^x)(\prod_{j \in v(p)} i^{(1-\sigma_j^z)/2}) \end{aligned} \tag{2.16}$$

The stars s, the plaquettes p and the edges coming out of the plaquettes p, denoted as $v(p)$ are shown in figure 2.2.7. Note that the periodicity of the model is defined

such that the left and right boundary are connected, as are the top and bottom boundaries.

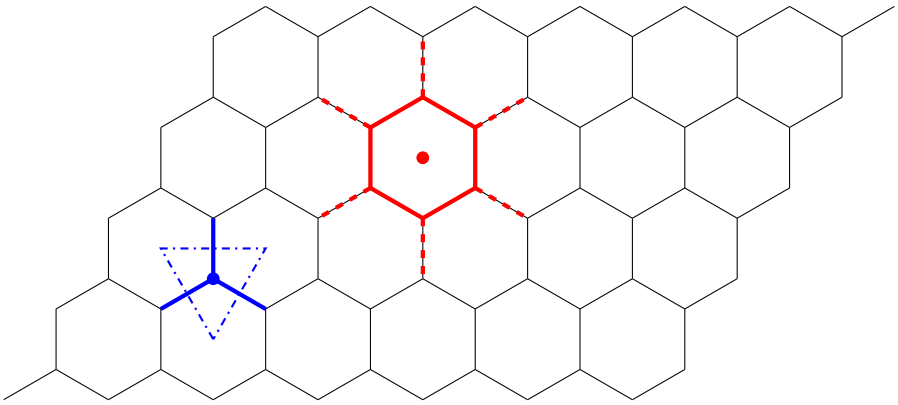


Figure 2.2.7: Operators in the DS-model. A_S shown in blue, with the part of the sub-lattice it affects shown with ".-" line. B_p is shown in red. The spins in the first factor in B_p are highlighted with a bold line, whilst the spins in the phase factor are highlighted with a dashed line

Importantly these operators do not commute amongst each other in the entire Hilbert space. However it can be shown that the following commutator properties hold: $[B_p, B_{p'}] \left|\text{loop}\right\rangle = 0$ and $[A_s, B_p] \left|\text{loop}\right\rangle = 0$. Where $\left|\text{loop}\right\rangle$ is any state which satisfies $A_s \left|\text{loop}\right\rangle = \left|\text{loop}\right\rangle, \quad \forall s$. This implies that we can minimize the A_s and B_p terms simultaneously, as constructing states with $A_s \left|\psi\right\rangle = \left|\psi\right\rangle$, will imply that the B_p operators commute in the $A_s = 1$ vector space. Thus we may construct ground states satisfying $A_s \left|\psi\right\rangle = \left|\psi\right\rangle$ and $B_p \left|\psi_0\right\rangle = -\left|\psi_0\right\rangle$ [3].

Like the TC model the GSM of the DS model has fourfold degeneracy [10]. We can derive them by following the same procedure as for the TC model from [3], where we will have to change the projection operator due to the sign difference in the B_p term in the Hamiltonian. The projection operator will therefore instead be given by:

$$\begin{aligned} \left|\psi_0\right\rangle &= \prod_p P_p \left|\text{loop}\right\rangle \\ P_p &= (-B_p + 1)/2 \end{aligned} \tag{2.17}$$

As with the TC model there will exist loops of even or odd parity in x and y direction. We will later more formally define the different ground states exploiting this fact.

The loop operators of the DC-model is given in [10]. The dual-lattice V-operators will be the same as with the TC-model, however the U-operators will differ due to the phase in the B_p operators. For these we will need to chose a direction along the loop such that we can meaningfully define the right and left side of the loop. Furthermore the U-operators come in two types ($+$ or $-$). Two examples of U and V operators are drawn in fig 2.2.8 and 2.2.9 respectively, and their expressions are:

$$
\begin{aligned}
V_{\mathcal{L}} &= \prod_{i\in\mathcal{L}} \sigma_i^z \\
U_{\mathcal{L}}^{\pm} &= \prod_{i\in\mathcal{L}} \sigma_i^x \prod_{k\in L} (-1)^{\frac{1}{4}(1-\sigma_i^z)(1+\sigma_j^z)} \prod_{l\in R} (\pm i)^{(1-\sigma_l^z)/2},
\end{aligned}
\tag{2.18}
$$

where L is the set of spins connecting to the loop from the left side, whilst R is the set of spins connecting to the loop from the right side. For the L factor i is the spin on the loop before the intersection with the spin k, whilst j is the spin after the intersection.

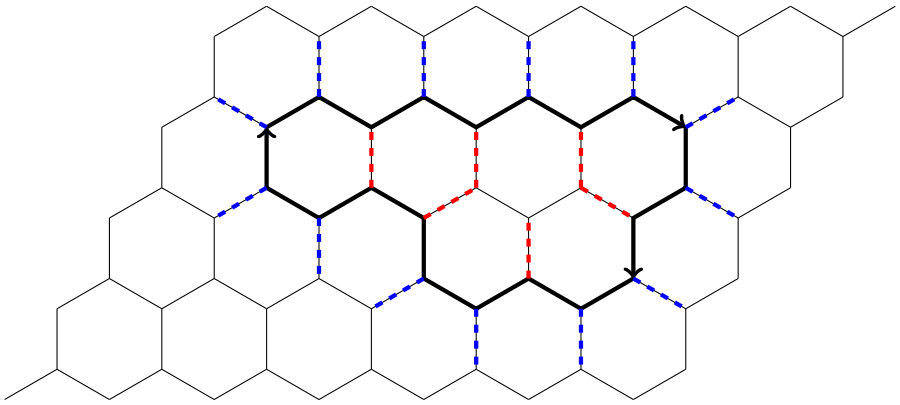


Figure 2.2.8: Drawing of a $U_{\mathcal{L}}^{+}$ operator. The $\mathcal{L}$ factor is shown as a thick black line, and its orientation is shown with arrows. The L factor in $U_{\mathcal{L}}^{+}$ is to the left of the loop, and is shown in blue. The R factor is to the right of the loop and is shown in red

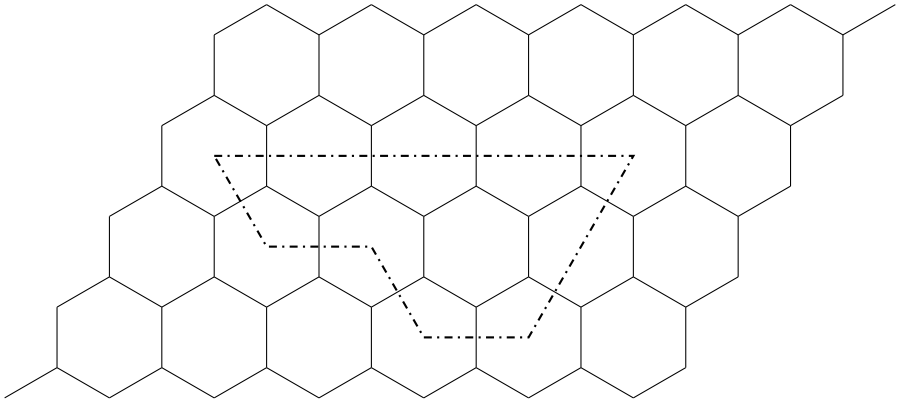
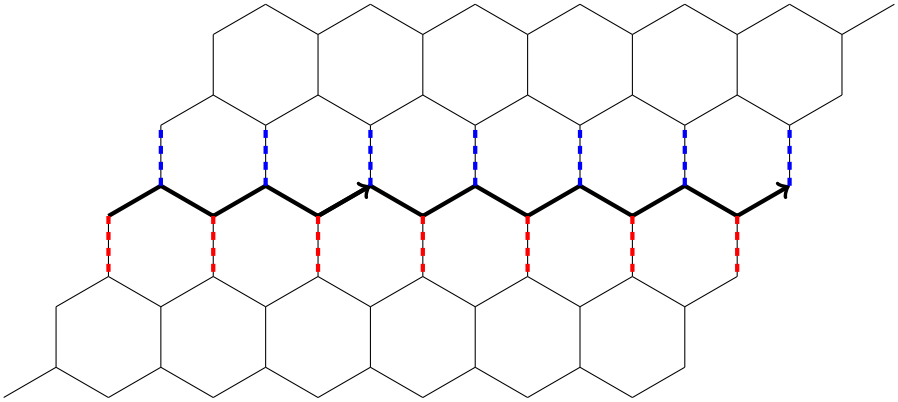
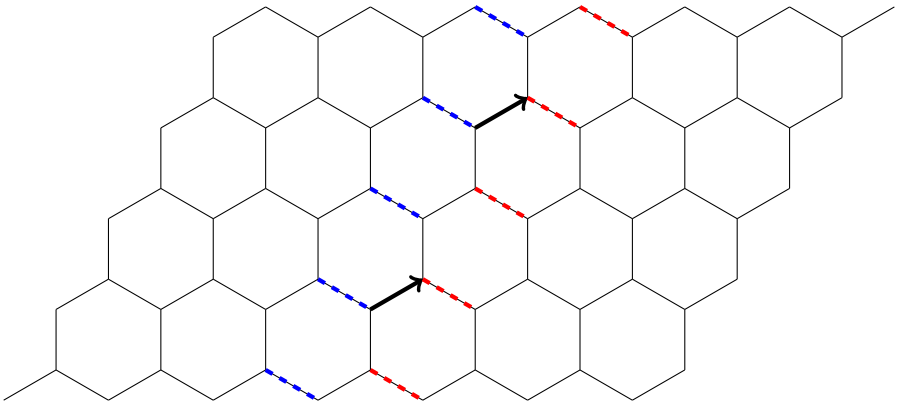


Figure 2.2.9: Drawing of a $V_{\mathcal{L}}$ operator. The involved spins are all the vertexes of the honeycomb which $V_{\mathcal{L}}$ passes through

The extra phase terms are needed in order for the $U_{\mathcal{L}}^{\pm}$ operators to commute with the Hamiltonian. As with the TC-model we may choose one loop in the x direction and one in the y direction on the lattice, examples for which are shown in fig 2.2.10.
We may do the same for the V operators. We do however get twice as many U operators in the DS model as in the TC model due to $\pm$ in the R-term. This results in a complicated set of operators which do not easily form a group. First of all, the



(a) The $U_x^{\pm}$*operator*. Direction is from left to right marked with arrows



(b) The $U_y^{\pm}$*operator*. Direction is from the bottom to the top marked with arrows

Figure 2.2.10: Visualization of the $U_x^{\pm}$ and $U_y^{\pm}$ operators. The spins belonging to the loop factors $(\mathcal{L}_x, \mathcal{L}_y)$ are shown in black. the L terms are to the left of the loops and are shown in blue. The R terms are to the right of the loops and are shown in red

U-operators are not self-inverses as they are in the TC-model. Furthermore the $U_x^{\pm}$ and $U_y^{\pm}$ operators do not commute. We will later construct a group containing all these operators, which we will use to derive properties of the DS-model.

2.2.5 Doubled semion ladder model

If we remove the periodicity in the y-direction for the DS-model, as we did for the TC model to get the TCL model, we get the doubled semion ladder (DSL) model. The Hamiltonian is then the same as for the DS model, however the operators change slightly due to there only being two layers and no y-periodicity. The resulting operators are shown in fig 2.2.11. We note that these operators should still commute in the $|\text{loop}\rangle$ state vector space, as the overlap of these operators are the exact same as if we would extend them to the DS model. We will therefore get ground states which are $+1$ eigenstates of the A_s operators and -1 eigenstates of the B_p operators.

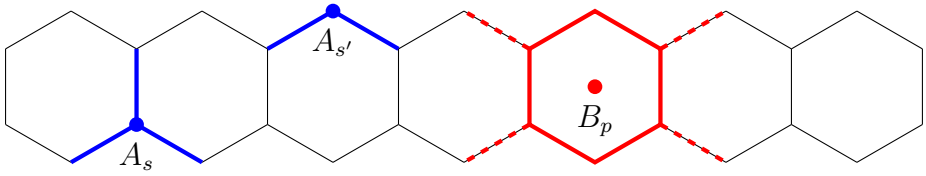


Figure 2.2.11: Operators in the TCL model. The B_p operator is shown in red, where the loop factor is shown as a bold line and the phase factor is shown as dashed. The A_S operators are shown in blue.

2.3 Matrix formalism and diagonalization

Quantum systems where the dimension of the Hamiltonian is not too large may be solved using exact diagonalization. To use exact diagonalization we need to use the matrix formalism of quantum mechanics, as outlined in [2]. To start out we need a a Hamiltonian operator, $\hat{H}$ and a complete, orthonormal set of quantum states $\{|\psi_i\rangle\}$ which we use as a basis. Then any state $|\phi\rangle \in \mathcal{H}$, can be represented as a column vector $\vec{\phi}$, where the i-th element of this vector is given by $\vec{\phi}_i = \langle\phi|\,\psi_i\rangle$. This is effectively equivalent to representing the state ϕ as a linear combination of basis states. Now we define the Hamiltonian matrix according to equation 2.19

$$\mathbf{H_{i,j}} = \langle\psi_i|\,\hat{H}\,|\psi_j\rangle \tag{2.19}$$

If this matrix is finite, and sufficiently small, we can now diagonalize this matrix. Sufficiently small depends on the computer power one has available (Ellaborate). The algorithm used for diagonalization scales as $O(n^3)$ [13], where n is the order of the matrix. Thus breaking up the problem into smaller matrices may significantly accelerate the process. Suppose the derived Hamiltonian matrix were block diagonal. In that case we would not need to diagonalize the entire matrix at once, instead we could diagonalize every block separately, and in parallel.
To get a block diagonal matrix we can exploit the symmetries of the Hamiltonian operator. If there exists a set of hermitian operators $\{\hat{A}_i\}$, which commutes with the Hamiltonian and mutually commute, i.e. $[\hat{H}, \hat{A}_i] = 0\ \forall i$ and $[\hat{A}_i, \hat{A}_j] = 0\ \forall i, j$ then we can create a set of simultaneous eigenstates of the operator s. When we construct the matrix using these states as our basis we will then be guaranteed to get a block diagonal matrix, Which we will now prove.

Assume we have constructed a set of eigenstates of the symmetry operators $\{\hat{A}_i\}$. We label these states by the quantum numbers $\{a_i\}$ corresponding to each of the operators, and a second number i which differentiates states with the same quantum number.

$$\hat{A}_k\,|a_1, a_2, ..., a_n; i\rangle = a_k\,|a_1, a_2, ..., a_n; i\rangle \qquad \forall i, k \tag{2.20}$$

We will now operate on this state with the Hamiltonian and any one of the $\hat{A}_i$ operators.

$$\hat{A}_k\hat{H}\left|a_1,a_2,...,a_n;i\right\rangle = \hat{H}\hat{A}_k\left|a_1,a_2,...,a_n;i\right\rangle = a_k\hat{H}\left|a_1,a_2,...,a_n;i\right\rangle \qquad \forall i,k \tag{2.21}$$

This equation shows that the state $|\phi\rangle = \hat{H}\left|a_1,a_2,...,a_n;i\right\rangle$ is also an eigenstate of all the $\hat{A}_i$ operators. We will make use of the spectral theorem for hermitian operators. One of the statements in that theorem is that eigenvectors of hermitian operators corresponding to different eigenvalues are orthogonal. Since the state $|\phi\rangle$ is an eigenstate of all the $\hat{A}_k$ operators, with the same quantum numbers as $\left|a_1,a_2,...,a_n;i\right\rangle$, we may conclude that

$$\begin{aligned}\left\langle a_1',a_2',...,a_n';i\right|\hat{H}\left|a_1,a_2,...,a_n;j\right\rangle &= \left\langle a_1',a_2',...,a_n';i\right|\phi\rangle \\ &= \delta_{a_i,a_i'}\left\langle a_1',a_2',...,a_n';i\right|\phi\rangle\end{aligned} \tag{2.22}$$

We can therefore now treat each set of basis vectors with the same eigenvalues as a separate Hilbert space, as each of these subspaces are invariant under the Hamiltonian. Each subspace will therefore have their own Hamiltonian matrix. We define each matrix according as:

$$\mathbf{H}_{\mathbf{i,j}}^{\mathbf{a_1},\mathbf{a_2},...,\mathbf{a_n}} = \left\langle a_1,a_2,...,a_n;i\right|\hat{H}\left|a_1,a_2,...,a_n;j\right\rangle, \tag{2.23}$$

with the subscript indicating the elements in the matrix and the superscript indicating the quantum numbers of the states in this block. These matrices can now be numerically diagonalized independently.

2.4 Group theory

In this section we will give a brief overview of the group theory used in the thesis. The main source for this section is [14]. A group is formally defined as a set of distinct elements $G = g_1, g_2...$ and an associated binary operator $\circ$ which follows the following four axioms:

(1) The group is closed under the binary operator:
$g \circ h \in G, \quad \forall g,h \in G.$

(2) The binary operator is associative on G.
$a \circ (b \circ c) = (a \circ b) \circ c, \quad \forall a,b,c \in G$.

(3) There exists an identity element, usually denoted E.
$\exists E \in G$ such that $E \circ g = g \circ E = g, \quad \forall g \in G.$

(4) Every element has an inverse.
$\forall g \in G, \ \exists g^{-1} \in G$ such that $g \circ g^{-1} = g^{-1} \circ g = E$

One of the simplest examples of a group is the real numbers with $+$ as their binary operator. The real numbers are closed under addition, so the first axiom is satisfied. $+$ is associative, so the second axiom is satisfied. 0 will act as an identity element, satisfying the third axiom and finally every real number x has $-x$ as its

inverse, as $x + (-x) = 0$, so the fourth axiom is satisfied. We note that the size of a group may vary greatly. A finite groups have a distinct number of elements, and we can have continuous infinite groups like $\mathbb{R}$ with $+$ as binary operator, and we can have discrete infinite groups, like $\mathbb{Z}$ with $+$ as its binary operator [14, p. 3]. In this case all the elements commute, however for some group G the products of the elements $g, h \in G$ given by $g \circ h$ and $h \circ g$, will not in general be equal.

In the context of physics we are often the most interested in what is called symmetry groups. The elements of symmetry groups are symmetry operators, which leaves some system invariant when they act upon it. An example of this is the rotation group C_{4v} which is the set of all reflections and rotations which leaves a 2D square invariant. An example of one of these elements is c_1 which is a $\pi/4$ rotation. In the context of quantum mechanics the symmetry group of a system may be defined as:

$$Sym(\hat{H}) = \{\hat{A} | [\hat{H}, \hat{A}] = 0\} \tag{2.24}$$

Hence the symmetry group of the Hamiltonian $Sym(\hat{H})$ is the set of all operators which commute with the Hamiltonian. This set is always a group. The binary operator is successive application of operators, which is associative. The set is closed, as for two operators $\hat{A}, \hat{B} \in sym(\hat{H})$, $[\hat{H}, \hat{A}\hat{B}] = 0$. The identity exists, as the "do nothing" operator: I or 1 will commute with $\hat{H}$. Finally symmetry transformations are reversible, so the inverse element for any symmetry operator must exist.

Now suppose a subset $S \subset G$ with the same binary operator as G also follows all the group axioms. If this is the case then S is called a subgroup of G. As an example we may let G be previously mentioned group $\mathbb{Z}$ with binary operator $+$. If we let S be the even numbers (positive and negative), then S is obviously a subset of G. This set also follows all the group axioms, so it is itself a group. On the other hand the set of odd numbers is not a group, as it does not have the identity, and it is not closed as the sum of two odd numbers is even.

Lastly we will consider the direct product of two groups G and H, which is given by the equation:

$$K = G \times H = \{g \circ h | g \in G, h \in H\} \tag{2.25}$$

We are essentially combining every element of H with the elements of G. For this to be a valid direct product every element of G must commute with every element of H. Furthermore the only overlap between the two groups must be the the identity operator [14, p. 17].

2.4.1 Isomorphisms and Homomorphisms

An isomorphism is a mapping, denoted as ϕ between two groups. It has the property that it preserves group structure. The group structure of a group G may be visualized in a character table. Suppose we numerate each element in G as g_1, g_2.... We can then produce a multiplication table of G, in which the element in the $i-th$ row and the $j-th$ column represents the product $g_i * g_j$. The multiplication table contains all the information about the group and it itself characterizes a group completely. An isomorphism is any mapping between groups

which preserves the multiplication table. Two isomorphic groups will have the same properties. For instance they will have the same irreducible representations, which we will define in the subsequent section.

A homomorphism is a generalization of an isomorphism. An isomorphism is bijective, whilst a homomorphism in general is not. A homomorphism ϕ is a mapping from a group G_1 to a group G_2 which satisfies the homomorphism condition

$$\begin{gathered} g, h \in G_1 \\ \phi(g)\phi(h) = \phi(g \circ h) \end{gathered} \tag{2.26}$$

The mapping will result in several elements in G_1 being sent to the same element in G_2, unless they have the same size, as the homomorphism would then be an isomorphism [14, pp. 18–19].

2.4.2 Representation theory for finite groups

Representation theory is an area of group theory which is very important for Quantum mechanics. It is the basis for the fingerprinting method, and it gives a way of explaining degeneracies in the spectrum of some quantum systems. We will in this section explain why this is the case. As all the groups we will investigate in this thesis are finite, we will only consider finite representation theory.

Firstly we need to define another group property. We let G be some group, and we let $a, b \in G$. These two group elements called conjugate if and only if:

$$\exists c \in G \text{ such that } a = c^{-1}bc \tag{2.27}$$

This property is reflexive. If a is conjugate with b the above equation holds. By inference we may conclude that likewise b is conjugate with a, as we may just replace c with c^{-1} and get:

$$\text{such that } b = cac^{-1} \tag{2.28}$$

Using the definition for conjugate elements we may now define the concept of a conjugacy class. Suppose we start out all the group elements in G. We then partition the set of all the group elements into subsets where each element in a given subset is conjugate to each other. These subsets are called the conjugacy classes of the group G [14, pp. 11–12]. We may label each conjugacy class by any one of the elements it contains, as given one element we may construct the whole conjugacy class by the following definition:

$$Conj(g) = \{g' \in G | \exists h \in G \text{ such that } g' = hgh^{-1}\} \tag{2.29}$$

We will denote the number of conjugacy classes of a group G as n_c. An important clarification about the conjugacy classes is that they are not groups, with only one exception. The identity element commutes with every other element, which results in there being no other element conjugate to it. This implies that none of the other conjugacy classes will contain the identity element, so they are not groups.

We may now define the representation of a finite group. We let G be some finite group with elements g_i. We may then define a representation of this group

as a set of matrices T, where we have one matrix associated with each group element. We write the matrix corresponding to the group element g_i as $T(g_i)$. These matrices need to be of the same size and they must be invertible. Finally we let the matrices have the property

$$\Gamma(g_i)\Gamma(g_j) = \Gamma(g_i g_j) \ \forall g_i, g_j \in G \tag{2.30}$$

If this property is satisfied then this set of matrices Γ is called a representation of G [14, pp. 58–59]. We call the order of the matrices the dimension the representation. Note that a representation need not capture all the multiplication properties of a group. The trivial representation $\Gamma(g_i) = 1 \ \forall g_i$ will be a valid representation for any group G [14, p. 60].

We may construct a representation of a set of operators by acting with the operators on some vector space which is invariant under the group operations. We denote a vector space with this property as L_n, where n is the dimension of the vector space. We let $\{|\phi_i\rangle\}_{i=1}^n$ be an orthonormal basis of L_n. We may write the action of an operator $\hat{g} \in G$ acting on a basis state $\langle\phi_i|$ as:

$$\hat{g}\,|\phi_j\rangle = \sum_{i=0}^{n} \Gamma(\hat{g})_{ij}\,|\phi_i\rangle \tag{2.31}$$

left-multiplying both sides of the equation with $\langle\phi_i|$ and switching the LHS and RHS yields:

$$\Gamma(\hat{g})_{ij} = \langle\phi_i|\,\hat{g}\,|\phi_j\rangle \tag{2.32}$$

The choice of an orthonormal basis makes it so that the resulting matrices are unitary as well. We could have chosen to start with a non-unitary basis, and used a transformation to turn the resulting non-unitary matrix into a unitary one, however for simplicity we will skip that step. As mentioned earlier the subspace L_n is closed under the group operations. Now we consider the case where the case where a proper subspace of L_n, which we call L_m, is also closed under the group operations. If that is the case then the representation is said to be reducible. If we chose another orthonormal basis L_n, which we call $\{|\psi_i\rangle\}$, where the first m states lie in L_m, and the rest lie outside, then the resulting representation will be block-diagonal.

$$\Gamma(A) = \begin{bmatrix} S^{(1)} & 0 \\ 0 & S^{(2)} \end{bmatrix}, \tag{2.33}$$

where $S^{(1)}$, $S^{(2)}$ are matrices of order m and $n-m$ respectively. If the subspace these matrices correspond to may be split into invariant subspace themselves, we repeat the procedure. If the subspace cannot be split into smaller invariant subspaces then the corresponding representation is said to be irreducible. We may transform any unitary representation Γ into a block diagonal form by unitary transformation, where the blocks correspond to the irreducible representations of the symmetry group. We may therefore write the reducible representation Γ as a direct sum of the irreduible representations of G

$$\Gamma = \oplus_\alpha n_\alpha \Gamma^(\alpha), \tag{2.34}$$

where $\Gamma^(\alpha)$ is the α-th irrep of G, and n_α is the number of times this irrep appears in Γ. The number of irreps a certain finite group has is equal to the number of conjugacy classes.

Irreducible representations are not unique, as we can transform the basis by unitary transformation and it would yield a different, but equivalent representation. One property that will be the same for any irreducible representation of a group however is the trace of the irreps, as the trace of a matrix is conserved under unitary transform. We call the trace of a representation the characters of the representation, and are denoted $\chi(g)$. The characters of the irreducible representations of a given group G are very important, and may be used to decompose any representation into irreducible representations, as we will show in the next section. Lastly the characters of a representation are class functions. This signifies that two conjugate elements will have the same characters.

Knowing the irreducible representations of the symmetry group $\mathrm{Sym}(\hat{H})$ gives us a tool for explaining the energy degeneracy in quantum systems. Suppose we start out with an eigenstate $|\phi\rangle$ of the Hamiltonian $\hat{H}$ with energy E. Then by operating on $|\phi_1\rangle$ with all the operators in $\mathrm{Sym}(\hat{H})$ we would create a set of n independent eigenstates $|\phi_n\rangle$. Using these states as a basis for a representation, the resulting representation would necessarily be irreducible. The problem with this procedure is that it is exceedingly difficult to know whether or not we have actually included all the symmetry operations in $\mathrm{Sym}(\hat{H})$. It is also possible, though unlikely that eigenstates belonging to two different representations share the same energy eigenvalue. This is called an accidental degeneracy.

Finally we will discuss a generalization of representations called projective representations. A projective representation does not satisfy the relation in equation 2.30. Instead it satisfies the following relation:

$$\Gamma(g_i)\Gamma(g_j) = \omega(g_i, g_j)\Gamma(g_i g_j) \ \forall g_i, g_j \in G \tag{2.35}$$

Where $\omega(g_i, g_j)$ is some complex phase. When this is the case we may need to create a covering of the group G. We say that when we get this behaviour then we say that we have a projective representation. Furthermore our group G is called a projectile linear group, which we will denote $PGL(V)$, where V is the vector space the group acts on. We then have a general linear group $GL(V)$ and the two are related by a homomorphism. We may write the relation as:

$$\frac{GL(V)}{F} = PGL(GSM) \tag{2.36}$$

Where F is the normal subgroup of $GL(V)$. In general the elements will be scalar multiples of the identity operator [15].

2.4.3 The fingerprinting method

The fingerprinting method is essentially just doing irreducible representation decomposition, where the vector space we are using to construct the representation is the ground state manifold (GSM). The procedure is outlined in [16]. Here they define the GSM as the set of ground states which become degenerate in the thermodynamic limit. For a finite size system, however the states may be degenerate. The GSM is defined as:

$$GSM = span(|\psi^i_{GS}\rangle) \tag{2.37}$$

Where $|\psi^i_{GS}\rangle$ are the the lowest energy states of some Hamiltonian $\hat{H}$, which become degenerate in the thermodynamic limit. We may then use this basis to create a representation of $\mathrm{Sym}(\hat{H})$. Let $g \in \mathrm{Sym}(\hat{H})$ and let U_g be the operator acting on the GSM which corresponds to the group element g. Then we may then write the resulting representation as:

$$\Gamma^{GSM}_{ij}(g) = \langle \phi^i_{GS}| \, U_g \, |\phi^j_{GS}\rangle \tag{2.38}$$

We may decompose this representation into a direct sum of irreps of $\mathrm{Sym}(\hat{H})$ as shown in equation $\mathrm{Sym}(\hat{H})$. The integers n_α may be written as an array, where the α-th element corresponds to how many times the α-th element appeared in the decomposition of Γ^{GSM}. We will call this array the fingerprint of the GSM, as it essentially classifies how the GSM transforms under the group operations. In order to calculate n_α we may use the character formula:

$$n_\alpha = \frac{1}{|G|} \sum_{g \in G} \overline{\chi(g)} Tr(\Gamma(g)) \tag{2.39}$$

If Γ^{GSM} turns out to be an irrep, then $n_\alpha = \delta_{i\alpha}$, where i corresponds to the i-th irrep of G. In general G should equal $\mathrm{Sym}(\hat{H})$. In reality, however, the size of $\mathrm{Sym}(\hat{H})$ may be so large that analyzing the system and determining the irreps become prohibitively difficult. A trick which we may use to simplify the problem is splitting the symmetry group into a direct product of two commuting subgroups. For instance for a model which conserves total spin and magnetization, like the Heisenberg model, the operators $\hat{S}^2_{tot}$, $\hat{S}^z_{tot}$ and the space group (SG) all mutually commute, which lets us define the symmetry group as the direct product $\{E, \hat{S}^2_{tot}\} \times E, \hat{S}^z_{tot} \times SG$. Since the irreps of a direct sum may be written simply in terms of the irreps of the component groups, we may focus on the smaller subgroups instead of the whole $\mathrm{Sym}(\hat{H})$ at the same time.

2.4.4 Intrinsic Group Properties

Every group has certain properties that can be determined either by direct calculation or by finding an isomorphism from the group to a known, tabulated group and looking up the properties of that group.

The number of isomorphically different groups of any given finite order is finite. Therefore, if we have a group with defined group elements and a binary operator, we can identify which of the tabulated groups of the same order it is isomorphic to by filtering out every candidate that has intrinsic group properties different from the group we are investigating.

There are several tools for doing this. For instance, the GAP programming language [17] has tabulated all groups of finite order up to about 2000 (with the exception of 1024, which is exceptionally large). In GAP, several properties of the listed groups can be found by a simple function call, while others need to be derived through a script. To derive the groups that are isomorphic to the symmetry groups of the TC and DS models, the following group properties will be measured. We can then filter for these properties to find the tabulated groups they

are isomorphic to. As the properties are only used to filter for possible groups, the explanations will be quite brief, and the source for the section is [18].

The *exponent* of a group G is defined as the smallest integer n such that $\forall g \in G, g^n = E$. In essence, this measures the largest order of an element in the group.

The *order of an element* is defined similarly to the exponent, but it is the minimal n such that for a given $g \in G$, $g^n = E$. Therefore, different elements may have different orders. It can be shown that elements of different orders cannot be in the same conjugacy class.

The *center* of a group is defined as $C_G = \{g \in G \mid \forall g' \in G, gg' = g'g\}$. Simply put, it is the subgroup of G comprised of all the elements in G that commute with every element in G. The size of this group can be used to eliminate several candidate groups, and if it has a specific group structure (cyclic, for instance), that can further limit the number of candidates.

The *derived subgroup*, or commutator subgroup, is defined as $D_G = \{g \in G \mid g = x^{-1}y^{-1}xy \text{ for some } x, y \in G\}$. It essentially encapsulates the commutativity of the group. For two commuting elements, the group commutator defined as $[x, y] = x^{-1}y^{-1}xy$ will be E. On the other hand, if there is some non-commutation between the elements, the result will be another group element.

The number of *conjugacy classes* and their sizes can also be used to filter out group candidates. The conjugacy classes of small groups can be derived computationally by brute force.

The *minimum number of generators* for a group can also be used. A *generator* of a group G is an element from which every element of the group can be obtained by repeated application of the group operation. If a group can be generated by a single element, it is called a *cyclic group*. More generally, a *set of generators* $S \subseteq G$ is a subset such that every element of G can be expressed as a combination of elements from S and their inverses. For example, in the cyclic group $\mathbb{Z}_5$ under addition modulo 5, the element 1 is a generator because $\{0, 1, 2, 3, 4\}$ can be obtained by repeatedly adding 1. In contrast, the dihedral group D_4 (symmetries of a square) is generated by the set $\{r, s\}$, where r is a 90-degree rotation and s is a reflection. Thus, every element of D_4 can be expressed as a combination of r and s.

The *Frattini subgroup* is the group of nongenerators and is defined as $F_G = \{g \in G \mid \forall S \subseteq G, \text{ if } S \cup \{g\} \text{ generates } G \implies S \text{ generates } G\}$. This means that no elements in the Frattini subgroup can help generate the entire group G if they are added to any generating set. Any proper generator of G will generate the elements in the Frattini subgroup. For instance, the identity element E is always part of the Frattini subgroup as it can be created by exponentiating any element to its highest order. Another example is in the cyclic group $\mathbb{Z}_4$, where the element 2 cannot generate the entire group and is thus in the Frattini subgroup.

CHAPTER

THREE

METHODS

3.1 Numerical derivation of fingerprints

We will investigate the fingerprint of the second nearest neighbour Hamiltonian by numerically deriving the eigenstates of the system. We will do so by exact diagonalization. For the purpose of investigating the GSM we will note that full, exact diagonalization is not the most efficient approach. The best numerical tool would likely be the Lanczos algorithm [13], which can derive the ground state and lowest lying eigenstates of an operator very efficiently. However for the purpose of spectrum analysis we will use exact diagonalization instead. Should the method be generalized in code for other models however, the Lanczos algorithm would probably be the best general method for deriving the GSM.

3.1.1 Generation of $|S, S_z; i\rangle$ spin basis

The J-J' Hamiltonian has several exploitable symmetries, as mentioned in 2.2.1. There is however a trade off in efficiency when block diagonalizing the Hamiltonian. If the construction of the given spin basis and the more complicated Hamiltonian matrix takes more compute time than we save by diagonalizing smaller blocks, then we are better off including less symmetries in our block diagonal basis. For spin-$\frac{1}{2}$ models one generally chooses to start out with an eigenbasis of the translational group and total magnetization operator ($\hat{S}^z_{tot}$), or an eigenbasis of the total spin operator (S^2_{tot}) and the total magnetization operator. For this project we will construct the latter as it will make spectrum analysis later slightly simpler. It is possible to do all three, although it is very difficult, and is done in [19]. In this thesis however, focuses on the properties of the ground states first, and derives the ground states as a means to an end, we will only generate a $|S, S_z; i\rangle$ spin basis. Lastly one should note that there exists libraries for diagonalization on arbitrary spin systems. In `python` there is `QuSpin` [20], and for `C++` there was previously the `ALPS` [21], however that is no longer supported due to lack of funding (although the legacy source code is still available). Both of these use translational symmetry and $\hat{S}^z_{tot}$ symmetry to do the block diagonalization. We will not use these libraries in order to investigate the $|S, S_z; i\rangle$ spin basis and to get a better understanding of the system.

We start by constructing a slightly simpler basis. The z-basis, which is the basis of all $\hat{S}_i^z$ operators, commonly written as $|\uparrow\downarrow ... \uparrow\rangle$ is also an eigenbasis of $\hat{S}_{tot}^z$. We may exploit this numerically by writing the states in the z-basis as binary numbers. We let spin-$\downarrow$ be represented as binary 0, and spin-$\uparrow$ be represented as binary 1. For the purpose of generating an eigenbasis of $|S, S_z; i\rangle$ the ordering of the spins does not matter, so we may choose whichever will be the simplest for the computation. For our case for example, we may chose the zeroth (we use zero-indexing henceforth) bit to be the zeroth spin in a spin chain, the first to be the first, and so on.

In the code implementation we will implement the basis states as a vector of boolean arrays of length N, however one could also implement the basis elements as integers directly. In any case we can associate each element with an integer either directly or by converting the boolean array to an integer. We construct the basis set then to be the integers from 0 to $2^N - 1$ in ascending order. We can now calculate the $\hat{S}_{tot}^z$ eigenvalue for each of the basis elements. We can then do what is called a stable sort of the elements with respect to the $\hat{S}_{tot}^z$ eigenvalue, which we will denote as S_z. A stable sorting algorithm is one that preserves the ordering of elements in the original list given that the parameter the sorting is based on is equivalent [22]. The algorithm we will choose is merge-sort as it is built into `np.argsort`. As an example we will consider the 4-spin basis, construct it first as the integers from 0 to 15, then determine the S_z eigenvalue for every state, then finally we perform merge-sort on the original list with respect to the S_z eigenvalue for each state.

Original Basis			**Sorted Basis**			
Index	Basis	S_z Eigenvalue	**Index**	Basis	S_z Eigenvalue	Index in Block
0	$\|0\rangle$	-2	**0**	$\|0\rangle$	-2	0
1	$\|1\rangle$	-1	**1**	$\|1\rangle$	-1	0
2	$\|2\rangle$	-1	**2**	$\|2\rangle$	-1	1
3	$\|3\rangle$	0	**3**	$\|4\rangle$	-1	2
4	$\|4\rangle$	-1	**4**	$\|8\rangle$	-1	3
5	$\|5\rangle$	0	**5**	$\|3\rangle$	0	0
6	$\|6\rangle$	0	**6**	$\|5\rangle$	0	1
7	$\|7\rangle$	1	**7**	$\|6\rangle$	0	2
8	$\|8\rangle$	-1	**8**	$\|9\rangle$	0	3
9	$\|9\rangle$	0	**9**	$\|10\rangle$	0	4
10	$\|10\rangle$	0	**10**	$\|12\rangle$	0	5
11	$\|11\rangle$	1	**11**	$\|7\rangle$	1	0
12	$\|12\rangle$	0	**12**	$\|11\rangle$	1	1
13	$\|13\rangle$	1	**13**	$\|13\rangle$	1	2
14	$\|14\rangle$	1	**14**	$\|14\rangle$	1	3
15	$\|15\rangle$	2	**15**	$\|15\rangle$	2	0

We may use the results from the table to define the $|S_z; i\rangle$ basis, where S_z is a conserved quantum number for the model in consideration, whilst i is simply a degeneracy index. As a note "degenerate" in this context does not mean that they will have the same energy, rather that there are several states with the same S_z quantum number. For the duration of this section we will take "degeneracy" of a

basis state to mean that there are multiple states with the same set of quantum numbers. We define the counting index i as the the "index in block" shown in the table, which is simply the index the element appears in the list minus the block start, which is the index where the first state with a given S_z quantum number appears.

There are two reasons for why we construct the S_z basis this way. Firstly it lets us use binary search to find a given state in the sorted basis. This algorithm is very fast, specifically it is of order $O(\ln_2(N))$, where N is the number of elements in the given block we search [22]. So if we implement our base operators (such as S_i^+) as operators transforming elements in the z-basis into other elements in the z-basis, we can find the element it was transformed into very quickly.

Secondly it lets us use less memory for the $|S, S_z; i\rangle$ basis. We note that a general quantum state may be written as a linear combination of the $|S_z; i\rangle$ basis due to it being complete. It would then be a linear combination of every S_z and every i:

$$|\psi\rangle = \sum_{S_z,i} C_{S_z,i} |S_z; i\rangle \tag{3.1}$$

A general state would therefore require one coefficient for every state in the basis, which for a N-spin-$\frac{1}{2}$ system is 2^N states. So the memory required to store the N elements in a general basis in terms of the z-basis would be a matrix of coefficients of size $2^N \times 2^N$, where each column represents a basis state. However if the basis we use is also an eigenbasis of $\hat{S}^z_{tot}$ then we may instead sum over only the elements in the same S_z block, and not over every state. This massively reduces the memory requirements of our basis, and will make our basis more efficient. So for instance the elements of the (S, S_z) basis may be written as:

$$|S, S_z; i\rangle = \sum_j C_j |S_z; j\rangle \tag{3.2}$$

We note that deriving the coefficients C_j is quite difficult. We will calculate them recursively using the Clebsch-Gordan (CG) coefficients [14], which are used to couple two separate (S, S_z) basses, by which we mean we have a Hilbert space $\mathcal{H}_A$ with a $|S, S_z; i\rangle_A$ basis, and a Hilbert space $\mathcal{H}_B$ with a $|S, S_z; i\rangle_B$ basis which we want to combine into a single $|S, S_z; i\rangle$ basis for the Hilbert space $\mathcal{H}_A \otimes \mathcal{H}_B$. In general the CG they look like

$$CG(J, M|j_1, j_2, m_1, m_2) = \langle JM|j_1 m_1 j_2 m_2\rangle \,, \tag{3.3}$$

where J signifies the quantum number for $\hat{S}^2_{tot}$, and M signifies the quantum number for $\hat{S}^z_{tot}$. The same is true for j_1, J_2, m_1 and m_2. $|JM\rangle$ is the coupled state, whilst $|j_1 m_1 j_2 m_2\rangle = |j_1 m_1\rangle \otimes |j_2 m_2\rangle$ is the uncoupled states. The CG coefficients are tabulated in several numerical programs like for instance `sympy` for `python`. The CG coefficients possess certain selection rules. For instance the inner product is only nonzero when $M = m_1 + m_2$. It is also only nonzero whenever $|j_1 - j_2| < J < |j_1 + j_2|$.

Suppose for instance we have derived a (S, S_z) basis for a system with N spins, which lies in $\mathcal{H}_A$. we then add another spin which lies in $\mathcal{H}_B$. The one spin system can be written trivially as a (S, S_z) eigenstate:

$$\begin{aligned}\hat{S}^2\left|\uparrow\right\rangle = \frac{1}{2}\left|\uparrow\right\rangle,\ \hat{S}_z\left|\uparrow\right\rangle = \frac{1}{2}\left|\uparrow\right\rangle \implies \left|\frac{1}{2},\frac{1}{2}\right\rangle = \left|\uparrow\right\rangle \\ \hat{S}^2\left|\uparrow\right\rangle = \frac{1}{2}\left|\uparrow\right\rangle,\ \hat{S}_z\left|\uparrow\right\rangle = \frac{1}{2}\left|\uparrow\right\rangle \implies \left|\frac{1}{2},-\frac{1}{2}\right\rangle = \left|\downarrow\right\rangle\end{aligned} \tag{3.4}$$

We therefore have two uncoupled total angular momentum eigenstates, which we can use CG-coefficients to couple. Every state in the $n-1$ spin basis can be combined with either $|\uparrow\rangle$ or $|\downarrow\rangle$, which in turn can be used to construct the n spin basis, denoted as $|\tilde{S},\tilde{S}_z;j\rangle$. Due to the selection rules for the CG coefficients each one of those combinations $|S,S_z;i\rangle\otimes|\uparrow\rangle$ and $|S,S_z;i\rangle\otimes|\downarrow\rangle$ will only give a nonzero CG coefficient when $\tilde{S}_z = S_z \pm \frac{1}{2}$, where sign depends on if the added spin is $|\uparrow\rangle$ or $|\downarrow\rangle$, and $\tilde{S} = S \pm \frac{1}{2}$ and $\tilde{S} \geq 0$. Therefore we get two nonzero CG coefficients for each combination of a $n-1$ spin $\frac{1}{2}$ state $|S,S_z;i\rangle$ and a single spin $\frac{1}{2}$ state, with the exception of if $\tilde{S}=0$ where we only get 1. This is because for $\tilde{S}=0$

This does not tell us which of the degenerate states a given spin $n-1$ state should contribute to. In order to determine this we need to exploit Catalans triangle [23]. This gives us a theoretical formula for the degeneracy of the states. Furthermore we may determine the degeneracy index i from the Catalans triangle. The degeneracy for a given total spin S at a given number of spins S is equivalent to the number of paths there are in 3.1.1.

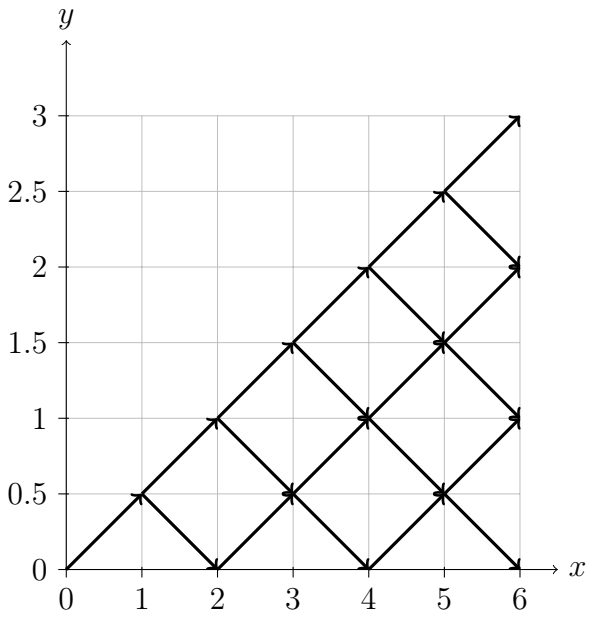


Figure 3.1.1: Paths to a total spin S, with a given N

We can create a binary number for each path by starting at spin $\frac{1}{2}$. We then define a path number by the following criterion. If the path goes from a node with spin S to a path with spin $S-\frac{1}{2}$, then we label this binary 0. If instead it goes up to $S+\frac{1}{2}$ then we denote the state with binary 1. We can therefore calculate the spin associated with a given path B with the equation:

$$S = \sum_{i}^{n-1}(-1)^{b_i+1}\frac{1}{2}, \tag{3.5}$$

where B_0 is the zeroth bit of the binary number B, B_1 is the first, and so on, and n is the number of spins. By this construction there are an equal number of path-numbers to a given spin S and number of spins n, which lets us associate each state with the same S and S_z with a unique path. A valid path cannot end with spin less than 0 nor can the path dip bellow zero at any point of the path. Therefore our criterion for filtering out the potential paths is the following: If for any binary number representing a path B, then the following inequality must be satisfied:

$$\forall m \in \mathbb{N},\ m < n-1 :\ \sum_{i}^{m} (-1)^{B_i+1} \frac{1}{2} \geq 0, \tag{3.6}$$

To associate each path with some number we can do either of the following. Either iteratively create every binary number which represents a valid path, or we can create every binary number up to $n-1$, and then filter out the invalid paths. Note first that we can calculate the spin associated with a given path by summing over the bits, adding $\frac{1}{2}$ if the bit is 1, and subtracting $\frac{1}{2}$ otherwise.

Having filtered out the invalid paths we now have a list of binary numbers representing valid paths in acceding order by interpreting the list of binary numbers as integers. Now we exploit a similar approach as we did when we created the S_z basis with the states $|S_z; i\rangle$ earlier. We sort the path numbers based on their corresponding spin according to equation 3.5. We can now determine the degeneracy index i in the state $|S, S_z; i\rangle$ by which path we needed to take in order to create it. We can binary search for the path in the sorted set of all paths (limiting ourselves to the block with the same S, as we did for the $|S_z; i\rangle$ state). Now, suppose we have a $n-1$ spin $\frac{1}{2}$ basis, and we add another spin which can either be $|\uparrow\rangle$ or $|\downarrow\rangle$. We know the paths for the $n-1$ spin basis, and furthermore we know for example when adding the extra spin $|\uparrow\rangle$ to the $n-1$ spin state $|S, S_z; i\rangle$ it will give nonzero contribution to $|S+\frac{1}{2}, S_z+\frac{1}{2}; k\rangle$ and $|S-\frac{1}{2}, S_z+\frac{1}{2}; l\rangle$ (the latter only getting a contribution if S is nonzero). To determine the degeneracy indices k and l from the degeneracy index for the $n-1$ spin state, i, let B be the binary number associated with the path for $|S, S_z; i\rangle$, which is determined by i. For the first state we are increasing S by $\frac{1}{2}$ which implies the binary number associated with its path is B, with a single extra 1 appended to the end. The same is true for the other state except we append a 0 instead. Now we can binary search for these new binary numbers to get the degeneracy indices k and l. This is all numerically implemented with the `add_spin_to_system` function in `Diagonalizer.py`. A significant strength is that we only need to compute the spinbasis once for a specific number of spins, as we have implemented a function to save the data to a file, and another function to reimport the data. This lets us effectively precompute the spinbasis and reuse it. A further note is that the spinbasis is represented in code as a linear combination of $|S_z; i\rangle$ states as shown in equation 3.2. Numerically it is represented using the matrix formalism for basis states

$$\vec{\phi}^{S,S_z,i} = \begin{pmatrix} \langle S_z, 0 \,|S, S_z, i\rangle \\ \langle S_z, 1 \,|S, S_z, i\rangle \\ \vdots \\ \langle S_z, M-1 \,|S, S_z, i\rangle \end{pmatrix} = \begin{pmatrix} C_0 \\ C_1 \\ \vdots \\ C_{M-1} \end{pmatrix}, \tag{3.7}$$

where M is the number of degenerate states in the S_z basis for a given S_z.

3.1.2 Diagonalization of the H-matrix

Assuming we have created the $|S, S_z; i\rangle$ basis, we can use it to derive the Hamiltonian matrix, or rather the Hamiltonian matrices, as each block will be independent. We start out by filling in the equation for a Hamiltonian matrix in a given block given in equation 2.23.

$$\mathbf{H}^{\mathbf{S},\mathbf{S_z}}_{\mathbf{i},\mathbf{j}} = \langle S, S_z; i| \hat{H} |S, S_z; j\rangle \tag{3.8}$$

Due to the way that the operators in $\hat{H}$ are defined, we cannot easily evaluate this inner product. We can get rid of this problem by exploiting the completeness relation for the S_z basis: $1 = \sum_i |S_z; i\rangle \langle S_z; i|$.

$$\mathbf{H}^{\mathbf{S},\mathbf{S_z}}_{\mathbf{i},\mathbf{j}} = \sum_{k,l} \langle S, S_z; i| S_z; k\rangle \langle S_z; k| \hat{H} |S_z; l\rangle \langle S_z; l |S, S_z; j\rangle \tag{3.9}$$

The matrix element $\mathbf{H}^{\mathbf{S_z}}_{\mathbf{k},\mathbf{l}} = \langle S_z; k| \hat{H} |S_z; l\rangle$ is significantly simpler to calculate, as constituent operators of the Hamiltonian matrix have these states as eigenstates or they are ladder operators of the states. A final trick we may use is to replace $\langle S, S_z; i|$ and $|S, S_z; j\rangle$ with a column vector and row vector respectively. This gives us an expression for the entire matrix and not just one element.

$$\mathbf{H}^{\mathbf{S},\mathbf{S_z}} = \sum_{k,l} \begin{pmatrix} \langle S, S_z; 0| S_z; k\rangle \\ \langle S, S_z; 1| S_z; k\rangle \\ \vdots \\ \langle S, S_z; L-1| S_z; k\rangle \end{pmatrix} \langle S_z; k| \hat{H} |S_z; l\rangle \begin{pmatrix} \langle S_z; l |S, S_z; 0\rangle \\ \langle S_z; l |S, S_z; 1\rangle \\ \vdots \\ \langle S_z; l |S, S_z; L-1\rangle \end{pmatrix}^T \tag{3.10}$$

This expression can be written as a matrix product. We define the matrices $\mathbf{\Phi}^{\mathbf{S,Sz}}$ and $\mathbf{H}^{\mathbf{S_z}}$ as:

$$\begin{aligned} \mathbf{\Phi}^{\mathbf{S},\mathbf{S_z}}_{\mathbf{i}} &\equiv \vec{\phi}^{S,S_z,i} \\ \mathbf{H}^{\mathbf{S_z}}_{\mathbf{ij}} &\equiv \langle S_z; i| \hat{H} |S_z; j\rangle . \end{aligned} \tag{3.11}$$

We will call $\mathbf{\Phi}^{\mathbf{S,Sz}}$ the coefficient data matrix, which is defined such that the i'th row of $\mathbf{\Phi}^{\mathbf{S},\mathbf{S_z}}$ is the vector $\vec{\phi}^{S,S_z,i}$, which is defined in equation 3.7. With this we can rewrite equation 3.10 as:

$$\mathbf{H}^{\mathbf{S},\mathbf{S_z}} = \mathbf{\Phi}^{\mathbf{S},\mathbf{S_z}} \mathbf{H}^{\mathbf{S_z}} {\mathbf{\Phi}^{\mathbf{S},\mathbf{S_z}}}^T, \tag{3.12}$$

where ${\mathbf{\Phi}^{\mathbf{S},\mathbf{S_z}}}^T$ is the matrix transpose of $\mathbf{\Phi}^{\mathbf{S},\mathbf{S_z}}$. Note that the coefficient data matrix is exactly how the basis data for a given S, S_z block is stored within the code. It is a $L \times M$ matrix, where M is the degeneracy of the $|S_z; i\rangle$ states, and L is the degeneracy of the $|S, S_z; i\rangle$ states. We note that M is a function of only

S_z whilst L is a function of only S. The $\mathbf{\Phi}$ data matrices are generated when the spin basis is initialized, and since the spin basis may be saved and reused we really only need to calculate them once if the number of spins we are working with is fixed. The H^{S_z} matrices needs to be calculated only once for each S_z. So for a N spin $\frac{1}{2}$ system we only need to calculate $N+1$ $\mathbf{H^{S_z}}$ matrices. All the $\mathbf{H^{S,S_z}}$ can then be calculated by simple matrix products. We further note that the fact that this approach uses matrix multiplication for most of the numerical calculations will make it so that we can calculate it very quickly, even in `python`, as matrix multiplication in numpy is very efficient due to it being based on `BLAS`/`LAPACK` [24].

An additional exploit for the J-J' model is that the operator may be split into $\hat{H}_0$ and $\hat{H}_1$, where $\hat{H}_0$ is only the nearest neighbour terms, whilst $\hat{H}_1$ is only the second nearest neighbour terms. We write the total Hamiltonian as:

$$\hat{H} = J\hat{H}_0 + J'\hat{H}_1 \tag{3.13}$$

Inserting this into the definition of $\mathbf{H^{S_z}}$ we get the following expression:

$$\begin{aligned} \mathbf{H^{S_z}_{kl}} &= J \langle S_z; k| \hat{H}_0 |S_z; l\rangle + J' \langle S_z; k| \hat{H}_1 |S_z; l\rangle \\ \mathbf{H^{S_z}} &= J\mathbf{H^{S_z}_0} + J'\mathbf{H^{S_z}_1} \end{aligned} \tag{3.14}$$

Inserting this into equation 3.12 yields the following matrix:

$$\begin{aligned} \mathbf{H^{S,S_z}} &= J\mathbf{\Phi^{S,S_z}}\mathbf{H^{S_z}_0}\mathbf{\Phi^{S,S_z}}^T + J'\mathbf{\Phi^{S,S_z}}\mathbf{H^{S_z}_1}\mathbf{\Phi^{S,S_z}}^T \\ &= J\mathbf{H^{S,S_z}_0} + J'\mathbf{H^{S,S_z}_1} \end{aligned} \tag{3.15}$$

Since $\mathbf{H^{S,S_z}_0}$ and $\mathbf{H^{S,S_z}_1}$ are the same for second nearest neighbour model of a given size, we only need to generate the matrices once. Then for any given J and J' we can calculate the corresponding $\mathbf{H^{S,S_z}}$ matrix by using equation 3.15. Note that this does not speed up the diagonalization step. If we want the eigenbasis for a given J-J' model with constants J and J', we cannot use the result from a J-J' model with different $\tilde{J}$ and $\tilde{J}'$ unless $\frac{\tilde{J}}{J} = \frac{\tilde{J}'}{J'}$. However if we want to investigate the properties of several different J-J' models with the same number of spins, the block-diagonalization step may be made much more efficient using this approach.

Having generated the set of independent matrices through block diagonalization the last step of the process is the actual diagonalization. Since the project is implemented in `python` a natural choice is using `np.scipy.eigh`, as it is designed to work with `numpy` arrays. It diagonalizes a given hermitian matrix, and by default it returns the eigenbasis, which we need to calculate the fingerprint.

3.1.3 Derivation of space-group irreps

Most space groups (SG) of a quantum system S can be written as a semi-direct product of the translation group T and the point group P:

$$S = T \rtimes P \tag{3.16}$$

Calculating the semi-direct product is much more complicated than calculating a direct product. In order to calculate the SG-irreps we will follow a method from [14], and implement it numerically. The procedure is quite involved, however the general thing to recognize is that we may construct the irreps of the SG from the irreps of the point group and the translation group. There are a few shortcuts we may take along the way, mainly exploiting that the point group P is simply $\{E, \Pi\}$, where the operator Π is the parity operator, which flips the x-axis, and the fact that the resulting space group will be symmorphic, as there are no scew axis or glide planes in 1D. The code implementation was written more general such that with some additional work it would be applicable to symmorphic 2D and 3D lattices as well, however we will focus on the 1D case.

In general we may write a space group element as $\{A|\vec{\tau}\}$. Where $\vec{\tau}$ is a translation vector and $A \in P$. The operator acts on a point in space $\vec{r}$ as such:

$$\{A|\vec{t}\}\vec{r} = A\vec{r} + \vec{t} \tag{3.17}$$

Using this definition we may find the inverse of the space group element to be:

$$\{A|\vec{t}\}^{-1} = \{A^{-1}|A^{-1}\vec{t}\} \tag{3.18}$$

In code these space group elements are implemented as objects, with member functions for multiplication, inverse and exponentiation defined in code. Note that in general $\vec{\tau}$ need not be a translation element, however the pure translations must be a subgroup of the space group. However, if the group is symmorphic then $\vec{\tau} \in T$.

Since the 1D lattice is symmorphic, this property makes it simple for us to construct the set of all space group elements as we can iterate over the point group and the translation group elements. These operators are implemented in the code such that when they are applied to a z-basis state $|\uparrow\downarrow ... \uparrow\rangle$, they transform the state accordingly. The behaviour of the simple translation by the lattice constant, a (which we will from now set equal to 1), is shown in 3.1.4. The translation shifts a z-state into another z-state. This will be useful later in order to determine the fingerprint when the SG-operators act on the GSM.

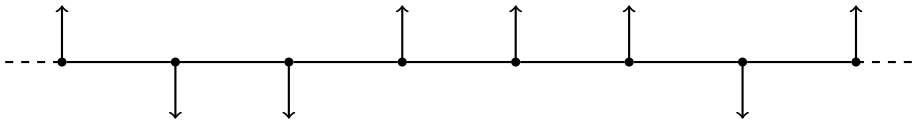


Figure 3.1.2: $|\phi\rangle$

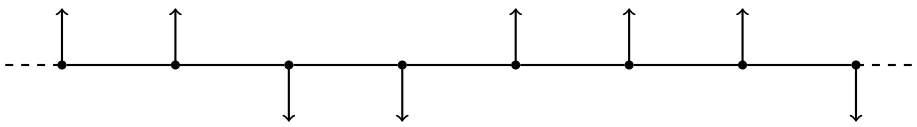


Figure 3.1.3: $\hat{T}\,|\phi\rangle$

Figure 3.1.4: Drawing showing how the translation operator by one ($\hat{T}$) transforms a z-basis state. Note that the index of the spin is the order in which they appear from left to right, starting at 0

The SG class is called `SpaceGroup`. The object has a member function `generate_irreps` which returns the irrep-data in a data class called `IrrepData`. The first step of the process is generating the conjugacy classes for the space group. This is done by the `generate_conjugacy_classes` member-function and it essentially partitions the space group into a list of conjugacy classes by determining which elements are conjugate to each other, according to the definition in 2.4.2. As an example throughout the process we will use the the space group for the 8-site chain lattice as an example.

The next step of the process involves determining how the k-vectors (wave vectors) transform under the point group. We will denote the set of k-vectors as K. The reciprocal lattice and the lattice always share the point group. The k-vectors in 1d can be written simply as

$$k_i = \frac{i2\pi}{N}, \tag{3.19}$$

where N is the number of lattice sites. In general we would also divide by a lattice constant a, however we set that equal to 1. This definition makes the k_i range from 0 to 2π. This makes numerics much simpler, however the conventional choice for the first Brillouin zone is $k \in [-\pi, \pi)$. We will use the more numerically favourable interval in this section. We may convert our value to the conventional broulin zone by subtracting by 2π for all $k > \pi$. Now we partition the set of k vectors by how they transform under the point group. This is done by the member function `partition_k_space`. We do this by calculating the "stars" of the wave vectors. The star of a wave vector k is defined as such:

$$\text{star}(k) = \{k' \in K | \exists A \in P,\ s.t:\ Ak = k'\}, \tag{3.20}$$

where equality is defined up to a reciprocal lattice vector G. For the 1D chain lattice the stars will either have one or two k vectors. Furthermore we are interested in systems with an even number of spins, as the odd will be frustrated systems, and may therefore not be dimerized until N gets very large. If we furthermore assume there are an even number of spins then we will get $2 + (N-2)/2$ stars. We prove this by acting with Π on a general k vector:

$$k' = \Pi k = -k \tag{3.21}$$

$-k$ is outside the range $[0, 2\pi]$ so we enforce periodicity by adding 2π. So the star will contain k (as it is equal to the identity operator acting on k) and k'. Generally these are not equal, so the star has two elements, however when $k = 0$ or $k = \pi$, we get $k' = k$. For an even number of sites $k = \pi$ will be a valid wave vector as $k_{(N/2)} = \pi$, and $N/2$ is a valid index when N is even. So we get one star for $k = 0$ and one for $k = \pi$. The other stars will have two elements $\text{star}(k) = \{k, -k\}$. This results in the reciprocal lattice being partitioned into $2 + (N-2)/2$ stars. In parallel with this we also calculate a set of unique point group elements which were needed in order to generate the star, which in code is called *unique_array*. For $k \notin \{0, \pi\}$ the we need both Π and E in order to generate the star, so the set of unique arrays will be $\{E, \Pi\}$, however for $k \in \{0, \pi\}$ we may choose the set to be $\{E\}$ or $\{\Pi\}$. So the array itself is not what is unique. It is rather the elements that each of the elements produces a unique k'-vector when acting on the k-vector used to generate the star.

The next step in calculating the irreps is iterating over the stars and its corresponding unique array. For each star we need to find the so called K-group of the star. This is the set of operators in the point group which leaves the elements of the star invariant up to addition of a reciprocal lattice vector G. We then need to find the irreducible representations associated with this star. This is one of the most tricky parts of the procedure, and would be difficult to generalize to space groups in 2D and 3D. The rest of the code is implemented such that if someone would like to continue working on our project, they would only need to implement this functionality. In 1D however, determining the little group and its irreps is trivial. The stars with 2 elements are only invariant under the identity. The identity has only one irrep, and it is simply 1. The stars with one element on the other hand are invariant under E, Π. This group has only two elements and is therefore isomorphic to $\mathbb{Z}_2$. $\mathbb{Z}_2$ has two irreps: Γ^1 and Γ^2, where:

$$\begin{aligned} \Gamma^1(E) = 1,\ \Gamma^1(\Pi) = 1 \\ \Gamma^2(E) = 1,\ \Gamma^2(\Pi) = -1 \end{aligned} \tag{3.22}$$

So as of now our function for getting the K-group and its irreps simply measures the number of elements in the star and returns either the data for $\mathbb{Z}_1$ or $\mathbb{Z}_2$ depending on if the number was two or one respectively.

The next step is constructing the irrep matrices. Suppose we partitioned the 1st Brillouin zone into m stars, where we cal the i-th star $Star_i$. We call its associated group K_i. There will be a number of irreps equal to the sum

$$\sum_i \text{len}(\text{Star}_i) \times \text{len}(\text{Irr}(K_i)), \tag{3.23}$$

where $\text{len}(\text{Star}_i)$ counts the number of elements in the star and $\text{len}(\text{Irr}(K_i))$ counts the number of irreps in K_i. This results in:

$$\text{len}(\text{Irr}(S)) = 4 + (N-2)/2 \tag{3.24}$$

So to get every irrep we iterate over every star and every irrep of the associated K-group. Suppose we pick a star denoted $\text{Star}(k)$ and a particular irrep of K denoted Γ. Then the resulting irrep of the space group $D(\{A|t\})$ may be written in terms of blocks:

$$D(\{A|t\}) = \begin{pmatrix} M_{11}(\{A|t\}) & \cdots & M_{1q}(\{A|t\}) \\ \vdots & \ddots & \vdots \\ M_{q1}(\{A|t\}) & \cdots & M_{qq}(\{A|t\}) \end{pmatrix} \tag{3.25}$$

Where q is the number of elements in $\text{star}(k)$. The dimension of the matrix blocks is the same as the irrep of K_i we chose. A property of the matrices $M_{ij}(\{A|t\})$ is that they will have an orthogonality relation. In $\text{Star}(k)$ we label each element k_l. Suppose we chose the i-th row of M. Then the following is true:

$$\begin{aligned} M_{ij}(\{A|t\}) = M_{im}(\{A|t\})\delta_{jm} \\ Ak_i = k_m \end{aligned} \tag{3.26}$$

So the column index in the non-vanishing block in the i-th row is determined by which vector in $\text{Star}(k)$ the point group element A sends k_i to. Finally the nonzero matrices M_{ij} are given by the following expression:

$$\begin{aligned} M_{ij}(\{A|t\}) &= D(B,\sigma) \\ D(B,\sigma) &= exp(ik\cdot\sigma)\Gamma(B), \end{aligned} \tag{3.27}$$

where $\{B|\sigma\}$ is another SG-element, k is the first element of the star(k) (and the vector we used to generate it) and Γ is the irrep of K we chose earlier, and is thus also a matrix. Finally we need a way of relating the space group elements $\{A|t\}$ and $\{B|\sigma\}$. For symmorphic space groups, $\{B|\sigma\}$ is given by

$$\{B,\sigma\} = \{A_i|0\}^{-1}\{A|t\}\{A_j|0\}, \tag{3.28}$$

where A_i and A_j are the i-th and j-th element in `unique_array`. Note that i and j are the same indices as in M_{ij}. $\{B,\sigma\}$ is therefore different for each block.
As mentioned, the the full justification for this method is given in [14]. With this we are now able derive the space group for the 1D lattice of any length. Furthermore, given a method in the code for deriving the irreps of a little group K given only K, the code should be able to handle general symmorphic space groups.

3.1.4 Calculating characters for J-J' model

Assuming we have numerically calculated the eigenstates of a J-J' Hamiltonian, and deduced which states belong to the GSM. Furthermore we assume we have managed to calculate the irreps of the space group. The irreducible representation decomposition is outlined in 3.1.3. This is implemented numerically in `Irrep_decomposition_method.py`. The function `get_fingerprint` takes in the GSM as a list and the irrep-data as the `IrrepData` class, which contains the irreps, characters and conjugacy classes of the space group. The function returns a `np.ndarray` of length equal to the number of irreps, where the i-th element of this array corresponds to the number of times the i-th irrep occurs in the GSM representation.

3.2 Analytical derivation of fingerprints

For the TC and DS models we are able to derive the ground states analytically. However deriving their symmetry groups (or specifically the subgroup of the symmetry group which will have broken symmetries) is less straight forward than for the space groups. As a note the space group is less relevant for the TC and DC model. This is because we can remove translational symmetry for both these systems without it affecting the ground states. For the TC system we can do this by redefining the Hamiltonian as:

$$\mathcal{H}_{TC,\text{no translation}} = -\sum_p J_p B_p - \sum_s J_s A_s, \tag{3.29}$$

where J_s and J_p are positive real numbers. This results in an a system without translational symmetry, but with the same ground states. The groups generated by B_p or A_s operators will not have broken symmetries either, as all the ground states are $+1$ eigenstates of these operators. This leads us to investigating the U and V loop operators instead. For the TC, TCL, DS and DSL system the following

method will be used to identify precisely what group will be generated by their respective U and V operators.

3.2.1 Determine a set of operators which should be in the symmetry group

This is generaly a highly nontrivial task, however for both the TC and DS system these relevant operators are known. For the TC model we need to form a group with the U_x, U_y, V_x and V_y operators. For the DC model we need a group which contains U_x^+, U_y^+, U_x^-, U_y^-, V_x and V_y. For the 1D ladder versions of these systems we will need to investigate which of the operators survive when we remove the periodicity in the y-direction. It will turn out that for all these systems we will be able to analytically derive the fingerprints.

3.2.2 Generate a closed set with these elements

From 2.4 we know products must be closed under the group operations in order to be a group. We can force this to be true if we start with elements which generate the group. We can always have more than the minimum number of generators for a group, so a nice starting point would be to use the operators we know should be in the group and then exponentiate them and multiply them amongst each other as shown in the following equation algorithm:

Let $\mathcal{G} = \{g_1, \ , g_2, \ g_3, ...g_n\}$ Be as set of generators for group G. Let n_i be the order of the generator g_i. Then $G = \{g | g = g_i^l g_j^m, \ 0 < m < n_m, \ 0 < m < n_m \quad g_i, \ g_j \in \mathcal{G}\}$

Note that a single element of G might be written several ways in this approach. The equations may be slightly simplified if it turns out that every one of the generators are independent in the sense that they cannot be generated from any of the other generators.

We can now prove analytically if what we derived is a group. We know the group will be closed due to the last step, identity E will exist as any generator to the power of its order will be E, and associativity follows from the fact that subsequent application of operators is associative. Lastly the inverse must exist as exponentiating any group element to its order results in the element becoming E.

3.2.3 Derive multiplication rules for the group

In order to derive properties of the group it is helpful to require all group elements to be in some standard form/order and then determining how the product of two elements behave if we enforce that ordering on it.

3.2.4 Determine which tabulated group the derived group is isomorphic to

We will derive which tabulated group the group we derived is isomorphic to. The TC and DS groups are of finite order, and it is the case that for any finite order there exists only a finite number of non-isomorphic groups. In order to derive which specific one our group corresponds to we can start out with a list of every group that has the same order. Then we may calculate certain properties of our group which is conserved under an isomorphism. This may for instance be the number of conjugacy classes, the number of elements of every order, if the group is abelian or not, and so on. The larger the order the more of these properties we will need in order to single out a specific group. We may then filter out all the tabulated groups which do not have the same properties as our derived group. When we are left with only one we will have found the correct tabulated group.

3.2.5 Investigate properties of the tabulated group

We will want to find the irreducible representations or characters of the tabulated group. In the software we will use for the filtering, GAP, it is possible to see the character of any of the groups it has listed.

Furthermore, when the GAP group id is known, which we will find from the filtering, we can use it to determine additional properties of the group, either through GAP directly, or by searching databases listing the properties of groups.

3.2.6 Derive the GSM fingerprint

The method for deriving the fingerprint is the same that we use numerically, but with two significant differences. In order to calculate the fingerprint we need to know how the operators in the groups operate on the ground states. When this is determined we may continue as usual. For these systems it will turn out that we need not use the character formula given in Eq. 2.39 in order to determine which irreps are present in the GSM representation. If we can show that the GSM itself is an irrep, where a sufficient condition is that the characters for the GSM matches exactly the characters of one of the irreps of the tabulated group which the GSM is isomorphic to, then the fingerprint will simply be the integer array N, where the elements are written as $N_i = \delta_{i\alpha}$ where Γ^α is the irrep with identical characters as the GSM representation.

CHAPTER

FOUR

RESULTS

4.1 Numerical results

4.1.1 Testing the numerical $|S, S_z; i\rangle$ basis generation

Before using the $|S, S_z; i\rangle$ basis, we should first validate that it has a few of the properties we expect. We should also verify that it reproduces some of the bases which we are able to calculate by hand. A simple first test is to verify that all the basis states in a few spin bases with different numbers of N all have a norm of 1. We are not manually setting the norm of the basis states equal to one, so all the states having a norm of 1 would be a massive coincidence if it turned out that the procedure was faulty. The function `normtest()` was implemented to test this and the result was within tolerance. A even a basis of 14 spins had a norm that was approximately equal to 1.

Next we may test if `get_spin_system(2)` produces the $|S, S_z\rangle$ states (i-index supressed as the states are unique). These states may be written in terms of the z-basis as:

$$\begin{aligned} |S=1, S_z=1\rangle &= |\uparrow\uparrow\rangle \\ |S=1, S_z=0\rangle &= \frac{1}{\sqrt{2}}(|\uparrow\downarrow\rangle + |\downarrow\uparrow\rangle) \\ |S=1, S_z=-1\rangle &= |\downarrow\downarrow\rangle \\ |S=0, S_z=0\rangle &= \frac{1}{\sqrt{2}}(|\uparrow\downarrow\rangle - |\downarrow\uparrow\rangle) \end{aligned} \tag{4.1}$$

Running the function `get_spin_system(2)`, and printing the result with `print_spin_system()`, we get the following result:

```
|S=0.0, Sz=-0.0; i=0> [-0.70710678  0.70710678]
|S=1.0, Sz=-1.0; i=0> [1.]
|S=1.0, Sz=0.0; i=0> [0.70710678 0.70710678]
|S=1.0, Sz=1.0; i=0> [1.]
```

This result is as expected, as $0.70710678 \approx \frac{1}{\sqrt{2}}$. Note that the right-hand side is $\vec{\phi}^{S,S_z,i}$, as defined in equation 3.7. In this case the function did not need to deal

with multiple states with the same quantum numbers. We may verify that the function system correctly handles this by deriving the degenerate states of the 3-spin basis. The numerical result for the 3-spin basis is the following:

```
|S=0.5, Sz=-0.5; i=0> [ 0.         -0.70710678  0.70710678]
|S=0.5, Sz=-0.5; i=1> [-0.81649658  0.40824829  0.40824829]
|S=0.5, Sz=0.5; i=0> [-0.70710678  0.70710678  0.        ]
|S=0.5, Sz=0.5; i=1> [-0.40824829 -0.40824829  0.81649658]
|S=1.5, Sz=-1.5; i=0> [1.]
|S=1.5, Sz=-0.5; i=0> [0.57735027 0.57735027 0.57735027]
|S=1.5, Sz=0.5; i=0> [0.57735027 0.57735027 0.57735027]
|S=1.5, Sz=1.5; i=0> [1.]
```

The $S = \frac{3}{2}$ states are unique, so we may focus on the $S = \frac{1}{2}$ states instead. $i = 0$ corresponds to the path from $S = \frac{1}{2}$ to $S = 0$ to $S = \frac{1}{2}$, and has the associated boolean number $B_0 = 01$. $i = 1$ corresponds to the path from $S = \frac{1}{2}$ to $S = 1$ to $S = \frac{1}{2}$, and has the associated boolean number $B_1 = 10$. As expected $B_0 < B_1$. There is only one 2-spin state with $S = 0$. This implies that the $i = 1$ states may be written as:

$$\begin{aligned}
|S = \frac{1}{2}, S_z = \frac{1}{2}; 0\rangle_3 &= \mathrm{CG}(\frac{1}{2}, \frac{1}{2}, 0, 0, \frac{1}{2}, \frac{1}{2}) \, |S = 0, S_z = 0\rangle_2 \times |\uparrow\rangle \\
&= 1\frac{1}{\sqrt{2}}(|\uparrow\downarrow\uparrow\rangle - |\downarrow\uparrow\uparrow\rangle) \\
|S = \frac{1}{2}, S_z = -\frac{1}{2}; 0\rangle_3 &= \mathrm{CG}(\frac{1}{2}, -\frac{1}{2}, 0, 0, \frac{1}{2}, -\frac{1}{2}) \, |S = 0, S_z = 0\rangle_2 \times |\downarrow\rangle \\
&= 1\frac{1}{\sqrt{2}}(|\uparrow\downarrow\downarrow\rangle - |\downarrow\uparrow\downarrow\rangle),
\end{aligned} \tag{4.2}$$

where $\mathrm{CG}(J, M, j, m, j', m')$ is the Clebsch-Gordan coefficient. Note that to differentiate spin bases with a different number of spins we include a subscript N to the states indicating the number of spins. Comparing this to the numeric results we find that the coeffients match. Now we may tackle the $i = 1$ states. In this case, due to the selection rules for the CG-coefficents we find that each state has two contributions:

$$\begin{aligned}
|S = \frac{1}{2}, S_z = \frac{1}{2}; 1\rangle_3 &= \mathrm{CG}(\frac{1}{2}, \frac{1}{2}, 1, 0, \frac{1}{2}, \frac{1}{2}) \, |S = 1, S_z = 0\rangle_2 \times |\uparrow\rangle \\
&\quad + \mathrm{CG}(\frac{1}{2}, \frac{1}{2}, 1, 1, \frac{1}{2}, \frac{1}{2}) \, |S = 1, S_z = 1\rangle_2 \times |\downarrow\rangle \\
&\approx -0.408(|\uparrow\downarrow\uparrow\rangle + |\downarrow\uparrow\uparrow\rangle) + 0.816 \, |\uparrow\uparrow\downarrow\rangle \\
|S = \frac{1}{2}, S_z = -\frac{1}{2}; 1\rangle_3 &= \mathrm{CG}(\frac{1}{2}, -\frac{1}{2}, 1, 0, \frac{1}{2}, -\frac{1}{2}) \, |S = 0, S_z = 0\rangle_2 \times |\downarrow\rangle \\
&\quad + \mathrm{CG}(\frac{1}{2}, \frac{1}{2}, 1, -1, \frac{1}{2}, \frac{1}{2}) \, |S = 1, S_z = -1\rangle_2 \times |\uparrow\rangle \\
&\approx 0.408(|\uparrow\downarrow\downarrow\rangle + |\downarrow\uparrow\downarrow\rangle) - 0.816 \, |\downarrow\downarrow\uparrow\rangle
\end{aligned} \tag{4.3}$$

This result corresponds to the numerical data, thus there is strong evidence that the function handles the multiple states with the same quantum numbers correctly.

We will get more evidence that the function creates the correct basis later when we use it to create a block diagonal matrix for the Majumdar-Gosh model, and correctly get an exact degeneracy in the ground state, as theoretically predicted.

Finally we should test the performance of the `get_spin_system` function, as it will give us an upper bound to which systems are feasible to work with. The exact times will vary depending on the computer, however the general growth should be similar. As the size of the Hilbert space is exponential in the number of spins, we would expect this function to be the same. The result when run on our computer was the following:

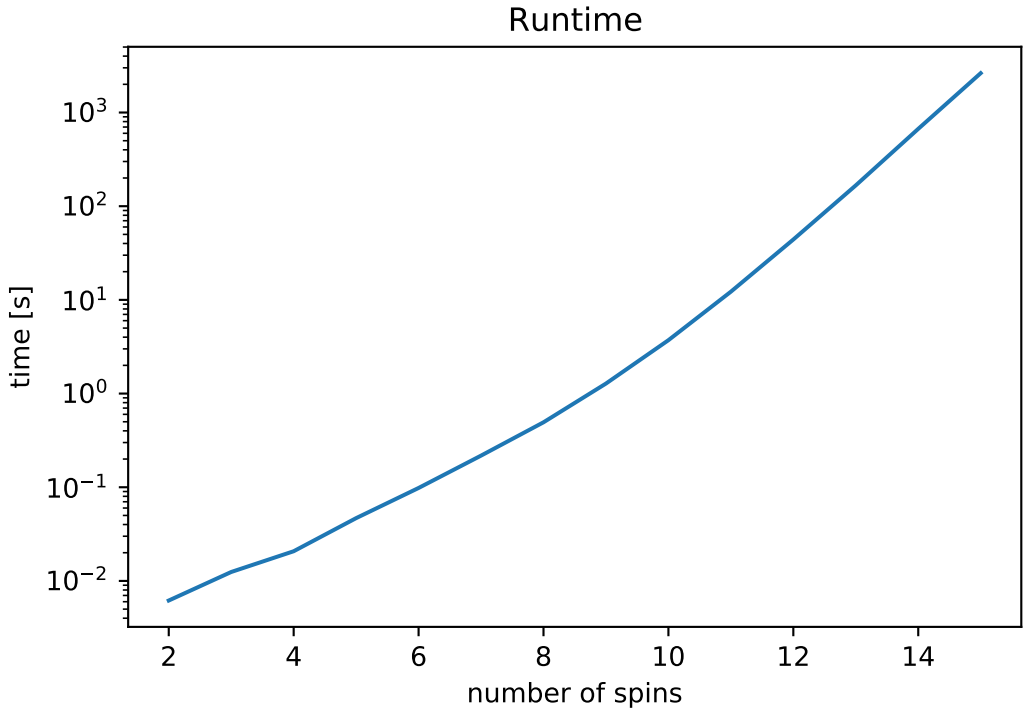


Figure 4.1.1: Runtime for `get_spin_system` with respect to the number of spins

As expected the runtime is exponential in N. Doing a regression of the logarithm of the data (excluding the first few data points to capture the large N behaviour) lets us predict the runtime of `get_spin_system` given N. For the computer used in this project the prediction function is

$$\text{Runtime}(N) \approx exp(1.37N - 12.61) \text{ seconds}, \tag{4.4}$$

which correspond to a straight line when plotted on logarithmic y-axis, as the plot in 4.1.1. This predicted the runtime for generating a 16 spin basis to be about 2.85 hours, which was fairly accurate. The theoretical times for a 18 and 20 spin basis would then be about two days and a month respectively. However, as implemented the computer would run out of memory long before that. This function is by far the least efficient part of the project, and is probably the part which would benefit the most by being converted to a compiled language, like `C++`. However due to the fact that we can save the data from the $|S, S_z; i\rangle$ basis this problem was not too detrimental. As a side-note the 16 spin system was the largest basis we generated

as part of the project, and due to the computer consistently crashing in the middle of generating matrices for the this basis (most likely a memory issue), it was never actually used in the end.

4.1.2 Matrix generation and diagonalization

As a test of the block diagonalization and diagonalization we check if the numerical diagonalization of the MG-model gives an exactly degenerate eigenstate as discussed in 2.2.1. We may either recalculate the $|S, S_z; i\rangle$ basis, or import one of the precalculated bases in the project file. Using this basis we can calculate the matrices for the nearest- and second nearest neighbour term with the `generate_hamiltonian_matrix` function, which takes in the number of spins, the spin basis and a Hamiltonian operator. The Hamiltonian operator is implemented such that it takes in a S_z quantum number and returns the $\mathbf{H^{S_z}}$ matrix, as defined in equation 3.11. To get the $\mathbf{H_0^{S,S_z}}$ matrices we input `H_heisenberg_chain` and to get $\mathbf{H_1^{S,S_z}}$ matrices we input `H_second_nearest`. `generate_hammiltonian_matrix` returns the matrix blocks in a nested dict, where `output`$[S][S_z] = \mathbf{H^{S,S_z}}$. To scale the dictionary of the $\mathbf{H_1^{S,S_z}}$ matrices by $\frac{1}{2}$ we use `scalar_mul_H_matrices`, and to add the two dictionaries use `add_H_matrices`. We finally use `fulldiag` to diagonalize the individual matrices, which runs `scipy.eigh` on all the individual blocks in parallel. The data from the diagonalization is stored in a new dictionary. To get get the full spectrum in a list we can use `get_fullspectrum_sorted`.

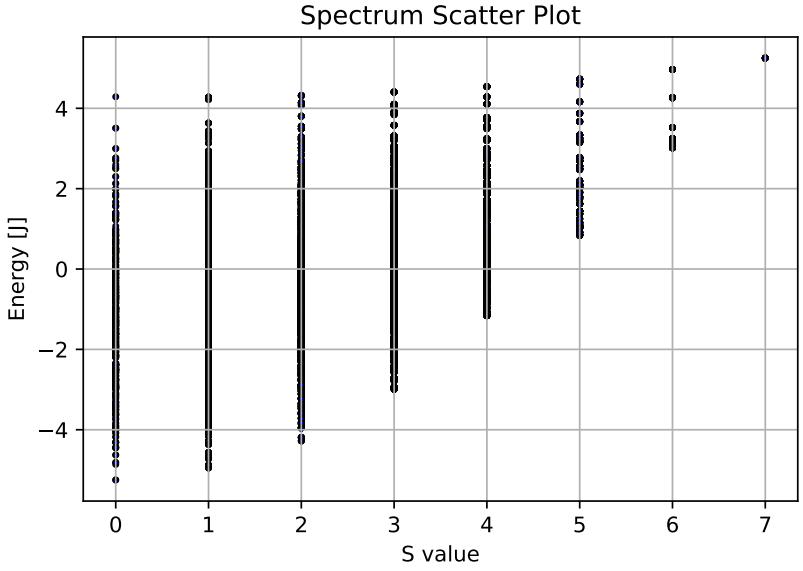


Figure 4.1.2: Spectrum of the MG-model with 14 spins

The result for the MG-model with 14 spins is that the two lowest energy eigenstates are exactly degenerate, with an energy eigenvalue of -5.25. Furthermore there is a gap $\Delta = 0.29925$ between the ground states and the first exited state. Using a scatter-plot we can show the spectrum for the MG model. In order to visualize the data more clearly we can plot the energy of each state against the S quantum number. We expect the lowest energy eigenstates to have $S = 0$ as the ground states of the MG model are dimerized. Looking at figure 4.1.2 we can see that

the ground state is indeed a $S = 0$ state, and from inspecting the spectrum data earlier we know there should be two states at that single point. Looking closely one can also see that the first exited state is a $S = 1$ state.

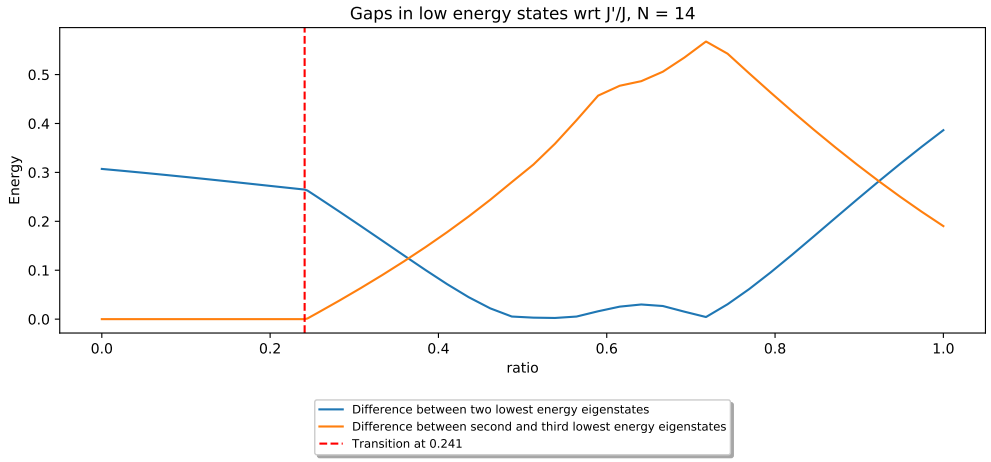
4.1.3 Spectrum analysis

We want to determine the fingerprint for the J-J'-model when it is in a dimer quantum phase. As mentioned in 2.2.1, the the J-J' model has two phases for $J'/J \in [0, 1]$, those being the quantum critical phase, when $J'/J < J_c$, and the dimer phase when $J' > J_c/J$ [3].

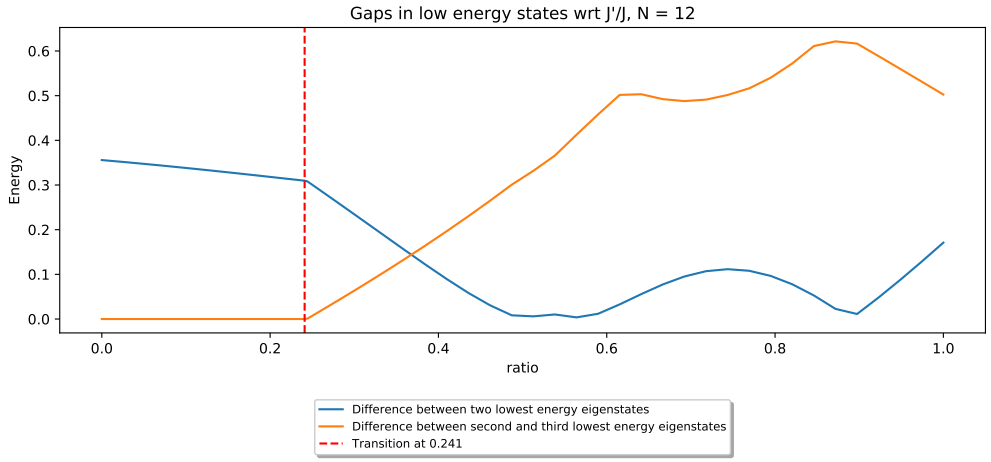
Determining which states in a finite model should be a part of the GSM or not is generally a nontrivial task. They are generally not degenerate, and we only say they are a part of the GSM in the sense that in the thermodynamic limit, the states become degenerate. So in order to determine when the system is in the dimer phase we will make the use of spectrum analysis similar to [25], and our knowledge about the system. We start with our knowledge about the system. When $J' = 0$ we simply have the Heisenberg chain model. This is known to be quantum critical, where the lowest energy eigenstate is a $S = 0, S_z = 0$ state, however the spectrum is continuous, and the first exited states are not $S = 0, S_z = 0$ eigenstates. We can reproduce this result numerically. When we diagonalize the Heisenberg chain model with 12 spins ($J = 1$,$J' = 0$), we find that the lowest energy eigenstate indeed is a $S = 0, S_z = 0$ eigenstate, whilst the next energy eigenstate is not.

At $J' = 0.5J$ we have the Majumdar-Gosh model. As mentioned in 2.2.1, the ground states of this model can be analytically derived, and are known to be exactly degenerate (for an even number of spins). The ground states are both $S = 0, S_z = 0$ eigenstates, and is in a dimer phase. So we know the phase at two points in the phase space. For some value for J'/J we must therefore have a phase transition. In order to find it we will investigate the three lowest energy eigenstates in the spectrum. In Fig. 4.1.3 we show the energy difference between the ground state and the second lowest energy state, and the difference between the second lowest energy and the third both with respect to J'/J. This was done for both 12 and 14 spins. The critical value J_c/J is marked in the figure by a red vertical line. We can see that the behaviour before and after this point is very different. Before the critical value there is no splitting between the second and third lowest energy state. From closer spectrum analysis this shows that the first exited states are highly degenerate, so the second lowest energy eigenstate has the same energy as the third. After the critical value however, it is instead another $S = 0, S_z = 0$ eigenstate which is the second lowest energy eigenstate. This lets us define a criterion for when the system is in a dimer state. Whenever the two lowest energy eigenstates are $S = 0, S_z = 0$ states in a finite size system, we assume them to become degenerate in the thermodynamic limit, and form a dimer phase.

Using this criterion lets us determine the critical value of a J-J'-model for a given N to high precision. To do this we use a bisection algorithm. We have an interval for J': $[0, 0.5J]$, where we know the system is in a dimer phase for $J' = 0.5J$, and we know it is not for $J' = 0$. Generally we want to have an interval $[a, b]$ where $J' = a$ is not a dimer phase, but $J' = b$ is. We then bisect the interval $[a, b]$ and diagonalize the system associated with with $J' = c = \frac{a+b}{2}$. If the point



(a) N=14 Spin System



(b) N=12 Spin System

Figure 4.1.3: Comparison of low energy splitting for J-J' model with respect to J'/J ratio

c is a dimer phase, then the phase transition is in the interval $[a, c]$. If it is not a dimer phase, then the phase transition is in the interval $[c, b]$. This procedure may be repeated as many times as one wishes (until the interval is as short as the numerical precision of the computer). Each iteration we halve the interval, so the algorithm rapidly converges to the critical value. Note that this assumes phase transition between two phases. Using this approach we can determine the phase transition with respect to the number of spins. The results of these calculations are shown in Fig. 4.1.4.

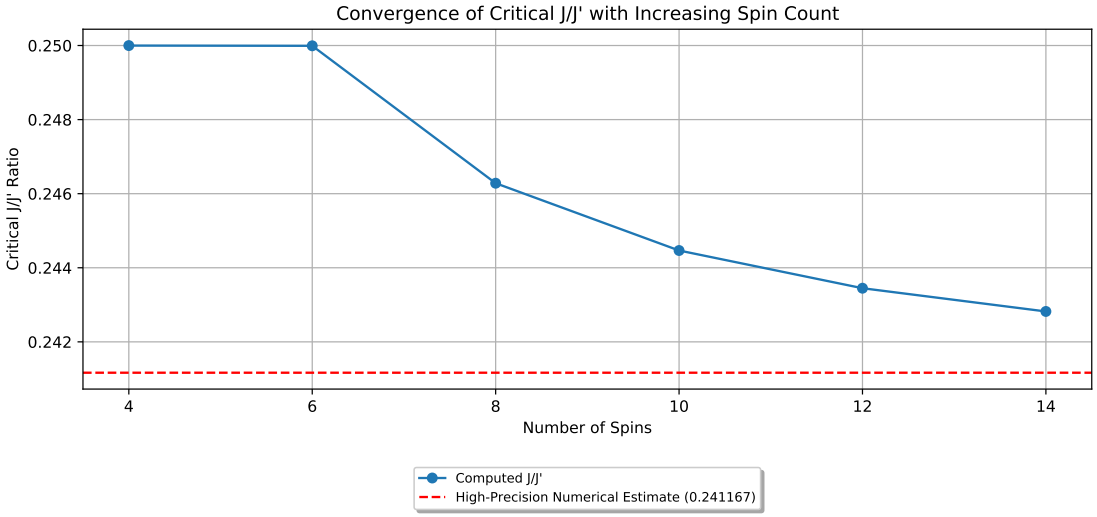


Figure 4.1.4: Estimation of critical value from diagonalization with respect to the number of spins

Our calculation approaches the high precision numerical estimate and is off by only $0.02J$. Considering we only used 14 spins this result is quite satisfying. With this result we will from now on assume that for the systems with $J'/J > J_c$ will have its two lowest energy eigenstates as its GSM.

4.1.4 Space group of the J-J' model

Using the method in section 3.1.3, we calculate the irreps of the space groups. Before we use these characters we may apply the method and compare the result to a known group. In general the finite 1D space group with reflection symmetry and n lattice sites will be isomorphic to the dihedral group D_{2n} [18]. We may therefore test the functions for deriving the space group irreps by comparing the character table of a small space group, for example a 4-spin chain, against the character table for D_8 in the GAP software. The numerical results for this derivation is shown in section 4.1.1. Comparing the results with the listed character table for D_8 we find that they are identical. Having verified that the function is working as intended, we may now use it to derive the characters for the J-J'-model in the dimer phase.

Irrep	$\{E\|0\}$	$\{E\|1\},\{E\|3\}$	$\{E\|2\}$	$\{\Pi\|0\},\{\Pi\|2\}$	$\{\Pi\|1\},\{\Pi\|3\}$
Irrep 1	1	1	1	1	1
Irrep 2	1	1	1	-1	-1
Irrep 3	1	-1	1	1	-1
Irrep 4	1	-1	1	-1	1
Irrep 5	2	0	-2	0	0

Table 4.1.1: Numerically calculated character table for the group D_8

4.1.5 Numerical fingerprint and analytical fingerprint for the MG model

We will now investigate the fingerprint of the J-J' model with 8 spins for a few different values of J' We may start with the MG-point, where $J' = 0.5$. Running the program `Irrep_decomposition_method.py` with $J = 0.5$ and $N = 8$ yields the fingerprint:

$$n_\alpha = [1, 0, 0, 1, 0, 0, 0] \tag{4.5}$$

So the first irrep, which is the trivial irrep, and the fourth, which is another 1D irrep, appears in the decomposition. Running the same program for $N = 12$ yields a similar fingerprint:

$$n_\alpha = [1, 0, 0, 1, 0, 0, 0, 0, 0] \tag{4.6}$$

Here too we get two 1D irreps, where one belongs to the trivial irrep and the other belongs to the fourth. We also see a pattern in the number of irreps of 1D and 2D. There will always be four 1D irreps. We found an equation for the number of irreps which is given in 3.23. The star of the $k = 0$ and $k = \pi$ vector has corresponding K-groups which are isomorphic to $\mathbf{Z}_2$, whilst the rest of the stars have K groups isomorphic to $\mathbf{Z}_1$. In both cases these groups have only 1D irreps. The dimension of the irreps depended on the dimension of the K group used to construct it and the number of elements in the star of k. The stars of $k = 0$ and $k = \pi$ have only one element each and produces therefore 1D irreps. The stars of the other k vectors on the other hand have two elements, which implies that the corresponding irreps will be two-dimensional. So the irreps that appear in the fingerprint of the $N = 8$ and $N = 12$ J-J' models at the Majumdar-Ghosh point is the trivial irrep and the irrep corresponding to the irrep of the space group constructed with $k = \pi$ and the second irrep of the K-group (equal to the point group), which transforms the point group as such:

$$\Gamma^2(E) = 1,\ \Gamma^2(\Pi) = -1 \tag{4.7}$$

Before we relate this result to the fingerprint of the analytically derived ground states of the MG-point we will investigate the fingerprint of the $N = 12$ J-J' model for a few different values of J'. Firstly we measure at $J' = 0.45J$. This could be interpreted as a small perturbation away from the MG-point. We then do the same for $J' = 0.35J$, and for $J' = 0.25$ where the system has just transitioned to the dimer phase. As a final test we attempt to measure the fingerprint at $J' = 0$. We do this by extracting the two lowest energy $S = 0, S_z = 0$ states. From the

spectrum analysis we know the system is not in the dimer phase at that point, however the exercise will turn out to be insightful. The resulting fingerprint for all these cases is:

$$n_\alpha = [1, 0, 0, 1, 0, 0, 0, 0, 0] \tag{4.8}$$

At first glance this result might seem strange, as the fingerprint seems to be the same for both phases, however the reason for this behaviour will turn out to be quite simple. It does however illustrate the importance of doing the spectrum analysis. If we simply picked the two lowest energy $S = 0$ $S_z = 0$ without doing so, one would come to the wrong conclusion that the fingerprint is the same for the quantum-critical phase and the dimer phase. For the $J' = 0$ model we picked a state to be in the GSM which should not be there. In order to investigate why we get this result we may attempt to calculate the characters for the two states separately. We are effectively making an ansatz that the ground states will be eigenstates of the space-group operators. This is generally a good guess, as the states have slightly different energy eigenvalues for most values of J' and are therefore non-degenerate. In those cases if an operator commutes with the operators we used to construct the basis (in this case that would be $\hat{S}^2$ and $\hat{S}_z$) then the resulting state will also be an eigenstate of that operator. This is generally not true when we have degeneracy, as we may simply create new eigenstates from the span of the old eigenstates which will not have this property.

Using this ansatz to calculate the fingerprint of the two lowest energy $S = 0, S_z = 0$ eigenstate we determine that with the exception of at the MG-point, the lowest energy eigenstate was automatically an eigenstate of the space group and belonged to the trivial irrep, whilst the second lowest energy $S = 0, S_z = 0$ eigenstate transformed under the SG like the fourth irrep. We may further test other values of J' by diagonalizing a set of systems with J'/J in the range $[0, 1]$. We pick out the lowest energy $S = 0, S_z = 0$ eigenstate in the spectrum and determine whether it transforms like the first or fourth irrep of the SG. We do the same for the second lowest energy $S = 0, S_z = 0$ state. The result of which is shown in Fig. 4.1.5. As predicted the two lowest energy $S = 0, S_z = 0$ eigenstates are indeed automatically eigenstates of the space group, the only exception for which is the MG-model. We could also construct those states for the MG-model, however it would not automatically be the case. We see that right around the MG-point the eigenstate of the fourth irrep briefly becomes the lowest energy eigenstate.

Lastly we will visualize how the ground states of the J-J'-model in the dimer phase transform under the SG transformations. We have already shown that the GSM has the same fingerprint in a general point in the dimer phase as in the GSM, however the real strength of the fingerprinting method is that we may relate the results of the numerical diagonalization to simpler pictures which do not exactly resemble the ground state, however the simplified picture will transform under the SG like the numerically calculated ground states. In order to do so we start with the ground states of the MG-model which are analytically derivable. These are visualized in figure 2.2.2. Using the figure we may apply the translation operator $\{E|a\}$, which shifts the lattice by one times the lattice constant. This transformation is visualized in fig 4.1.6.
Looking at the figure it is clear that the states break translational symmetry, as translating the lattice by $r = 1$ transforms the first state into the second and vice

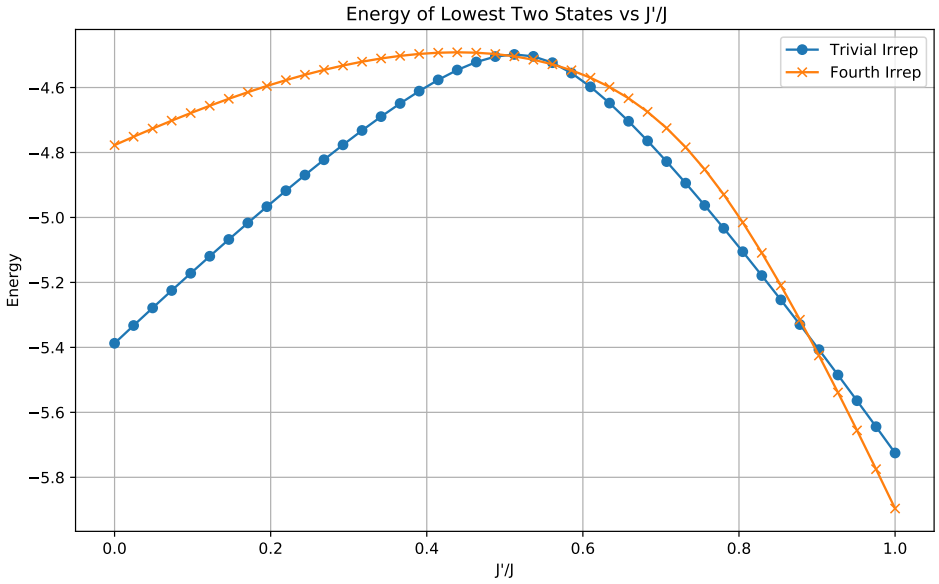
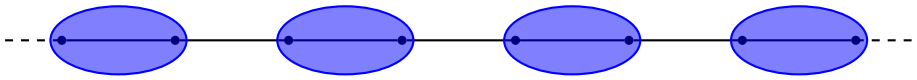
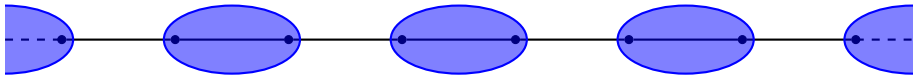


Figure 4.1.5: Energy spectrum of the two lowest energy $S = 0$, $S_z = 0$ states



(a) $|\phi_1\rangle$



(b) $\{E|a\}\,|\phi_1\rangle = |\phi_2\rangle$

Figure 4.1.6: Applying a shift by the lattice constant a to the state $|\phi_1\rangle$.

versa. This may be extended to

$$\begin{aligned} \{E, r\}\,|\phi_1\rangle &= |\phi_2\rangle\,, \\ \{E, r\}\,|\phi_2\rangle &= |\phi_1\rangle\,, \end{aligned} \tag{4.9}$$

where $r = N$, and N is odd. Operating on one ground state with the point group element Π will also transform the state into the other. In general a space group element $\{A|Na\}$ will switch between the two states if N is odd and $A = E$, or N is even and $A = \Pi$. Otherwise it will act as the identity operator. With this we could derive the GSM representation for the space group. One potential complication however is that these states are in general not orthogonal for a finite system. In the thermodynamic limit the overlap between the states goes to zero, however for a finite number of sites there is some overlap between the states. We want an orthonormal basis for the representation, and furthermore converting the states $|\phi_1\rangle$ and $|\phi_2\rangle$ into eigenstates of the space group would be desirable.

$$
\begin{aligned}
|\psi_1\rangle &= \frac{1}{\sqrt{2}}(|\phi_1\rangle + |\phi_2\rangle) \\
|\psi_2\rangle &= \frac{1}{\sqrt{2}}(|\phi_1\rangle - |\phi_2\rangle)
\end{aligned}
\tag{4.10}
$$

Let $\hat{A}$ be one of the SG-elements which we determined would not switch between the $|\phi_1\rangle$ and the $|\phi_2\rangle$ state and let $\hat{B}$ be one of the SG-elements that do. We operate with these operators on $|\psi_1\rangle$ and $|\psi_2\rangle$:

$$
\begin{aligned}
\hat{A}\,|\psi_1\rangle &= \frac{1}{\sqrt{2}}(\hat{A}\,|\phi_1\rangle + |\hat{A}\phi_2\rangle) \\
&= \frac{1}{\sqrt{2}}(|\phi_1\rangle + |\phi_2\rangle) = +1\,|\psi_1\rangle \\
\hat{B}\,|\psi_1\rangle &= \frac{1}{\sqrt{2}}(\hat{B}\,|\phi_1\rangle + \hat{B}\,|\phi_2\rangle) \\
&= \frac{1}{\sqrt{2}}(|\phi_2\rangle + |\phi_1\rangle) = +1\,|\psi_1\rangle \\
\hat{A}\,|\psi_2\rangle &= \frac{1}{\sqrt{2}}(\hat{A}\,|\phi_1\rangle - |\hat{A}\phi_2\rangle) \\
&= \frac{1}{\sqrt{2}}(|\phi_1\rangle - |\phi_2\rangle) = +1\,|\psi_2\rangle \\
\hat{B}\,|\psi_1\rangle &= \frac{1}{\sqrt{2}}(\hat{B}\,|\phi_1\rangle - \hat{B}\,|\phi_2\rangle) \\
&= \frac{1}{\sqrt{2}}(|\phi_1\rangle - |\phi_2\rangle) = -1\,|\psi_2\rangle\,.
\end{aligned}
\tag{4.11}
$$

From this we may conclude that the first ground state $|\phi_1\rangle$ transforms under the SG-transformations like the trivial irrep. Proving that the second ground states belongs to the fourth 1D irrep is more tricky. Recall that the fourth 1D irrep of a 1D space group with an even number of lattice sites was constructed from the star of $k = \pi$, and the second irrep of the associated little group $K = \{E, \Pi\}$. We may therefore write this irrep as:

$$
\begin{aligned}
M(\{A|t\}) &= D(B, \sigma) = D(A, t) \\
&= \exp(i\pi \cdot t)\Gamma(A),
\end{aligned}
\tag{4.12}
$$

This equation is a simplification of 3.27, where we exploit that the irrep will only contain one block, as the star of $k = \pi$ only has one element, and that $\{A|t\} = \{B|\sigma\}$ as A_i and A_j in equation 3.28 are both the identity.

$$
\exp(i\pi \cdot t) = \begin{cases} -1 & \text{when } t \text{ is odd,} \\ 1 & \text{when } t \text{ is even.} \end{cases}
\tag{4.13}
$$

$$
\Gamma(A) = \begin{cases} 1 & \text{when } A = E, \\ -1 & \text{when } A = \Pi \end{cases}
\tag{4.14}
$$

Combining the two equations we get:

$$M(\{A|t\}) = \begin{cases} -1 & \text{when } A = E \text{ and } t \text{ is odd} \\ -1 & \text{when } A = \Pi \text{ and } t \text{ is even} \\ 1 & \text{Otherwise} \end{cases} \tag{4.15}$$

This implies that the $|\phi_2\rangle$ state belongs to the irrep of the space group constructed by the star of $k = \pi$, and the second irrep of its associated little group $K = \{E, \Pi\}$. This fingerprint matches the numerical data for the J-J' model in dimer phase (for 12 and 8 spins), as we found that the representation of those GSM's contained one instance of the trivial irrep and one instance of the irrep generated by $k = \pi$, and the second irrep of its associated little group $K = \{E, \Pi\}$ just like the analytical results for the MG GSM. Thus the MG-ground states may be used to picture the behaviour of a the ground states of all the GSM's for the J-J' model in the dimer phase, even though they are exact eigenstates of only the MG point.

4.2 Analytical results

4.2.1 TC model

4.2.1.1 The UV-group

The equations for the U-operators and V-operators are given in equations 2.13 and 2.14 respectively. These operators have the property that when U_x and V_y, or U_y and V_x are moved past each other, they pick up a -1 phase shift. Therefore, in addition to all our other elements, we will need to include a -1 phase shift as a group operator. We use this to define a group element as follows

$$\begin{aligned} G_{UV} &= \{g = (b_0, b_1, b_2, b_3, b_4) \mid \forall i,\ b_i \text{ are booleans}\} \\ U_g &= (-1)^{b_0} U_x^{b_1} U_y^{b_2} V_x^{b_3} V_y^{b_4} \end{aligned} \tag{4.16}$$

We write an abstract group element as an array of five booleans and associate this abstract group element with its operator in the GSM vector space, which is given by U_g. In this notation, exponentiating the group elements to the zeroth order yields the identity element, which is 1. Note that a standard ordering of the operators must be chosen. This order can be arbitrary, but it must be consistent to differentiate the anticommuting elements. We now need to define the binary group operator. We do so by setting up two group elements in the U_g form and moving the operators that are in the wrong order to the standard form. This is shown in equation 4.17. Note that $\oplus$ signifies the binary XOR operator, and & signifies the binary AND operator.

$$\begin{aligned} U_g * U_{g'} &= (-1)^{b_0} U_x^{b_1} U_y^{b_2} V_x^{b_3} V_y^{b_4} (-1)^{\beta_0} U_x^{\beta_1} U_y^{\beta_2} V_x^{\beta_3} V_y^{\beta_4} \\ &= (-1)^{b_0 \oplus \beta_0 \oplus (b_3 \& \beta_2) \oplus (b_4 \& \beta_1)} U_x^{b_1 \oplus \beta_1} U_y^{b_2 \oplus \beta_2} V_x^{b_3 \oplus \beta_3} V_y^{b_4 \oplus \beta_4} \end{aligned} \tag{4.17}$$

We can see from the equation that the multiplication rules for U_x, U_y, V_x, and V_y are quite simple, involving only the XOR operator due to its equivalence to addition mod 2. However, the -1 term is slightly more complex. It is essentially a sum of four sources of -1 phase shift. This phase shift can come from b_0 or

β_0, or it can result from moving either a U_x operator past a U_y operator or vice versa. The "And" terms in the exponents handle the latter case. For example, if there is both a V_x operator in U_g and a U_y operator in $U_{g'}$, then $b_2 \& \beta_3 =$ **True**. Using the multiplication rules for the operators, we can define the abstract group multiplication as follows:

$$g * g' = (b_0 \oplus \beta_0 \oplus [b_3 \& \beta_2] \oplus [b_4 \& \beta_1], b_1 \oplus \beta_1, b_2 \oplus \beta_2, b_3 \oplus \beta_3, b_4 \oplus \beta_4) \quad (4.18)$$

With the group element and group order defined, we can now prove that this is indeed a group. For it to be a group, the four group axioms must be satisfied: closure, associativity, the existence of an identity element, and the existence of an inverse for every element.

The group is closed because every combination of five binary numbers forms a valid group element, and the multiplication defined in 4.18 uses only boolean operators, which cannot produce anything other than another boolean value. The group is associative because it is based on the successive application of operators, which is inherently associative. The identity element of the group is $(0, 0, 0, 0, 0)$. Lastly, the inverse of every element exists. This can be verified by examining the multiplication table in figure 4.2.1 or by noting that every element is either its own inverse or results in $(-1, 0, 0, 0, 0)$ when squared. If an element g squared equals $(-1, 0, 0, 0, 0)$, then g^3 must be the inverse of g. Therefore, every element has an inverse, thus satisfying the last group axiom.

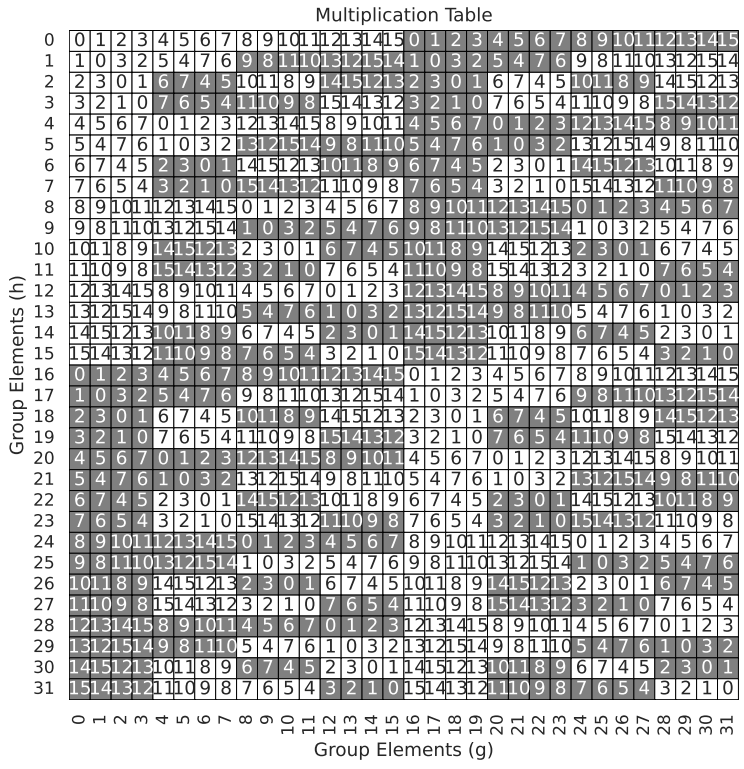


Figure 4.2.1: Multiplication table for the UV-group

The multiplication table for the group was calculated in Python as described in 2.4. For easier visualization, each group element is assigned a number using the formula $n = \sum_{i=1}^{4} 2^{i-1}$, and a color representing its phase (white for $+1$, gray for -1). This allows for easier distinction between the phase and the operators, and it will later enable us to define a homomorphism to an underlying group.
To learn more about the UV-group, including its irreducible representations, we may search a database of groups. We know the UV-group is of order 32, as the group elements consist of 5 booleans that can each take two values, yielding a group size of 2^5. The challenge is that for groups of order 2^n, there are many isomorphically different groups. Specifically, there are 51 groups of order 32, and the UV-group will be isomorphic to one of them. To determine which one, we will derive certain properties of the UV-group to filter out groups it cannot be isomorphic to. This process is conducted using the programming language GAP [17].

To derive the tabulated group to which the TC is isomorphic, we will start with every group of order 51 and filter for properties that are easy to derive and verify in GAP. First, the UV-group is non-abelian, reducing the number of candidates to 44. The UV-group has a minimum number of generators equal to 4. This can be easily seen from the U_g form of the group elements, as the operators U_x, U_y, V_x, and V_y can be exponentiated to generate every other group element. The -1 phase shift does not need to be a generator, as it can be generated from the others. For instance, $(U_x V_y)^2 = -1$. Moreover, we cannot have fewer than four generators, as removing any of the generators U_x, U_y, V_x, or V_y makes it impossible to generate the whole group. Filtering for groups with a minimum of four generators leaves us with only four possible candidates.

To further refine the search, we derive the conjugacy classes of the UV-group. Conjugacy classes help in limiting the candidates by comparing the total number of conjugacy classes in the UV-group to those in the candidates. The result of separating the group elements into conjugacy classes yields 17 classes, which narrows down the candidates to two. Finally, we examine the number of self-inverse elements of the UV-group. This can be determined by looking at the multiplication table and counting how many times the identity element occurs on the diagonal. It turns out that 20 of the 32 elements are self-inverse, leaving us with a single group. In GAP, this group is `Smallgroup(32,49)`, also known as "The + type extraspecial group of order 2^5".

Extraspecial groups have the property that the center, derived subgroup, and Frattini subgroup all coincide [26]. This is evident from the multiplication table for the UV-group. The only two elements that commute with every element are 1 and -1, making them the elements of the center. Furthermore, it can be seen from the multiplication table that 1 and -1 are the only results of the operation $x^{-1}y^{-1}xy$ for all $x, y \in G$. Therefore, they are the only elements in the derived subgroup. The same holds for the Frattini subgroup, as 1 and -1 can both be generated by the other four generators, but any other element can also be used as a generator. These are not formal proofs, but we know them to be true since the UV-group is isomorphic to the extraspecial + group of order 32.

Knowing the group allows us to use the tabulated character table for the group in GAP to derive the fingerprint of the UV-group. In general, every property listed for the `Smallgroup(32,49)` group in GAP now applies to the UV-group.

Before moving on, we will note that any choice of loops in the U_x, U_y, V_x, and V_y operators (if they have the same parity) is equally valid. The $U_{\mathcal{L}}$ operators are composed of σ_i^x operators with loops on the lattice. We demonstrated this in Fig. 2.2.4, where we deformed loop A into loop B. This implies that the group of all loop operators in the lattice can be written as a direct product of the U-group and all the $\{E, B_p\}$ operators. The analogous statement is true for the dual-lattice loops V_x and V_y and the star operator groups $\{E, A_s\}$. These relations are shown in the following equations:

$$\begin{aligned} G_{\text{All lattice loops}} &= G_U \times_{p\in P\setminus\{\tilde{p}\}} \{E, B_p\} \\ G_{\text{All dual-lattice loops}} &= G_V \times_{s\in S\setminus\{\tilde{s}\}} \{E, A_s\} \\ G_{\text{All loops}} &= G_{UV} \times_{p\in P\setminus\{\tilde{p}\}} \{E, B_p\} \times_{s\in S\setminus\{\tilde{s}\}} \{E, A_s\} \end{aligned} \tag{4.19}$$

The group of all loops on the lattice and dual-lattice combined can be written as a direct product ($\times$) of the UV-group and the groups of A_s and B_p operators. Note that in the notation for the direct product, we are taking the direct product over every star s in the set of stars S, and every plaquette p in the set of plaquettes P, with the exception of a single plaquette and a single star. This is because the stars and plaquettes are not completely independent. If they were, we would only have a single ground state. We can remove any one single B_p operator, as it can be written as a product of all the other B_p operators: $\prod_{p\in P\setminus\tilde{p}} B_p = B_{\tilde{p}}$. The same is true for the stars, so in conclusion, we remove any one of the stars and plaquettes, and the direct product becomes correct.

Irreducible representations of direct product groups are simple to derive; however, in this case, we need not do that. We know that the ground state will belong to the trivial irrep of all the $\{E, A_s\}$ and $\{E, B_p\}$ groups, as the ground states are the $+1$ eigenstates of all these operators. Therefore, we can choose just four U_x, U_y, V_x, and V_y operators and ignore the rest. Furthermore, we can choose these operators to be particularly simple, such as a straight loop in the x-direction for U_x and V_x, and similarly in the y-direction for U_y and V_y. This will slightly simplify the coming analysis.

Lastly, we can investigate an alternative way of looking at the UV-group. We have elected to include the -1 element in the group; however, this is not strictly necessary. Let's call this underlying group H_{UV}. This is now considered an abstract group whose elements are defined as

$$H_{UV} = \{(b_1, b_2, b_3, b_4) | b_i \text{ are booleans } \forall i\}. \tag{4.20}$$

The group is essentially counting how many operators there are of each type in a general UV-group element, ignoring the -1 sign. With this simplification, the group multiplication becomes much simpler, as we no longer need to account for the -1 phase shift that can occur when certain group elements are exchanged. The group should then have the same multiplication rules as the UV-group but with the b_0 bit removed. This implies

$$g, h \in H_{UV}, \quad g * h \equiv g \oplus h \tag{4.21}$$

where $\oplus$ is the element-wise XOR. First of all, this is a group for the same reason that the UV-group is a group. Secondly, this group is Abelian. This is because the four booleans are completely independent, and because the XOR operator is

commutative. Furthermore, every element is self-inverse. This can be derived analytically or can simply be seen from the multiplication table of the UV-group, which I calculated in `python`. The multiplication is shown in Figure 4.2.2.

Multiplication Table

Group Elements (h) \ Group Elements (g)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	0	3	2	5	4	7	6	9	8	11	10	13	12	15	14
2	2	3	0	1	6	7	4	5	10	11	8	9	14	15	12	13
3	3	2	1	0	7	6	5	4	11	10	9	8	15	14	13	12
4	4	5	6	7	0	1	2	3	12	13	14	15	8	9	10	11
5	5	4	7	6	1	0	3	2	13	12	15	14	9	8	11	10
6	6	7	4	5	2	3	0	1	14	15	12	13	10	11	8	9
7	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8
8	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7
9	9	8	11	10	13	12	15	14	1	0	3	2	5	4	7	6
10	10	11	8	9	14	15	12	13	2	3	0	1	6	7	4	5
11	11	10	9	8	15	14	13	12	3	2	1	0	7	6	5	4
12	12	13	14	15	8	9	10	11	4	5	6	7	0	1	2	3
13	13	12	15	14	9	8	11	10	5	4	7	6	1	0	3	2
14	14	15	12	13	10	11	8	9	6	7	4	5	2	3	0	1
15	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Figure 4.2.2: Multiplication table for H_{UV}

This is sufficient information to conclude that $H_{UV} \cong (\mathbb{Z}_2)^4$, i.e., the H_{UV} group is isomorphic to the direct product of four $\mathbb{Z}_2$ groups. The reason this all still makes sense is because we can incorporate the -1 anticommutation by treating the H_{UV} group as the projective linear group (PGL) of the UV-group, which would then be called a general linear group. This concept is defined in Section 2.4.2. Therefore, there must exist a homomorphism between G_{UV} and H_{UV}. How these groups are related is given in Equation 2.35, which, when we insert G_{UV} for the general linear group, H_{UV} for the PGL, and $F = \{E, -E\}$, we get

$$\frac{G_{UV}(GSM)}{\{E, -E\}} = H_{UV}(GSM), \tag{4.22}$$

which is indeed satisfied if we define the homomorphism to be the removal of the first bit from a group element $g \in G_{UV}$. This is shown by the following equation

$$\forall g \in G: \; h[g] = h[(b_0, b_1, b_2, b_3, b_4)] = (b_1, b_2, b_3, b_4). \tag{4.23}$$

We can now treat the U and V operators as a projective representation of the PGL. This represents a significant change from before. Previously, we could treat the U and V operators as themselves being group elements (although formally

they would also be representations of the group in this case). Now, however, we need to treat the operators U_g not as group elements themselves, but instead as representations of group elements $\Gamma(g)$ for $g \in G$. These representations will not satisfy the usual representation criterion as given in equation 2.30. Instead, they will satisfy the relation 2.35. In this case, ω is either $+1$ or -1. We can derive a formula for ω by considering when we get a -1 phase shift in the UV-group (for the elements without a -1 phase). This can be taken directly from the multiplication formula for the UV-group in equation 4.17. Thus, we get that:

$$\omega(g,h) = (-1)^{(h_1 \& g_4) \oplus (h_2 \& g_3)} \tag{4.24}$$

Using this formula, we can calculate ω for any two elements in the group. I implemented this in Python, and the result is shown in Figure 4.2.3.

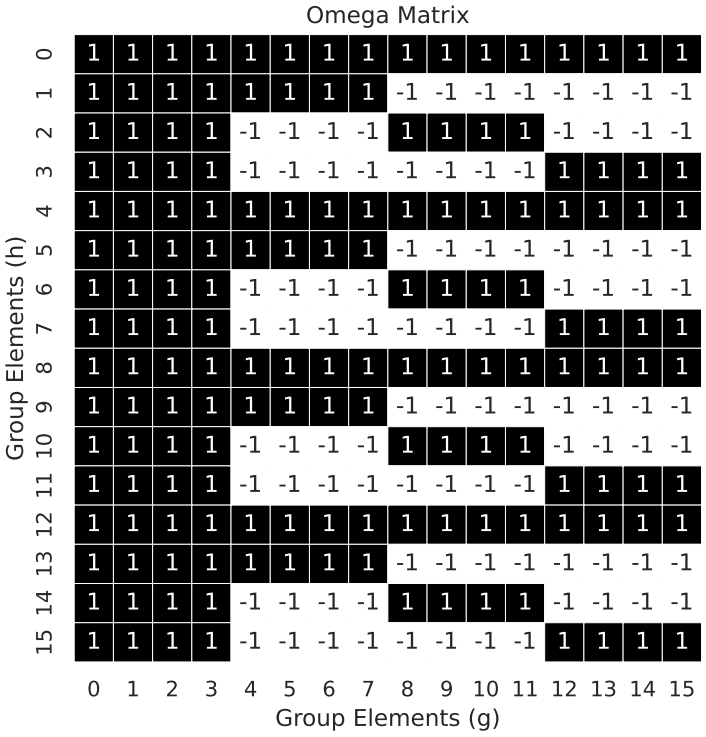


Figure 4.2.3: Omega matrix showing phase change when commuting elements

An interesting detail arises when comparing the multiplication table of the UV-group, the multiplication table of the PGL, and the ω-phase. We find that the pattern of the ω-phase can be seen in the multiplication table of the UV-group. If we simply remove the -1 phase in the multiplication table of the UV-group, we obtain the multiplication table of the PGL. This formally proves that this is a homomorphism, as removing the phase maps the elements to elements in the projective group in a way that preserves the group multiplication.

4.2.1.2 Characters of UV group

Now let's derive the effect of the UV-operators on the GSM. We will start by defining the state with loop parity numbers $(+1, +1)$. Since the parity can only be $+1$ or -1, we will from now on denote the two simply as $+$ or $-$, respectively. We define the $|+,+\rangle$ ground state first.

$$|+,+\rangle = \left(\prod_i P_i\right) |\uparrow\uparrow\uparrow \ldots \uparrow\rangle \tag{4.25}$$

where P_i are the projection operators onto $B_p = 1$. Since P_i are composed of only B_p operators, the operator $(\prod_i P_i)$ will commute with all the operators in the UV-group. Thus, we define the $|+,+\rangle$ state as the projection of the no-loop (all spin-up) state onto the $B_p = 1\ \forall p$ subspace. Now, we operate on this state with U_x.

$$\begin{aligned} U_x |+,+\rangle &= U_x(\prod_i P_i) |\uparrow\uparrow\uparrow \ldots \uparrow\rangle \\ &= (\prod_i P_i) U_x |\uparrow\uparrow\uparrow \ldots \uparrow\rangle \\ &= |-,+\rangle \end{aligned} \tag{4.26}$$

Since $U_x |\uparrow\uparrow\uparrow \ldots \uparrow\rangle$ is a loop state with parity $(-,+)$, we arrive at the fact that $U_x |+,+\rangle = |-,+\rangle$. The same proof holds for $U_y |+,+\rangle = |+,-\rangle$. Furthermore, since the U_x and U_y operators are self-inverse, we find that $U_x |-,+\rangle = |+,+\rangle$ and $U_y |+,-\rangle = |+,+\rangle$. Thus, the U_x operators toggle the parity of the ground states. Now we are left with deriving $V_x |+,+\rangle$. This turns out to be quite simple, as a similar method of proof can be used.

$$\begin{aligned} V_x |+,+\rangle &= V_x(\prod_i P_i) |\uparrow\uparrow\uparrow \ldots \uparrow\rangle \\ &= (\prod_i P_i) V_x |\uparrow\uparrow\uparrow \ldots \uparrow\rangle \\ &= (\prod_i P_i) |\uparrow\uparrow\uparrow \ldots \uparrow\rangle \\ &= |+,+\rangle \end{aligned} \tag{4.27}$$

The crucial detail to notice here is that $|\uparrow\uparrow\uparrow \ldots \uparrow\rangle$ is an eigenstate of V_x and V_y with eigenvalue 1. This is because $|\uparrow\uparrow\uparrow \ldots \uparrow\rangle$ is composed entirely of spin-up particles, so when the σ_i^z operators comprising V_x and V_y act on it, they do not transform the state or impart any -1 phase shifts.

This is enough information to derive the expectation value of any group element in the GSM. However, to simplify derivations, I will also derive the behavior of V_x and V_y on the other states.

$$\begin{aligned} V_x \left|-,+\right\rangle &= V_x U_x (\prod_i P_i) \left|\uparrow\uparrow\uparrow ... \uparrow\right\rangle \\ &= (\prod_i P_i) U_x V_x \left|\uparrow\uparrow\uparrow ... \uparrow\right\rangle \\ &= (\prod_i P_i) \left|\uparrow\uparrow\uparrow ... \uparrow\right\rangle \\ &= \left|-,+\right\rangle \end{aligned} \tag{4.28}$$

$$\begin{aligned} V_x \left|+,-\right\rangle &= V_x U_y (\prod_i P_i) \left|\uparrow\uparrow\uparrow ... \uparrow\right\rangle \\ &= (\prod_i P_i) - U_y V_x \left|\uparrow\uparrow\uparrow ... \uparrow\right\rangle \\ &= -(\prod_i P_i) \left|\uparrow\uparrow\uparrow ... \uparrow\right\rangle \\ &= - \left|+,-\right\rangle \end{aligned} \tag{4.29}$$

From these two equations, we see that the ground states are eigenstates of the V-operators. Furthermore, the eigenvalues are the same as the parity number of the state in the opposite axis.
Now that the effect of the operators on the ground states has been determined, we can derive the characters of the GSM. The character of a group element can be written as

$$\begin{aligned} \chi(g) &= Tr(\Gamma(g)) \\ &= \left\langle +,+\right| U_g \left|+,+\right\rangle + \left\langle -,+\right| U_g \left|-,+\right\rangle \\ &+ \left\langle +,-\right| U_g \left|+,-\right\rangle + \left\langle -,-\right| U_g \left|-,-\right\rangle \\ &= \langle U_g \rangle_{+,+} + \langle U_x U_g U_x \rangle_{+,+} \langle U_y U_g U_y \rangle_{+,+} + \langle U_y U_x U_g U_x U_y \rangle_{+,+}, \end{aligned} \tag{4.30}$$

where $\langle U_g \rangle_{(+,+)} = \left\langle +,+\right| U_g \left|+,+\right\rangle$. As a small note, the operators are unitary (by definition), therefore $U_x^\dagger = U_x^{-1} = U_x$ since U_x is self-inverse. On the other hand, $(U_x U_y)^{-1} = U_y U_x$. To derive the characters, we first note that the characters of any group are class functions, which means that any two group elements in the same class have the same character. Secondly, we can simplify the derivation of the characters by noting that the expectation value of any group operator U_g that contains either U_x or U_y is zero. Furthermore, for any other group element $U_{g'}$, the expectation value $\langle U_{g'}^{-1} U_g U_{g'} \rangle = 0$. I will now prove this fact.
Assume $g \in G$, where $g = (b_0, b_1, b_2, b_3, b_4)$, and either $b_1 = 1$ or $b_2 = 1$ (or both). Then:

$$\left\langle +,+\right| U_g \left|+,+\right\rangle = c \left\langle +,+\right| \left|(-1)^{b_1}, (-1)^{b_2}\right\rangle\rangle = 0 \tag{4.31}$$

where c is either +1 or -1. This evaluates to zero because $\left|+,+\right\rangle \neq \left|(-1)^{b_1}, (-1)^{b_2}\right\rangle$ (we assumed either b_1, b_2 are not both zero by assumption), so orthogonality implies the inner product is 0. Now let $g' \in G$. Then

$$\begin{aligned} \langle U_{g'}^{-1} U_g U_{g'} \rangle &= \langle c U_g U_{g'}^{-1} U_{g'} \rangle \\ &= c \langle U_g \rangle = 0. \end{aligned} \tag{4.32}$$

These proofs both rely on the fact that commuting two U_g operators will only result in a possible sign change. This can be seen by looking at the multiplication rules for the UV-group shown in 4.17. Knowing this, we can go back to deriving the characters of the UV-group. The only five conjugacy classes that are not zero by this fact alone are $\{1\}$[1], $\{-1\}$, $\{\pm V_x\}$, $\{\pm V_y\}$, and $\{\pm V_x V_y\}$. Inserting either 1 or -1 into the character equation 4.30 we get $\chi(1) = 4$ and $\chi(-1) = -4$. Next, we derive the characters for the last three classes. To make the equations more readable, I have suppressed the $(+,+)$ notation in $\langle U_g \rangle_{(+,+)}$. However, it is still implied.

$$\begin{aligned}
\chi(V_x) &= \langle V_x \rangle + \langle U_x V_x U_x \rangle + \langle U_y V_x U_y \rangle + \langle U_y U_x V_x U_x U_y \rangle \\
&= \langle V_x \rangle + \langle V_x \cancel{U_x U_x} \rangle - \langle V_x \cancel{U_y U_y} \rangle - \langle V_x \cancel{U_y U_x U_x U_y} \rangle \\
&= \langle V_x \rangle - \langle V_x \rangle - \langle V_x \rangle + \langle V_x \rangle = 0
\end{aligned} \tag{4.33}$$

$$\begin{aligned}
\chi(V_y) &= \langle V_y \rangle + \langle U_x V_y U_x \rangle + \langle U_y V_y U_y \rangle + \langle U_y U_x V_y U_x U_y \rangle \\
&= \langle V_y \rangle - \langle V_y \cancel{U_x U_x} \rangle + \langle V_y \cancel{U_y U_y} \rangle - \langle V_y \cancel{U_y U_x U_x U_y} \rangle \\
&= \langle V_y \rangle - \langle V_y \rangle - \langle V_y \rangle + \langle V_y \rangle = 0
\end{aligned} \tag{4.34}$$

$$\begin{aligned}
\chi(V_x V_y) &= \langle V_x V_y \rangle + \langle U_x V_x V_y U_x \rangle + \langle U_y V_x V_y U_y \rangle \\
&\quad + \langle U_y U_x V_x V_y U_x U_y \rangle \\
&= \langle V_x V_y \rangle - \langle V_x V_y \cancel{U_x U_x} \rangle - \langle V_x V_y \cancel{U_y U_y} \rangle \\
&\quad + \langle V_x V_y \cancel{U_y U_x U_x U_y} \rangle \\
&= \langle V_x V_y \rangle - \langle V_x V_y \rangle - \langle V_x V_y \rangle + \langle V_x V_y \rangle = 0
\end{aligned} \tag{4.35}$$

In conclusion, we find that the character $\chi(c)$ for any class c, with the exception of $\{1\}$ and $\{-1\}$, is 0. We can now compare this result with the tabulated group in GAP. It turns out that the four-dimensional irreducible representation of the tabulated group has characters that are exactly the same as the representation we derived for the UV-group. This result is particularly noteworthy for two reasons.

Firstly, it means that the fingerprint is trivial to derive. The 4D irrep of the tabulated group occurs once in the irrep decomposition of the UV-group representation on the GSM. Since the GSM is four-dimensional, there are no other irreps that also contribute.

Secondly, this explains the degeneracy of the GSM from a group-theoretical perspective. The GSM transforms under the group transformations like the four-dimensional irreducible representation of the UV-group. This also implies that the GSM spontaneously breaks the symmetries of the UV-group; otherwise, we would expect the character decomposition to result in the group transforming under the group transformations like a direct sum of four of the trivial irreps. As a final note, the ground states transforming like an irrep of some group was mentioned already in the founding paper by Kitaev [9]. So we have reproduced that result and found a specific group that gives rise to this irrep.

[1] E = 1

4.2.2 TCL model

The TCL model is very similar to the TC model; however, the removal of the periodicity in the x-axis will naturally change the behavior of the system. Firstly, let's investigate the ground states of the TCL model.

4.2.2.1 Ground States and the UV-Group

The argument for the number of ground states for the TC-model relied on how many topologically distinct loops there were on the lattice. That is, how many types of loops there are that cannot be deformed into each other by successive application of B_p operators. It turned out that for each direction, we had a loop-parity number ($+1$ or -1), which indicated if a loop state wound an even or odd number of times around a given axis.

The ladder model is not periodic in the y-direction. The U_y operator from before no longer commutes with the star operators due to them only sharing a single spin (which results in anticommutation). This, in turn, means that we have to remove U_y as one of the generators of the group. It also means that we only have one parity number, which counts if the loop winds an even or odd number of times around the x-axis. This also removes one of the loop parity numbers, which in turn results in there only being two loops that cannot be deformed into each other by the use of B_p operators. This implies there are only two ground states, one for each loop. We call these ground states $|+\rangle$ and $|-\rangle$.

We will now consider the V_y operator. It turns out that this operator will remain as a symmetry operator for the group. This is because it consists of σ_i^z operators, so it commutes trivially with the A_s operators. For any B_p, it shares either two or no spins, so it will also commute with B_p.

So our current working group should be generated by U_x, V_x, and V_y. However, this is actually more complicated than necessary. Recall for the UV-group that the reason we could discard all U_x and U_y loops that were different from the ones we chose was because the UV-group is really a subgroup of a group of all loops which contains all the non-commutative behavior. It turns out that we can, by the same argument, remove the V_x operator and instead include it in an analogous direct product at the end. This is because V_x commutes with both V_y and U_x. Therefore, we can write the group generated by U_x, V_x, and V_y, which we will call $G'_{UV,\text{reduced}}$, as a direct product of the group generated by U_x and V_y (which we will call $G_{UV,\text{reduced}}$) and the group $\{E, V_x\}$.

We will now derive what group $G_{UV,\text{reduced}}$ should be. We start by defining a general unitary operator of $G_{UV,\text{reduced}}$ as given in equation 4.36. As for the UV-group, the ordering here is strict.

$$\begin{gathered} U_g = (-1)^{b_0} U_x^{b_1} V_y^{b_2} \\ \text{where } g = (b_0, b_1, b_2) \\ \text{and } b_1,\ b_2,\ b_3 \text{ are booleans} \end{gathered} \tag{4.36}$$

As with the UV-group, the source of the anticommutation in the reduced UV-group is whenever we move a U_x operator past a V_y operator. Ignoring the induced phase, all elements commute. An interesting note is that all the operators in the reduced UV-group are operators in the UV-group. The U_x operator is identical,

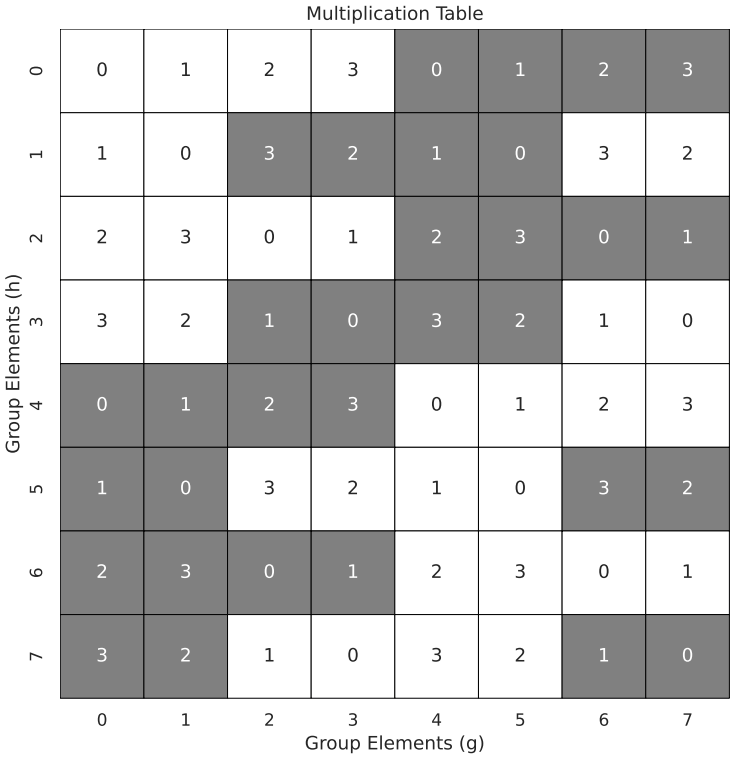


Figure 4.2.4: $G_{UV,\text{reduced}}$ multiplication table

and the V_y operator is identical to the one for the 2D model, only shrunk down to 1D. Because of this, we can infer that the reduced UV-group will be a subgroup of the UV-group, specifically the subgroup:

$$G_{UV,\text{reduced}} = \{g \in G_{UV} \mid g_2, g_3 = 0\}. \tag{4.37}$$

Equation 4.36 now lets us define the group multiplication rules, which are given by:

$$\begin{aligned} &\text{Let } b, \beta \in G_{UV,reduced} : \\ &U_b U_\beta = (-1)^{b_0 \oplus \beta_0 \oplus (b_2 \& \beta_1)} U_x^{b_1 \oplus \beta_1} V_y^{b_2 \oplus \beta_2} \\ &b * \beta = (b_0 \oplus \beta_0 \oplus (b_2 \& \beta_1), b_1 \oplus \beta_1, b_2 \oplus \beta_2) \end{aligned} \tag{4.38}$$

This formula and the definition was programmed in python in order to derive a multiplication table. This is shown in Fig. 4.2.4. As with the UV group we assign each group element a number based on the bits b_1 and b_2 according to equation 4.39, and either the color white or black depending on its phase being 1 or -1 respectively.

$$n_g = \prod_{i=1}^{2} 2^{g_i} \tag{4.39}$$

With the group elements and multiplication defined we can now find out which tabulated group the reduced UV-group is isomorphic to. However i will first define the symmetry group of all the loops as i did for the TC-model. This will be slightly different for the TCL-model as we have taken out V_x from the group due to it commuting with every other element of the reduced UV-group.

$$\begin{aligned} G_{\text{All lattice loops},TCL} &= U_x \times_{p\in P} \{E, B_p\} \\ G_{\text{All dual-lattice loops},TCL} &= \{E, V_x\} \times \{E, V_y\} \times_{s\in S\setminus\{\tilde{s}\}} \{E, A_s\} \\ G_{\text{All loops},TCL} &= G_{UV,reduced} \times \{E, V_x\} \times_{p\in P} \{E, B_p\} \times_{s\in S\setminus\{\tilde{s}\}} \{E, A_s\} \end{aligned} \tag{4.40}$$

These equations differ from the ones for the UV-group given in equation 4.19. The stars are not completely independent, as before; however, in the ladder case, the plaquette operators are completely independent. Thus, we need to remove one of the stars for the equations to be correct, but we keep all the plaquettes. The second difference is that $\{E, V_x\}$ is not included in $G_{UV,\text{reduced}}$ and is instead included in $G_{\text{All loops, TCL}}$ by taking a direct product.

We can now derive the tabulated group the reduced UV-group corresponds to. The group is of order 8, which gives us 5 candidate groups. We know the group is non-abelian, which reduces the possible candidates to only 2. These two groups are quite similar and are the quaternion group Q_8 and the dihedral group D_8. Both these groups have the same exponent (4) and the same number of conjugacy classes (5). What will separate the two is investigating the orders of the elements. If we consider only the number of self-inverse elements in each group, we find that 6 of the 8 elements of the reduced UV-group are self-inverse. This implies that there should be one element of order 1 (this is always true), and seven of order 2. This criterion implies that the reduced UV-group is isomorphic to the dihedral group D_8 [17].

It is interesting that the group we end up with is the dihedral group because it is strongly related to the tabulated group to which the UV-group is isomorphic. The dihedral group can alternatively be called the + type extraspecial group of order 2^3. The extraspecial groups of prime order 2 (order mod 2 = 0) will, in general, be of order 2^{2r+1} where r is some positive integer (nonzero). The smallest extraspecial groups of prime order 2 are therefore of order 8, and they are D_8 (+ type) and the quaternion group Q_8 (- type). All other extraspecial groups of prime order 2 can be written as a central product of D_8 and Q_8 groups. For the case of the + type extraspecial group of order 32, it can be written either as a central product of two D_8 groups or two Q_8 groups. Generally, the + or − type refers to whether one needs an odd or even number of Q_8 groups in the central product to construct it.

So the UV-group and the reduced UV-group are heavily related. They also have very similar PGLs. If we repeat the same procedure as we did for the UV-group with the reduced UV-group, we can define $H_{UV,\text{reduced}}$ as follows:

$$H_{UV,\text{reduced}} = \{(b_1, b_2) \mid b_1,\ b_2 \text{ are booleans}\} \tag{4.41}$$

The multiplication rules for the two groups will be identical:

$$g, h \in H_{UV,\text{reduced}}, \quad g * h \equiv g \oplus h \tag{4.42}$$

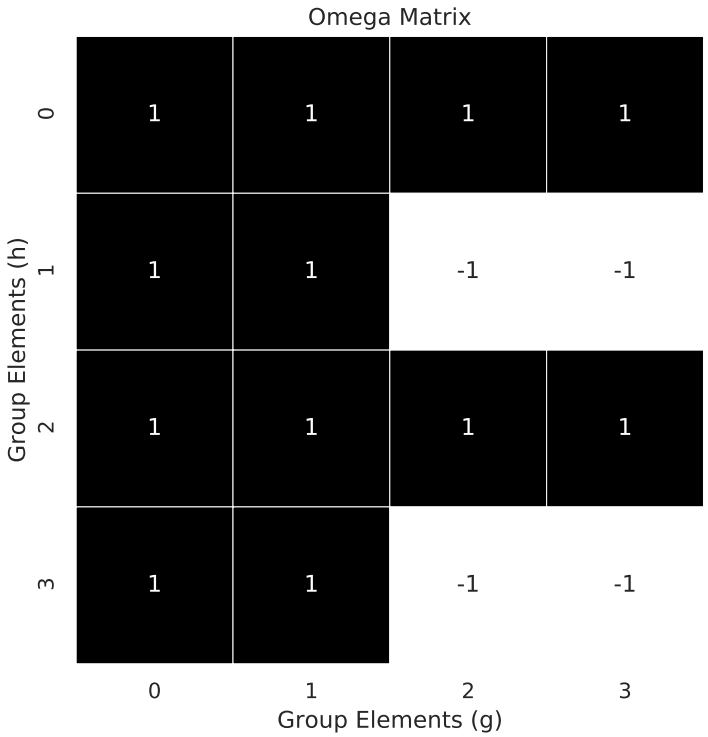


Figure 4.2.5: Omega matrix for the reduced UV-group

Due to the fact that the booleans are completely independent, as is also the case for H_{UV}, we can conclude that the group is isomorphic to the direct product of two $\mathbb{Z}_2$ groups ($H_{UV,\text{reduced}} = (\mathbb{Z}_2)^2$). The homomorphism condition is also identical:

$$\frac{G_{UV,\text{reduced}}(GSM)}{\{E, -E\}} = H_{UV,\text{reduced}}(GSM) \tag{4.43}$$

We define the homomorphism function in a similar way, by simply removing the sign bit of the reduced UV-group.

$$\forall g \in G_{UV,\text{reduced}} : \; h[g] = h[(b_0, b_1, b_2)] = (b_1, b_2) \tag{4.44}$$

We can now write the operators which will operate on the GSM U_g as $\Gamma(\tilde{g})$ where $\tilde{g} = h(g)$ for $g \in G_{UV,\text{reduced}}$. These representations should also satisfy a similar equation as for the UV-group (Eq. 4.23), with its own ω phase function defined as:

$$\omega(g, h) = (-1)^{(h_1 \wedge g_2)} \tag{4.45}$$

This function was also defined in Python and plotted to show the pattern, which, as with the UV-group, can be recognized in the multiplication table for the reduced UV-group.

4.2.2.2 Characters of the reduced UV group

The derivation of the characters for the reduced UV-group is also very similar to the derivation for the characters for the UV-group. We start by defining the ground state with even parity ($|+\rangle$) as follows:

$$|+\rangle = \left(\prod_i P_i\right) |\uparrow\uparrow\uparrow \dots \uparrow\rangle \tag{4.46}$$

The method is the same as for 4.25, exploiting the fact that $(\prod_i P_i)$ commutes with the group operations. We can operate on this state with the U_x operator:

$$\begin{aligned} U_x |+\rangle &= U_x \left(\prod_i P_i\right) |\uparrow\uparrow\uparrow \dots \uparrow\rangle \\ &= \left(\prod_i P_i\right) U_x |\uparrow\uparrow\uparrow \dots \uparrow\rangle = |-\rangle \end{aligned} \tag{4.47}$$

So, as in the TC-model, the U_x operator switches the loop parity in the x-axis. We will now derive $V_y |+\rangle$ and $V_y |-\rangle$.

$$\begin{aligned} V_y |+\rangle &= V_y(\prod_i P_i) |\uparrow\uparrow\uparrow \dots \uparrow\rangle \\ &= (\prod_i P_i) V_y |\uparrow\uparrow\uparrow \dots \uparrow\rangle = |+\rangle \\ V_y |-\rangle &= V_y(\prod_i P_i) U_x |\uparrow\uparrow\uparrow \dots \uparrow\rangle \\ &= (\prod_i P_i) - U_x V_x |\uparrow\uparrow\uparrow \dots \uparrow\rangle = - |-\rangle \end{aligned} \tag{4.48}$$

So, as in the TC-model, the loop parity in the x-axis is equivalent to the quantum number for V_y, and the states $|+\rangle$ and $|-\rangle$ are eigenstates of V_y. Now we are equipped to derive the characters of the reduced UV-group in the GSM vector space. We can write the general character of some group element g as follows:

$$\begin{aligned} \chi(g) &= \mathrm{Tr}(\Gamma(g)) \\ &= \langle +| U_g |+\rangle + \langle -| U_g |-\rangle \\ &= \langle +| U_g |+\rangle + \langle +| U_x U_g U_x |+\rangle \end{aligned} \tag{4.49}$$

From the multiplication table of the reduced UV-group, we can see, as for the UV-group, that when we exchange any two elements, the number of U_x and V_x operators does not change. Only the sign may change under such an operation. Therefore, we can conclude, as for the UV-group, that any group element $g \in G_{UV,\text{reduced}}$ where g_1 is nonzero will result in $\chi(g) = 0$. This leaves us with three classes which may have nonzero characters: $\{1\}$, $\{-1\}$, and $\{\pm V_y\}$. The first two are simple. Since these are simply scalars, their characters are:

$$\begin{aligned} \chi(1) &= \langle +| + \rangle + \langle -| - \rangle = 2 \\ \chi(-1) &= - \langle +| + \rangle - \langle -| - \rangle = -2 \end{aligned} \tag{4.50}$$

This leaves us finally with the character for V_y. This character will also be zero:

$$\begin{aligned}\chi(V_y) &= \langle +| V_y |+\rangle + \langle +| U_x V_y U_x |+\rangle \\ &= 1 + \langle +| U_x V_y U_x |+\rangle \\ &= 1 + \langle +| V_y |+\rangle \\ &= 1 - 1 \\ &= 0\end{aligned} \tag{4.51}$$

We can now compare the characters for the reduced UV-group in the GSM vector space with the tabulated characters for the D_8 group. We find that the characters correspond exactly to the two-dimensional irrep of D_8, which thus explains the degeneracy of the TCL-model from a group theoretical standpoint, as the GSM transforms under the group operations of the reduced UV-group like the 2D irrep of D_8.

4.2.3 DS-Model

In this section, we will derive the group structure for the DS-model and derive the characters of the group elements in the GSM vector space.

4.2.3.1 The UU-Group

To derive the structure of the UU-group, a name for which the reasoning will become apparent later, we first start by investigating the group element U_x^+. We define this as the shortest loop we can make around the x-axis. We will later argue why the other odd parity loops in the x-direction can be neglected.

We treat this element as a generator, so we will start by determining what will result if we exponentiate it. The U operators are defined in equation 2.18. We use this definition to find $(U_x^\pm)^2$:

$$\begin{aligned}(U_x^\pm)^2 = &\prod_{i\in\mathcal{L}_x}\sigma_i^x \prod_{k\in L}(-1)^{\frac{1}{4}(1-\sigma_i^z)(1+\sigma_j^z)} \prod_{l\in R}(\pm i)^{(1-\sigma_l^z)/2} \\ &\times \prod_{i'\in\mathcal{L}_x}\sigma_i^x \prod_{k'\in L}(-1)^{\frac{1}{4}(1-\sigma_i'^z)(1+\sigma_j'^z)} \prod_{l'\in R}(\pm i)^{(1-\sigma_{l'}^z)/2}\end{aligned} \tag{4.52}$$

To determine this product, we have to be careful about anti-commuting elements. The simplest term to deal with is the R term. The spins in the R term are shared by none of the spins in the loop term, so they will commute with everything and can be combined. The first and second terms share all their spins, so they will anticommute.

$$\begin{aligned}(U_x^\pm)^2 &= \prod_{k\in L}(-1)^{\frac{1}{4}(1+\sigma_i^z)(1-\sigma_j^z)} \prod_{i\in\mathcal{L}_x}\sigma_i^x \cancel{\prod_{i'\in\mathcal{L}_x}\sigma_i^x} \\ &\quad\times \prod_{k'\in L}(-1)^{\frac{1}{4}(1-\sigma_i'^z)(1+\sigma_j'^z)} \prod_{l'\in R}(\pm i)^{(1-\sigma_l^z)} \\ &= \prod_{k\in L}(-1)^{\frac{1}{4}(1+\sigma_i^z)(1-\sigma_j^z)+\frac{1}{4}(1-\sigma_i'^z)(1+\sigma_j'^z)} \prod_{l'\in R}\sigma_l^z \\ &= \prod_{k\in L}(-1) \prod_{l'\in R}\sigma_l^z \\ &= (-1)^{N_x} V_x\end{aligned} \tag{4.53}$$

The result of the exponentiation is that we produce a V_x with a sign depending on whether the number of spins in the x-direction is even. The shape of V_x is shown in Figure 4.2.6.

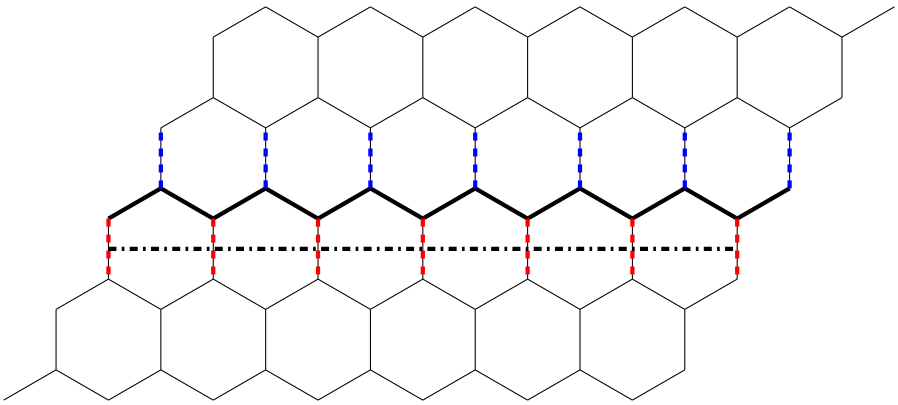


Figure 4.2.6: Drawing of the U_x^+ operator and its corresponding and V_x operator (shown as dot-dashed line).

It turns out that for the system to be periodic, N_x has to be even. We can see from Figure 4.2.6 that the spins in the zig-zag line can be joined in pairs, which implies that N_x has to be even. This also holds true for N_y. One of the steps needs additional justification: $(\pm i)^{(1-\sigma_l^z)} = \prod_{l' \in R} \sigma_l^z$. The fact that this is true can be seen by how $(\pm i)^{(1-\sigma^z)}$ operates on the spin states $|\uparrow\rangle$ and $|\downarrow\rangle$.

$$\begin{aligned} (\pm i)^{(1-\sigma^z)} |\uparrow\rangle &= (\pm i)^{(1-1)} |\uparrow\rangle = |\uparrow\rangle \\ (\pm i)^{(1-\sigma^z)} |\downarrow\rangle &= (\pm i)^{(1+1)} |\downarrow\rangle = -|\downarrow\rangle \end{aligned} \tag{4.54}$$

Replacing x with y in equation 4.53 yields the exact same result for $U_y^\pm$. So we have determined that $(U_x^\pm)^2 = V_x$ and $(U_y^\pm)^2 = V_y$. However, we know that V_x and V_y are self-inverses. This leads to the following result:

$$\begin{aligned} & (V_x)^2 = 1, \text{ and } (V_y)^2 = 1 \\ \implies & (U_x^\pm)^4 = E \\ \implies & (U_y^\pm)^4 = E \\ \implies & (U_x^\pm)^3 = (U_x^\pm)^{-1} \\ \implies & (U_y^\pm)^3 = (U_x^\pm)^{-1} \end{aligned} \tag{4.55}$$

Now we will derive the operator $U_x^+ V_x$. We will utilize the relation $\sigma^z = i^{(1-\sigma^z)}$ in order to calculate this. Note that U_x^+ and V_x commute, as V_x only shares spins with the R-term of U_x^+.

$$\begin{aligned} U_x^+ V_x &= \prod_{i \in \mathcal{L}_x} \sigma_i^x \prod_{k \in L} (-1)^{\frac{1}{4}(1-\sigma_i^z)(1+\sigma_j^z)} \prod_{l \in R} i^{(1-\sigma_l^z)/2} \prod_{l' \in R} i^{(1-\sigma_l^z)} \\ &= \prod_{i \in \mathcal{L}_x} \sigma_i^x \prod_{k \in L} (-1)^{\frac{1}{4}(1-\sigma_i^z)(1+\sigma_j^z)} \prod_{l \in R} (i^3)^{(1-\sigma_l^z)/2} \\ &= U_x^- \end{aligned} \tag{4.56}$$

The derivation in equation 4.56 implies that U_x^+ and U_x^- can be turned into each other by exponentiating the other to the third power, as $V_x = (U_x^\pm)^2$. It also

implies that they are each other's inverses. This fact can also be derived from the fact that they are unitary operations, and therefore $U_x^{+\dagger} = (U_x^+)^{-1}$. Only the R term in U_x has a complex part. If we flip the sign of the i's in the R term for $U_x^\pm$, we get $U_x^\mp$, so we find that $(U_x^\pm)^\dagger = (U_x^\pm)^{-1} = U_x^\mp$.

The $U_y^\pm$ operators are virtually identical to the $U_x^\pm$ operators, hence they will have the same properties when exponentiated. It seems now by exponentiating U_x^+ and U_y^- we get all the group elements which we assumed to be a part of the group. We will therefore, for now, assume that the UU-group, as I will call it, is the group generated by U_x^+ and U_y^-. We will later argue why this assumption is valid by deriving the group of all the loop operators later.

To derive the group structure, we need to derive one final property of the operators. We need to determine the multiplicative commutator of U_x^+ and U_y^-. Before we do that, we will start off with a simpler example, which is to derive the multiplicative commutator of U_x^+ and $(U_y^-)^2$. It turns out here that the answer is the same as for the TC-model. Looking at Figure 4.2.7, we can see that the two operators share a single spin.

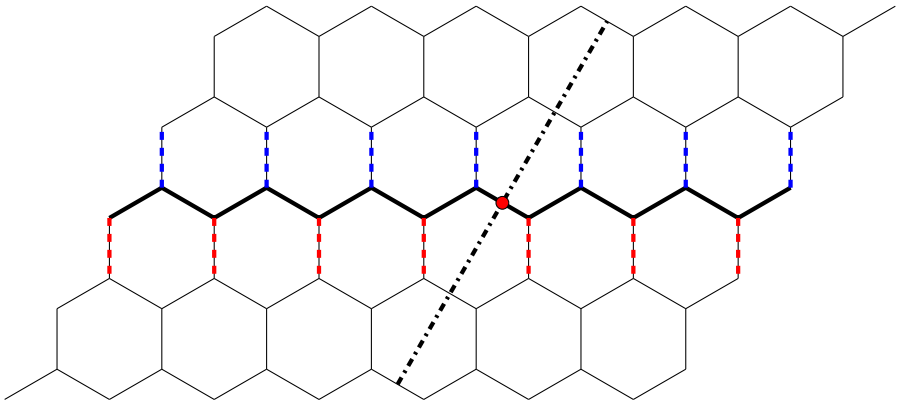


Figure 4.2.7: Visualization of the U_x^+ operator and V_y operator (shown as dot-dashed line). They share a single spin, marked with a red dot, which will lead to a -1 sign change when they are exchanged

The spin which is shared is part of the loop term in U_x^+, which implies there will be a factor of -1 from this term when the operators commute. Thus, we find that, as with the toric code, -1 should be a group element. Now, this suggests something interesting. If switching the places of two U_x^+ operators and a U_y^+ operator yields a phase shift of -1, then doing the same with only a single U_x^+ operator and a U_y^+ operator could yield a phase shift of i or $-i$. If we assume exchanging U_x and U_y yields either $\pm i$ or a phase operator that behaves like $\pm i$ and commutes with every operator in the group, then that would satisfy the following equation:

$$\begin{aligned} U_x^+(U_y^-)^2 &= (U_y^-)^2 U_x^+ \\ \implies U_x^+(U_y^-)^2 &= \phi U_y^- U_x^+ U_y^- = \phi^2 (U_y^-)^2 U_x^+ \end{aligned} \tag{4.57}$$

It turns out the operators will have this behaviour [10], so thus we will assume the operators U_x^+ and U_x^- generates a $\pm i$ phase when exchanged. In order to not decide which, we will use ϕ instead to represent either $+i$ or $-i$. We can now write a general group operator (strict ordering) and its corresponding group element as follows:

$$\begin{aligned} U_g &= \phi^{g_0} U_x^{g_1} U_y^{g_2} \\ g &= (g_0, g_1, g_2) \\ G &= (g \in \mathbb{N}^3 | g = g \ mod \ 4) \end{aligned} \tag{4.58}$$

To find a multiplication formula, we use the same strategy as for the UV-group. We start with the operators for the group elements and apply the rule that $U_x U_y = iU_y U_x$.

$$U_g U_h = \phi^{g_0} U_x^{g_1} U_y^{g_2} \phi^{h_0} U_x^{h_1} U_y^{h_2} \tag{4.59}$$

In order to write equation 4.59 in the standard form given in equation 4.58, we need to exchange a certain number of U_x operators with U_y operators. Each exchange produces a $\phi^{g_2 h_1}$ phase shift. Once the operators are in the correct order, we can simply add their exponents. This leads to the following multiplication rules:

$$\begin{aligned} U_g U_h &= \phi^{(g_0+h_0+g_2h_1) \ mod \ 4} U_x^{(g_1+h_1) \ mod \ 4} U_y^{(g_2+h_2) \ mod \ 4} \\ g * h &= (g_0 + h_0 + g_2 h_1, g_1 + h_1, g_2 + h_2) \ mod \ 4 \end{aligned} \tag{4.60}$$

Note that the modulo 4 in the operator form is not strictly necessary, as $U_x^{k+4} = U_x^k$. However, restricting the exponents to natural numbers smaller than 4 makes it easier to see if two group elements are identical.
To derive several properties of the group, we implemented the multiplication function in Python. Since the group is much larger than the UV-group, visualizing the entire multiplication table becomes too difficult. Instead, we will only show the multiplication table for the elements without a phase. We associate each group element with a number and a color. The number is given by the equation:

$$n_g = g_1 + 4g_2 \tag{4.61}$$

The color is determined by the phase, where 1 is white, -1 is dark gray, ϕ is light gray, and $-\phi$ is black. This yields the multiplication table shown in Fig. 4.2.8.
By closely examining the first upper quadrant of Fig. 4.2.1, we observe that if we were to remove the complex factor—thereby sending i to 1 and $-i$ to -1—the tables would be identical. Therefore, we may conclude that there exists a homomorphism from the UU group to the UV group.
Now we can start filtering for the group to which the UU-group is isomorphic. There are 267 groups of order 64 up to isomorphism, making the task of finding the tabulated group which the UU-group is isomorphic to more challenging. We will use the same filtering strategy as for the UV-group and the reduced UV-group; however, we will need to derive more information about the UU-group manually than we did earlier.

The UU-group is non-abelian. Filtering for this reduces the number of candidates to 256 groups. The UU-group has a minimum number of generators equal to 2. This can be seen from the definition of U_g in equation 4.58, as U_x and U_y generate the group. Filtering for this reduces the number of candidates to 50.

The conjugacy classes are needed later to derive the characters. They were derived in Python using a similar script as for the UV-group. We end up with 22 conjugacy classes, which reduces the number of candidates to 15. Examining

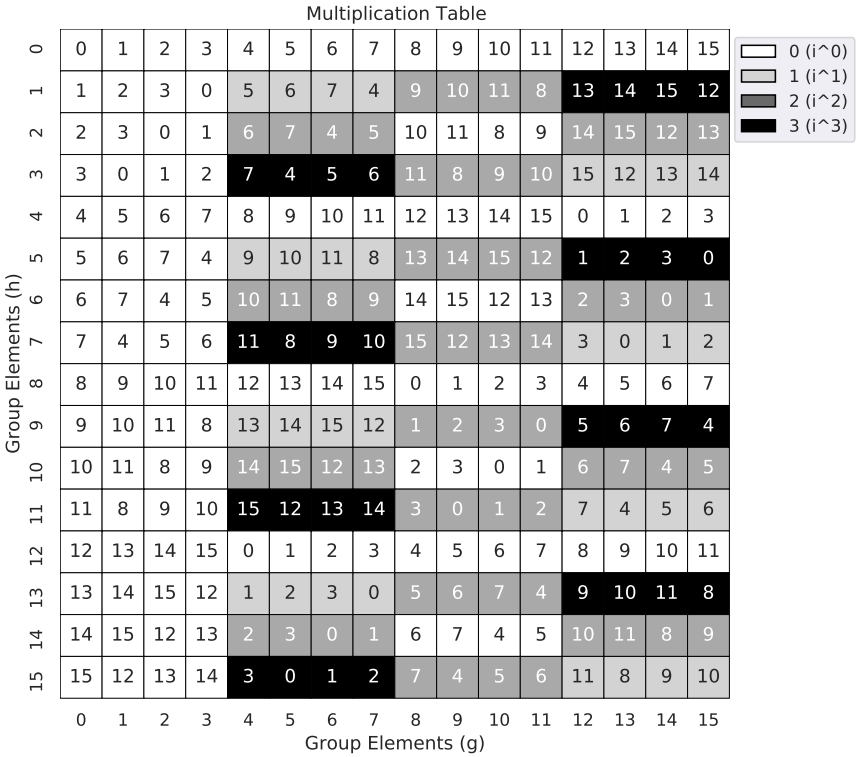


Figure 4.2.8: Multiplication table for the UU-group

the multiplication table, we find that the highest order of the elements is 8. One example is the element U_xU_y, which has order 8:

$$\begin{aligned} (U_xU_y)^2 &= U_xU_yU_xU_y \\ &= \phi V_xV_y \\ \implies (U_xU_y)^4 &= -1 \\ \implies (U_xU_y)^8 &= 1 \end{aligned} \tag{4.62}$$

Filtering for groups with exponent 8 leaves us with 6 candidates. An interesting property to investigate would be the center, derived subgroup, and Frattini subgroup of the UU-group. For the UV-group, these all coincided, allowing us to call it extra special. The UU-group, however, cannot be extra special, as every extra special group of prime order 2 has order given by p^{1+2n}, which the UU-group is not. It would still be interesting to see which of these subgroups will coincide and which will not.

The center is the simplest to determine. The only elements that commute with every other element are the pure phases $C = \{1, -1, i, -i\}$. This can be seen either from the multiplication table or from the fact that these elements are alone in their conjugacy classes (they have only one element), which is an equivalent property. Furthermore, we can see that the center group is isomorphic to the cyclic group C_4, as it is abelian and exponentiating i generates the entire group. Filtering for groups where the center group is C_4 leaves us with 3 candidates.

The derived subgroup is also quite simple to generate. I implemented it in code, and the result shows that the derived subgroup is identical to the center. This can be explained by noting that the elements of the derived subgroup are all the elements in the group which can be written as a multiplicative commutator $g = [x, y] = x^{-1}y^{-1}xy$. However, we find that switching any two group elements will only have the effect of picking up a phase change. So the derived subgroup is only the phase elements $D = \{1, -1, \phi, -\phi\} = C$. Filtering for groups with a derived subgroup isomorphic to C_4 leaves us with only two candidates.

The fact that the derived subgroup and center coincide also implies that the Frattini subgroup cannot be the same subgroup. I will not derive the full Frattini subgroup, but an argument for the fact that it is not equal to the other two is that the V_x and V_y operators also have to be a part of the Frattini subgroup as they too are non-generators. The two remaining candidates are very similar; however, we can separate them by determining the number of elements of each order in the group. This was calculated in Python, with the resulting terminal output:

```
1 element of order 1
7 elements of order 2
40 elements of order 4
16 elements of order 16
```

This distribution is matched by only one of the tabulated groups, which is the unitriangular matrix group $UT(3, \mathbb{Z}_4)$ [17]. This group has two 4-dimensional irreps and can therefore potentially explain the degeneracy of the GSM.

We can create a PGL for the UU group, analogous to how we did it for the TC and TCL models. In the UU-group, we have four phase elements due to the presence of ϕ in the group. Thus, the PGL will contain only 16 elements. We can define the homomorphism relation between the UU-group (G_{UU}) and the PGL (H_{UU}) as follows:

$$\frac{G_{UU}(GSM)}{\{E, -E, \phi, -\phi\}} = H_{UU}(GSM) \tag{4.63}$$

where the homomorphism $h(g)$ is defined as:

$$\forall g \in G_{UU} : \; h[g] = h[(g_0, g_1, g_2] = (g_1, g_2) \tag{4.64}$$

This results in the underlying subgroup H_{UU} being isomorphic to the group $\mathbb{Z}_4 \times \mathbb{Z}_4$.

Lastly, before calculating the characters of the UU-group, we will investigate the full loop operator group for the DS-model, similar to what we did for the TC model. One caveat here is that the B_p operators do not actually commute. However, we are only interested in the behavior of the operators on the GSM. Since the commutator $[B_p, B_{p'}]$ has the property that $[B_p, B_{p'}] |\text{loop}\rangle = 0$, where $|\text{loop}\rangle$ is any state satisfying $A_s |\text{loop}\rangle = |\text{loop}\rangle$, $\forall s$, we can conclude that we can treat the B_p operators as commuting as long as we are operating on the GSM. This allows us to split up the full loop group as a direct product of B_p operators and A_s operators as we did for the TC model. It will not be exactly a direct product, as the operators do not generally commute, but we can approximate them as commuting. Before giving the expression for the full loop operator group,

we will consider two operators which we could have included in the UU-group, but we will not due to their behavior already being captured by the UU-group.

The B_p operators themselves are slightly different than the A_s operators in the TC model. In the TC model, the ground state is the $+1$ eigenstate of the B_p operators, which implies that they act as the identity on the GSM. However, for the DS model, the ground states are instead -1 eigenstates of the B_p operators. This implies that B_p operators will act as the negative identity operator on the GSM. However, we have shown that there is already such an operator which is part of the GSM, namely -1. So the behavior of the B_p operator is already captured by the UU-group, so we need not include them in the UU-group.

When we selected our operators to represent either positive or negative chirality loops around the x and y axes, we needed to specify a direction to define the left-hand side and right-hand side of the loop. For the $U_x^{\pm}$ loops, we chose right to left, and for the $U_y^{\pm}$ loops, we chose top to bottom. We will now investigate if loops traveling in the opposite directions need to be included in the UU-group. We have already created a closed group; however, if these operators bring some interesting behavior to the UU-group, we would need to include them. It turns out it is not necessary, due to the fact that a loop traveling around the x-axis in the opposite direction with positive chirality acts on the GSM like either U_x^- or $-U_x^-$. We illustrate this by considering the following figure:

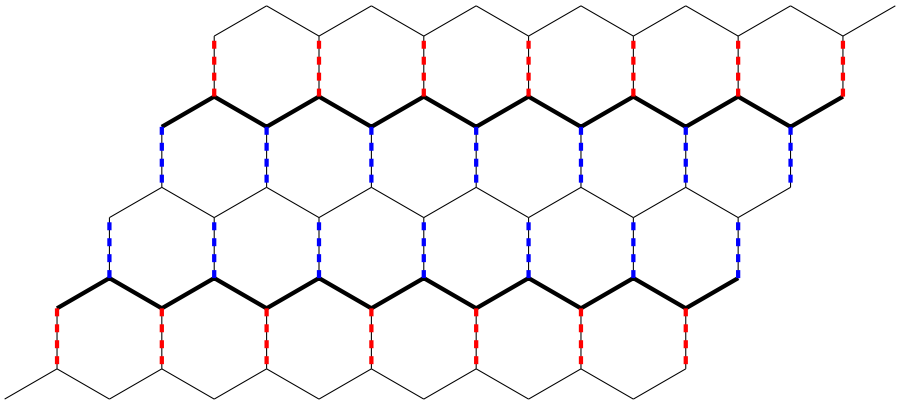


Figure 4.2.9

Investigating the figure, we see that the region enclosed by the two loops is comprised of N hexagons. We know that the B_p operators can deform loops, and furthermore, we can write this loop as a product of B_p operators:

$$U_x^+ U_{\mathcal{L}} \simeq \prod_{p \in C} B_p \tag{4.65}$$

where $U_{\mathcal{L}}$ is the right-to-left loop operator, C is the enclosed region and $\simeq$ denotes equal in the GSM. We note that we cannot say they are exactly equal, as this behavior is only exactly true in the GSM. From before, we know that the B_p operators act on the GSM like -1. So this operator will be equivalent to $(-1)^N$ where N is the number of enclosed hexagons. If $U_x^+ U_{\mathcal{L}} \simeq 1$, then $U_{\mathcal{L}}$ behaves like the inverse of U_x, and if $U_x^+ U_{\mathcal{L}} \simeq -1$, then $U_{\mathcal{L}}$ behaves like $-1(U_x^+)^{-1}$. In both cases, there are already operators in the UU-group that capture this behavior, so we need not directly include them in the UU-group. They will, however, be a part of the larger group of all loop operators, defined as:

$$G_{\text{All loops}} = G_{UU} \times_{p\in P\setminus\{\tilde{p}\}} \{E, B_p\} \times_{s\in S\setminus\{\tilde{s}\}} \{E, A_s\} \tag{4.66}$$

We note again that the use of the direct product is really only valid whenever the operators are applied to $|\text{loop}\rangle$ states. As with the TC-model, we need to remove any single one of the B_p and A_s operators as they are not completely independent.

4.2.3.2 Characters of the UU-group

To derive the characters, we will first redefine the GSM more carefully. We can define the $(++)$ ground state as we did for the TC model:

$$|+,+\rangle = \left(\prod_i P_i\right) |\uparrow\uparrow\uparrow \ldots \uparrow\rangle \tag{4.67}$$

We note that the projection operators in this case are not the same as for the TC model. We will treat the other ground states slightly differently. For the TC model, we argued that applying the U-operators should flip the loop parities. We then proved that not only would the states be flipped into each other, but we would also not get any phase shift when doing so. In this case, we will not argue in the same way. Instead, we will avoid the issue of a potential phase change by defining the other states in terms of applying U_x and U_y operators to the $|+,+\rangle$ state. This is a subtle change, but it is done to carefully account for when the ϕ operator will produce a given phase change. Thus, we define all the ground states as follows:

$$\begin{aligned} |+,+\rangle &= \left[\prod_p P_p\right] |\uparrow\uparrow\uparrow \ldots \uparrow\rangle\,, \\ |-,+\rangle &= U_x\,|+,+\rangle\,, \\ |+,-\rangle &= U_y\,|+,+\rangle\,, \\ |-,-\rangle &= U_xU_y\,|+,+\rangle\,. \end{aligned} \tag{4.68}$$

Note that the ordering for $|-,-\rangle$ is strict, as exchanging these would result in picking up a ϕ operator. Similar to before, we can determine that the ground states are eigenstates of the V_x and V_y operators. This is most easily seen by applying the operators to $|+,+\rangle$.

$$\begin{aligned} V_x\,|+,+\rangle &= V_x[\prod_p P_p]\,|\uparrow\uparrow\uparrow \ldots \uparrow\rangle \\ &= [\prod_p P_p]V_x\,|\uparrow\uparrow\uparrow \ldots \uparrow\rangle = |+,+\rangle \end{aligned} \tag{4.69}$$

The same applies to V_y, and we can reason in the same way as we did for the TC model to conclude that the GSM are eigenstates of the V operators. We now start to derive the characters for the UU group. The character of any given group element in the GSM is as follows:

$$\begin{aligned}\chi(g) &= \mathrm{Tr}(\Gamma(g)) \\ &= \langle +,+|\, U_g \,|+,+\rangle + \langle -,+|\, U_g \,|-,+\rangle \\ &\quad + \langle +,-|\, U_g \,|+,-\rangle + \langle -,-|\, U_g \,|-,-\rangle \\ &= \langle U_g\rangle_{+,+} + \langle U_x^{-1}U_gU_x\rangle_{+,+} \\ &\quad + \langle U_y^{-1}U_gU_y\rangle_{+,+} + \langle U_y^{-1}U_x^{-1}U_gU_xU_y\rangle_{+,+}\end{aligned} \tag{4.70}$$

The inverse elements appear because the U-operators are unitary, so $U_x^\dagger = U_x^{-1}$. Furthermore, $U_x^{-1} = U_x^3$. From now on, we will drop the $(+,+)$ subscript in the expectation values. As with the TC model, we can eliminate most of the characters by exploiting the fact that U_x or U_y to an odd power will change one ground state into another. We can generalize this by calculating the expectation value:

$$\langle U_g\rangle = \langle +,+|\, \phi^{g_0}U_x^{g_1}U_y^{g_2} \,|+,+\rangle = 0. \tag{4.71}$$

We assume that either g_1 or g_2 (or both) are odd. This will transform the ket into a different state than the bra, resulting in the expectation value being 0. Secondly, we can investigate the product $h^{-1}gh$. If $h = (h_0, h_1, h_2)$, then we automatically know the second and third numbers in h^{-1}. This is because these are simply additive mod 4. So in order for $hh^{-1} = (0,0,0)$, we must have that $h^{-1} = (x, (4-h_1) \mod 4, (4-h_2) \mod 4)$. In general, when two UU-group elements are exchanged, their second and third numbers are conserved, but there may be a ϕ phase as a result. We therefore find that $h^{-1}gh = \phi^n g$. This implies that if g is comprised of either an odd number of U_x operators or an odd number of U_y operators, its character is 0. As with the TC model, this significantly reduces the number of conjugacy classes that may have a nonzero character. The ones we are left with are: $\{1\}$, $\{-1\}$, $\{\phi\}$, $\{-\phi\}$, $\{\pm V_x\}$, $\{\pm V_y\}$, $\{\pm V_xV_y\}$, $\{\pm\phi V_x\}$, $\{\pm\phi V_y\}$, and $\{\pm\phi V_xV_y\}$.

We can first see that ϕV_x will have the same character as V_x due to $\langle\phi V_x\rangle = \langle\phi\rangle\chi(V_x)$. So we only need to consider the first 7 conjugacy classes. We can furthermore argue that the V-operators will have character 0 for the same reason as the TC-model had character 0 for these. So we are left with 4 nonzero characters:

$$\chi(1) = 4\chi(-1) = -4\chi(\phi) = 4\langle\phi\rangle\chi(-\phi) = -4\langle\phi\rangle. \tag{4.72}$$

We can first see that ϕV_x will have the same character as V_x due to $\langle\phi V_x\rangle = \langle\phi\rangle\chi(V_x)$. So we only need to consider the first 7 conjugacy classes. Furthermore, we can argue that the V-operators will have character 0 for the same reason that the TC model had character 0 for these. Thus, we are left with 4 nonzero characters:

$$\begin{aligned}\chi(1) &= 4, \\ \chi(-1) &= -4, \\ \chi(\phi) &= 4\langle\phi\rangle, \\ \chi(-\phi) &= -4\langle\phi\rangle.\end{aligned} \tag{4.73}$$

where $\phi = \pm i$. The characters could thus correspond to either of the two 4D irreps of the UU-group. These two irreps are practically identical, as the only difference in the characters is a sign change for the characters of ϕ and $-\phi$. The

important part is that we have determined the symmetry group of the GSM, and we have found that the GSM transforms under group operations in the UU-group like one of its four-dimensional irreps, which implies that we have explained the degeneracy.

4.2.4 DSL - Model

The DS-model, like the TC-model, can be transformed into a 1D system by removing the periodicity in the y-direction and arranging the system into a series of honeycombs in a line. We will start with the full UU-group and consider which of the operators will remain as symmetry operators and which will not.

4.2.4.1 Ground States and Reduced UU-group

We first note that the U_x^+ operator remains as a symmetry operator. This can be seen by the fact that it shares two spins with every A_s operator, and it commutes with B_p as the overlap between the two operators is the same as in the DS model. This is shown in the following figure, where the phase terms have been suppressed, as they only involve σ_z and thus trivially commute with the A_s operators.

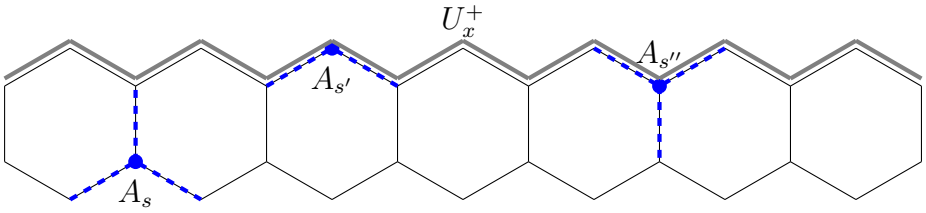


Figure 4.2.10: The overlap between U_x^+ and a few representative A_s. The A_s operators all share either zero or two spins with U_x^+

In conclusion, the parity number in the x-direction remains a conserved quantum number, giving us at least twofold degeneracy. On the other hand, U_y^- will not remain as a symmetry operator. This is evident from the following figure: the marked A_s operators share only a single spin with U_y^-, and therefore, will not commute with it.

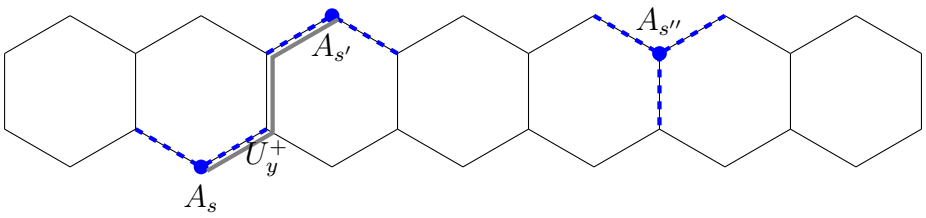


Figure 4.2.11: The overlap between U_y^+ and a few representative A_s. $A_{s''}$ commutes, due to not sharing spins, however $A_{s'}$ and A_s share only one spin with U_y^+ and will thus not commute with it

This implies that there will not be a conserved loop parity number in the y-direction, which in turn will imply there is only a twofold degeneracy in the GSM. On the other hand, $(U_y^+)^2 = V_y$ will remain a symmetry operator, as it commutes

trivially with every A_s operator and shares either zero or two spins with every B_p operator. The implications of this are the following: we will not get the ϕ operator as a member of the symmetry group. Furthermore, the generators of the group will be U_x^+ and V_y. In that sense, this looks more like the reduced UV-group for the TCL model. The important difference between the two is that the V_x element cannot be removed from the group, as it is generated by $(U_x^\pm)^2$. As with the UU-group, $U_x^\pm V_y = -V_y U_x^\pm$, which implies that the -1 phase element will remain a part of the group. All this together lets us define a general group element as:

$$\begin{aligned} G_{UU,\text{Reduced}} &\equiv \{g = (g_0, g_1, g_2) \mid g_0, g_2 \in \{0,1\}, g_1 \in \{0,1,2,3\}\}, \\ U_g &\equiv (-1)^{g_0} (U_x^+)^{g_1} (V_y)^{g_2}. \end{aligned} \tag{4.74}$$

This is a 16-element subgroup of the UU-group for the TC model. The commutation relations for the operators are known, so we can derive the multiplication rules for two general operators:

$$\begin{aligned} &\forall g, g' \in G_{UU,\text{Reduced}}, \\ &U_g U_{g'} = (-1)^{g_0} (U_x^+)^{g_1} (V_y)^{g_2} (-1)^{g_0'} (U_x^+)^{g_1'} (V_y)^{g_2'} \\ &\qquad = (-1)^{(g_0 + g_0' + g_1' g_2) \mod 2} (U_x^+)^{(g_1 + g_1') \mod 4} V_y^{(g_2 + g_2') \mod 2} \\ \implies &\; g * g' = ([g_0 + g_0' + g_1' g_2] \mod 2, [g_1 + g_1'] \mod 4, [g_2 + g_2'] \mod 2)\,. \end{aligned} \tag{4.75}$$

As before, we can prove this is a group by recognizing it is closed due to the modulo operators in the multiplication rules. Since it is closed, and the identity element exists ($E = (0,0,0)$), we can prove the last axiom by simply creating a multiplication table and verifying that every element has an inverse. The multiplication table is given in Fig. 4.2.12.
Inspecting the multiplication table, we can deduce that the multiplication rules we defined for the reduced UU-group yield a valid group. With this information, we can identify which of the tabulated groups of order 16 is isomorphic to the reduced UU-group.

The number of groups of order 16 is 14. The reduced UU-group is non-abelian, and when we filter for the non-abelian groups of order 16, we are left with 9 candidates. The UU-group has a minimum number of generators equal to 2; in our case, we use U_x^+ and V_y as those generators. Filtering for this, we are left with 6 candidate groups. As with the other groups we have investigated, we can derive the conjugacy classes of a group given we know the elements and the binary operator. The reduced UU-group has 10 conjugacy classes, which leaves us with 3 candidate groups. Finally, we can inspect how many elements there are of each order to identify the group. The reduced UU-group has 1 element of order 1, 7 elements of order 2, and 8 elements of order 8. This leaves us with a single possible group, which in GAP is called SmallGroup(16,3) [17]. Alternatively, the group may be defined as the semidirect product $(\mathbb{Z}_2 \times \mathbb{Z}_2) \rtimes \mathbb{Z}_4$.

This group is different from the other groups we have derived in several ways. Examining the group in GAP, we find that the derived subgroup and the Frattini subgroup are both isomorphic to the same group. We can start by investigating the center. We know the phase elements E and $-E$ should be part of the center;

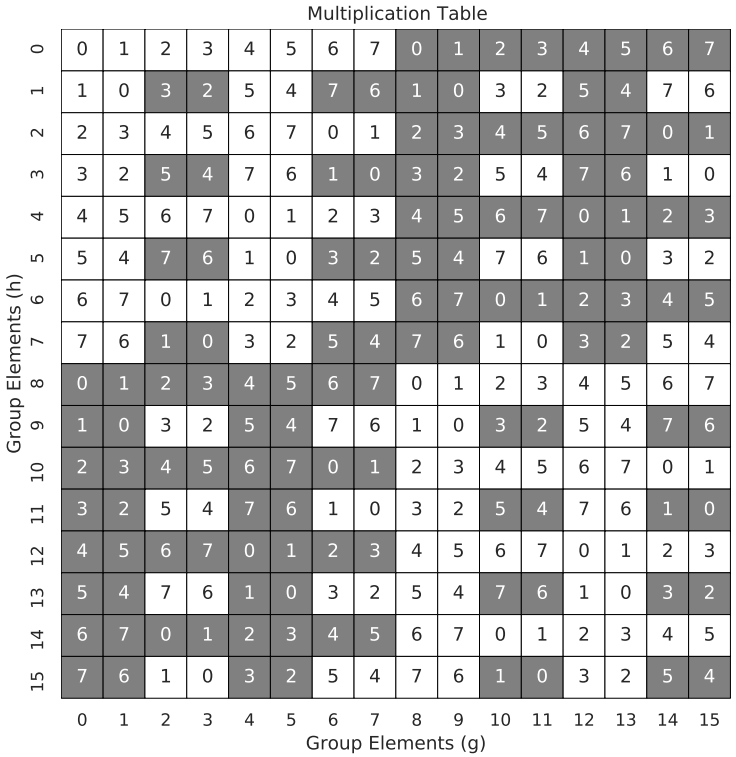


Figure 4.2.12: Multiplication table for the reduced UU group. The numbers are calculated as $n = 2g_1 + g_2$, and the color signifies the phase, where white is $+1$ and gray is -1.

however, in this case, it turns out there is one more member. V_x commutes with every element of the group and is therefore also part of the center. This means that the center becomes $C = \{E, -E\} \times \{E, V_x\}$, which is isomorphic to the Klein four-group. The derived subgroup, as with all the other groups, is just the phase elements $\{E, -E\}$, which is isomorphic to the cyclic group $\mathbb{Z}_2$. Finally, the Frattini subgroup will have the phase elements as members of the group, and it will have V_x, as it is generated by U_x (and U_x in turn cannot be generated by V_x). So it will be equal to the center. This is different from the full UU-group, as in that case, the center coincided with the derived subgroup, not the Frattini subgroup.

4.2.4.2 Characters of the Reduced UU-Group

As mentioned before, there are two ground states of the DSL model. We define the two as:

$$\begin{aligned} |+\rangle &= \prod P_i \, |\uparrow\uparrow \ldots \uparrow\rangle \,, \\ |-\rangle &= U_x^+ \, |+\rangle \,. \end{aligned} \tag{4.76}$$

As for the TCL model, $|+\rangle$ is the $+1$ eigenstate of V_y while $|-\rangle$ is the -1 eigenstate of V_y. $U_x^{\pm}$ can be used to toggle between the two eigenstates. Finally, V_x acts on the two ground states as the identity operator. This can be understood by the

fact that the loop states comprising the ground states are essentially the same as for the 2D model, except the -1 parity loops in the y-direction are not legal loop states in the 1D case. This implies V_x will share an even number of spins with all of them, which in turn implies V_x will act as the identity operator. Finally, we can exploit a similar property for the UU-group to simplify calculations. If we start with the group element h, and perform the transformation: $g^{-1}hg = h'$, then h' retains the same h_1 and h_2. This implies if h_1 is odd, the operator $U_g^{-1}U_hU_g$ will transform one ground state into the other, yielding a zero matrix element.

This implies there are six conjugacy classes that might have nonzero characters. In operator form, those conjugacy classes are: $\{1\}$, $\{-1\}$, $\{V_x\}$, $\{-V_x\}$, $\{\pm V_y\}$, $\{\pm V_xV_y\}$. The first four are the simplest. $\chi(1) = 2$ and $\chi(-1) = -2$. V_x acts on the GSM as the identity, which implies $\chi(V_x) = 2$ and $\chi(-V_x) = -2$. Finally, we calculate the characters for the last two conjugacy classes:

$$\begin{aligned}\chi(V_y) &= \langle +|\, V_y \,|+\rangle + \langle -|\, V_y \,|-\rangle \\ &= 1 - 1 = 0, \\ \chi(V_xV_y) &= \langle +|\, \cancel{V_x} V_y \,|+\rangle + \langle -|\, \cancel{V_x} V_y \,|-\rangle \\ &= \langle +|\, V_y \,|+\rangle + \langle -|\, V_y \,|-\rangle = 0.\end{aligned} \tag{4.77}$$

In conclusion, the character for every element except for 1, -1, V_x, and $-V_x$ is zero. When compared to the listed characters for the tabulated group which is isomorphic to the reduced UU-group, we find that these characters correspond exactly to one of the two-dimensional irreducible representations of the tabulated group.

CHAPTER

FIVE

CONCLUSIONS

5.1 Conclusions

In this thesis, we applied group theoretical methods to both analytical and numerical models, significantly enhancing our understanding and efficiency in these areas. These methods were particularly useful in identifying symmetries and degeneracies in complex quantum systems.

The application of group theoretical methods significantly aided both analytical and numerical approaches. In the numerical case, it helped speed up the calculations and allowed us to analyze larger systems, which facilitated easier spectrum analysis. We used the fingerprinting method to find the irreducible representations of the ground state manifold (GSM), allowing us to associate the GSM symmetries with the exactly solvable Majumdar-Ghosh eigenstates for all points in the dimer phase.

For the analytical models, group theoretical methods successfully explained the degeneracies of both systems. For the Toric Code (TC) model, the four-fold degeneracy stems from the 4-dimensional irreducible representation (irrep) of the UV-group, which is isomorphic to "the inner holomorph of D_8". For the Doubled Semion (DS) model, the degeneracy is explained by the UU-group, which is isomorphic to "The unitriangular matrix group: $UT(3, \mathbb{Z}_4)$". Although the systems are similar, they have quite different symmetry groups. However, we discovered a holomorph from the UU-group to the UV-group. In both cases, the groups could be explained as projective representations of the UV- and UU-groups respectively. Additionally, we were able to express the group of all loop operators for the TC model as a direct product of the groups $\{E, A_s\}$, $\{E, B_p\}$, and G_{UV}, and for the DS model as a direct product of $\{E, A_s\}$, $\{E, B_p\}$, and G_{UU}. Since this was a direct product, investigating only the smaller "core" groups proved fruitful.

The TCL and DSL models inherited many properties from their parent systems, but not all. Both models had half the degeneracy due to losing the U-loop operator in the y-direction. However, the reduced groups still had 2-dimensional irreps due to the V_y operator, which led to anti-commutation. The reduced UV-group inherited the "extraspecial" property from the UV-group, as it was isomorphic to D_8. Initially, this property was also considered for the DS model, but it turned out that the DS models retained only a few "extraspecial" properties. The DS model had the center and derived subgroup coincide, while the DSL model

had its Frattini subgroup and center coincide.

REFERENCES

[1] Lakshya Bhardwaj et al. *Lectures on Generalized Symmetries*. 2023. arXiv: 2307.07547 [hep-th].

[2] Jacob Linder. *Intermediate Quantum Mechanics*. Peer reviewed by Dr. Takehito Yokoyama, Tokyo Institute of Technology, Japan. Bookboon, 2017. ISBN: 978-87-403-1783-1. URL: https://bookboon.com.

[3] Steven M. Girvin and Kun Yang. *Modern Condensed Matter Physics*. First published 2019, Printed in the United Kingdom by TJ International Ltd, Padstow, Cornwall. Cambridge, UK; New York, NY: Cambridge University Press, 2019. ISBN: 978-1-107-13739-4. DOI: 10.1017/9781316480649. URL: https://www.cambridge.org/9781107137394.

[4] Nicholas Chancellor and Stephan Haas. "Using the J1–J2 quantum spin chain as an adiabatic quantum data bus". In: *New Journal of Physics* 14.9 (Sept. 2012), p. 095025. DOI: 10.1088/1367-2630/14/9/095025. URL: https://dx.doi.org/10.1088/1367-2630/14/9/095025.

[5] Sebastian Eggert. "Numerical evidence for multiplicative logarithmic corrections from marginal operators". In: *Phys. Rev. B* 54 (14 Oct. 1996), R9612–R9615. DOI: 10.1103/PhysRevB.54.R9612. URL: https://link.aps.org/doi/10.1103/PhysRevB.54.R9612.

[6] Kiyomi Okamoto and Kiyohide Nomura. "Fluid-dimer critical point in S = 12 antiferromagnetic Heisenberg chain with next nearest neighbor interactions". In: *Physics Letters A* 169.6 (1992), pp. 433–437. ISSN: 0375-9601. DOI: https://doi.org/10.1016/0375-9601(92)90823-5. URL: https://www.sciencedirect.com/science/article/pii/0375960192908235.

[7] Chanchal K. Majumdar and Dipan K. Ghosh. "On Next-Nearest-Neighbor Interaction in Linear Chain. II". In: *Journal of Mathematical Physics* 10.8 (Aug. 1969), pp. 1399–1402. ISSN: 0022-2488. DOI: 10.1063/1.1664979. eprint: https://pubs.aip.org/aip/jmp/article-pdf/10/8/1399/19219508/1399_1_online.pdf. URL: https://doi.org/10.1063/1.1664979.

[8] Austin G. Fowler et al. "Surface codes: Towards practical large-scale quantum computation". In: *Physical Review A* 86.3 (Sept. 2012). ISSN: 1094-1622. DOI: 10.1103/physreva.86.032324. URL: http://dx.doi.org/10.1103/PhysRevA.86.032324.

[9] A.Yu. Kitaev. "Fault-tolerant quantum computation by anyons". In: *Annals of Physics* 303.1 (Jan. 2003), pp. 2–30. ISSN: 0003-4916. DOI: 10.1016/s0003-4916(02)00018-0. URL: http://dx.doi.org/10.1016/S0003-4916(02)00018-0.

[10] C. W. von Keyserlingk, F. J. Burnell, and S. H. Simon. "Three-dimensional topological lattice models with surface anyons". In: *Physical Review B* 87.4 (Jan. 2013). ISSN: 1550-235X. DOI: 10.1103/physrevb.87.045107. URL: http://dx.doi.org/10.1103/PhysRevB.87.045107.

[11] Michael A. Levin and Xiao-Gang Wen. "String-net condensation: A physical mechanism for topological phases". In: *Physical Review B* 71.4 (Jan. 2005). ISSN: 1550-235X. DOI: 10.1103/physrevb.71.045110. URL: http://dx.doi.org/10.1103/PhysRevB.71.045110.

[12] K. J. Satzinger et al. "Realizing topologically ordered states on a quantum processor". In: *Science* 374.6572 (2021), pp. 1237–1241. DOI: 10.1126/science.abi8378. eprint: https://www.science.org/doi/pdf/10.1126/science.abi8378. URL: https://www.science.org/doi/abs/10.1126/science.abi8378.

[13] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Vol. 3. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2013. ISBN: 1421407949,9781421407944.

[14] A. W. Joshi. *Elements of Group Theory for Physicists*. Third. Fourth Reprint, 1988. New Delhi, India: Wiley Eastern Limited, 1982. ISBN: 0-85226-448-8.

[15] T. Gannon. *Moonshine beyond the Monster: The Bridge Connecting Algebra, Modular Forms and Physics*. Cambridge Monographs on Mathematical Physics. Cambridge University Press. ISBN: 9781139457804.

[16] Alexander Wietek, Michael Schuler, and Andreas M. Läuchli. *Studying Continuous Symmetry Breaking using Energy Level Spectroscopy*. 2017. arXiv: 1704.08622 [cond-mat.str-el].

[17] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.11.1*. https://www.gap-system.org. 2021.

[18] Richard M. Foote David S. Dummit. *Abstract Algebra*. John Wiley and Sons, Inc., 2004. ISBN: 0-471 -43334-9.

[19] Tjark Heitmann and Jürgen Schnack. "Combined use of translational and spin-rotational invariance for spin systems". In: *Physical Review B* 99.13 (Apr. 2019). ISSN: 2469-9969. DOI: 10.1103/physrevb.99.134405. URL: http://dx.doi.org/10.1103/PhysRevB.99.134405.

[20] Phillip Weinberg and Marin Bukov. "QuSpin: a Python package for dynamics and exact diagonalisation of quantum many body systems. Part II: bosons, fermions and higher spins". In: *SciPost Phys.* 7 (2019), p. 020. DOI: 10.21468/SciPostPhys.7.2.020. URL: https://scipost.org/10.21468/SciPostPhys.7.2.020.

[21] B Bauer et al. “The ALPS project release 2.0: open source software for strongly correlated systems”. In: *Journal of Statistical Mechanics: Theory and Experiment* 2011.05 (May 2011), P05001. DOI: 10.1088/1742-5468/2011/05/P05001. URL: https://dx.doi.org/10.1088/1742-5468/2011/05/P05001.

[22] Thomas H. Cormen et al. *Introduction to Algorithms*. MIT press, 2009. ISBN: 978-0-262-03384-8.

[23] Manuel Calixto, Alberto Mayorgas, and Julio Guerrero. “Role of mixed permutation symmetry sectors in the thermodynamic limit of critical three-level Lipkin-Meshkov-Glick atom models”. In: *Phys. Rev. E* 103 (1 Jan. 2021), p. 012116. DOI: 10.1103/PhysRevE.103.012116. URL: https://link.aps.org/doi/10.1103/PhysRevE.103.012116.

[24] NumPy Developers. *NumPy*. Accessed: 2024-05-15. NumPy Developers. 2024. URL: https://numpy.org/doc/stable/.

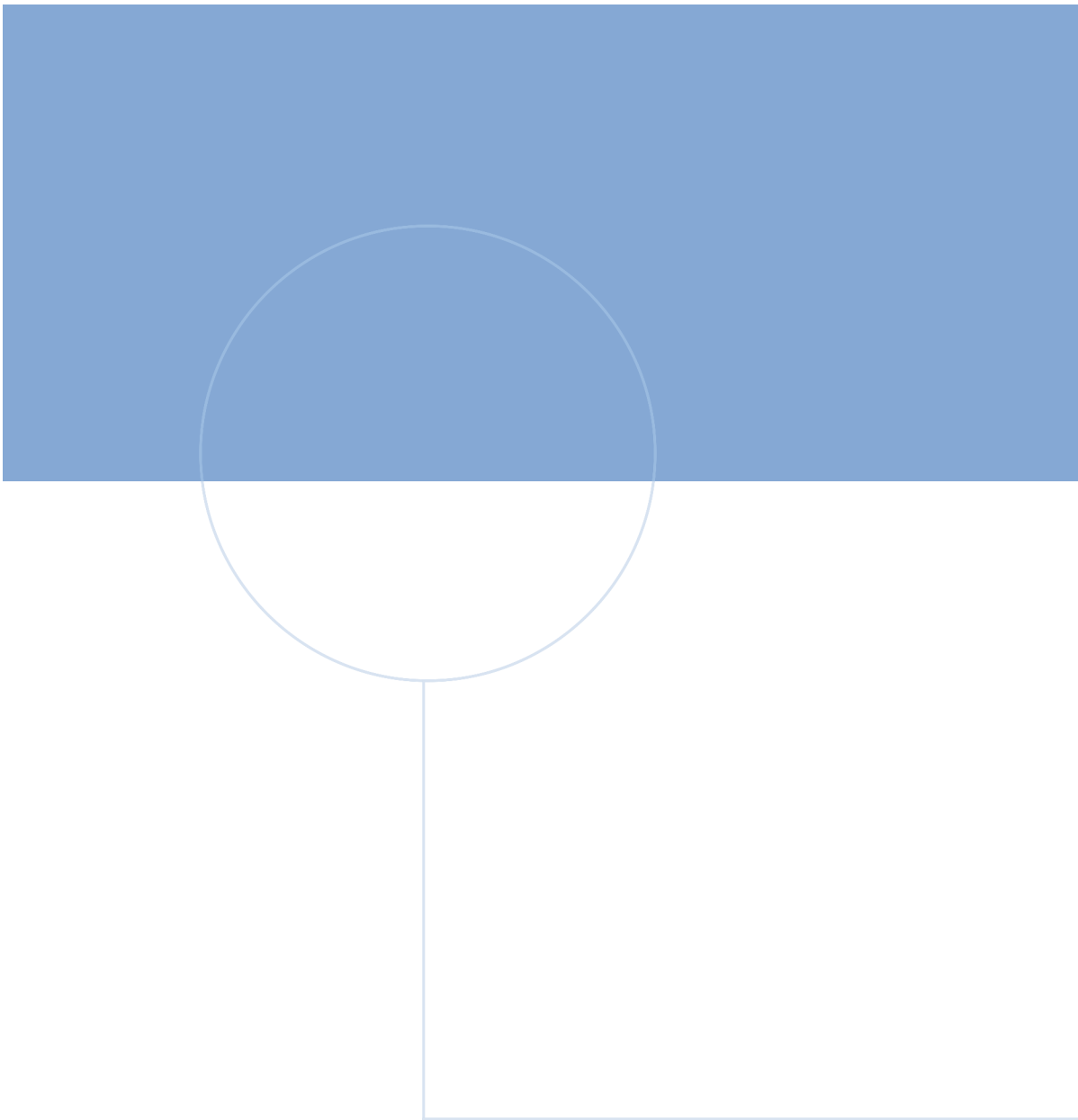
[25] Recep Eryiğit, Resul Eryigit, and Yigit Gunduc. “Quantum Phase Transitions and Entanglement in J1-J2 Model”. In: *International Journal of Modern Physics C - IJMPC* 15 (Oct. 2004), pp. 1095–1103. DOI: 10.1142/S0129183104006558.

[26] Mark L. Lewis. *Semi-extraspecial Groups*. 2017. arXiv: 1709.03857 [math.GR].

APPENDICES

A - GITHUB REPOSITORY

The codebase used for the project together with figures and some experimental scripts are available publicly on Github. The main files used for the project is `diagonalizer.py` for diagonalization, `irrep_generator.py` for numericaly creating irreps, `Irrep_decompositon_method.py` for preforming the fingerprinting method and finally `UV_group.py` and `UU_DS_group.py` for analysis. The ".g" files are GAP scripts used for group filtering.



NTNU

Norwegian University of
Science and Technology