Large-scale simulation of deep neural quantum states

Ao Chen

Angaben zur Veröffentlichung / Publication details:

Chen, Ao. 2025. “Large-scale simulation of deep neural quantum states.” Augsburg: Universität Augsburg.

Large-scale simulation of deep neural quantum states

Dissertation

zur Erlangung des akademischen Grades
Dr. rer. nat.

eingereicht an der
Mathematisch-Naturwissenschaftlich-Technischen Fakultät
der Universität Augsburg

von
Ao Chen

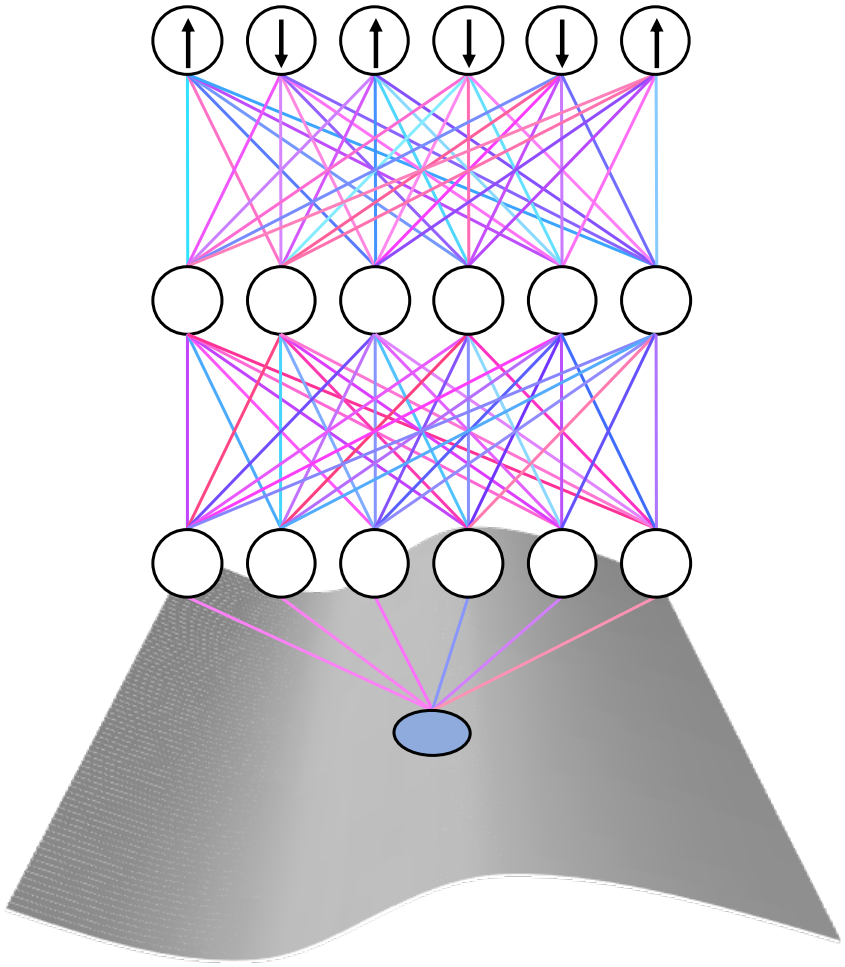
Augsburg, März 2025

Supervisor: Markus Heyl

Reviewers:

- Markus Heyl
- Christoph Weber
- Lode Pollet

Date of oral examination: 28.04.2025



Abstract

The study of quantum many-body systems remains a central challenge in condensed matter physics due to the exponential growth of the Hilbert space with system size. Traditional numerical approaches, such as exact diagonalization, quantum Monte Carlo, and tensor networks, often struggle with respective computational limitations, particularly in two and three-dimensional systems with strong correlations. In recent years, deep learning has emerged as a powerful alternative, offering new perspectives on quantum state representation and simulation. This thesis explores the large-scale simulation of deep neural quantum states (NQSs), leveraging artificial neural networks to approximate quantum many-body wave functions efficiently.

We begin by introducing the foundational concepts of NQS and their potential in overcoming the curse of dimensionality in quantum many-body problems. Variational Monte Carlo (VMC) is employed as the primary optimization framework, allowing the ground-state properties of complex Hamiltonians to be explored stochastically. The stochastic reconfiguration (SR) and minimum-norm SR (MinSR) are then introduced for accurate and efficient optimization of deep NQSs. We further investigate various neural network architectures, including feed-forward neural networks, restricted Boltzmann machines, convolutional neural networks, and transformer-based wave functions, assessing their expressivity and efficiency in encoding quantum correlations in strongly correlated quantum matters. The symmetry projection of NQSs starting from the group theory is introduced in detail, and particular attention is given to symmetry-preserving network architectures.

With the help of VMC, MinSR, and symmetry projection, the NQS achieves outstanding accuracy and outperforms existing numerical methods in several benchmark models of spin systems. We then show its application in quantum spin liquids (QSLs), the study of which relies heavily on the advances of computational methods. The NQS provides an accurate estimation of phase diagrams and energy gaps of several QSL candidates in the square J_1-J_2 model, the triangular J_1-J_2 model, and the Shastry-Sutherland model, leading to a deeper understanding of the emergence of QSL in frustrated magnets.

In fermion systems, fermionic mean-field wavefunctions are introduced to combine with the NQS for efficient expression of fermion sign structures. We then discuss the application of fermionic NQSs in the Fermi-Hubbard model, which reflects the mechanism of the Mott-insulator transition and probably high-temperature superconductivity. The fermionic NQS successfully reproduces the insulator transition and the spin density wave predicted by other numerical methods and further observes the possible superconducting order in the Hubbard model.

These findings highlight the immense potential of deep NQSs in quantum many-body physics. By combining large-scale neural networks with variational quantum algorithms, we make the NQS a powerful tool in solving quantum many-body systems, paving the way for future advancements in the computational study of quantum materials.

Publications and preprints

- Ao Chen and Markus Heyl
  "Empowering deep neural quantum states through efficient optimization"
  Nat. Phys. 20, 1476 (2024)

- Ao Chen*, Vighnesh Naik*, and Markus Heyl
  "Convolutional transformer wave functions"
  arXiv:2503.10462 (2025)

- AO Scheie, Minseong Lee, Kevin Wang, P Laurell, ES Choi, D Pajerowski, Qingming Zhang, Jie Ma, HD Zhou, Sangyun Lee, SM Thomas, MO Ajeesh, PFS Rosa, Ao Chen, Vivien S Zapf, M Heyl, CD Batista, E Dagotto, JE Moore, and D Alan Tennant
  "Spectrum and low-energy gap in triangular quantum spin liquid $NaYbSe_2$"
  arXiv:2406.17773 (2024)

- Hongzheng Zhao, Ao Chen, Shu-Wei Liu, Marin Bukov, Markus Heyl, and Roderich Moessner
  "Learning effective Hamiltonians for adaptive time-evolution quantum algorithms"
  arXiv:2406.06198 (2024)

Contents

Chapter 1

Introduction

1.1 Quantum many-body problem

It has been an ever-persisting quest in condensed matter and quantum many-body physics to capture the essence of quantum many-body systems that is covered behind their exponential complexity. It concerns the study of systems composed of a large number of interacting quantum particles, such as quantum spins and electrons. The quantum many-body problem is of fundamental importance because it underpins a wide range of physical applications, including superconductivity, quantum Hall effects, and topological insulators. Understanding and predicting the properties of such systems is one of the most profound challenges in theoretical and computational physics.

The study of quantum many-body problems dates back to the early days of quantum mechanics, especially in condensed matter physics. Landau's Fermi liquid theory [1], for example, provided an adiabatic framework for understanding interacting fermions in metals. Further examples include the theory of Landau levels revealing the underlying quantum nature of the integer quantum Hall effect [2], or the BCS theory explaining the phenomenon of conventional superconductivity [3]. These early successes demonstrated the power of the quantum many-body theory in explaining complex phenomena in terms of simple underlying quasi-particles as well as their adiabatic continuation.

However, during the exploration of strongly interacting systems, such as the high-temperature superconductors, it became clear that the simple picture of uncorrelated particles is no longer an appropriate description. The complexity of quantum many-body systems arises from the interplay between quantum-mechanical effects and particle interactions. While single-particle quantum mechanics can be solved exactly in most cases, the introduction of interactions often leads to correlations and entanglement that cannot be treated perturbatively. These features give rise to emergent phenomena, where the collective behavior of the system is qualitatively different from its constituents [4]. Therefore, understanding and predicting these emergent properties is one of the primary goals of modern condensed matter theory. Although there are a few successful examples, such as the explanation of the fractional quantum Hall effect based on Laughlin wavefunctions [5], the non-perturbative treatment of quantum many-body interactions is still generally beyond the reach of modern quantum theories. Therefore, computational approaches are necessary in the study of quantum many-body systems.

The central challenge in computational quantum physics is the exponential growth of the Hilbert space with the number of particles. For a system of N spin-1/2 particles, for example, the dimension of the Hilbert space is 2^N, making exact diagonalization (ED) feasible only for small systems. This "exponential wall" or "curse of dimensionality" necessitates the use of approximate methods and numerical techniques to study larger systems.

Mean-field theories, such as Hartree-Fock and density functional theory (DFT) [6], provide a starting point for understanding many-body systems by approximating the interactions between particles with an effective potential. After the mean-field treatment, the interacting systems are simplified to the non-interacting case which can be solved exactly. While these methods have been highly successful in describing weakly correlated systems, they often fail to capture the essential physics of strongly correlated systems, where the interactions dominate and lead to non-perturbative effects.

To address these limitations, a variety of advanced techniques have been developed. Quantum Monte Carlo (QMC) methods [7], for example, use stochastic sampling to estimate the properties of many-body systems, providing accurate results for certain classes of problems. However, QMC methods are often plagued by the "sign problem" [8] when frustration or fermion statistics lead to negative or complex phases in quantum systems, which is a fundamental property distinguishing a general quantum system from its classical counterparts. The "sign problem" results in an exponential computational cost in simulating many quantum systems with strong interactions.

Tensor network (TN) methods, such as the matrix product state (MPS) optimized by density matrix renormalization group (DMRG) [9], have also emerged as powerful tools for studying one-dimensional (1D) or quasi-1D systems. These methods exploit the structure of entanglement in quantum many-body states to reduce the effective size of the Hilbert space, enabling the study of systems with hundreds or even thousands of particles. Higher-dimensional tensor networks, such as projected entangled pair states (PEPS) [10] and entangled plaquette states (EPS) [11], have shown promising ability in capturing the correlations of many-body systems beyond 1D. Nevertheless, they also suffer from a high computational complexity in the contraction of high-dimensional tensors [12].

In summary, powerful computational methods have been developed to target various quantum systems. For large classes of interacting 2D or 3D quantum matter, however, they face critical limitations hindering a deeper understanding of this regime where most of the intriguing quantum many-body phenomena emerge. For instance, in Chapter 6 and Chapter 9 we discuss the quantum spin liquids and the Fermi-Hubbard model, which cannot be solved to satisfactory accuracy by traditional numerical methods. Therefore, it has become a central topic of modern quantum physics to search for new quantum many-body algorithms.

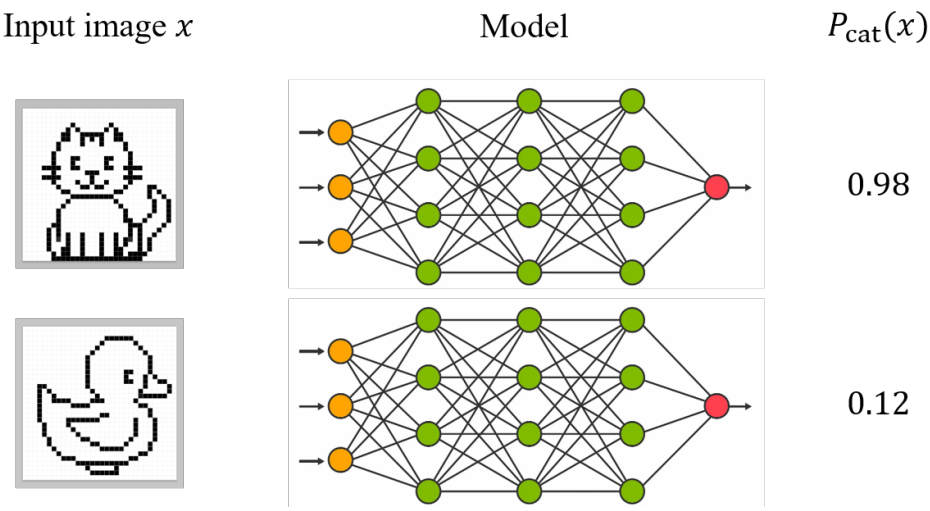


Figure 1.1: A simple exemplary computer vision task

1.2 Neural quantum state

1.2.1 Artificial neural network

The last 10 years have witnessed an impressive breakthrough of machine learning (ML) and artificial intelligence (AI) [13], pioneered by the progress in computer vision (CV). The most general task of CV is image recognition and classification through a computational model. For an input image x in the dataset, the task of CV algorithms is to provide probabilities $P_i(x)$ as an estimation for the likelihood of x belonging to the category i. In Fig. 1.1, we show a simple and paradigmatic CV task, in which the model is employed to process an input image x and provide the estimated probability of x being an image of a cat. Although these image recognition tasks appear simple for human beings, they can be difficult for brute-force approaches on computers. For an image with N pixels, the possible amount of images reaches 2^N even if the pixel color is just binary, black or white. The CV problem hence suffers from an "exponential wall" similar to quantum many-body problems.

A popular dataset for testing the performance of CV algorithms is ImageNet [14] which contains more than 14 million images and 20000 categories. The image classification of x is considered successful if the correct category i appears in the highest 5 probabilities $P_i(x)$, and this Top-5 accuracy is often utilized to evaluate the performance of a CV model. Before 2012, ImageNet benchmarks were dominated by traditional machine learning algorithms, typically support vector machine (SVM), and the best Top-5 accuracy was around 70% [15]. In 2012, a model named AlexNet [16] utilized a deep artificial neural network (ANN) to push the Top-5 accuracy to 84.7%, marking the beginning of this wave of deep learning and AI that has not yet ended. After that, the Top-5 accuracy of ImageNet has been pushed to 99% by ANNs in a decade.

The compelling power of ANN can be straightforwardly generalized to other tasks, for instance, the game of Go, an ancient strategy board game. In the game of Go, two players, one holding black pieces and the other one holding white, take turns to place pieces on the board, and the player controlling more space in the end is the winner. Despite simple

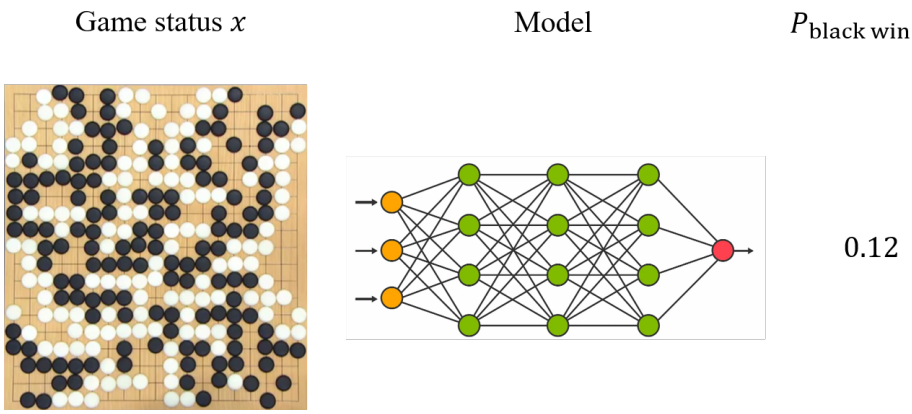


Figure 1.2: Estimating situations in the game of Go

rules, the game of Go has appeared extremely difficult for AI algorithms due to its huge search space. Although AI has been capable of beating human champions at chess in 1997, it could only reach the level of amateur Go players before 2016. Playing the game of Go can be decomposed into multiple tasks, one of which is estimating situations in the game. As shown in Fig. 1.2, the model should estimate a winning rate for an input situation similar to a CV task. Therefore, ANNs can also be utilized here to provide an estimated probability. With a suitable reinforcement learning strategy, ANNs can be trained to improve the estimation of game situations according to historical training matches. Based on this idea, AlphaGo [17] was created and for the first time defeated human champions through the power of AI in this ancient game. It was then further shown that AI can learn how to play the game purely from the rules without mimicking human players [18].

Inspired by the success of ANNs in computer science tasks such as CV and the game of Go, it is the key hope and promise that ANNs can also be utilized to solve the notorious quantum many-body problem, especially if we consider its ability to discover underlying structures without prior human knowledge.

1.2.2 Neural quantum state

To express a many-body quantum state Ψ as computable quantities, a common choice is to convert it to wavefunctions expressed on natural bases, for instance, the S_z bases in spin systems or the occupation bases in fermionic or bosonic systems. Then the quantum many-body problem can be formulated as a problem of finding the wavefunction component $\psi(s) = \langle s|\Psi\rangle$ for any input configuration s. With this map established, the full quantum state can be constructed as

$$|\Psi\rangle = \sum_s \psi(s)\,|s\rangle\,. \tag{1.1}$$

An exact expression of a quantum state $|\Psi\rangle$ requires all information in the exponential amount of wavefunction components $\psi(s)$, reflecting the "curse of dimensionality". Nevertheless, one can view $\psi(s)$ as a function of the input Fock state s and employ a model with a polynomial amount of parameters $\boldsymbol{\theta}$ to encode the essential information in this

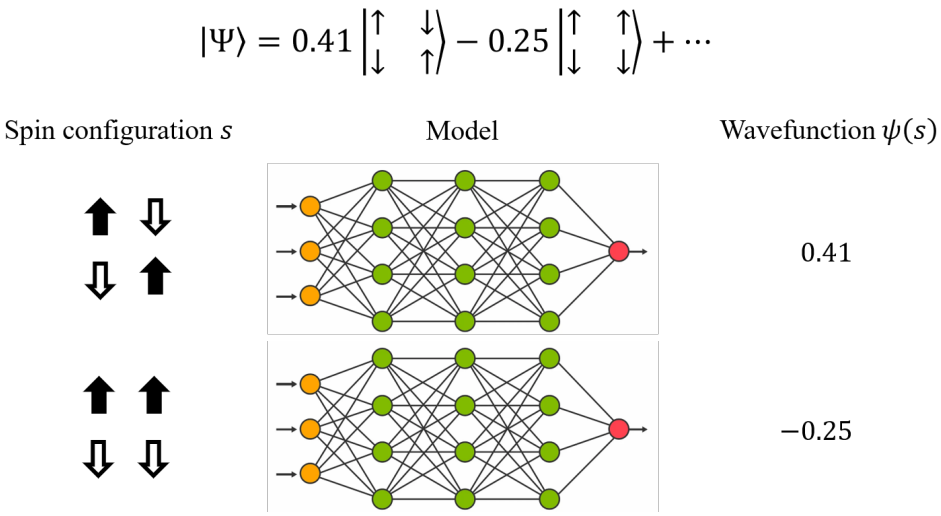


Figure 1.3: Illustration of neural quantum states

function. Then the quantum state becomes

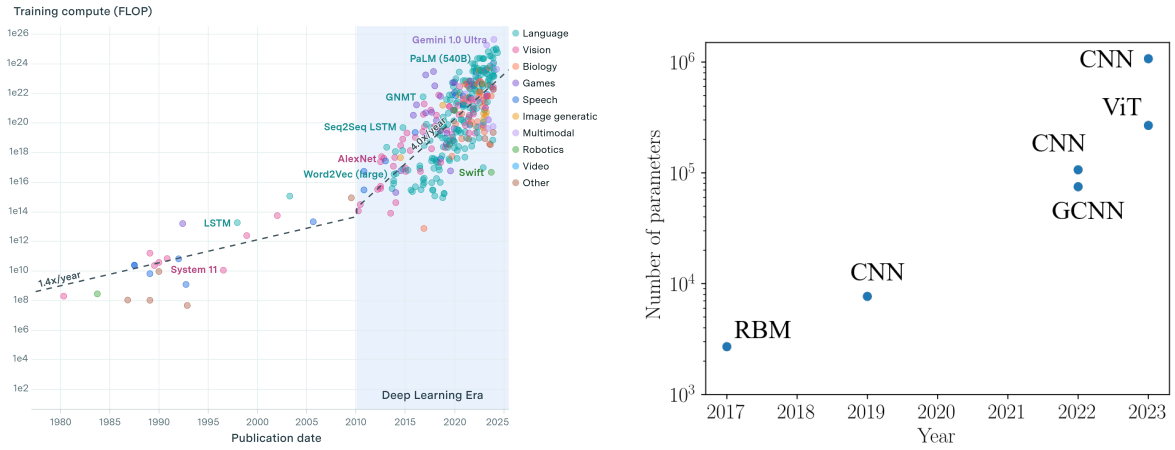
$$|\Psi_\theta\rangle = \sum_s \psi_\theta(s)\,|s\rangle\,, \tag{1.2}$$

which is controlled by the parameters $\boldsymbol{\theta}$ in ANNs. As illustrated in Fig. 1.3, the quantum many-body state formulated in this way can be described by ANNs similar to the tasks in CV and the game of Go, hence named the neural quantum state (NQS) [19]. The major difference between NQS and usual CV tasks is that NQS should produce a wavefunction $\psi(s)$ instead of a probability $P(s)$ in the output.

There is an exponential amount of Fock states s in the Hilbert space, while the number of parameters $\boldsymbol{\theta}$ in ANNs is limited. Therefore, NQS can be viewed as a compression of the full quantum state in large quantum systems. This strategy is also employed by tensor networks which compress the quantum information into a linear structure to cover the portion of the Hilbert space with area-law entanglement [9]. On the other hand, the compression done by NQS is mostly empirical without strict proof, although several works provide conclusions in some corner cases [20–31].

The expressive power of ANNs does not come from some specific physical background, but it is consistent with the "more is different" [4] philosophy in condensed matter physics. Although the computation between artificial neurons is extremely simple, a combination of an enormous amount of neurons can lead to emergent expressive power that enables ANNs to capture complex underlying structures in the studied system. In the AI community, there is a similar philosophy named connectionism, which attributes the appearance of intelligence to enormous neuron connections and leads to the prosperity of modern deep learning [13]. Therefore, deep NQS becomes necessary to provide new insights into quantum many-body problems beyond the reach of traditional methods such as DMRG. In Fig. 1.4, we show the growth of the number of parameters over time in general machine learning tasks and in NQS. The rapid growth of trainable network sizes has provided great potential for the development of ANN in multiple different fields.

In this thesis, we will provide a comprehensive introduction to the large-scale simulation of NQS with applications to spin and fermion systems. One can also read Ref. [32–35] for an overall summary of recent NQS progress.



(a) Machine learning tasks (taken from Ref. [36])

(b) Neural quantum state

Figure 1.4: The growth of ANN scale over time. The ANN scale is evaluated by the floating-point operation (FLOP) in panel (a) and the number of parameters in panel (b).

1.3 Outline

In this first chapter of the thesis, we have discussed the motivation of utilizing deep NQS in quantum many-body problems. In the following chapters, different aspects of deep NQS will be explained as outlined below.

Part I: Algorithm. Algorithms of NQS are discussed, especially in the ground state search task. The following questions are explained in this chapter.

- **Section 2: Variational Monte Carlo**. How to search for an unknown ground state $|\Psi_{\mathrm{GS}}\rangle$ given a Hamiltonian $\hat{\mathcal{H}}$ without any reference training set?
- **Section 3: Optimization**. How to compute a reliable gradient direction $\dot{\boldsymbol{\theta}}$ in variational Monte Carlo?
- **Section 4: Symmetry**. How to encode symmetries of quantum many-body systems into a general quantum state to produce a symmetrized state $|\Psi^{\mathrm{symm}}\rangle$?

Part II: Quantum Spin System. The application of NQS in quantum spin systems is discussed. The following questions are explained in this chapter.

- **Section 5: Neural quantum state architecture**. What is a good design choice of neural network architectures for the application of NQS in quantum spin systems?
- **Section 6: Neural quantum states for quantum spin liquids**. What can NQS do to help deepen our understanding of quantum spin liquids?

Part III: Fermion System. The application of NQS in fermion systems is discussed. The following questions are explained in this chapter.

- **Section 7: Mean-field fermionic wavefunction**. What are the mean-field determinant and pfaffian states $|\Psi_0\rangle$?
- **Section 8: Fermionic neural quantum state**. How to combine neural networks with mean-field fermion states $|\Psi_0\rangle$ to produce fermionic neural quantum states $|\Psi\rangle$?
- **Section 9: Neural quantum states for Fermi-Hubbard models**. What can fermionic NQS do to help deepen our understanding of the Fermi-Hubbard model and high-T_c superconductivity?

Section 10: Conclusion and outlooks. The current achievement of deep NQS in quantum many-body problems is summarized, and the outlooks of NQS development in the future are presented.

Part I

Algorithm

Chapter 2

Variational Monte Carlo

Having introduced the basic idea of NQS in the previous chapter, it is now the next step to discuss how to train an NQS to approximate the ground state of a given many-body problem. This is achieved by means of the conventional Rayleigh-Ritz method, in quantum physics usually known as the variational method. Considering a state $|\Psi_\theta\rangle$ controlled by parameters $\boldsymbol{\theta}$ as shown in Eq. (1.2), the variational method aims at finding the unknown ground-state wavefunction by tuning $\boldsymbol{\theta}$ and minimizing the energy of $|\Psi_\theta\rangle$ given by

$$E_\theta = \frac{\langle\Psi_\theta|\hat{\mathcal{H}}|\Psi_\theta\rangle}{\langle\Psi_\theta|\Psi_\theta\rangle}, \tag{2.1}$$

where $\hat{\mathcal{H}}$ denotes the Hamiltonian. Then the energy E_θ acts as the loss function in machine learning. The challenge in this optimization is that E_θ involves a summation over the full Hilbert space, leading to exponential computing cost in conventional bases. Nevertheless, the Monte Carlo method can help us to evaluate E_θ from an affordable amount of samples as we will explain in Sec. 2.1, making the variational method available even for large systems. This method is named variational Monte Carlo (VMC) [37] which helps to approximate unknown ground states in large quantum many-body systems.

2.1 Monte Carlo method

2.1.1 Classical statistical system

We denote s as configurations of the spin system. For example, in a system with 2 spin-1/2 sites, s can take 4 possible values

$$\sigma = \uparrow\uparrow, \uparrow\downarrow, \downarrow\uparrow \text{ or } \downarrow\downarrow\ . \tag{2.2}$$

Given the Hamiltonian $H(s)$ of a classical spin system, the Boltzmann weight of a specific configuration s is

$$w(s) = e^{-\beta H(s)}, \tag{2.3}$$

where $\beta = 1/T$ is the inverse temperature. The partition function is

$$Z = \sum_s w(s), \tag{2.4}$$

and the expectation value of an observable O can be expressed as

$$\langle O \rangle = \frac{1}{Z} \sum_s w(s) O(s). \tag{2.5}$$

In principle, one can sum over all configurations s to obtain the exact partition function and observables, but this is not practical when pushing to a large number of spins because the total number of configurations grows exponentially with the number of spins. The spin-1/2 system with N sites, for instance, has 2^N possible configurations. It is therefore impossible to perform a full sum to obtain exact expectation values in these systems.

2.1.2 Monte Carlo sampling

The spirit of the Monte Carlo method is to use suitable samples to estimate observables thereby avoiding the cost of brute-force methods. Usually, the aim of the Monte Carlo method in classical statistical systems is to generate samples $\{s^{\mathrm{MC}}\}$ with probability proportional to $w(s)$ such that the observables can be written as

$$\langle O \rangle = \frac{1}{N_s} \sum_{s \in \{s^{\mathrm{MC}}\}} O(s), \tag{2.6}$$

where N_s is the total number of samples.

The remaining problem is how to generate samples with the target probability. To achieve this, we begin with a random configuration s_0 and generate new configurations from old ones, denoted as $s_i \to s_{i+1}$. For every transition from s_i to s_{i+1}, there is a transition probability $T(s_{i+1}|s_i)$, represented as

$$s_0 \overset{T(s_1|s_0)}{\longrightarrow} s_1 \to \ldots \to s_i \overset{T(s_{i+1}|s_i)}{\longrightarrow} s_{i+1} \to \ldots \tag{2.7}$$

Ideally, the occurrence probability of every configuration s at time i should be proportional to $w(s)$ when i is large enough, and then we can take them as the required configurations generated by a suitable probability distribution. This process is also called Markov-chain Monte Carlo (MCMC) in which the combination of configurations $s_0 \to s_1 \to s_2 \to \ldots$ forms a chain of configurations [38].

To get a suitable transition probability $T(s_{i+1}|s_i)$, the principle of detailed balance has to be satisfied, which states that if the transition rate satisfies

$$w(s)T(s'|s) = w(s')T(s|s'), \tag{2.8}$$

then the configurations s will be distributed according to weights $w(s)$ after the Markov chain is equilibrated. To understand Eq. (2.8), we can imagine an ensemble with many spin systems in equilibrium. In every Markov step, the number of systems flowing from configuration s to s' will equal the number of inverse flows if Eq. (2.8) is satisfied, so the weights in this ensemble are left unchanged. To reach the detailed balance, there should be enough preparation steps to thermalize the ensemble and the transition process should be ergodic.

Another problem is how to propose suitable new configurations s' among an exponential amount of possible configurations. Usually, we propose new samples by flipping one

Algorithm 1 Monte Carlo method for estimating expectation values

Input: Quantity to measure $O(s)$, thermalization steps N_{thermal}, sample interval N_{interval}, total number of samples N_s
Output: Expectation value $\langle O \rangle$

```
Generate a random initial state σ
Ō ← 0
for i ← 1 to N_thermal + N_internval × N_s do
    Propose a new configuration s'
    A ← e^{-β(H(s')-H(s))}
    Generate a random number R ∈ (0, 1) with uniform distribution
    if R < A then
        s ← s'
    end if
    if i > N_thermal and (i − N_thermal) mod N_interval == 0 then
        Ō ← Ō + O(s)
    end if
end for
⟨O⟩ ← Ō/N_s
```

spin or switching neighboring spins in the existing configuration so that $w(s') \approx w(s)$. Then the number of possible s' is in line with the system size. We should also choose a suitable transition probability $T(s'|s)$ for the Markov chain. A common choice is to separate the transition probability into two parts, namely proposal probability $P(s'|s)$ and acceptance probability $A(s'|s)$. Then

$$T(s'|s) = P(s'|s)A(s'|s). \tag{2.9}$$

If s' is proposed as a local flip or neighbor exchange of spins in s, we should have $P(s|s') = P(s'|s)$, so the detailed balance equation Eq. (2.8) is simplified as

$$w(s)A(s'|s) = w(s')A(s|s'). \tag{2.10}$$

We hope the acceptance probability can be as large as possible to allow more transitions to happen, so the common choice is

$$A(s'|s) = \min\left(1, \frac{w(s')}{w(s)}\right) = \min\left(1, e^{-\beta(H(s')-H(s))}\right). \tag{2.11}$$

Then Eq. (2.10) is satisfied. As the new configuration is proposed by a local change of the existing configuration, the neighbor configurations s_i and s_{i+1} on the Markov chain in Eq. (2.7) are strongly correlated, while the Monte Carlo sampling in principle requires uncorrelated samples. To alleviate this auto-correlation problem, one usually selects spin configurations far from each other on the Markov chain.

The algorithm introduced above is the Metropolis algorithm, named after Nicholas Metropolis who proposed it in 1953 [39].

2.2 Variational method

In quantum systems with sign problems, a direct stochastic evaluation of their exact properties becomes unaffordable due to exponential computing costs. However, it does not prevent us from obtaining a result with controlled bias, which can be achieved through many methods including variational Monte Carlo (VMC).

2.2.1 Variational principle

VMC is based on the variational principle. Consider a quantum system with Hamiltonian $\hat{\mathcal{H}}$ and eigenstates $|\Psi_0\rangle, |\Psi_1\rangle, ...$ with eigenvalues $E_0 < E_1 < ...$ To find the ground state $|\Psi_0\rangle$, one can approximate it with a parametrized variational state $|\Psi_\theta\rangle$ with variational parameters $\boldsymbol{\theta}$. The variational principle states that the state $|\Psi_\theta\rangle$ can approximate the ground state if it minimizes the variational energy, as discussed in Eq. (2.1), given by

$$E_\theta = \frac{\langle\Psi_\theta|\hat{\mathcal{H}}|\Psi_\theta\rangle}{\langle\Psi_\theta|\Psi_\theta\rangle}. \tag{2.12}$$

The variational principle can be understood as follows. As the energy eigenstates provide a complete basis of the Hilbert space, one can decompose the variational state to be

$$|\Psi_\theta\rangle = \sum_n \alpha_n |\Psi_n\rangle, \tag{2.13}$$

where $\alpha_n = \langle\Psi_n|\Psi_\theta\rangle$. Therefore, the variational energy is given by

$$E_\theta = \frac{\sum_n |\alpha_n|^2 E_n}{\sum_n |\alpha_n|^2} \geq E_0, \tag{2.14}$$

and $E_\theta = E_0$ if and only if all $\alpha_n = 0$ for all $n > 0$ which means $|\Psi_\theta\rangle$ is the ground state.

In practice, it is usually impossible to hit the exact ground state. However, a lower variational energy E_θ hints at a more accurate variational state. When $E_0 < E_\theta < E_1$, the variational energy provides a lower bound of fidelity as

$$\mathcal{F} = \frac{\langle\Psi_0|\Psi_\theta\rangle\langle\Psi_\theta|\Psi_0\rangle}{\langle\Psi_0|\Psi_0\rangle\langle\Psi_\theta|\Psi_\theta\rangle} = \frac{|\alpha_0|^2}{\sum_n |\alpha_n|^2} \geq \frac{E_1 - E_\theta}{E_1 - E_0}. \tag{2.15}$$

A popular quantity for estimating the error of variational state is the relative error of variational energy, defined as

$$\epsilon_{\text{rel}} = \frac{E_\theta - E_0}{|E_0|}. \tag{2.16}$$

2.2.2 Stochastic estimation

The variational principle itself does not solve the problem of exponential cost, since the evaluation of variational energy E_θ in Eq. (2.12) still needs to sum over the exponentially large Hilbert space. Fortunately, one can estimate E_θ in a stochastic manner through Monte Carlo sampling as explained in Sec. 2.1.

VMC firstly requires a choice of variational basis $\{|s\rangle\}$. It is usually chosen as spin configurations or fermion occupation numbers. In a system with 2 spins, for instance, the basis is $\{|\uparrow\uparrow\rangle, |\uparrow\downarrow\rangle, |\downarrow\uparrow\rangle, |\downarrow\downarrow\rangle\}$. In this given basis, one has

$$|\Psi_\theta\rangle = \sum_s \psi_\theta(s)\,|s\rangle\,, \tag{2.17}$$

$$\hat{\mathcal{H}} = \sum_{s,s'} H_{s,s'}\,|s\rangle\,\langle s'|\,, \tag{2.18}$$

where $\psi_\theta(s) = \langle s|\Psi_\theta\rangle$, $H_{s,s'} = \langle s|\hat{\mathcal{H}}|s'\rangle$. Then Eq. (2.12) can be rewritten as

$$E_\theta = \frac{\sum_{s,s'}\psi_\theta(s)H_{s,s'}\psi_\theta(s')}{\sum_s|\psi_\theta(s)|^2} = \sum_s \frac{|\psi_\theta(s)|^2}{||\Psi_\theta||^2}\sum_{s'}\frac{\psi_\theta(s')}{\psi_\theta(s)}H_{s,s'}, \tag{2.19}$$

where $||\Psi_\theta||^2 = \sum_s|\psi_\theta(s)|^2$. To evaluate E_θ, one can generate Monte Carlo samples s with probability $p(s) = |\psi_\theta(s)|^2/||\Psi_\theta||^2$, then

$$E_\theta = \left\langle \sum_{s'}\frac{\psi_\theta(s')}{\psi_\theta(s)}H_{s,s'}\right\rangle = \langle E_\theta^{\mathrm{loc}}\rangle\,, \tag{2.20}$$

where

$$E_\theta^{\mathrm{loc}}(s) = \sum_{s'}\frac{\psi_\theta(s')}{\psi_\theta(s)}H_{s,s'} = \frac{\langle s|\hat{\mathcal{H}}|\Psi_\theta\rangle}{\langle s|\Psi_\theta\rangle} \tag{2.21}$$

is the local energy. Although $\sum_{s'}$ seems to cover the full Hilbert space, one only needs to perform the summation for s' with non-zero $H_{s,s'}$, which usually amounts to polynomial costs. The complexity is $\mathcal{O}(N)$ for Hamiltonian with neighbor couplings and $\mathcal{O}(N^2)$ for Hamiltonian with long-range couplings, where N is the system size.

Other observables can be estimated similarly. For a general observable $\hat{A}$, its expectation value can be estimated by

$$\langle\hat{A}\rangle = \langle A_\theta^{\mathrm{loc}}\rangle\,, \tag{2.22}$$

where the left bracket represents quantum expectation, while the right bracket represents the statistical average in Monte Carlo samples, and

$$A_\theta^{\mathrm{loc}}(s) = \sum_{s'}\frac{\psi_\theta(s')}{\psi_\theta(s)}A_{s,s'} = \frac{\langle s|\hat{A}|\Psi_\theta\rangle}{\langle s|\Psi_\theta\rangle}. \tag{2.23}$$

Although the variational method in principle provides an approach to approximate the ground state of a given Hamiltonian $\hat{\mathcal{H}}$ by energy minimization, it is still challenging to find optimal parameters $\boldsymbol{\theta}$ to minimize the energy of the variational state $|\Psi_\theta\rangle$. This problem will be discussed in Chapter 3.

2.3 Time-efficient simulation

2.3.1 Time complexity

As discussed in previous sections, the VMC evaluation of observables consists of two steps. One should firstly generate N_s samples $\{s_{\mathrm{MC}}\}$ from Monte Carlo sampling, and then compute $E_\theta^{\mathrm{loc}}(s)$ for every $s \in \{s_{\mathrm{MC}}\}$ to obtain $E_\theta = \langle E_\theta^{\mathrm{loc}}\rangle$.

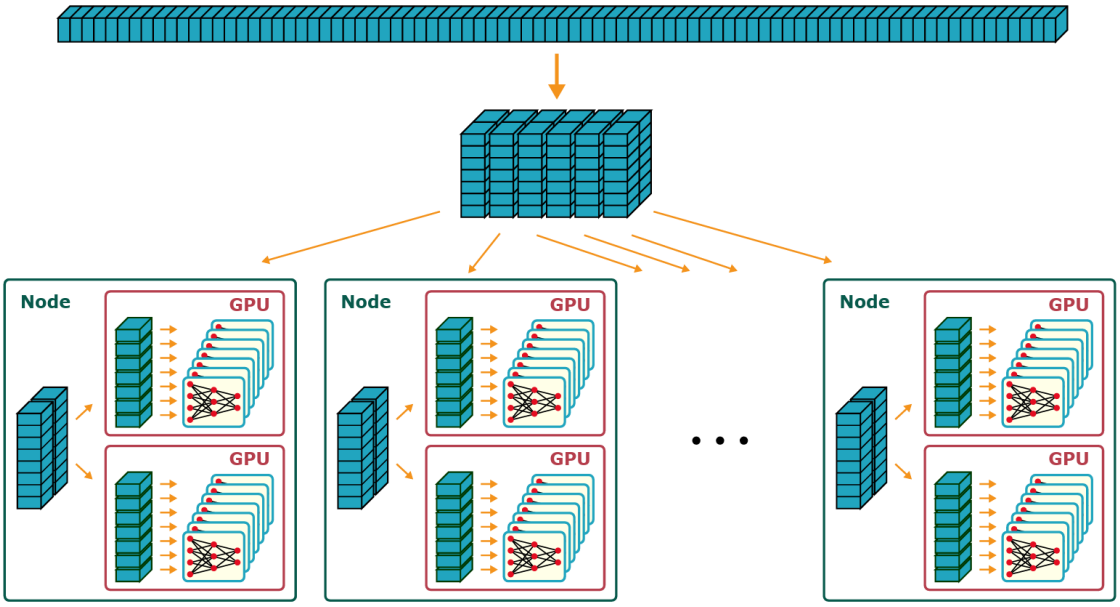


Figure 2.1: Parallelization scheme (taken from Ref. [40]). Each small block represents a Monte Carlo sample s.

In Monte Carlo sampling, one needs $\mathcal{O}(N)$ local updates to obtain a global update, where N is the system size. Variational wavefunctions $\psi_\theta(s)$ should be evaluated for every sample in every local update. Assuming the time complexity of a single evaluation is $\mathcal{O}(F)$, the total time complexity of Monte Carlo sampling is $\mathcal{O}(N_s N F)$. In the computation of local energy, as we have discussed, there are $\mathcal{O}(N)$ evaluations of $\psi_\theta(s')$ if the Hamiltonian contains only local interactions. Hence, the time complexity of computing local energy is also $\mathcal{O}(N_s N F)$. On the other hand, the evaluation of gradients in VMC, as we will discuss in Chapter 3, only has time complexity $\mathcal{O}(N_s F)$. Therefore, the Monte Carlo sampling and the computation of local energy, both with complexity $\mathcal{O}(N_s N F)$, are the time bottleneck of VMC. In a large-scale simulation of deep NQS, multiple devices should be employed to distribute the computational load of sampling and local energy to accelerate the VMC simulation.

2.3.2 Parallelization

The high-performance computing of large-scale NQS is mostly based on a Python package named JAX [41], which supports efficient parallelization in modern hardware, especially the graphical processing unit (GPU). The usage of JAX is similar to NumPy, but it supports some extra functions useful for high-performance computing on GPUs, including `jax.jit` for just-in-time compilation, `jax.grad` for automatic differentiation, and `jax.vmap` for auto-vectorization. Based on JAX, efficient simulation of NQS and VMC is implemented in multiple packages, including NetKet [42], jVMC [40], and Quantax [43], the last of which is mainly established by me for flexible NQS simulations. Here we provide a concrete summary of time-efficient VMC for large-scale GPU parallelization.

The typical parallelization scheme is depicted in Fig. 2.1. There are multiple computing nodes, each containing multiple GPUs. As discussed in Sec. 2.1, each Markov chain is independent in Monte Carlo sampling, and hence no data communication is required

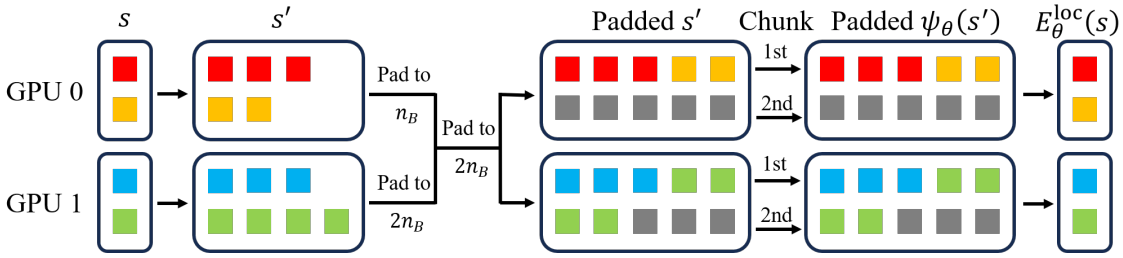


Figure 2.2: Parallel computation of the local energy. We illustrate a simple case with the number of samples $N_s = 4$, distributed on 2 GPUs. Each colored square represents a spin configuration or wavefunction, in which the grey square is the padded one. The chunked batch size is set to $n_B = 5$, and s' is padded to $2n_B$ in both devices. The computation of $\psi_\theta(s')$ is chunked to two evaluations.

among chains. Therefore, we can evenly distribute all Markov chains on all GPUs, and each GPU generates Monte Carlo samples in parallel by `jax.vmap`.

The parallelization of the local energy computation is more complex. Consider a Heisenberg model with nearest-neighbor interactions for example, whose Hamiltonian is

$$\hat{\mathcal{H}} = J \sum_{\langle i,j \rangle} \hat{\mathbf{S}}_i \cdot \hat{\mathbf{S}}_j = J \sum_{\langle i,j \rangle} \left[\hat{S}_i^z \hat{S}_j^z + \frac{1}{2} \left(\hat{S}_i^+ \hat{S}_j^- + \hat{S}_i^- \hat{S}_j^+ \right) \right], \tag{2.24}$$

where $\langle i, j \rangle$ represents nearest neighbors. For each $s \in \{s_{\mathrm{MC}}\}$, one should compute all $\psi(s')$ with $s' \neq s$ and non-zero $H_{s,s'}$. In the Heisenberg model, however, the number of non-zero s' is different for different s. For instance, when $N = 2$, $s = (\uparrow, \uparrow)$ has no s', while $s = (\uparrow, \downarrow)$ has $s' = (\downarrow, \uparrow)$. The varying amount of s' makes it harder to compile the code using `jax.jit` since the compiled function only applies to inputs with fixed shapes.

To solve this problem, we fix the number of allowed input configurations in each evaluation, namely the chunked batch size n_B. The chunked batch is also necessary to avoid memory overflow in large-scale simulations. In the local energy computation, we collect all $\{s' | s' \neq s, H_{s,s'} \neq 0, s \in \{s_{\mathrm{MC}}^{(d)}\}\}$ for samples $\{s_{\mathrm{MC}}^{(d)}\}$ in the same device d, and pad the number of s' to a multiple of n_B. When there are multiple devices, the padding number is determined by the one with the most s' so that the array shapes are the same on different devices. The padding is performed implicitly by the function `jax.numpy.nonzero` with a specified `size` argument. Finally, the local energy $E_\theta^{\mathrm{loc}}(s)$ is computed from the wavefunction $\psi_\theta(s')$ of padded s' by `jax.ops.segment_sum`. As illustrated in Fig. 2.2, the evaluation of the local energy as described above can be performed in parallel on different GPUs. The only communication across devices is for determining the padding number, and the communicated variables are only a few scalar numbers. The forward pass of unnecessary s' in padding (grey squares) leads to an efficiency loss, but it helps to avoid recompiling functions in JAX. Furthermore, the efficiency loss is usually small in a large-scale simulation with a suitable n_B. Therefore, the parallelization scheme in Fig. 2.2 reaches a good balance between the compilation and evaluation efficiency.

2.4 Zero-variance property

2.4.1 Energy variance

An important quantity in VMC is the energy variance

$$\sigma^2 = \langle (\hat{\mathcal{H}} - E_\theta)^2 \rangle = \langle \hat{\mathcal{H}}^2 \rangle - E_\theta^2. \tag{2.25}$$

In general, evaluating an expectation value of the form $\langle \hat{A}\hat{B} \rangle$ can be written as

$$\langle \hat{A}\hat{B} \rangle = \frac{\langle \Psi_\theta | \hat{A}\hat{B} | \Psi_\theta \rangle}{\langle \Psi_\theta | \Psi_\theta \rangle} = \sum_s \frac{|\psi_\theta(s)|^2}{||\Psi_\theta||^2} (A_\theta^{\mathrm{loc}}(s))^* \, B_\theta^{\mathrm{loc}}(s) = \langle (A_\theta^{\mathrm{loc}})^* \, B_\theta^{\mathrm{loc}} \rangle \, . \tag{2.26}$$

Therefore,

$$\sigma^2 = \langle |E_\theta^{\mathrm{loc}}|^2 \rangle - |\, \langle E_\theta^{\mathrm{loc}} \rangle \, |^2. \tag{2.27}$$

An important property of VMC is that when $|\Psi_\theta\rangle = |\Psi_0\rangle$,

$$E_\theta^{\mathrm{loc}}(s) = \frac{\langle s | \hat{\mathcal{H}} | \Psi_\theta \rangle}{\langle s | \Psi_\theta \rangle} = E_0, \tag{2.28}$$

so in this case the local energy always equals the ground-state energy for every sample s, and one also has $\sigma^2 = 0$. Consequently, lower energy variance can also serve as a signal of better variational accuracy [44]. Motivated by this, the V-score [45] is also proposed as a quantity proportional to the energy variance to measure the variational accuracy across different systems and different methods. The V-score is designed in such a way that it remains invariant under a transformation $\hat{\mathcal{H}} \to a\hat{\mathcal{H}} + b$,

$$\text{V-score} = \frac{N\sigma^2}{(E_\theta - E_\infty)^2}, \tag{2.29}$$

where N is the system size and E_∞ is the energy of the system when temperature $T \to \infty$.

2.4.2 Zero-variance extrapolation

VMC provides an inexact estimation of observables with controlled bias due to the fundamental constraint of the sign problem. As shown in Eq.(2.20) and Eq.(2.22), and also shown in numerical experiments, this bias manifests itself mainly as an inexact variational state $|\Psi_\theta\rangle$. The error in $|\Psi_\theta\rangle$ results in the bias in observables, among which the most common is the overestimation of the ground-state energy due to the variational principle. Another example is the overestimation of the spin order in quantum spin liquids. Therefore, the zero-variance extrapolation is often utilized to reduce the bias in VMC.

The main idea of zero-variance extrapolation is to generate multiple variational states with different energies E_θ variances σ^2. As σ^2 indicates the variational error, an extrapolation to $\sigma^2 \to 0$ provides a better estimation of the ground-state energy [46, 47].

Assuming the normalized variational state $|\Psi_\theta\rangle$ only deviates slightly from the exact ground state $|\Psi_0\rangle$, one can express it as

$$|\Psi_\theta\rangle = \sqrt{1-\lambda^2}\, |\Psi_0\rangle + \lambda \, |\Psi_e\rangle \, , \tag{2.30}$$

where $|\Psi_e\rangle$ represents an error state orthogonal to $|\Psi_0\rangle$ and λ is a small positive number indicating the error strength. Denoting $E_e = \langle\Psi_e|\hat{\mathcal{H}}|\Psi_e\rangle$ and $\langle\hat{\mathcal{H}}^2\rangle_e = \langle\Psi_e|\hat{\mathcal{H}}^2|\Psi_e\rangle$, one can express the variational energy as

$$E_\theta = E_0 + \lambda^2(E_e - E_0), \tag{2.31}$$

and the energy variance as

$$\sigma^2 = \lambda^2(\langle\hat{\mathcal{H}}^2\rangle_e - 2E_0E_e + E_0^2) + \mathcal{O}(\lambda^4). \tag{2.32}$$

If the error state $|\Psi_e\rangle$ does not change substantially in different variational states, there is a linear relation for small λ

$$(E_\theta - E_0) \propto \sigma^2, \tag{2.33}$$

which reveals $E_\theta \to E_0$ under an extrapolation to $\sigma^2 \to 0$.

For a general observable $\hat{A}$ with $A_0 = \langle\Psi_0|\hat{A}|\Psi_0\rangle$ and $A_e = \langle\Psi_e|\hat{A}|\Psi_e\rangle$, the expectation value in Eq. (2.31) should be modified to

$$A_\theta = \langle\Psi_\theta|\hat{A}|\Psi_\theta\rangle = A_0 + 2\lambda\sqrt{1-\lambda^2}\,\mathrm{Re}\,\langle\Psi_e|\hat{A}|\Psi_0\rangle + \lambda^2(A_e - A_0). \tag{2.34}$$

If $\mathrm{Re}\,\langle\Psi_e|\hat{A}|\Psi_0\rangle \neq 0$, the linear relation for small λ should be modified to

$$(A_\theta - A_0) \propto \sigma. \tag{2.35}$$

2.5 Lanczos step

The Lanczos step (LS) is a popular method in VMC to improve variational accuracy [46, 48, 49], recently also introduced for NQS simulations [50, 51]. The key idea of LS is to construct new states $\{|\Psi_1\rangle, |\Psi_2\rangle, ...\}$ orthogonal to the well-trained variational wavefunction $|\Psi_0\rangle = |\Psi_\theta\rangle$ and minimize the energy of the state formed by a linear combination of $\{|\Psi_0\rangle, |\Psi_1\rangle, |\Psi_2\rangle, ...\}$. The new energy is then guaranteed to be lower than the initial energy $E_0 = E_\theta$.

Here we explain how to perform one Lanczos step. The $|\psi_1\rangle$ satisfying $\langle\psi_0|\psi_1\rangle = 0$ is given by

$$|\psi_1\rangle = \frac{\hat{\mathcal{H}} - E_0}{\sigma}|\psi_0\rangle. \tag{2.36}$$

The new variational state given by the linear combination of $|\psi_0\rangle$ and $|\psi_1\rangle$ is

$$|\psi_\alpha\rangle = |\psi_0\rangle + \alpha|\psi_1\rangle, \tag{2.37}$$

whose energy is

$$E_\alpha = E_0 + \frac{\langle\psi_\alpha|(\hat{\mathcal{H}} - E_0)|\psi_\alpha\rangle}{\langle\psi_\alpha|\psi_\alpha\rangle} = E_0 + \sigma\frac{\alpha^2\mu_3 + 2\alpha}{\alpha^2 + 1}, \tag{2.38}$$

where

$$\mu_n = \frac{\langle\psi_0|(\hat{\mathcal{H}} - E_0)^n|\psi_0\rangle}{\sigma^n}. \tag{2.39}$$

The minimal energy is achieved at

$$\alpha_* = \frac{\mu_3 - \sqrt{\mu_3^2 + 4}}{2}, \tag{2.40}$$

and the lowest energy is

$$E_{\alpha_*} = E_0 + \sigma \frac{\alpha_*^2 \mu_3 + 2\alpha_*}{\alpha_*^2 + 1} = E_0 + \sigma \alpha_*. \tag{2.41}$$

2.5.1 Initial guess

A direct way to compute μ_n is by measuring suitable quantities as expectation values of the initial state $|\psi_0\rangle$. However, the measurement becomes more accurate if it is performed with a state $|\psi_{\alpha_0}\rangle$ closer to the ground state. There are multiple ways to guess a suitable α_0. For instance, one can perform a short simulation to obtain an estimated $\alpha_\star$, and set α_0 to this value in the following simulations. This step can be performed iteratively for better α_0 estimation. Then from Eq. (2.38) one can compute μ_3 as

$$\mu_3 = \frac{(\alpha_0^2 + 1)(E_{\alpha_0} - E_0)/\sigma - 2\alpha_0}{\alpha_0^2}, \tag{2.42}$$

where E_{α_0} can be measured by Monte Carlo sampling. The optimal α_* can be derived from μ_3 by Eq. (2.40) and the lowest energy is then given by Eq. (2.41).

2.5.2 Energy variance

To compute the energy variance of $|\psi_\alpha\rangle$, we start with an intermediate quantity

$$v_\alpha = \frac{\langle\psi_\alpha| (\hat{\mathcal{H}} - E_0)^2 |\psi_\alpha\rangle}{\sigma^2 \langle\psi_\alpha|\psi_\alpha\rangle} = \frac{\alpha^2 \mu_4 + 2\alpha\mu_3 + 1}{\alpha^2 + 1}. \tag{2.43}$$

Similar to Eq. (2.42), one can measure v_{α_0} by Monte Carlo sampling and determine μ_4 as

$$\mu_4 = \frac{(\alpha_0^2 + 1)v_{\alpha_0} - 2\alpha_0\mu_3 - 1}{\alpha_0^2}. \tag{2.44}$$

Then v_{α_*} can be computed given μ_3 and μ_4, which gives the required energy variance as

$$\sigma_{\alpha_*}^2 = \frac{\langle\psi_{\alpha_*}| (\hat{\mathcal{H}} - E_{\alpha_*})^2 |\psi_{\alpha_*}\rangle}{\langle\psi_{\alpha_*}|\psi_{\alpha_*}\rangle} = \sigma^2 v_{\alpha_*} - (E_0 - E_{\alpha_*})^2. \tag{2.45}$$

Chapter 3

Optimization

While Chapter 2 explained how to evaluate the variational energy E_θ through Monte Carlo sampling, the optimization of the variational parameters has not yet been discussed. In this chapter, we will start with the direct stochastic gradient descent (SGD) optimization, and then introduce the more popular stochastic reconfiguration (SR) and minimum-norm stochastic reconfiguration (MinSR) methods and their details.

This chapter is partially related to my following publication.

- Ao Chen and Markus Heyl
  "Empowering deep neural quantum states through efficient optimization"
  Nat. Phys. 20, 1476 (2024)

3.1 Stochastic gradient descent

Stochastic gradient descent (SGD) is the most common optimization method in modern machine learning applications and the most direct method for minimizing the variational energy in VMC. SGD makes a small change to all parameters $\boldsymbol{\theta}$ in each step, i.e.,

$$\boldsymbol{\theta}' = \boldsymbol{\theta} + \tau\dot{\boldsymbol{\theta}}, \tag{3.1}$$

where τ is a small training rate and $\dot{\boldsymbol{\theta}}$ is obtained by moving parameters toward the energy descending direction. While E_θ is always real, $\boldsymbol{\theta}$ might be real or complex, so there are 2 cases as listed below.

1. $\boldsymbol{\theta}$ is real. Then we have the most common SGD equation in machine learning tasks

$$\dot{\boldsymbol{\theta}} = -\frac{\partial E_\theta}{\partial \boldsymbol{\theta}^T}. \tag{3.2}$$

2. $\boldsymbol{\theta}$ is complex. According to the complex gradient explained in Appendix A,

$$\dot{\boldsymbol{\theta}} = -\frac{\partial E_\theta}{\partial \boldsymbol{\theta}^\dagger}. \tag{3.3}$$

By defining $\partial E_\theta/\partial \boldsymbol{\theta}^\dagger = \partial E_\theta/\partial \boldsymbol{\theta}^T$ when $\boldsymbol{\theta}$ is real, one can use Eq. (3.3) in general cases.

Evaluating the exact gradient in Eq. (3.3) requires exponential time, but one can utilize Eq. (2.12) to rewrite it as

$$\begin{aligned}
\dot{\boldsymbol{\theta}} &= -\frac{\partial E_\theta}{\partial \boldsymbol{\theta}^\dagger} \\
&= -\frac{\partial \langle\Psi_\theta|}{\partial \boldsymbol{\theta}^\dagger}\frac{\partial E_\theta}{\partial \langle\Psi_\theta|} \\
&= -\frac{\partial \langle\Psi_\theta|}{\partial \boldsymbol{\theta}^\dagger}\left(\frac{\hat{\mathcal{H}}|\Psi_\theta\rangle}{||\Psi_\theta||^2} - \frac{\langle\Psi_\theta|\hat{\mathcal{H}}|\Psi_\theta\rangle|\Psi_\theta\rangle}{||\Psi_\theta||^4}\right) \\
&= -\frac{\partial \langle\Psi_\theta|}{\partial \boldsymbol{\theta}^\dagger}(\hat{\mathcal{H}} - E_\theta)\frac{|\Psi_\theta\rangle}{||\Psi_\theta||^2}.
\end{aligned} \tag{3.4}$$

To compute $\dot{\boldsymbol{\theta}}$ using VMC, we expand it using the Fock basis $|s\rangle$ to obtain

$$\begin{aligned}
\dot{\boldsymbol{\theta}} &= -\sum_s \frac{1}{||\Psi_\theta||^2}\frac{\partial \langle\Psi_\theta|s\rangle}{\partial \boldsymbol{\theta}^\dagger}\langle s|(\hat{\mathcal{H}} - E_\theta)|\Psi_\theta\rangle \\
&= -\sum_s \frac{\langle\Psi_\theta|s\rangle\langle s|\Psi_\theta\rangle}{||\Psi_\theta||^2}\left(\frac{1}{\langle\Psi_\theta|s\rangle}\frac{\partial \langle\Psi_\theta|s\rangle}{\partial \boldsymbol{\theta}^\dagger}\right)(E_\theta^{\mathrm{loc}}(s) - E_\theta) \\
&= -\sum_s \frac{|\psi_\theta(s)|^2}{||\Psi_\theta||^2}\left(\frac{1}{\psi_\theta^*(s)}\frac{\partial \psi_\theta^*(s)}{\partial \boldsymbol{\theta}^\dagger}\right)(E_\theta^{\mathrm{loc}}(s) - \langle E_\theta^{\mathrm{loc}}\rangle) \\
&= -\left\langle \frac{1}{\psi_\theta^*}\frac{\partial \psi_\theta^*}{\partial \boldsymbol{\theta}^\dagger}(E_\theta^{\mathrm{loc}} - \langle E_\theta^{\mathrm{loc}}\rangle)\right\rangle \\
&= -\left\langle \left(\frac{1}{\psi_\theta^*}\frac{\partial \psi_\theta^*}{\partial \boldsymbol{\theta}^\dagger} - \left\langle \frac{1}{\psi_\theta^*}\frac{\partial \psi_\theta^*}{\partial \boldsymbol{\theta}^\dagger}\right\rangle\right)(E_\theta^{\mathrm{loc}} - \langle E_\theta^{\mathrm{loc}}\rangle)\right\rangle
\end{aligned} \tag{3.5}$$

where the local energy E_θ^{loc} is defined in Eq. (2.21) and we have utilized the relation $E_\theta = \langle E_\theta^{\mathrm{loc}}\rangle$ in Eq. (2.20). In the last step, we have used $\langle A(B - \langle B\rangle)\rangle = \langle (A - \langle A\rangle)(B - \langle B\rangle)\rangle$. Therefore, the energy gradient can also be obtained from stochastic estimation. To further simplify the expression, we consider the optimization involving N_s Monte Carlo samples $\{s^{\mathrm{MC}}\}$ and N_p variational parameters $\boldsymbol{\theta}$. One can define a vector $\bar{\boldsymbol{\epsilon}}$ with N_s entries

$$\bar{\epsilon}_n = \frac{1}{\sqrt{N_s}}(E_\theta^{\mathrm{loc}}(s_n^{\mathrm{MC}}) - \langle E_\theta^{\mathrm{loc}}\rangle), \tag{3.6}$$

and an $N_s \times N_p$ matrix

$$\bar{\mathbf{O}} = \frac{1}{\sqrt{N_s}}(\mathbf{O} - \langle \mathbf{O}\rangle), \tag{3.7}$$

where $\mathbf{O}$ is also an $N_s \times N_p$ matrix with elements

$$O_{nk} = \frac{1}{\psi_\theta(s_n^{\mathrm{MC}})}\frac{\partial \psi_\theta(s_n^{\mathrm{MC}})}{\partial \theta_k}. \tag{3.8}$$

Then Eq. (3.5) becomes

$$\dot{\theta}_k = -\sum_{n=1}^{N_s} \bar{O}_{nk}^*\,\bar{\epsilon}_n, \tag{3.9}$$

Algorithm 2 VMC with SGD

Input: Initial variational state $|\Psi_\theta\rangle$, training rate τ, training epochs $n_{\max}$
Output: Final variational state $|\Psi_\theta\rangle$
  for $n \leftarrow 1$ to $n_{\max}$ **do**
    Generate Monte Carlo samples $\{s^{\mathrm{MC}}\}$ with probability $p_\theta(s) = |\psi_\theta(s)|^2/||\Psi_\theta||^2$
    Compute $\bar{\boldsymbol{\epsilon}}$ from Eq. (3.6) and Eq. (2.21)
    Compute $\bar{\mathbf{O}}$ from Eq. (3.7) and Eq. (3.8)
    $\dot{\boldsymbol{\theta}} \leftarrow -\bar{\mathbf{O}}^\dagger \bar{\boldsymbol{\epsilon}}$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \tau \dot{\boldsymbol{\theta}}$
  end for

or equivalently,

$$\dot{\boldsymbol{\theta}} = -\mathbf{F} = -\bar{\mathbf{O}}^\dagger \bar{\boldsymbol{\epsilon}}, \tag{3.10}$$

where $\mathbf{F} = \bar{\mathbf{O}}^\dagger \bar{\boldsymbol{\epsilon}}$ is an effective force on parameters. The full SGD algorithm of VMC optimization is shown above.

3.2 Stochastic reconfiguration

Although the naive SGD is powerful in machine learning tasks like computer vision (CV), it does not work well in VMC. The reason for this difference is still an open problem, but it is possibly explained by the following arguments.

- CV usually tolerates 1% to 10% error, while VMC requires a much lower error ($\epsilon_{\mathrm{rel}} < 10^{-3}$), so a more accurate optimizer is needed for VMC.
- For every input s, CV only needs a probability $P(s)$, while VMC needs a wavefunction $\psi(s)$ which possibly contains a sign structure making the variational space harder to explore.
- The variational space of VMC is more rugged than CV. For instance, $\psi(s)$ and $\psi(s')$ can be quite different even if s and s' are only different by a spin flip, while the change of a few pixels in an image usually leads to similar $P(s)$.

Stochastic reconfiguration (SR) is proposed as a solution to these problems [46, 52–55].

3.2.1 Imaginary-time evolution

The motivation of SR is to utilize the more reliable imaginary-time evolution instead of direct gradient descent for better training accuracy. Consider an initial state $|\Psi_0\rangle = |\Psi(t_0)\rangle$ evolved into $|\Psi(t)\rangle = |\Psi(t_0 + dt)\rangle$ under imaginary-time evolution, then

$$|\Psi(t)\rangle = e^{-\hat{\mathcal{H}}dt} |\Psi_0\rangle, \quad |\dot{\Psi}\rangle = -\hat{\mathcal{H}} |\Psi\rangle. \tag{3.11}$$

In practice, two quantum states are equivalent if they differ by a global constant. Therefore, the evolution of $|\Psi\rangle$ contains redundancy because $|\Psi\rangle$ is not normalized. To remove

this redundancy and fix the gauge, we define a normalized state

$$|\tilde{\Psi}(t)\rangle = \frac{||\Psi_0||}{\langle\Psi_0|\Psi(t)\rangle}|\Psi(t)\rangle\,, \tag{3.12}$$

which satisfies the following properties.

- Fidelity $\mathcal{F}(\Psi,\tilde{\Psi}) = \frac{\langle\Psi|\tilde{\Psi}\rangle\langle\tilde{\Psi}|\Psi\rangle}{\langle\tilde{\Psi}|\tilde{\Psi}\rangle\langle\Psi|\Psi\rangle} = 1$, so $|\tilde{\Psi}\rangle$ represents the same physical state as $|\Psi\rangle$.
- At $t = t_0$, $|\tilde{\Psi}\rangle$ is a normalized state with $||\tilde{\Psi}(t_0)|| = 1$.
- $|\tilde{\Psi}\rangle$ remains unchanged under $|\Psi\rangle \to ae^{i\varphi}|\Psi\rangle$.

The usual normalized state $|\Psi\rangle/||\Psi||$ is not utilized here because it does not satisfy the third property when $e^{i\varphi} \neq 1$. The time evolution of $|\tilde{\Psi}(t)\rangle$ at $t = t_0$ is given by

$$|\dot{\tilde{\Psi}}\rangle = \frac{||\Psi_0||}{\langle\Psi_0|\Psi\rangle}|\dot{\Psi}\rangle - \frac{||\Psi_0||\langle\Psi_0|\dot{\Psi}\rangle}{\langle\Psi_0|\Psi\rangle^2}|\Psi\rangle = \frac{|\dot{\Psi}\rangle}{||\Psi||} - \frac{\langle\Psi|\dot{\Psi}\rangle}{||\Psi||^2}\frac{|\Psi\rangle}{||\Psi||}, \tag{3.13}$$

where we have utilized $|\Psi\rangle = |\Psi_0\rangle$ at $t = t_0$. Combined with Eq. (3.11), we obtain

$$|\dot{\tilde{\Psi}}\rangle = -\frac{(\hat{\mathcal{H}} - E)|\Psi\rangle}{||\Psi||}. \tag{3.14}$$

On the other hand, the quantum state gradient of the variational energy E in the Hilbert space, as explained in Appendix A, is given by

$$|g_\Psi\rangle = -||\Psi||\frac{\partial E}{\partial\langle\Psi|} = -||\Psi||\left(\frac{\hat{\mathcal{H}}|\Psi\rangle}{\langle\Psi|\Psi\rangle} - \frac{\langle\Psi|\hat{\mathcal{H}}|\Psi\rangle}{\langle\Psi|\Psi\rangle^2}|\Psi\rangle\right) = -\frac{(\hat{\mathcal{H}} - E)|\Psi\rangle}{||\Psi||}. \tag{3.15}$$

The equivalence of Eq. (3.14) and Eq. (3.15) shows that the imaginary-time evolution and the quantum state gradient are mathematically equivalent. In machine learning, this kind of gradient descent in a proper space is called the natural gradient [56, 57].

3.2.2 Variational state evolution

In VMC, the evolution of a variational state $|\Psi_\theta\rangle$ is controlled by parameters $\boldsymbol{\theta}$. If $\psi_\theta(s)$ is a real or complex-holomorphic function of $\boldsymbol{\theta}$, the time derivative is given by

$$|\dot{\Psi}_\theta\rangle = \frac{\partial|\Psi_\theta\rangle}{\partial\boldsymbol{\theta}}\cdot\dot{\boldsymbol{\theta}}. \tag{3.16}$$

To remove the redundancy of global constants, we also need the normalized state as defined in Eq. (3.12),

$$|\dot{\tilde{\Psi}}_\theta\rangle = \frac{|\dot{\Psi}_\theta\rangle}{||\Psi_\theta||} - \frac{\langle\Psi_\theta|\dot{\Psi}_\theta\rangle}{||\Psi_\theta||^2}\frac{|\Psi_\theta\rangle}{||\Psi_\theta||} = \frac{(\partial_\theta - \langle\partial_\theta\rangle)|\Psi_\theta\rangle}{||\Psi_\theta||}\cdot\dot{\boldsymbol{\theta}}, \tag{3.17}$$

where $\langle\partial_\theta\rangle = \langle\Psi_\theta|\partial_\theta|\Psi_\theta\rangle / \langle\Psi_\theta|\Psi_\theta\rangle$.

Assume $|\Psi(t_0)\rangle = |\Psi_{\theta(t_0)}\rangle$ at a starting time t_0. An ideal optimization step $\dot{\boldsymbol{\theta}}$ will generate a variational evolution $|\dot{\tilde{\Psi}}_\theta\rangle$ equivalent to the exact imaginary-time evolution $|\dot{\tilde{\Psi}}\rangle$ in Eq. (3.14). However, this is in general impossible because the degrees of freedom in $|\dot{\tilde{\Psi}}\rangle$ is the Hilbert space dimension, while the degrees of freedom in $|\dot{\tilde{\Psi}}_\theta\rangle$ is only the number of parameters N_p. With a polynomial N_p in $|\dot{\tilde{\Psi}}_\theta\rangle$, one cannot catch all possible evolutions in the exponentially large Hilbert space. As an alternative, one can minimize the difference of $|\dot{\tilde{\Psi}}_\theta\rangle$ and $|\dot{\tilde{\Psi}}\rangle$ given by

$$\begin{aligned} d^2 &= \left(\langle\dot{\tilde{\Psi}}_\theta| - \langle\dot{\tilde{\Psi}}|\right)\left(|\dot{\tilde{\Psi}}_\theta\rangle - |\dot{\tilde{\Psi}}\rangle\right) \\ &= \sum_s \frac{|\psi_\theta(s)|^2}{||\Psi_\theta||^2}\left|\left(\frac{1}{\psi_\theta}\frac{\partial\psi_\theta}{\partial\boldsymbol{\theta}} - \left\langle\frac{1}{\psi_\theta}\frac{\partial\psi_\theta}{\partial\boldsymbol{\theta}}\right\rangle\right)\dot{\boldsymbol{\theta}} - (E_\theta^{\mathrm{loc}}(s) - E_\theta)\right|^2 \\ &= \left\langle\left|\left(\frac{1}{\psi_\theta}\frac{\partial\psi_\theta}{\partial\boldsymbol{\theta}} - \left\langle\frac{1}{\psi_\theta}\frac{\partial\psi_\theta}{\partial\boldsymbol{\theta}}\right\rangle\right)\dot{\boldsymbol{\theta}} - (E_\theta^{\mathrm{loc}} + E_\theta)\right|^2\right\rangle, \end{aligned} \tag{3.18}$$

When there are in total N_s Monte Carlo samples $\{s^{\mathrm{MC}}\}$,

$$d^2 = ||\bar{\mathbf{O}}\dot{\boldsymbol{\theta}} + \bar{\boldsymbol{\epsilon}}||^2, \tag{3.19}$$

where $\bar{\mathbf{O}}$ and $\bar{\boldsymbol{\epsilon}}$ are defined in Eq. (3.7) and Eq. (3.6), respectively. Therefore, an optimal optimization step $\dot{\boldsymbol{\theta}}$ can be obtained by finding the least-squares solution of the linear equation

$$\bar{\mathbf{O}}\dot{\boldsymbol{\theta}} = -\bar{\boldsymbol{\epsilon}}. \tag{3.20}$$

Conceptually, the left side of the linear equation gives the evolution of the variational state, and the right side gives the imaginary-time evolution, so the least-squares solution minimizing their difference provides an optimal training step.

3.2.3 SR and MinSR

In traditional VMC, typically the variational state only contains a few parameters, so $N_s > N_p$ and Eq. (3.20) is an overdetermined linear equation. According to the complex gradient explained in Appendix A, the least-squares solution can be obtained by

$$\mathbf{0} = \frac{\partial(d^2)}{\partial(\dot{\boldsymbol{\theta}}^\dagger)} = \bar{\mathbf{O}}^\dagger\bar{\mathbf{O}}\dot{\boldsymbol{\theta}} + \bar{\mathbf{O}}^\dagger\bar{\boldsymbol{\epsilon}}, \tag{3.21}$$

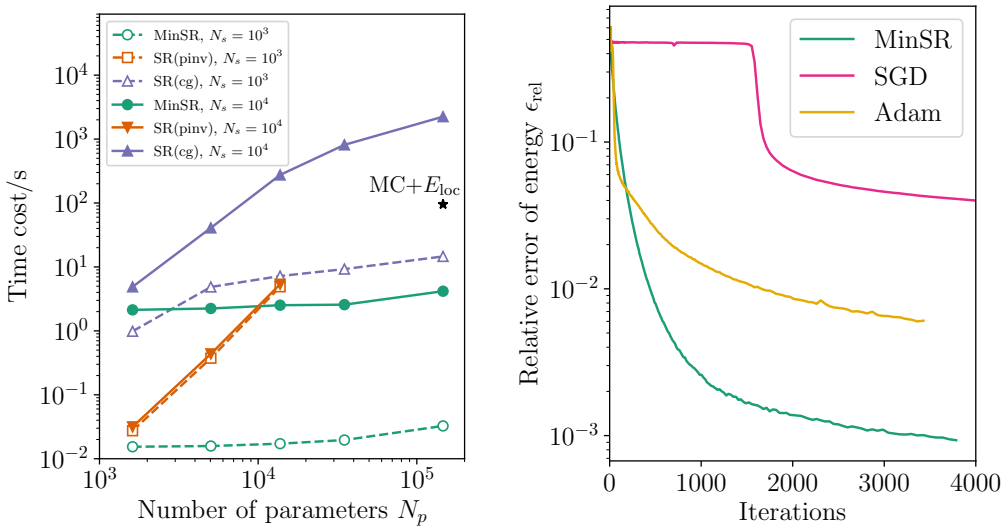
and the solution is

$$\dot{\boldsymbol{\theta}} = -(\bar{\mathbf{O}}^\dagger\bar{\mathbf{O}})^{-1}\bar{\mathbf{O}}^\dagger\bar{\boldsymbol{\epsilon}} = -\mathbf{S}^{-1}\mathbf{F}, \tag{3.22}$$

where $\mathbf{F}$ is the effective force defined in Eq.(3.10), and

$$\mathbf{S} = \bar{\mathbf{O}}^\dagger\bar{\mathbf{O}} \tag{3.23}$$

is an $N_p \times N_p$ quantum Fisher information matrix [57]. Therefore, Eq. (3.22) can also be understood as a gradient descent with a metric $\mathbf{S}$. The formula given in Eq. (3.22) is the stochastic reconfiguration (SR), which has been utilized for various kinds of variational states including fermionic mean-field wavefunctions and tensor networks.



(a) Time cost

(b) Training curve

Figure 3.1: Performance of optimization methods on the 10×10 square Heisenberg J_1-J_2 model (taken from Ref. [58]). (a) The time cost of solving Eq. (3.20). "MC+$E_{\rm loc}$" shows the time cost of the remaining VMC computation in each iteration on 4 A100 GPUs. (b) The training curve with $N_s = 10^4$ and $N_p \approx 10^6$.

When it comes to deep NQS, however, the deep network has a huge number of parameters and typically $N_s < N_p$. In this case, Eq. (3.20) becomes underdetermined. Therefore, there are typically an infinite amount of solutions satisfying $\bar{\mathbf{O}}\dot{\boldsymbol{\theta}} = -\bar{\boldsymbol{\epsilon}}$. To obtain a unique solution, we select the solution with minimum norm $||\dot{\boldsymbol{\theta}}||$, which also helps to reduce the higher-order effects and improve the stability in the optimization. One can obtain the least-squares minimum-norm solution by introducing a Lagrangian multiplayer $\boldsymbol{\alpha}$. Then the Lagrangian function is

$$\mathcal{L}(\dot{\boldsymbol{\theta}}, \boldsymbol{\alpha}) = \dot{\boldsymbol{\theta}}^\dagger \dot{\boldsymbol{\theta}} - \boldsymbol{\alpha}^\dagger(\bar{\mathbf{O}}\dot{\boldsymbol{\theta}} + \bar{\boldsymbol{\epsilon}}) - (\dot{\boldsymbol{\theta}}^\dagger \bar{\mathbf{O}}^\dagger + \bar{\boldsymbol{\epsilon}}^\dagger)\boldsymbol{\alpha}. \tag{3.24}$$

From $\partial\mathcal{L}/\partial(\dot{\boldsymbol{\theta}}^\dagger) = 0$, one obtains

$$\dot{\boldsymbol{\theta}} = \bar{\mathbf{O}}^\dagger \boldsymbol{\alpha}. \tag{3.25}$$

Multiplying $\bar{\mathbf{O}}$ in both sides and utilizing $\bar{\mathbf{O}}\dot{\boldsymbol{\theta}} = -\bar{\boldsymbol{\epsilon}}$, one can solve $\boldsymbol{\alpha}$ to be

$$\boldsymbol{\alpha} = (\bar{\mathbf{O}}\bar{\mathbf{O}}^\dagger)^{-1}\bar{\boldsymbol{\epsilon}}. \tag{3.26}$$

Putting this back into Eq. (3.25) gives the solution

$$\dot{\boldsymbol{\theta}} = -\bar{\mathbf{O}}^\dagger(\bar{\mathbf{O}}\bar{\mathbf{O}}^\dagger)^{-1}\bar{\boldsymbol{\epsilon}} = -\bar{\mathbf{O}}^\dagger \mathbf{T}^{-1}\bar{\boldsymbol{\epsilon}}, \tag{3.27}$$

where $\mathbf{T}$ is an $N_s \times N_s$ matrix called neural tangent kernel [59]. This is the minimum-norm stochastic reconfiguration (MinSR) method that I introduced in Ref. [58] to solve the optimization issue in deep NQSs.

An alternative way to obtain SR and MinSR formulas is based on the matrix pseudo-inverse as demonstrated in Appendix B.1. The least-squares minimum-norm solution of the linear equation Eq. (3.20) given by the pseudo-inverse method is

$$\dot{\boldsymbol{\theta}} = -\bar{\mathbf{O}}^{-1}\bar{\boldsymbol{\epsilon}} = -(\bar{\mathbf{O}}^\dagger\bar{\mathbf{O}})^{-1}\bar{\mathbf{O}}^\dagger\bar{\boldsymbol{\epsilon}} = -\bar{\mathbf{O}}^\dagger(\bar{\mathbf{O}}\bar{\mathbf{O}}^\dagger)^{-1}\bar{\boldsymbol{\epsilon}}, \tag{3.28}$$

Algorithm 3 VMC with (Min)SR

Input: Initial variational state $|\Psi_\theta\rangle$, training rate τ, training epochs $n_{\max}$
Output: Final variational state $|\Psi_\theta\rangle$
for $n \leftarrow 1$ to $n_{\max}$ **do**
  Generate Monte Carlo samples $\{s^{\mathrm{MC}}\}$ with probability $P_\theta(s) \propto |\psi_\theta(s)|^2$
  Compute $\bar{\boldsymbol{\epsilon}}$ from Eq. (3.6)
  Compute $\bar{\mathbf{O}}$ from Eq. (3.7)
  if $N_s > N_p$ **then**
    $\dot{\boldsymbol{\theta}} \leftarrow -(\bar{\mathbf{O}}^\dagger \bar{\mathbf{O}})^{-1} \bar{\mathbf{O}}^\dagger \bar{\boldsymbol{\epsilon}}$
  else
    $\dot{\boldsymbol{\theta}} \leftarrow -\bar{\mathbf{O}}^\dagger (\bar{\mathbf{O}} \bar{\mathbf{O}}^\dagger)^{-1} \bar{\boldsymbol{\epsilon}}$
  end if
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \tau \dot{\boldsymbol{\theta}}$
end for

where the matrix inverses are pseudo-inverse. The solutions here are the same as Eq. (3.22) and Eq. (3.27). Therefore, the MinSR equation can be viewed as a natural formalism of SR in an underdetermined situation.

The complexity of SR and MinSR are $\mathcal{O}(N_p^3 + N_p^2 N_s)$ and $\mathcal{O}(N_s^3 + N_s^2 N_p)$, respectively, which reduces to $\mathcal{O}(N_p^3)$ and $\mathcal{O}(N_p)$ when $N_s \ll N_p$. Although iterative solvers like the conjugate gradient [60] can be utilized to reduce the complexity of SR to $\mathcal{O}(N_p)$, these solvers instead rely on non-parallel iterations that significantly slow down the computation on modern GPU hardware. Therefore, MinSR provides great acceleration for the application of NQS in VMC. As shown in Fig. 3.1, the time cost of MinSR is much lower than SR when $N_s \ll N_p$. In VMC with $N_s = 10^4$ and $N_p > 10^5$, SR becomes the time bottleneck while MinSR is not. Furthermore, MinSR reaches the same level of accuracy compared with SR and significantly outperforms simple methods including SGD and Adam. The full VMC optimization with SR and MinSR algorithm is presented above.

3.3 Practical details

3.3.1 Regularization

In SR and MinSR, one has to solve matrix inverses $\mathbf{S}^{-1}$ and $\mathbf{T}^{-1}$, which might cause instability in the simulation. For instance, $\mathbf{S}$ is singular when $\psi_\theta(s)$ does not depend on some redundant parameters, and $\mathbf{T}$ is always singular [61] because for a vector $\mathbf{e}$ with all entries equal to 1 there is

$$\mathbf{T}\mathbf{e} = \bar{\mathbf{O}}\bar{\mathbf{O}}^\dagger \mathbf{e} = \bar{\mathbf{O}}\mathbf{O}^\dagger \left(\mathbf{1} - \frac{\mathbf{e}\mathbf{e}^T}{N_s}\right) \mathbf{e} = \bar{\mathbf{O}}\mathbf{O}^\dagger (\mathbf{e} - \mathbf{e}) = \mathbf{0}, \tag{3.29}$$

where $\mathbf{O}$ is defined in Eq.(3.8).

One way to regularize the singular matrix is the pseudo-inverse [58] as we have shown in Eq. (3.28). To show how to perform numerical pseudo-inverse, we start with the

singular value decomposition

$$\bar{\mathbf{O}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\dagger, \tag{3.30}$$

where $\mathbf{U}, \mathbf{V}$ are unitary matrices, and $\mathbf{\Sigma}$ is a diagonal matrix with non-negative elements. Then

$$\mathbf{S} = \bar{\mathbf{O}}^\dagger\bar{\mathbf{O}} = \mathbf{V}\mathbf{D}_\mathbf{S}\mathbf{V}^\dagger, \tag{3.31}$$

$$\mathbf{T} = \bar{\mathbf{O}}\bar{\mathbf{O}}^\dagger = \mathbf{U}\mathbf{D}_\mathbf{T}\mathbf{U}^\dagger, \tag{3.32}$$

where $\mathbf{D}_\mathbf{S} = \mathbf{\Sigma}^T\mathbf{\Sigma}$ and $\mathbf{D}_\mathbf{T} = \mathbf{\Sigma}\mathbf{\Sigma}^T$ are diagonal matrices with non-negative elements. This formula is exactly the diagonalization of Hermitian matrices, and it shows that the eigenvalues of $\mathbf{S}$ and $\mathbf{T}$ are non-negative. Therefore, the pseudo-inverse can be performed by

$$\mathbf{S}^{-1} = \mathbf{V}\mathbf{D}_\mathbf{S}{}^{-1}\mathbf{V}^\dagger, \tag{3.33}$$

$$\mathbf{T}^{-1} = \mathbf{U}\mathbf{D}_\mathbf{T}{}^{-1}\mathbf{U}^\dagger, \tag{3.34}$$

where

$$(\mathbf{D}^{-1})_{ii} = \begin{cases} 1/\mathbf{D}_{ii}, & \mathbf{D}_{ii} > r_{\text{tol}} \times d_m, \\ 0, & \mathbf{D}_{ii} \leq r_{\text{tol}} \times d_m, \end{cases} \tag{3.35}$$

where d_m is the maximum value in $\mathbf{D}$, and r_{tol} is the relative tolerance, usually specified to be 10^{-12} for double precision. As small eigenvalues are truncated, the pseudo-inverse ensures the stability of the matrix inverse in VMC simulations.

A different way of regularization is the diagonal shift [62]. The SR and MinSR solution with diagonal shift is given by

$$\dot{\boldsymbol{\theta}} = -(\mathbf{S} + \lambda\mathbf{1})^{-1}\bar{\mathbf{O}}^\dagger\bar{\boldsymbol{\epsilon}} = -\bar{\mathbf{O}}^\dagger(\mathbf{T} + \lambda\mathbf{1})^{-1}\bar{\boldsymbol{\epsilon}}, \tag{3.36}$$

where λ is the strength of the diagonal shift. As $\mathbf{S}+\lambda\mathbf{1}$ and $\mathbf{T}+\lambda\mathbf{1}$ are Hermitian positive-definite matrices, one can decompose them into $\mathbf{L}\mathbf{L}^\dagger$ using Cholesky decomposition with $\mathbf{L}$ a lower triangular matrix, and the matrix inverse can be performed faster than the pseudo-inverse. However, the diagonal shift might cause a loss of training accuracy because it tends to suppress too many small eigenvalues.

3.3.2 Kaczmarz method

The MinSR method utilizes an additional minimum-norm condition to determine a unique least-squares solution. However, one is free to choose other reasonable conditions, among which a popular one is minimizing $(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{\theta}}_0)^\dagger(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{\theta}}_0)$ with $\dot{\boldsymbol{\theta}}_0$ the last optimization step. In this case, the optimization employs the information from previous training steps to alleviate the problem of insufficient samples. In practice, the dependence on training history becomes too strong if one only minimizes $(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{\theta}}_0)^\dagger(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{\theta}}_0)$. An improvement can be made by introducing a momentum μ and minimizing a more balanced quantity $(1-\mu)\dot{\boldsymbol{\theta}}^\dagger\dot{\boldsymbol{\theta}} + \mu(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{\theta}}_0)^\dagger(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{\theta}}_0)$. The Lagrangian function in this case becomes

$$\mathcal{L}(\dot{\boldsymbol{\theta}}, \boldsymbol{\alpha}) = (1-\mu)\dot{\boldsymbol{\theta}}^\dagger\dot{\boldsymbol{\theta}} + \mu(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{\theta}}_0)^\dagger(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{\theta}}_0) - \boldsymbol{\alpha}^\dagger(\bar{\mathbf{O}}\dot{\boldsymbol{\theta}} + \bar{\boldsymbol{\epsilon}}) - (\dot{\boldsymbol{\theta}}^\dagger\bar{\mathbf{O}}^\dagger + \bar{\boldsymbol{\epsilon}}^\dagger)\boldsymbol{\alpha}, \tag{3.37}$$

and the solution is

$$\dot{\boldsymbol{\theta}} = \bar{\mathbf{O}}^\dagger\mathbf{T}^{-1}(-\bar{\boldsymbol{\epsilon}} - \mu\bar{\mathbf{O}}\dot{\boldsymbol{\theta}}_0) + \mu\dot{\boldsymbol{\theta}}_0. \tag{3.38}$$

This method is intrinsically based on the Kaczmarz method for solving linear equations [63], in VMC known as the Subsampled Projected-Increment Natural Gradient Descent (SPRING) [61]. In our simulations, we find it most stable to choose $\mu = 0.5$.

3.3.3 Non-holomorphic function

The previous derivation is correct if $\psi_\theta(s)$ is a real or complex-holomorphic function of $\boldsymbol{\theta}$. In VMC simulations, we might also encounter the following situations.

1. $\boldsymbol{\theta}$ is real, but $\psi_\theta(s)$ is complex.
2. $\boldsymbol{\theta}$ and $\psi_\theta(s)$ are both complex, but $\psi_\theta(s)$ is a non-holomorphic function of $\boldsymbol{\theta}$.

$\boldsymbol{\theta}$ in case 2 can be viewed as a combination of real parameters $\boldsymbol{\theta}_r, \boldsymbol{\theta}_i$ and $\boldsymbol{\theta} = \boldsymbol{\theta}_r + i\boldsymbol{\theta}_i$. Therefore, it can be simplified to case 1.

In case 1, one can rewrite Eq. (3.19) as

$$d^2 = ||\bar{\mathbf{O}}\dot{\boldsymbol{\theta}} + \bar{\boldsymbol{\epsilon}}||^2 = ||\bar{\mathbf{O}}'\dot{\boldsymbol{\theta}} + \bar{\boldsymbol{\epsilon}}'||^2, \tag{3.39}$$

where

$$\bar{\mathbf{O}}' = \begin{pmatrix} \mathrm{Re}[\bar{\mathbf{O}}] \\ \mathrm{Im}[\bar{\mathbf{O}}] \end{pmatrix}, \bar{\boldsymbol{\epsilon}}' = \begin{pmatrix} \mathrm{Re}[\bar{\boldsymbol{\epsilon}}] \\ \mathrm{Im}[\bar{\boldsymbol{\epsilon}}] \end{pmatrix} \tag{3.40}$$

are both real-valued. Then one can construct the SR and MinSR solution as shown in Eq. (3.22) and Eq. (3.27). The SR solution becomes

$$\dot{\boldsymbol{\theta}} = -(\bar{\mathbf{O}}'^\dagger \bar{\mathbf{O}}')^{-1} \bar{\mathbf{O}}'^\dagger \bar{\boldsymbol{\epsilon}}' = -\mathrm{Re}[\mathbf{S}]^{-1}\mathrm{Re}[\mathbf{F}], \tag{3.41}$$

and the size of $\mathbf{S}' = \mathrm{Re}[\mathbf{S}]$ is still $N_p \times N_p$. On the other hand, the MinSR solution in this case is

$$\dot{\boldsymbol{\theta}} = -\bar{\mathbf{O}}'^\dagger (\bar{\mathbf{O}}'\bar{\mathbf{O}}'^\dagger)^{-1} \bar{\boldsymbol{\epsilon}}', \tag{3.42}$$

in which the size of $\mathbf{T}' = \bar{\mathbf{O}}'\bar{\mathbf{O}}'^\dagger$ becomes $2N_s \times 2N_s$.

3.3.4 Real-time evolution

The SR equation can be derived from the imaginary-time evolution. To simulate the real-time unitary evolution, one only needs to perform a rotation $\bar{\boldsymbol{\epsilon}} \to i\bar{\boldsymbol{\epsilon}}$, and then one obtains

$$\dot{\boldsymbol{\theta}} = -i\,\mathbf{S}^{-1}\mathbf{F}, \tag{3.43}$$

which is commonly known as the time-dependent variational principle (TDVP) [64, 65].

In the non-holomorphic case, the direct generalization of Eq. (3.42) is

$$\dot{\boldsymbol{\theta}} = -\mathrm{Re}[\mathbf{S}]^{-1}\mathrm{Re}[i\mathbf{F}]. \tag{3.44}$$

If the equation is instead derived from the principle of least action, one obtains a different equation [40, 64, 66]

$$\dot{\boldsymbol{\theta}} = -\mathrm{Im}[\mathbf{S}]^{-1}\mathrm{Im}[i\mathbf{F}], \tag{3.45}$$

which is more useful in TDVP since it ensures energy conservation in unitary dynamics.

The NQS method can also be applied to TDVP for the simulation of unitary dynamics [19, 67–71].

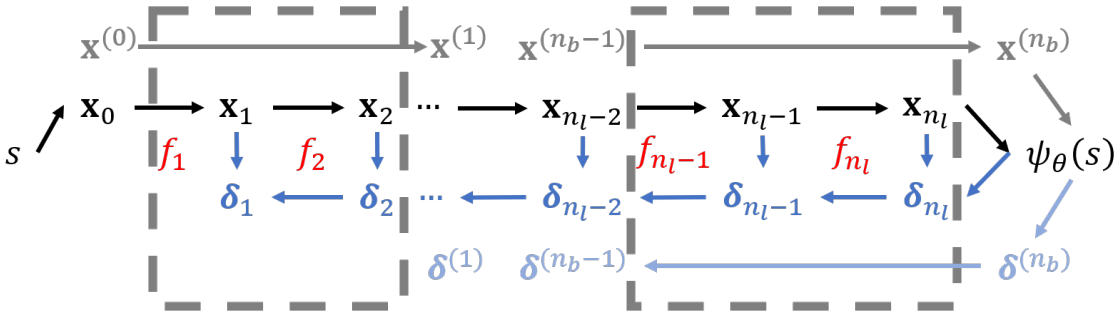


Figure 3.2: Forward and backward pass. The black part is the forward pass, and the blue part is the backward pass. Arrows show the dependence of variables in computation. The grey part shows the forward and backward pass with blocks.

3.4 Memory-efficient gradients

3.4.1 Forward and backward pass

Consider a variational wavefunction expressed by a layered structure, which is the common design choice of NQS as we will discuss in Sec. 5.1. One can express the computation of wavefunction components as

$$\begin{aligned} \mathbf{x}_0 &= s, \\ \mathbf{x}_l &= f_l(\boldsymbol{\theta}_l, \mathbf{x}_{l-1}), \\ \psi_\theta(s) &= \mathbf{x}_{n_l}, \end{aligned} \tag{3.46}$$

where $\mathbf{x}_l$, $\boldsymbol{\theta}_l$, and f_l are respectively the output, parameters, and forward function at the l'th layer, and there are in total n_l layers. This is the forward pass that computes the wavefunction $\psi_\theta(s)$ from the input s. In the forward pass, one can free the memory of $\mathbf{x}_{l-1}$ if $\mathbf{x}_l$ has been computed. Therefore, only one $\mathbf{x}_l$ has to be stored in memory. Consider a layered NQS in which the memory cost of $\mathbf{x}_l$ is typically in line with the system size $\mathcal{O}(N)$. Then the memory complexity of a forward pass is also $\mathcal{O}(N)$.

On the other hand, one can express the computation of gradients as

$$\begin{aligned} \boldsymbol{\delta}_{n_l} &= \frac{\partial \psi_\theta(s)}{\partial \mathbf{x}_{n_l}}, \\ \boldsymbol{\delta}_{l-1} &= \boldsymbol{\delta}_l \frac{\partial f_l(\boldsymbol{\theta}_l, \mathbf{x}_{l-1})}{\partial \mathbf{x}_{l-1}}, \\ \frac{\partial \psi_\theta(s)}{\partial \boldsymbol{\theta}_l} &= \boldsymbol{\delta}_l \frac{\partial f_l(\boldsymbol{\theta}_l, \mathbf{x}_{l-1})}{\partial \boldsymbol{\theta}_l}, \end{aligned} \tag{3.47}$$

where $\boldsymbol{\delta}_l = \partial \psi_\theta(s)/\partial \mathbf{x}_l$ is the neuron gradient at the l'th layer. This is the backward pass, which computes the gradient $\partial \psi_\theta(s)/\partial \boldsymbol{\theta}$ after a forward pass. As shown in Eq. (3.47) and also illustrated in Fig. 3.2, the forward pass should store every $\mathbf{x}_l$ in memory if a backward pass will be performed after the forward pass. Therefore, the computation of gradients has memory complexity $\mathcal{O}(Nn_l)$.

The time cost of gradients is usually negligible in VMC, because it only requires one forward and backward pass per sample while the Monte Carlo sampling and the local energy computation require $\mathcal{O}(N)$ forward passes in most cases. However, the memory complexity $\mathcal{O}(Nn_l)$ in the computation of gradients might be a memory bottleneck.

3.4.2 Gradient checkpointing

The gradient checkpointing is a popular method in deep learning to reduce the gradient memory cost $\mathcal{O}(Nn_l)$. In this method, the whole network is split into multiple blocks, each block containing multiple layers, as also presented in Fig. 3.2. The forward pass is still given by Eq. (3.46), and we can rewrite it into an equivalent form with blocks

$$\begin{aligned} \mathbf{x}^{(0)} &= s, \\ \mathbf{x}^{(b)} &= f^{(b)}(\boldsymbol{\theta}^{(b)}, \mathbf{x}^{(b-1)}), \\ \psi_\theta(s) &= \mathbf{x}^{(n_b)}, \end{aligned} \tag{3.48}$$

where $\mathbf{x}^{(b)}$, $\boldsymbol{\theta}^{(b)}$, and $f^{(b)}$ are the output, parameters, and forward function of the b'th block, and there are n_b blocks in total. In the gradient checkpointing, one only has to store $\mathbf{x}^{(b)}$ instead of all $\mathbf{x}_l$ for computing gradients. Therefore, the memory cost is reduced from $\mathcal{O}(Nn_l)$ to $\mathcal{O}(Nn_b)$.

The backward pass with blocks becomes

$$\begin{aligned} \boldsymbol{\delta}^{(n_b)} &= \frac{\partial \psi_\theta(s)}{\partial \mathbf{x}^{(n_b)}}, \\ \boldsymbol{\delta}^{(b-1)} &= \boldsymbol{\delta}^{(b)} \frac{\partial f^{(b)}(\boldsymbol{\theta}^{(b)}, \mathbf{x}^{(b-1)})}{\partial \mathbf{x}^{(b-1)}}, \\ \frac{\partial \psi_\theta(s)}{\partial \boldsymbol{\theta}^{(b)}} &= \boldsymbol{\delta}^{(b)} \frac{\partial f^{(b)}(\boldsymbol{\theta}^{(b)}, \mathbf{x}^{(b-1)})}{\partial \boldsymbol{\theta}^{(b)}}. \end{aligned} \tag{3.49}$$

In the backward pass, $\mathbf{x}^{(b)}$ is not enough for all gradients because it is only the neuron in the last layer of a block. One still needs all neuron values in intermediate layers for gradients, which amounts to $\mathcal{O}(Nn_l^{(b)})$ memory complexity with $n_l^{(b)}$ the number of layers in a block b. It also requires an additional forward pass in each block to recover the intermediate values, which amounts to the time cost of an additional forward pass of the whole network. The total memory complexity, including the cost of storing $\mathbf{x}^{(b)}$, is $\mathcal{O}(N(n_b + n_l^{(b)}))$. Considering the relation $n_l = \sum_{b=1}^{n_b} n_l^{(b)} \sim n_b n_l^{(b)}$, an optimal choice is $n_b \sim n_l^{(b)} \sim \sqrt{n_l}$ so that the total memory complexity becomes $\mathcal{O}(N\sqrt{n_l})$. In the computation of Jacobian or parallel gradients with N_s samples, the cost is $\mathcal{O}(N_s N\sqrt{n_l})$

In summary, the gradient checkpointing method reduces the memory complexity from $\mathcal{O}(N_s N n_l)$ to $\mathcal{O}(N_s N\sqrt{n_l})$ by utilizing an additional forward pass. As the gradient computation is usually a memory bottleneck instead of a time bottleneck, this method helps to alleviate memory issues in VMC with a negligible additional time cost.

3.4.3 Memory-efficient SR

The memory bottleneck in SR is storing the $N_s \times N_p$ matrix $\bar{\mathbf{O}}$ in a simulation with N_s Monte Carlo samples and N_p parameters, for which the direct memory complexity is $\mathcal{O}(N_s N_p)$. Typically N_p is proportional to n_l, so the memory cost of storing $\bar{\mathbf{O}}$ can become larger than the memory cost of gradients. Fortunately, one does not have to store the full $\bar{\mathbf{O}}$ in the memory. Here we discuss how to construct $\mathbf{S} = \bar{\mathbf{O}}^\dagger \bar{\mathbf{O}}$ without storing the full $\bar{\mathbf{O}}$, and one can play the same trick in the computation of $\mathbf{F} = \bar{\mathbf{O}}^\dagger \bar{\boldsymbol{\epsilon}}$.

The elements of $\mathbf{S}$ is given by $S_{k,k'} = \sum_s \bar{O}^*_{s,k}\bar{O}_{s,k'}$, where s iterates over Monte Carlo samples. One can split all samples into multiple batches $\{B\}$ and rewrite

$$S_{k,k'} = \sum_B \sum_{s_B \in B} \bar{O}^*_{s_B,k}\bar{O}_{s_B,k'}. \tag{3.50}$$

After the computation of one batch, one can free the memory of $O_{s_B,k}$ and start the next batch. Therefore, one only needs to store $O_{s_B,k}$ in memory, and the memory complexity is reduced to $\mathcal{O}(N_s N_p/n_B)$, where n_B is the number of batches. Similarly, gradient memory cost with batches is reduced to $\mathcal{O}(N_s N\sqrt{n_l}/n_B)$ if gradient checkpointing is utilized. Although the batched Jacobian computation seems to be slower than the fully parallelized one, it does not lead to a significant slowdown in practice, since the floating-point operation of computing devices usually reaches the maximum power even for a small batch.

In parallel computing over multiple devices, the sum over s or s_B requires data communication among devices, which may become the time bottleneck. To solve this problem, one can perform a sum firstly over samples on the same device and construct $S^{(d)}_{k,k'}$ on each device d, and then sum over different devices to obtain $S_{k,k'} = \sum_d S^{(d)}_{k,k'}$. When there are multiple batches, this method avoids multi-device data communication within each batch.

In conclusion, the memory complexity of SR, under a suitable choice of algorithms, can be reduced to $\mathcal{O}(N_s/n_B \times (N_p + N\sqrt{n_l}))$.

3.4.4 Memory-efficient MinSR

The memory bottleneck in MinSR is also the matrix $\bar{\mathbf{O}}$ with $\mathcal{O}(N_s N_p)$ memory complexity. Similar to the SR case, one does not need to store the full matrix and hence the memory can be saved. The matrix $\mathbf{T}$ constructed from $\bar{\mathbf{O}}$ has elements

$$T_{s,s'} = \sum_k \bar{O}_{s,k}\bar{O}^*_{s',k}. \tag{3.51}$$

Since the sum is over different parameters instead of different samples, the trick in SR cannot be applied here. Nevertheless, we can similarly utilize

$$T_{s,s'} = \sum_l \sum_{k_l} \bar{O}_{s,k_l}\bar{O}^*_{s',k_l}, \tag{3.52}$$

where k_l iterates over parameters in the l'th layer. Therefore, one only needs to store $\bar{O}_{s,k_l}$, and the memory complexity is $\mathcal{O}(N_s N_p/n_l)$, where n_l is the number of parametrized layers and we assume each parametrized layer has a similar amount of parameters. This method is named structured derivatives [72].

It is useful to combine structured derivatives with gradient checkpointing. The full algorithm is shown above, where $x^{(b)}_l$, $\boldsymbol{\theta}^{(b)}_l$, and $f^{(b)}_l$ are used to represent the output, parameters, and forward function at the l'th layer of the b'th block. The memory complexity combining both methods is $\mathcal{O}(N_s(N_p/n_l + N\sqrt{n_l}))$. If we assume $N_p \sim n_l$ in the utilized neural network, the memory cost of memory-efficient MinSR is proportional to

Algorithm 4 Memory-efficient MinSR

Input: Monte Carlo samples s
Output: Neural tangent kernel $\mathbf{T}$

$\mathbf{T} \leftarrow \mathbf{0}_{N_s \times N_s}$
$\mathbf{x}^{(0)} \leftarrow s$
for $b \leftarrow 1$ to n_b **do**
  $\mathbf{x}^{(b)} \leftarrow f^{(b)}(\boldsymbol{\theta}^{(b)}, \mathbf{x}^{(b-1)})$
end for
$\psi \leftarrow \mathbf{x}^{(n_b)}$
$\boldsymbol{\delta} \leftarrow \partial\psi / \partial \mathbf{x}^{(n_b)}$
for $b \leftarrow n_b$ to 1 **do**
  $\mathbf{x}_0^{(b)} \leftarrow \mathbf{x}^{(b-1)}$
  for $l \leftarrow 1$ to $n_l^{(b)}$ **do**
    $\mathbf{x}_l^{(b)} \leftarrow f_l^{(b)}(\boldsymbol{\theta}_l^{(b)}, \mathbf{x}_{l-1}^{(b)})$
  end for
  for $l \leftarrow n_l^{(b)}$ to 1 **do**
    $\mathbf{O} \leftarrow \boldsymbol{\delta} \cdot \partial f_l^{(b)} / \partial \boldsymbol{\theta}_l^{(b)}$
    $\bar{\mathbf{O}} \leftarrow (\mathbf{O} - \langle \mathbf{O} \rangle) / \sqrt{N_s}$
    $\mathbf{T} \leftarrow \mathbf{T} + \bar{\mathbf{O}} \bar{\mathbf{O}}^\dagger$
    $\boldsymbol{\delta} \leftarrow \boldsymbol{\delta} \cdot \partial f_l^{(b)} / \partial \mathbf{x}_{l-1}^{(b)}$
  end for
end for

$\sqrt{N_p}$ instead of N_p in a direct Jacobian computation, which is extremely helpful for deep networks.

Multi-device data communication is unavoidable in structured derivatives, so it may become a time-consuming problem. In most cases, however, this cost is negligible compared to the time cost of Monte Carlo sampling and local energy computation in VMC.

The full algorithm of memory-efficient MinSR is shown above.

Chapter 4

Symmetry

In the previous chapter, it has been outlined how an NQS can be optimized for the purpose of ground state searches. In the following, we will focus on improving the performance of NQSs by imposing symmetries. The symmetry of quantum systems is usually important in VMC for reducing the search space and improving the variational accuracy. In this chapter, we will discuss how to project variational states into the target symmetry sector by utilizing the group theory in quantum physics.

4.1 Group theory

Symmetry plays an essential role in almost all fields of quantum mechanics, including neural quantum states. As the mathematical language of symmetry, the group theory is what we need to understand symmetry.

4.1.1 Definition of group

Let's start with a square as an example. It allows several operations leaving the square unchanged, including rotations by 90°, 180°, 270°, mirror flips in horizontal or vertical directions, and an arbitrary combination of them. In mathematics, the group is a useful structure for these symmetry operations.

Definition 4.1.1 (Group)**.** A group is a set G in which a multiplication operator "$\cdot$" is defined for any two elements $a, b \in G$ such that $g = a \cdot b$ is also an element of G. The following three requirements should also be satisfied.

1. **Associativity**: $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ for every $a, b, c \in G$.
2. **Identity**: There exists a unique identity element $e \in G$ such that $g \cdot e = e \cdot g = g$ for every $g \in G$.
3. **Inverse**: For every $g \in G$, there exists a unique inverse element $g^{-1} \in G$ such that $g \cdot g^{-1} = g^{-1} \cdot g = e$.

Table 4.1: Multiplication table of the C_{4v} group

C_{4v}	e	C_4^1	C_4^2	C_4^3	σ_v	σ_v'	σ_d	σ_d'
e	e	C_4^1	C_4^2	C_4^3	σ_v	σ_v'	σ_d	σ_d'
C_4^1	C_4^1	C_4^2	C_4^3	e	σ_d'	σ_d	σ_v	σ_v'
C_4^2	C_4^2	C_4^3	e	C_4^1	σ_v'	σ_v	σ_d'	σ_d
C_4^3	C_4^3	e	C_4^1	C_4^2	σ_d	σ_d'	σ_v'	σ_v
σ_v	σ_v	σ_d	σ_v'	σ_d'	e	C_4^2	C_4^1	C_4^3
σ_v'	σ_v'	σ_d'	σ_v	σ_d	C_4^2	e	C_4^3	C_4^1
σ_d	σ_d	σ_v'	σ_d'	σ_v	C_4^3	C_4^1	e	C_4^2
σ_d'	σ_d'	σ_v	σ_d	σ_v'	C_4^1	C_4^3	C_4^2	e

Table 4.2: Character table of the C_{4v} group

C_{4v}	E	$2C_4$	C_2	$2\sigma_v$	$2\sigma_d$
A_1	$+1$	$+1$	$+1$	$+1$	$+1$
A_2	$+1$	$+1$	$+1$	-1	-1
B_1	$+1$	-1	$+1$	$+1$	-1
B_2	$+1$	-1	$+1$	-1	$+1$
E	$+2$	0	-2	0	0

The multiplication operator can be viewed as a superposition of symmetry operations. For illustration, in Table 4.1 we show the multiplication table of our exemplary square symmetry operation group, usually named the C_{4v} group. In the table, we use C_4^1, C_4^2, C_4^3 to represent rotations by 90°, 180°, 270°, σ_v, σ_d' to represent vertical and horizontal mirror flips, and σ_d, σ_d' to represent two diagonal mirror flips. The full group G can be generated by multiplications of C_4^1 and σ_v, which is where the name C_{4v} comes from.

4.1.2 Representation

Definition 4.1.2 (Representation)**.** A representation of a group G on a linear space V is a map from any $g \in G$ to a linear operator $\hat{T}_g \in V$, such that

$$\hat{T}_{g_1\, g_2} = \hat{T}_{g_1}\, \hat{T}_{g_2} \tag{4.1}$$

for any $g_1, g_2 \in G$.

One can easily stack two representations $\hat{T}_g^{(1)}$ and $\hat{T}_g^{(2)}$ to form the third representation $\hat{T}_g^{(3)} = \hat{T}_g^{(1)} \oplus \hat{T}_g^{(2)}$ which still satisfies

$$\hat{T}_{g_1 g_2}^{(3)} = \begin{pmatrix} \hat{T}_{g_1}^{(1)}\hat{T}_{g_2}^{(1)} & \\ & \hat{T}_{g_1}^{(2)}\hat{T}_{g_2}^{(2)} \end{pmatrix} = \begin{pmatrix} \hat{T}_{g_1}^{(1)} & \\ & \hat{T}_{g_1}^{(2)} \end{pmatrix} \begin{pmatrix} \hat{T}_{g_2}^{(1)} & \\ & \hat{T}_{g_2}^{(2)} \end{pmatrix} = \hat{T}_{g_1}^{(3)}\hat{T}_{g_2}^{(3)}. \tag{4.2}$$

A new representation can also be constructed by a similarity transformation $\hat{T}_g' = \hat{U}\hat{T}_g\hat{U}^{-1}$. One can perform this construction inversely for decomposition. A complex representation may be decomposed into multiple simple representations by choosing a suitable matrix $\hat{U}$ to block-diagonalize $\hat{T}_g$ for all $g \in G$.

Table 4.3: Character table of the D_6 group

D_6	E	$2C_6$	$2C_3$	C_2	$3C_2'$	$3C_2''$
A_1	$+1$	$+1$	$+1$	$+1$	$+1$	$+1$
A_2	$+1$	$+1$	$+1$	$+1$	-1	-1
B_1	$+1$	-1	$+1$	-1	$+1$	-1
B_2	$+1$	-1	$+1$	-1	-1	$+1$
E_1	$+2$	$+1$	-1	-2	0	0
E_2	$+2$	-1	-1	-2	0	0

Definition 4.1.3. A representation $\hat{T}_g$ is reducible if matrices $\hat{T}_g$ for all $g \in G$ can be simultaneously block-diagonalizable. Otherwise, it is an irreducible representation or irrep.

One can further define the character of a representation invariant under the transformation $\hat{T}'_g = \hat{U}\hat{T}_g\hat{U}^{-1}$ according to the property of matrix trace

$$\operatorname{tr}\hat{T}'_g = \operatorname{tr}(\hat{U}\hat{T}_g\hat{U}^{-1}) = \operatorname{tr}\hat{T}_g. \tag{4.3}$$

Definition 4.1.4 (Character)**.** The character of a representation $\hat{T}_g$ is $\chi(g) = \operatorname{tr}\hat{T}_g$.

Definition 4.1.5 (Conjugacy class)**.** Two group elements g and g' in a group G are in the same conjugacy class, denoted as $g \sim g'$, if there exists $h \in G$ such that $h^{-1}gh = g'$.

It can be directly verified that

$$\chi(g') = \operatorname{tr}\hat{T}_{g'} = \operatorname{tr}(\hat{T}_{h^{-1}}\hat{T}_g\hat{T}_h) = \operatorname{tr}\hat{T}_g = \chi(g). \tag{4.4}$$

Therefore, the elements in the same class have the same character.

Every finite group has a finite amount of irreps and each irrep has different characters. Table 4.2 shows the character table of the C_{4v} group. The symbols in the first row represent the conjugacy class of group elements, where E stands for the conjugacy class of element e, $2C_4$ for C_4^1 and C_4^3, C_2 for C_4^2, $2\sigma_v$ for σ_v and σ'_v, and $2\sigma_d$ for σ_d and σ'_d. The symbols in the first column stand for different irreducible representations. There are in total 5 irreps. For future reference, in Table 4.3 we also show the character table of the D_6 group.

Here we provide several important theorems of group representations without proof.

Theorem 4.1.1 (Schur's lemma)**.** *If $\hat{T}_g$ is an irreducible representation of a group G and a linear operator $\hat{A}$ satisfies*

$$\hat{A}\hat{T}_g = \hat{T}_g\hat{A} \tag{4.5}$$

for all $g \in G$, then $\hat{A} = \lambda\hat{1}$ for some constant λ.

Theorem 4.1.2 (Great orthogonality theorem)**.** *Given two d_r-dimensional irreps $\hat{T}^{(r)}$ and $\hat{T}^{(s)}$ of a group G, we have*

$$\sum_{g\in G} T^{(r)*}_{i,j}(g)T^{(s)}_{l,m}(g) = \frac{|G|}{d_r}\delta_{i,l}\delta_{j,m}\delta_{r,s}, \tag{4.6}$$

where $|G|$ is the number of elements in group G, also called the order of G, and $T^{(r)}_{i,j}(g)$ and $T^{(s)}_{l,m}(g)$ are the matrix elements of $\hat{T}^{(r)}_g$ and $\hat{T}^{(s)}_g$.

Theorem 4.1.3 (Row orthogonality). *Given two irreps r and s of a group G with characters $\chi^{(r)}$ and $\chi^{(s)}$, we have*

$$\sum_{g\in G}\chi^{(r)*}(g)\chi^{(s)}(g) = |G|\delta_{r,s}. \tag{4.7}$$

Theorem 4.1.4 (Column orthogonality). *Given all irreps $\{r\}$ of a group G with characters $\{\chi^{(r)}\}$, we have*

$$\sum_{r}\chi^{(r)*}(g)\chi^{(r)}(h) = \begin{cases} |G|/|c_g|, & g\sim h, \\ 0, & \text{otherwise}, \end{cases} \tag{4.8}$$

where $|c_g|$ is the number of elements in the conjugacy class of the group element g.

4.2 Symmetry in quantum mechanics

4.2.1 Group representation in quantum mechanics

In quantum mechanics, the symmetry of the Hamiltonian can also be studied by groups. For instance, the Hamiltonian $\hat{\mathcal{H}}$ has C_{4v} symmetry if it consists of isometric and homogeneous terms on a square lattice. The group representation $\hat{T}_g$ is a quantum operator acting on the Hilbert space with the following properties.

1. $\hat{T}_g$ does nothing on a vacuum state, i.e., $\hat{T}_g|0\rangle = |0\rangle$.
2. For any quantum state $|\alpha\rangle$, $\hat{T}_g|\alpha\rangle$ does not change the norm of the state, so $\langle\alpha|\hat{T}_g^\dagger\hat{T}_g|\alpha\rangle = \langle\alpha|\alpha\rangle$. Then $\hat{T}_g^\dagger\hat{T}_g = \hat{1}$, or $\hat{T}_g^\dagger = \hat{T}_g^{-1}$.
3. The symmetry transformation of $\hat{\mathcal{H}}$ leaves $\hat{\mathcal{H}}$ invariant, so $\hat{T}_g^\dagger\hat{\mathcal{H}}\hat{T}_g = \hat{\mathcal{H}}$, or $\hat{\mathcal{H}}\hat{T}_g = \hat{T}_g\hat{\mathcal{H}}$.

We can find a suitable basis in which $\hat{T}_g$ is block-diagonal, i.e.,

$$\hat{T}_g = \begin{pmatrix} \ddots & & & & & & \\ & \hat{T}_g^{(r)} & & & & & \\ & & \hat{T}_g^{(r)} & & & & \\ & & & \ddots & & & \\ & & & & \hat{T}_g^{(s)} & & \\ & & & & & \ddots & \end{pmatrix}, \tag{4.9}$$

where $\hat{T}_g^{(r/s)}$ are irreducible representations. Each irreducible representation may appear multiple times in $\hat{T}_g$. According to Schur's lemma Eq. (4.5), the Hamiltonian takes the

form

$$\hat{\mathcal{H}} = \begin{pmatrix} \ddots & & & & & & \\ & E^{(r,1)}\hat{1} & & & & & \\ & & E^{(r,2)}\hat{1} & & & & \\ & & & \ddots & & & \\ & & & & E^{(s,1)}\hat{1} & & \\ & & & & & \ddots & \end{pmatrix}. \tag{4.10}$$

Therefore, the eigenstates live in the subspace of the Hilbert space that furnishes irreducible representations.

4.2.2 Symmetry projection

As we know, the dimension of Hilbert space grows exponentially. Therefore, it is impractical to search for the basis of irreducible representations directly in large systems. However, one can still perform a projection to reduce the effective dimension of the Hilbert space in ED and VMC. In this thesis, we utilize QuSpin [73, 74] for efficient ED in reduced Hilbert space.

Consider a state $|\psi_i^{(s)}\rangle$ which has only one non-zero entry at the i'th row of the irrep $\hat{T}^{(s)}$. Then

$$\hat{T}_g |\psi_i^{(s)}\rangle = \sum_{j=1}^{d_r} T_{j,i}^{(s)}(g) |\psi_j^{(s)}\rangle . \tag{4.11}$$

Multiplying both sides of Eq. (4.11) by $T_{k,k}^{(r)*}$ and summing over g, one obtains

$$\begin{aligned} \sum_{g\in G} T_{k,k}^{(r)*}(g)\hat{T}_g |\psi_i^{(s)}\rangle &= \sum_{j=1}^{d_r} \left(\sum_{g\in G} T_{k,k}^{(r)*}(g) T_{j,i}^{(s)}(g) \right) |\psi_j^{(s)}\rangle \\ &= \frac{|G|}{d_r} \sum_{j=1}^{d_r} \delta_{k,j}\delta_{k,i}\delta_{r,s} |\psi_j^{(s)}\rangle \\ &= \frac{|G|}{d_r} \delta_{k,i}\delta_{r,s} |\psi_i^{(s)}\rangle , \end{aligned} \tag{4.12}$$

where we have utilized the great orthogonality theorem in Eq. (4.6). Then we can define a projection operator

$$\hat{P}_k^{(r)} = \frac{d_r}{|G|} \sum_{g\in G} T_{k,k}^{(r)*}(g)\hat{T}_g, \tag{4.13}$$

which projects any state to the k'th row of the irrep r because it satisfies

$$\hat{P}_k^{(r)} |\psi_i^{(s)}\rangle = \delta_{k,i}\delta_{r,s} |\psi_i^{(s)}\rangle . \tag{4.14}$$

We can also project any state to the irrep r by summing over k, i.e.,

$$\hat{P}^{(r)} = \sum_k \hat{P}_k^{(r)} = \frac{d_r}{|G|} \sum_{g\in G} \chi^{(r)*}(g)\hat{T}_g. \tag{4.15}$$

Consider a representation shown in Eq. (4.9) and an arbitrary state $|\Psi\rangle$ for example. Then we have

$$\hat{P}^{(r)}|\Psi\rangle = \hat{P}^{(r)}\begin{pmatrix}\vdots\\ \psi^{(r,1)}\\ \psi^{(r,2)}\\ \vdots\\ \psi^{(s,1)}\\ \vdots\end{pmatrix} = \begin{pmatrix}\vdots\\ \psi^{(r,1)}\\ \psi^{(r,2)}\\ \vdots\\ 0\\ \vdots\end{pmatrix}. \tag{4.16}$$

Through a suitable choice of sampling methods and network designs, one can also impose SU(2) and gauge symmetries in NQS to further improve the accuracy of simulations [75–78].

4.2.3 Superposition of symmetry groups

Consider two symmetry groups G and G' with $gg' = g'g$ for any $g \in G$ and $g' \in G'$. Then one can define a superposition $H = \{gg' \,|\, g \in G, g' \in G'\}$. It is not hard to verify that H is also a group. Furthermore, if $\hat{T}_g\hat{\mathcal{H}} = \hat{\mathcal{H}}\hat{T}_g$ and $\hat{T}_{g'}\hat{\mathcal{H}} = \hat{\mathcal{H}}\hat{T}_{g'}$ for any $g \in G$ and $g' \in G'$, then

$$\hat{T}_h\hat{\mathcal{H}} = \hat{T}_g\hat{T}_{g'}\hat{\mathcal{H}} = \hat{\mathcal{H}}\hat{T}_g\hat{T}_{g'} = \hat{\mathcal{H}}\hat{T}_h \tag{4.17}$$

for any $h \in H$, so H is also a symmetry group of the Hamiltonian $\hat{\mathcal{H}}$. A typical example of this superposition is the quantum spin system satisfying both total spin conservation $[\hat{S}, \hat{\mathcal{H}}] = 0$ and spin conservation in z-axis $[\hat{S}_z, \hat{\mathcal{H}}] = 0$. Since $[\hat{S}, \hat{S}_z] = 0$, one can combine them for a larger symmetry group whose irreps act on eigenstates. The irrep of the new group H can be constructed as

$$\hat{T}_h^{(r)} = \hat{T}_g^{(s)} \otimes \hat{T}_{g'}^{(t)}, \tag{4.18}$$

and the character is therefore

$$\chi^{(r)}(h) = \mathrm{tr}(\hat{T}_g^{(s)} \otimes \hat{T}_{g'}^{(t)}) = \mathrm{tr}(\hat{T}_g^{(s)})\mathrm{tr}(\hat{T}_{g'}^{(t)}) = \chi^{(s)}(g)\chi^{(t)}(g'). \tag{4.19}$$

Combining Eq. (4.19) with Eq. (4.15), we obtain the projector of the superposition of symmetry groups

$$\hat{P}^{(r)} = \frac{d_s d_t}{|G|\,|G'|}\sum_{g\in G}\sum_{g'\in G'}\chi^{(s)*}(g)\chi^{(t)*}(g')\hat{T}_{gg'}, \tag{4.20}$$

requiring us to sum over $|G|\,|G'|$ terms.

Another important case is the superposition of point-group and translation symmetries, which generally do not commute with each other. However, their superposition is important in the study of lattice systems. Consider a translation symmetry group whose irrep is $\hat{T}_{\mathbf{R}}^{(\mathbf{k})} = e^{i\mathbf{k}\cdot\mathbf{R}}$, where $\mathbf{R}$ is the translation generated by the operator and $\mathbf{k}$ is the momentum. The point group must be a little group leaving $\mathbf{k}$ unchanged. For instance, the whole C_{4v} group is the little group with respect to $\mathbf{k} = (0, 0)$, while the subgroup $\{e, \sigma_v'\}$ is the little group with respect to $\mathbf{k} = (k_x, 0)$.

4.3 Symmetry in lattice models

4.3.1 Spin system

Consider a symmetry group G with group elements g. Its action on a quantum state $|\Psi\rangle$ is given by the group representation $\hat{T}_g$ as

$$|\Psi'\rangle = \hat{T}_g |\Psi\rangle , \tag{4.21}$$

and the action on a spin configuration s is given by the group representation D_g as

$$s' = D_g\, s. \tag{4.22}$$

The two representations are related to each other by

$$\hat{T}_g |s\rangle = |D_g s\rangle . \tag{4.23}$$

For instance, when g is the flip of all spin configurations, $D_g(\uparrow\downarrow\downarrow\downarrow) = \downarrow\uparrow\uparrow\uparrow$, and $\hat{T}_g |\uparrow\downarrow\downarrow\downarrow\rangle = |\downarrow\uparrow\uparrow\uparrow\rangle$. The wave function transformed by g is given by

$$\psi'(s) = \langle s|\hat{T}_g|\Psi\rangle = \langle D_g^{-1}s|\Psi\rangle = \psi(D_g^{-1}s). \tag{4.24}$$

Therefore, the symmetrized wave function given by the symmetry projection in Eq. (4.15) is

$$\psi^{\text{symm}}(s) = \langle s|\hat{P}^{(r)}|\Psi\rangle = \frac{d_r}{|G|}\sum_{g\in G}\chi^{(r)*}(g)\,\psi(D_g^{-1}s). \tag{4.25}$$

By applying this symmetry projection to the variational wavefunctions in VMC, one not only obtains a more accurate variational state but also gets access to excited states as the lowest energy states in specific symmetry sectors, which is a common practice in VMC and NQS [79–82]. If the symmetry group is given by a superposition of symmetries as shown in Eq. (4.20), the symmetrized wave function is

$$\psi^{\text{symm}}(s) = \frac{d_s d_t}{|G|\,|G'|}\sum_{g\in G}\sum_{g'\in G'}\chi^{(s)*}(g)\chi^{(t)*}(g')\,\psi(D_{gg'}^{-1}s). \tag{4.26}$$

4.3.2 Fermion system

For convenience, we define fermion operators in spinful fermion systems as

$$\hat{c}_i = \begin{cases} \hat{c}_{i,\uparrow}, & i \le N, \\ \hat{c}_{i,\downarrow}, & i > N, \end{cases} \tag{4.27}$$

where N is the number of lattice sites. The Fock state in the fermion system is defined as

$$|n\rangle = \prod_{i=1}^{N_e}\hat{c}^\dagger(r_i)\,|0\rangle , \tag{4.28}$$

where N_e is the number of fermions, r_i are indices occupied by fermions, and $|0\rangle$ is the vacuum state. For example, $|\uparrow,0,\uparrow\downarrow,0\rangle = |1,0,1,0,0,0,1,0\rangle = c_1^\dagger c_3^\dagger c_7^\dagger\,|0\rangle$.

In fermion systems, one cannot directly apply the symmetry operator as $\hat{T}_g |n\rangle = |D_g n\rangle$, because additional minus signs might be generated. The correct way to define symmetry operators is

$$\hat{T}_g \hat{c}_i^\dagger \hat{T}_g^\dagger = \hat{c}_{g(i)}^\dagger, \tag{4.29}$$

$$\hat{T}_g \hat{c}_i \hat{T}_g^\dagger = \hat{c}_{g(i)}, \tag{4.30}$$

where $g(i)$ transforms the position i to a new position according to the action of the group element g. As $\hat{T}_g |0\rangle = |0\rangle$ and $\hat{T}_g^\dagger = \hat{T}_g^{-1}$, we have its action on a Fock state $|n\rangle$ as

$$\hat{T}_g |n\rangle = \prod_{i=1}^{N_e} (\hat{T}_g \hat{c}_{r_i}^\dagger \hat{T}_g^\dagger) |0\rangle = \prod_{i=1}^{N_e} \hat{c}_{g(r_i)}^\dagger |0\rangle = \mathrm{sign}(g, n) |D_g n\rangle , \tag{4.31}$$

where D_g is the group representation furnished by the space of Fock states, and $\mathrm{sign}(g, n) = \pm 1$ is determined by the permutation on n. For example, consider a symmetry operation performing a mirror symmetry on a chain, whose action on a Fock state $|\uparrow, 0, \uparrow, 0\rangle$ is

$$\hat{T}_g |\uparrow, 0, \uparrow, 0\rangle = \hat{T}_g \hat{c}_1^\dagger \hat{c}_3^\dagger |0\rangle = (\hat{T}_g \hat{c}_1^\dagger \hat{T}_g^\dagger)(\hat{T}_g \hat{c}_3^\dagger \hat{T}_g^\dagger) |0\rangle = \hat{c}_4^\dagger \hat{c}_2^\dagger |0\rangle = -\hat{c}_2^\dagger \hat{c}_4^\dagger |0\rangle = - |0, \uparrow, 0, \uparrow\rangle , \tag{4.32}$$

in which $\mathrm{sign}(g, n) = -1$.

The symmetrized wave function given by Eq. (4.15) is

$$\psi^{\mathrm{symm}}(n) = \langle n | \hat{P}^{(r)} | \Psi \rangle = \frac{d_r}{|G|} \sum_{g \in G} \mathrm{sign}(g, n)\, \chi^{(r)*}(g)\, \psi(D_g^{-1} n), \tag{4.33}$$

where we have utilized $\mathrm{sign}(g, n) = \mathrm{sign}(g^{-1}, n)$. The only difference compared with spin systems is the additional sign factor $\mathrm{sign}(g, n)$. The symmetrized wave function with superposition of symmetry groups given in Eq (4.20) is

$$\psi^{\mathrm{symm}}(n) = \frac{d_s d_t}{|G|\,|G'|} \sum_{g \in G} \sum_{g' \in G'} \mathrm{sign}(gg', n) \chi^{(s)*}(g) \chi^{(t)*}(g')\, \psi(D_{gg'}^{-1} n). \tag{4.34}$$

4.3.3 Boundary condition

Many symmetries, for instance, the translation symmetry, can generate particles outside the finite lattice, which are generally not allowed with the open boundary condition (OBC). Alternatively, one can map these particles back into the finite lattice by utilizing the periodic boundary condition (PBC), which moves particles outside the lattice to the other side of it. Consider a spin state $|\uparrow, \uparrow, \uparrow, \downarrow\rangle$ for example. The operator $\hat{T}$ translating the spin configuration to the right acts on this state as $\hat{T} |\uparrow, \uparrow, \uparrow, \downarrow\rangle = |\downarrow, \uparrow, \uparrow, \uparrow\rangle$, in which the rightmost spin is translated to the leftmost position due to PBC. In a one-dimensional fermion system, the PBC is expressed by

$$c_{R+L} = c_R, \tag{4.35}$$

in which L is the length of the system. In higher-dimensional systems, the PBC can be applied similarly to all boundaries.

However, the PBC in spin systems might correspond to a different boundary condition, namely the anti-periodic boundary condition (APBC), in fermion systems. To understand APBC, it is convenient to utilize a different fermion ordering

$$\begin{aligned} \hat{c}_{i,\uparrow} &= \hat{c}_{2i-1}, \\ \hat{c}_{i,\downarrow} &= \hat{c}_{2i}, \end{aligned} \tag{4.36}$$

which is different from the ordering in Eq. (4.27) but does not change the physical system. Consider a spin chain with L sites, which can also be taken as a fermion chain with L electrons and no double occupancy. As we know, the translation operator might generate an additional minus sign in fermion systems. For example, consider $L = 4$ and a state $|\uparrow\uparrow\uparrow\downarrow\rangle$. The translation in a spin system with PBC acts as

$$\hat{T}^{\text{spin(PBC)}} |\uparrow\uparrow\uparrow\downarrow\rangle = |\downarrow\uparrow\uparrow\uparrow\rangle . \tag{4.37}$$

Nevertheless, the translation in a fermion system with PBC acts as

$$\begin{aligned} \hat{T}^{\text{fermion(PBC)}} |\uparrow\uparrow\uparrow\downarrow\rangle &= \hat{T}^{\text{fermion(PBC)}} \hat{c}^\dagger_{1,\uparrow}\hat{c}^\dagger_{2,\uparrow}\hat{c}^\dagger_{3,\uparrow}\hat{c}^\dagger_{4,\downarrow} |0\rangle \\ &= \hat{c}^\dagger_{2,\uparrow}\hat{c}^\dagger_{3,\uparrow}\hat{c}^\dagger_{4,\uparrow}\hat{c}^\dagger_{5,\downarrow} |0\rangle \\ &= \hat{c}^\dagger_{2,\uparrow}\hat{c}^\dagger_{3,\uparrow}\hat{c}^\dagger_{4,\uparrow}\hat{c}^\dagger_{1,\downarrow} |0\rangle \\ &= -\hat{c}^\dagger_{1,\downarrow}\hat{c}^\dagger_{2,\uparrow}\hat{c}^\dagger_{3,\uparrow}\hat{c}^\dagger_{4,\uparrow} |0\rangle \\ &= -|\downarrow\uparrow\uparrow\uparrow\rangle . \end{aligned} \tag{4.38}$$

It often happens that the translation should be made the same in spin and fermion systems. Therefore, one can define the APBC as

$$\hat{c}_{R+L} = -\hat{c}_R, \tag{4.39}$$

which leads to

$$\begin{aligned} \hat{T}^{\text{fermion(APBC)}} |\uparrow\uparrow\uparrow\downarrow\rangle &= \hat{T}^{\text{fermion(APBC)}} \hat{c}^\dagger_{1,\uparrow}\hat{c}^\dagger_{2,\uparrow}\hat{c}^\dagger_{3,\uparrow}\hat{c}^\dagger_{4,\downarrow} |0\rangle \\ &= \hat{c}^\dagger_{2,\uparrow}\hat{c}^\dagger_{3,\uparrow}\hat{c}^\dagger_{4,\uparrow}\hat{c}^\dagger_{5,\downarrow} |0\rangle \\ &= -\hat{c}^\dagger_{2,\uparrow}\hat{c}^\dagger_{3,\uparrow}\hat{c}^\dagger_{4,\uparrow}\hat{c}^\dagger_{1,\downarrow} |0\rangle \\ &= \hat{c}^\dagger_{1,\downarrow}\hat{c}^\dagger_{2,\uparrow}\hat{c}^\dagger_{3,\uparrow}\hat{c}^\dagger_{4,\uparrow} |0\rangle \\ &= |\downarrow\uparrow\uparrow\uparrow\rangle . \end{aligned} \tag{4.40}$$

One can easily verify that $\hat{T}^{\text{spin(PBC)}} |s\rangle = -\hat{T}^{\text{fermion(PBC)}} |s\rangle = \hat{T}^{\text{fermion(APBC)}} |s\rangle$ if L is even and $\hat{T}^{\text{spin(PBC)}} |s\rangle = \hat{T}^{\text{fermion(PBC)}} |s\rangle = -\hat{T}^{\text{fermion(APBC)}} |s\rangle$ if L is odd. The choice of PBC and APBC is also discussed in Ref. [74].

Part II

Quantum Spin System

Chapter 5

Neural quantum state architecture

As discussed in Sec. 1.2, artificial neural networks (ANNs) have been widely applied in various fields including quantum many-body physics. In previous chapters, we have introduced the algorithm for training variational states. In this chapter, we will discuss suitable ANN architectures for constructing performant NQSs for studying spin systems.

This chapter is partially related to my following publication and preprint.

- Ao Chen and Markus Heyl
  "Empowering deep neural quantum states through efficient optimization"
  Nat. Phys. 20, 1476 (2024)
- Ao Chen, Vighnesh Naik, and Markus Heyl
  "Convolutional transformer wave functions"
  arXiv:2503.10462 (2025)

5.1 Feed-forward neural network

Here we introduce the most common ANN architecture, the feed-forward neural network (FFNN). FFNN has a multi-layer structure with non-linear maps connecting two neighbor layers. Each layer takes the output of the last layer as its input and produces outputs to serve as the input of the next layer. In Fig. 5.1, we illustrate the structure of a FFNN with n_l layers. Every layer $\mathbf{x}_i$ can be an array with arbitrary dimensions. $\mathbf{x}_0$ is the input of the whole network, and the value of each layer after that is given by

$$\mathbf{x}_n = f_n(\boldsymbol{\theta}_n, \mathbf{x}_{n-1}), \tag{5.1}$$

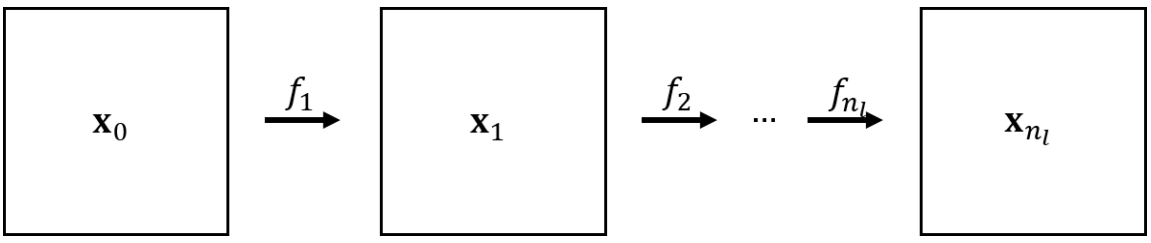


Figure 5.1: Feed-forward neural network

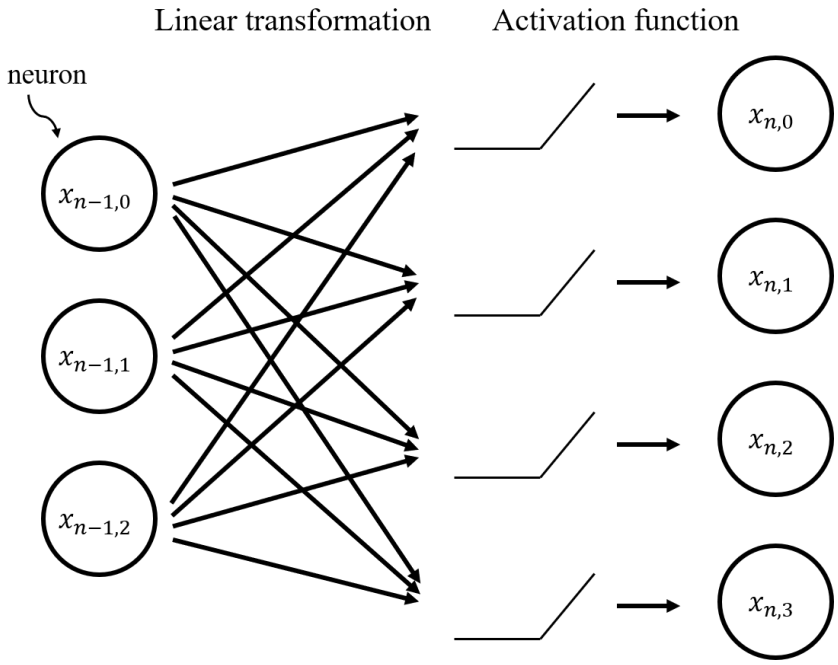


Figure 5.2: Fully-connected layer. The arrows in the linear transformation show the weights connecting different neurons. $x_{n,i}$ represents the i'th neuron at the n'th layer.

where f_n and $\boldsymbol{\theta}_n$ are the forward function and the parameter of the n'th layer. In the end, the last layer $\mathbf{x}_{n_l}$ is taken as the output of the network. The whole process from the input at the first layer to the output at the last layer is called a forward pass, as we have discussed in Sec. 3.4.

A common and convenient implementation of a non-linear forward function f is the fully-connected structure, as shown in Fig. 5.2. Consider a common case in which $\mathbf{x}_n$ and $\mathbf{x}_{n-1}$ are vectors. Analogous to the biological neural networks, we call them neurons because they are nodes of an ANN. The first part of the fully connected layer is a linear transformation connecting neurons in neighbor layers as

$$\mathbf{y} = \mathbf{W}_n \mathbf{x}_{n-1} + \mathbf{b}_n, \tag{5.2}$$

where $\mathbf{W}_n$ and $\mathbf{b}_n$ are trainable weights and bias implementing the linear transformation. The second part is a non-linear activation function g applied to each element of $\mathbf{y}$ to produce a non-linear output

$$\mathbf{x}_n = g(\mathbf{y}). \tag{5.3}$$

The activation function ensures that the full function f is non-linear. Common activation functions include the rectified linear unit (ReLU) [83]

$$\mathrm{ReLU}(x) = \max(0, \mathrm{x}), \tag{5.4}$$

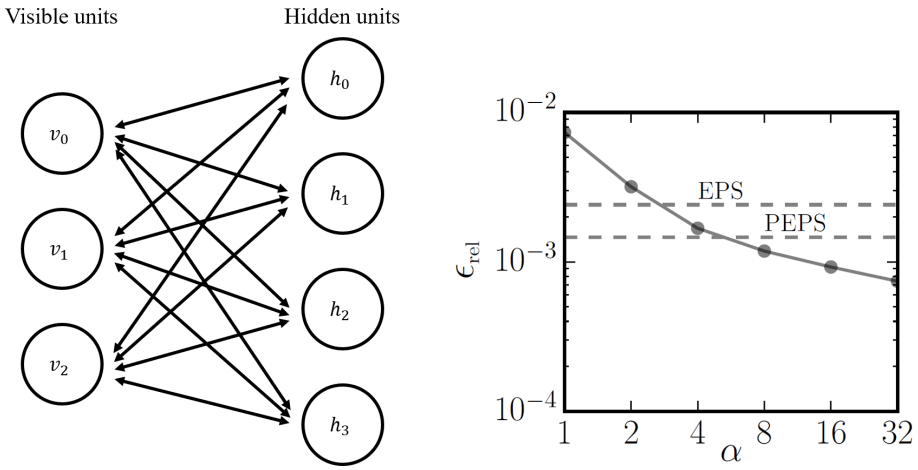
and the Gaussian error linear unit (GELU) [84]

$$\mathrm{GELU}(x) = x\Phi(x), \tag{5.5}$$

where $\Phi(x)$ is the standard Gaussian cumulative distribution function.

Combining Eq. (5.2) and Eq. (5.3), a fully-connected layer can be expressed as

$$\mathbf{x}_n = f_n(\{\mathbf{W}_n, \mathbf{b}_n\}, \mathbf{x}_{n-1}) = g(\mathbf{W}_n \mathbf{x}_{n-1} + \mathbf{b}_n), \tag{5.6}$$



(a) Network structure (b) Variational result

Figure 5.3: Restricted Boltzmann machine. In panel (b), we show the error of variational energy in 10×10 square antiferromagnetic Heisenberg model (taken from Ref. [19]). The x-axis is the ratio between hidden and visible units $\alpha = M/N$. The exact ground state energy is provided by QMC [89].

where $\{\mathbf{W}_n, \mathbf{b}_n\} = \boldsymbol{\theta}_n$ are trainable parameters. The FFNN with several linear layers is sometimes also known as the multi-layer perception (MLP). The linear transformation and the non-linear activation function both allow parallel computation, ensuring efficient implementation on modern GPU hardware.

There have been many theoretical studies on the expressive power of FFNN, among which the most famous one is the universal approximation theorem [85] which states that the multi-layer FFNN can approximate any continuous function. Therefore, the FFNN architecture or other ANN variants are also expected to approximate physical wave functions in quantum many-body problems up to suitable design choices. The FFNN has been utilized in several early works of NQS [86–88].

5.2 Restricted Boltzmann machine

The early studies of NQS started with a focus on the restricted Boltzmann machine (RBM) [19], a simple shallow structure with an extraordinary ability to capture the long-range entanglement in quantum systems [20].

5.2.1 RBM in machine learning

The original idea of RBM comes from a classical Ising model with restricted connections. Consider an Ising model separated into two parts, namely visible units and hidden units [90]. Connections only exist between a visible unit and a hidden unit, but not between two visible units or two hidden units, as shown in Fig. 5.3(a). The energy of this Ising model is

$$E(\mathbf{v}, \mathbf{h}) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_{i,j} W_{i,j} v_i h_j, \tag{5.7}$$

where v_i and h_j are visible and hidden units, a_i and b_j are magnetic fields on visible and hidden units, and $W_{i,j}$ is the interaction strength between v_i and h_j. The probability distribution of this model in a canonical ensemble is

$$P(\mathbf{v}, \mathbf{h}) = \frac{e^{-E(\mathbf{v},\mathbf{h})}}{Z}, \tag{5.8}$$

where we set temperature $T = 1$ for convenience, and Z is the partition function. In machine learning, $\mathbf{v}$ is taken as the input. In computer vision tasks, for instance, $\mathbf{v}$ is an input image. The probability corresponding to the input $\mathbf{v}$ is

$$P(\mathbf{v}) = \sum_{\{h_j=\pm 1\}} P(\mathbf{v}, \mathbf{h}) = \frac{e^{\sum_i a_i v_i}}{Z} \prod_j 2\cosh\left(\sum_i W_{i,j} v_i + b_j\right). \tag{5.9}$$

The advantage of the RBM architecture is it allows efficient sampling. As discussed in Sec. 2.1, one usually proposes new Monte Carlo samples by flipping one spin or swapping neighbor spins. In RBM, nevertheless, the sampling can be performed globally. One can fix all $\mathbf{h}$ and obtain a conditional probability of $\mathbf{v}$ as

$$P(\mathbf{v}|\mathbf{h}) = \frac{e^{\sum_j b_j h_j}}{Z} \prod_i \exp\left[\left(\sum_j W_{i,j} h_j + a_i\right) v_i\right]. \tag{5.10}$$

Therefore, one can sample every v_i with a probability $\propto \exp[(W_{i,j}h_j + a_i)v_i]$ independent of other visible units. Similarly, the conditional probability of $\mathbf{h}$ given $\mathbf{v}$ is

$$P(\mathbf{h}|\mathbf{v}) = \frac{e^{\sum_i a_i v_i}}{Z} \prod_j \exp\left[\left(\sum_i W_{i,j} v_j + b_j\right) h_j\right], \tag{5.11}$$

and h_j can also be sampled independent of other hidden units. Therefore, the samples with a joint distribution given in Eq. (5.8) can be generated by multiple iterations of sampling $\mathbf{v}$ with $\mathbf{h}$ fixed and sampling $\mathbf{h}$ with $\mathbf{v}$ fixed. Due to its simple structure and efficient sampling, RBM has been widely applied to quantum physics [91].

5.2.2 RBM as NQS

When taken as an NQS, the RBM directly takes the probability $P(\mathbf{v})$ in Eq. (5.9) as the wave function $\psi(s)$, where the input $\mathbf{v}$ is the Fock state s. It is common to set all $a_i = 0$ and neglect global constants, so the wave function is given by

$$\psi_\theta(s) = \prod_j \cosh\left(\sum_i W_{i,j} s_i + b_j\right), \tag{5.12}$$

where $\boldsymbol{\theta} = \{W_{i,j}, b_j\}$ is the parameters of the wave function. All parameters are in general complex-valued so that RBM can represent wave functions with both amplitude and phase. In the non-frustrated systems where all $\psi_\theta(s)$ are positive in a suitable basis, the parameters can be set real-valued to accelerate the optimization.

RBM can be viewed as a network with one FFNN layer with the activation function $g(x) = \cosh(x)$. In practice, translation symmetries are often implemented in $W_{i,j}$ to increase the accuracy of the optimization result. In this case, RBM is essentially a 1-layer convolutional neural network, which we will introduce Sec. 5.3. The architecture of RBM contains many long-range connections, which helps RBM to capture long-range entanglement despite its simple architecture [20, 23]. Therefore, RBM reaches extraordinary accuracy in various quantum systems and has been widely applied in NQS studies [19, 28, 80, 81, 92–101].

As an example, in Fig. 5.3(b) the numerical results from Ref. [19] are shown in the two-dimensional antiferromagnetic Heisenberg model

$$\hat{\mathcal{H}} = J \sum_{\langle i,j \rangle} \hat{\mathbf{S}}_i \cdot \hat{\mathbf{S}}_j. \tag{5.13}$$

As the number of hidden units M grows, the network architecture becomes more complicated and the relative error of energy in Eq. (2.16) improves systematically. RBM reaches an error lower than 10^{-3}, outperforming the results given by EPS [11] and PEPS [102] at that time.

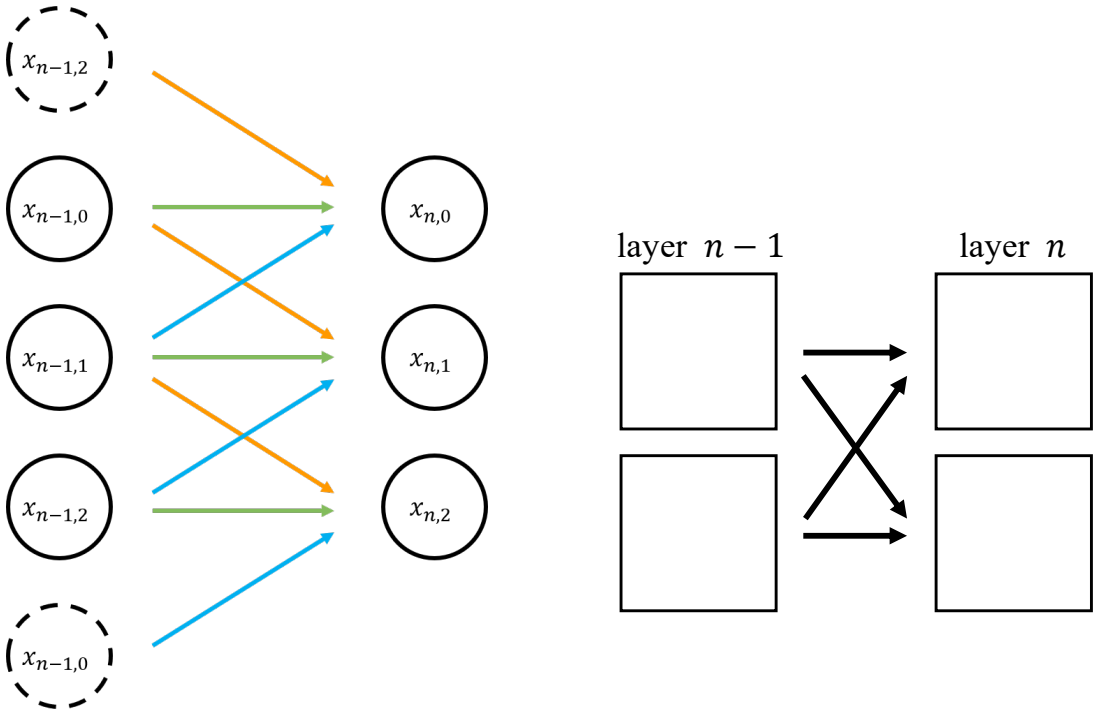
5.3 Convolutional neural network

5.3.1 Convolutional layer

RBM is a simple architecture in the early years of machine learning but shows great performance in NQS studies. As a next step, it is natural to consider the convolutional neural network (CNN) widely used in modern deep learning [103], which directly implements translation symmetry into its structures to reduce the number of parameters and improve accuracy. In Fig. 5.4(a) we illustrate a simple case where each layer only contains 3 neurons. There are three kinds of connections distinguished by the spatial coordinates of neurons, each represented by a different color. Intuitively, the connections with the same color should be similar according to the translation symmetry of the system. The central idea of CNN is to have shared values in weights with the same color, so the number of weights connecting two layers with N neurons is reduced from N^2 to N. This is called a convolution kernel, named after the convolution operation in signal processing. A convolution kernel can be formulated as

$$y(\mathbf{r}') = \sum_{\mathbf{r}} W(\mathbf{r}' - \mathbf{r}) x(\mathbf{r}) + b, \tag{5.14}$$

where W and b are the weight and bias of the kernel, and y and x are the input and output. To further reduce the number of parameters, one can let $W(\mathbf{r}' - \mathbf{r})$ to be non-zero only for several small $|\mathbf{r}' - \mathbf{r}|$, hence not encoding long-range correlations directly in a single layer. Similar to FFNN, the activation function $g(x)$ in Eq. (5.3) is also used after the linear transformation to implement the non-linearity in CNN. The design of a convolutional kernel contains other hyperparameters including stride, padding, and dilation, for which one can read a more detailed introduction in Ref. [103].



(a) Weight sharing (b) Channel

Figure 5.4: Illustration of a CNN layer. (a) The weights with the same color share the same value. The neurons with dashed curves are padded to strictly implement PBC. (b) An example of a CNN layer with 2 input channels and 2 output channels. The weights are not shared between different channels.

The periodic boundary condition (PBC) discussed in Sec. 4.3 can be implemented by padding neurons with dashed lines in Fig. 5.4(a). Consider a translation on inputs $x'(\mathbf{r}) = x(T(\mathbf{r}))$, leading to the same translation on outputs as

$$y'(\mathbf{r}') = \sum_{\mathbf{r}} W(\mathbf{r}' - \mathbf{r})x'(\mathbf{r}) + b = \sum_{\mathbf{r}} W(T(\mathbf{r}') - T(\mathbf{r}))x(T(\mathbf{r})) + b = y(T(\mathbf{r}')), \quad (5.15)$$

where we have utilized $W(\mathbf{r}' - \mathbf{r}) = W(T(\mathbf{r}') - T(\mathbf{r}))$ in the convolutional kernel with PBC. Then the convolutional layer can be interpreted as a translation covariant operation. This relation applies to each convolutional layer, so the translation covariance can be propagated from CNN inputs to outputs. In the last layer of an NQS, a dimension-reducing operation such as

$$\psi(s) = \frac{1}{N} \sum_{\mathbf{r}} e^{i\mathbf{k}\cdot\mathbf{r}} z(s, \mathbf{r}) \quad (5.16)$$

is usually imposed to produce the output wavefunction, where N is the number of lattice sites, $\mathbf{k}$ is the momentum, and $z(s, \mathbf{r})$ are the neurons at the position $\mathbf{r}$ of the last layer when the input Fock state is s. According to the propagation of translation covariance over layers, we have

$$\psi(T_{\mathbf{t}}(s)) = \frac{1}{N} \sum_{\mathbf{r}} e^{i\mathbf{k}\cdot\mathbf{r}} z(T_{\mathbf{t}}(s), \mathbf{r}) = \frac{1}{N} \sum_{\mathbf{r}} e^{i\mathbf{k}\cdot\mathbf{r}} z(s, T_{\mathbf{t}}(\mathbf{r})) = \frac{1}{N} \sum_{\mathbf{r}} e^{i\mathbf{k}\cdot\mathbf{r}} z(\mathbf{r} + \mathbf{t}). \quad (5.17)$$

By a substitution of $\mathbf{r}' = \mathbf{r} - \mathbf{t}$, we have

$$\psi(T_{\mathbf{t}}(s)) = \sum_{\mathbf{r}} e^{i\mathbf{k}\cdot\mathbf{r}} z(T_{\mathbf{t}}(\mathbf{r})) = \sum_{\mathbf{r}'} e^{i\mathbf{k}\cdot(\mathbf{r}'-\mathbf{t})} z(\mathbf{r}') = e^{-i\mathbf{k}\cdot\mathbf{t}} \psi(s), \quad (5.18)$$

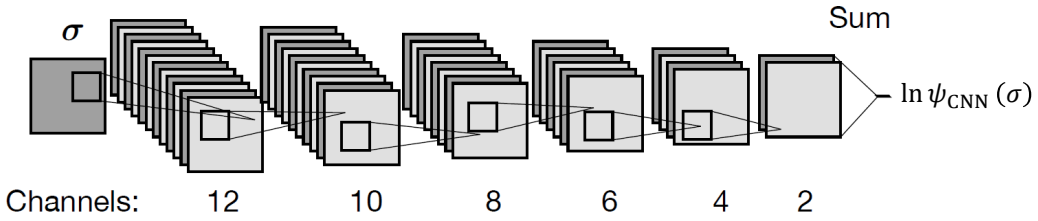


Figure 5.5: CNN architecture in Ref. [104]. The network contains 6 convolutional layers with the number of channels reduced in each layer.

Therefore, the translation symmetry is directly implemented in the CNN architecture and one does not need to apply the symmetry projection in Eq. (4.25), which reduces the computational complexity by $\mathcal{O}(N)$.

The simple network in Fig. 5.4(a) only has one channel in each layer, while a general CNN can have multiple channels, which can be viewed as a new dimension of neurons that differs from the spatial dimensions. In Fig. 5.4(b), a simple CNN layer with two input channels and two output channels is illustrated. Every arrow represents a convolution kernel with shared weights, while different kernels do not share weights for better expressive power. The convolution layer with channels can be formulated as

$$y_{c'}(\mathbf{r}') = \sum_{c,\mathbf{r}} W_{c',c}(\mathbf{r}' - \mathbf{r}) x_c(\mathbf{r}) + b_{c'}, \tag{5.19}$$

where c' and c represent channels of output and input neurons, respectively. After the convolution kernel, the same non-linear activation function is applied to each output channel.

5.3.2 CNN as NQS

The translation-invariant feature of CNN proves to be extremely helpful in solving physical problems where symmetries play a central role, including NQS simulations [58, 104–109]. In Ref. [104], the CNN architecture shown in Fig. 5.5 is utilized to represent quantum states. The network takes a Fock state $|s\rangle$ (in the paper denoted as $|\sigma\rangle$) as the input, and produces $\ln \psi(s)$ as the output. The choice of $\ln \psi(s)$ is utilized here to improve the stability in case $\psi(s)$ becomes too large. The parameters are complex-valued to represent the amplitudes as well as the phases so that the network can solve frustrated systems, and the total number of complex parameters reaches 3838. In Ref. [107], a deeper CNN is used to achieve a better accuracy with 106529 parameters.

In deep learning, an important building block is the residual network (ResNet) [110, 111]. As a deep neural network has a large number of layers, the information of the original input can be lost in the forward pass. Therefore, the residual connection is proposed to restore the information of the input in each layer. If a regular layer is expressed as $\mathbf{x}_n = f_n(\boldsymbol{\theta}_n, \mathbf{x}_{n-1})$, then a layer with a residual connection is

$$\mathbf{x}_n = f_n(\boldsymbol{\theta}_n, \mathbf{x}_{n-1}) + \mathbf{x}_{n-1}, \tag{5.20}$$

which is also translation covariant. Although it appears to be a simple change, the residual connection has become indispensable for building deep networks. In Ref. [58],

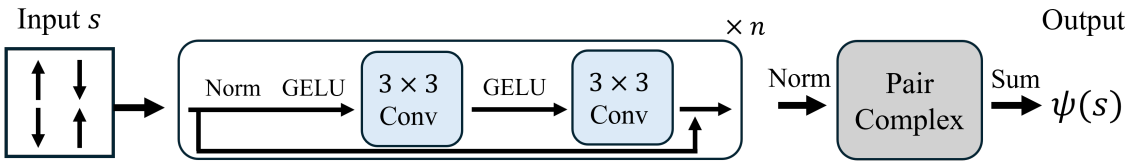


Figure 5.6: CNN architecture in Ref. [58]. Each block contains 2 convolutional layers with a shortcut for the residual connection in Eq. (5.20). The number of channels is kept unchanged across different layers. The activation function is ReLU in the original paper, while the GELU activation function shown here is an improvement. The network is formed by a stack of n blocks. In the end, there is a pair-complex function to be explained in Eq. (5.39) and a summation as shown in Eq. (5.16).

the deep residual network is utilized to achieve unprecedented accuracy in the benchmark frustrated spin system. There are two network architectures shown in the paper, ResNet1 with 64 layers and 146320 parameters, and ResNet2 with 32 layers and 1071488 parameters. The ResNet2 architecture is shown in Fig. 5.6.

5.4 Group convolutional neural network

The group convolutional neural network (GCNN) is a generalization of CNN, which includes not only the translation symmetry but also a general symmetry group G in the convolution. For a brief introduction to the group theory and symmetry in quantum mechanics, one can reference Chapter 4. In a regular convolution kernel as shown in Eq. (5.14), the input $x(\mathbf{r})$ and output $y(\mathbf{r}')$ can be viewed as translation symmetry group elements $T_{\mathbf{r}}$ and $T_{\mathbf{r}'}$, and the weight $W(\mathbf{r}'-\mathbf{r})$ represents the group operation $T_{\mathbf{r}'-\mathbf{r}} = T_{\mathbf{r}}^{-1}T_{\mathbf{r}'}$. The group convolution kernel generalizes this formulation to a general group operation as

$$y(g') = \sum_{g\in G} W(g^{-1}g')x(g) + b. \tag{5.21}$$

If $x(g)$ is transformed by a symmetry group element h to be $x'(g) = x(hg)$, the output $y(g')$ is similarly transformed to be

$$\begin{aligned} y'(g') &= \sum_{g\in G} W(g^{-1}g')x(hg) + b \\ &= \sum_{t\in G} W((h^{-1}t)^{-1}g')x(t) + b \\ &= \sum_{t\in G} W(t^{-1}hg')x(t) + b \\ &= y(hg'), \end{aligned} \tag{5.22}$$

where we let $t = hg$. The group convolution kernel is hence covariant under a group transformation. The CNN channels in Eq. (5.19) can also be implemented similarly in GCNN.

Similar to Eq. (5.16), the summation in the last layer of GCNN with characters $\chi^{(r)}$

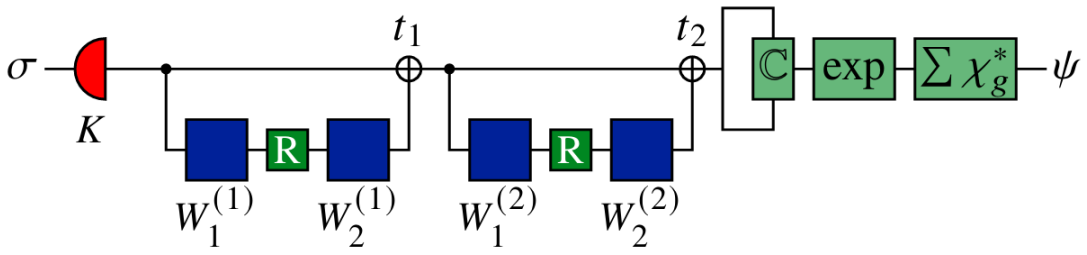


Figure 5.7: GCNN architecture (taken from Ref. [112]). The red box represents an embedding layer. Each blue block represents a group convolution kernel as shown in Eq. (5.21), sandwiching a ReLU activation function. The shortcut is created for every two group convolution layers for the residual connection in Eq. (5.20). In the end, there is a pair-complex function to be explained in Eq. (5.39) and a summation in Eq. (5.23).

is given by

$$\psi(s) = \frac{d_r}{|G|} \sum_{g \in G} \chi^{(r)*}(g) z(s, g^{-1}). \tag{5.23}$$

To show that GCNN provides a symmetrized wave function, we denote $\psi_0(s) = z(s, e)$ when the input at the first layer is s, where e is the identity element of the symmetry group G. Utilizing the propagation of group element operation over layers in Eq. (5.22), we have

$$\psi_0(D_g s) = z(D_g s, e) = z(s, g), \tag{5.24}$$

where D_g is the representation of g furnished by the space of Fock states. Combining Eq. (5.23) and Eq. (5.24), we have

$$\psi(s) = \frac{d_r}{|G|} \sum_{g \in G} \chi^{(r)*}(g) \psi_0(D_g^{-1} s), \tag{5.25}$$

which is equivalent to the symmetry projection on the wave function ψ_0 as shown in Eq. (4.25). Therefore, GCNN itself represents a symmetrized wave function without explicit symmetry projection in Eq. (4.25), which helps to reduce the computational complexity by $\mathcal{O}(|G|)$.

In Fig. 5.7, we show the GCNN architecture in Ref. [112], where the residual structure in Eq. (5.20) is also employed. This architecture successfully approximates several lowest energy states in different irreps of the square and triangular J_1-J_2 Heisenberg models [112] and has also been utilized in several other systems of frustrated magnets [113, 114].

5.5 Transformer

In the field of machine learning, transformer neural networks have developed into the most powerful architecture for many tasks [115, 116]. Consequently, it is a natural immediate question to which extent such transformer architectures could have a similar powerful potential for NQS.

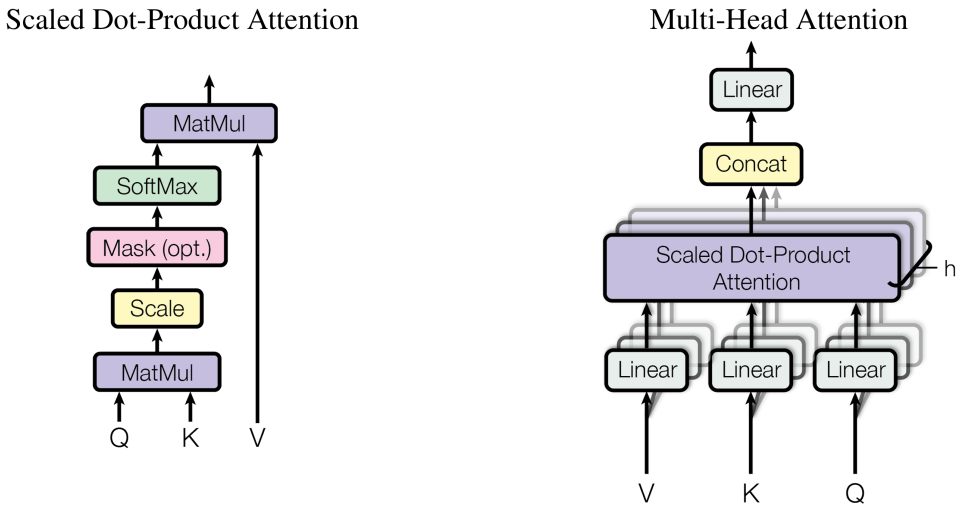


Figure 5.8: Illustration of multi-head self-attention (taken from Ref. [115]).

5.5.1 Multi-head self-attention

The core ingredient of the transformer is the multi-head self-attention (MHSA), which can be viewed as a parallelization of h self-attention heads [115]. The computation within each attention head can be decomposed into the following steps.

1. Each attention head takes the same input $\mathbf{x} \in \mathbb{R}^{N\times c}$, where N represents the flattened spatial dimension and c represents the embedded token dimension (or channel dimension in the context of CNN).

2. For each site i, three vectors, namely query $\mathbf{Q}_i$, key $\mathbf{K}_i$, and value $\mathbf{V}_i$, are computed by a linear transformation of $\mathbf{x}_i$ as

$$\begin{aligned} \mathbf{Q} &= \mathbf{x}\mathbf{W}^Q, \\ \mathbf{K} &= \mathbf{x}\mathbf{W}^K, \\ \mathbf{V} &= \mathbf{x}\mathbf{W}^V, \end{aligned} \tag{5.26}$$

   where $\mathbf{W}^Q \in \mathbb{R}^{c\times d_K}$, $\mathbf{W}^K \in \mathbb{R}^{c\times d_K}$, and $\mathbf{W}^V \in \mathbb{R}^{c\times d_V}$ are trainable parameters. Then the shapes of $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are respectively $N \times d_K$, $N \times d_K$, and $N \times d_V$.

3. The dot products between each query $\mathbf{Q}_i$ and key $\mathbf{K}_j$ are computed for the correlation between site i and j, which is neatly expressed as $(\mathbf{Q}\mathbf{K}^T)_{ij}$.

4. The attention coefficient $\boldsymbol{\alpha}_{ij}$ between site i and j is computed as

$$\boldsymbol{\alpha} = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_K}}\right), \tag{5.27}$$

   where

$$\mathrm{softmax}(\mathbf{X})_{ij} = \frac{\exp(\mathbf{X}_{ij})}{\sum_k \exp(\mathbf{X}_{ik})}. \tag{5.28}$$

   Then the shape of $\boldsymbol{\alpha}$ is $N \times N$.

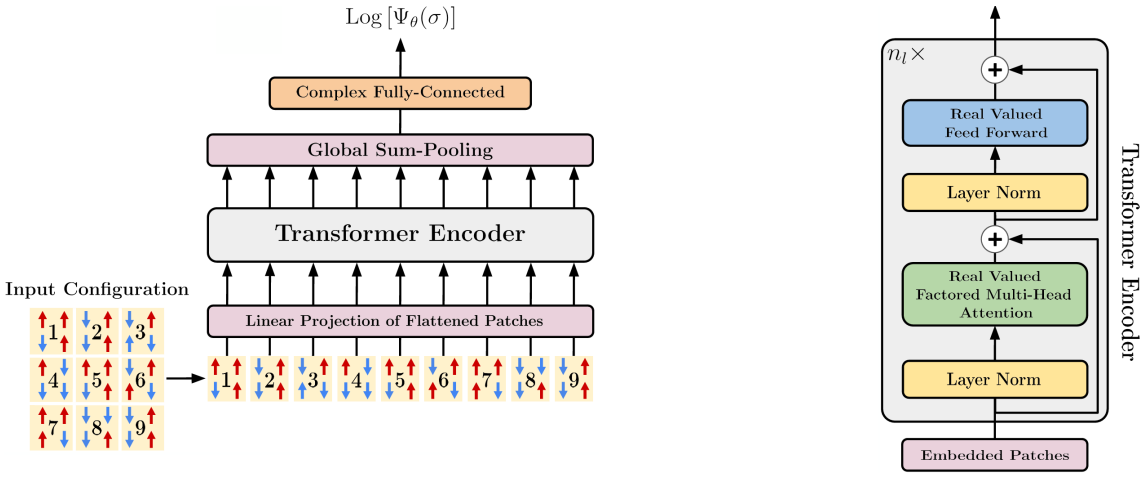


Figure 5.9: Architecture of vision transformer NQS with factored attention (taken from Ref. [117]). Each transformer encoding block contains a factored attention in Eq. (5.34) and a feed-forward layer (sometimes called 1×1 convolution) with shortcuts for the residual connection in Eq. (5.20). After all blocks, there is a sum-pooling and an RBM-like fully-connected layer, which will be further explained in Sec. 5.6.

5. For each site i, the attention $\mathbf{A}_i$ is encoded as

$$\mathbf{A} = \boldsymbol{\alpha}\mathbf{V}, \tag{5.29}$$

whose shape is $N \times d_V$.

In MHSA, the attention $\mathbf{A}$ from h different heads are collected and concatenated into a big matrix $\mathbf{A} \in \mathbb{R}^{N \times d_V h}$, and then transformed into

$$\mathbf{y} = \mathbf{A}\mathbf{W}^O, \tag{5.30}$$

where $\mathbf{W}^O \in \mathbb{R}^{d_V h \times c}$ are trainable parameters, and $\mathbf{y} \in \mathbb{R}^{N \times c}$ are outputs of the MHSA. Combining Eq. (5.26) – (5.30), we can summarize the MHSA as

$$\mathbf{y} = \mathrm{MHSA}(\mathbf{W}, \mathbf{x}), \tag{5.31}$$

where $\mathbf{W} = \{\mathbf{W}_1^Q, \mathbf{W}_1^K, \mathbf{W}_1^V, ..., \mathbf{W}_h^Q, \mathbf{W}_h^K, \mathbf{W}_h^V, \mathbf{W}^O\}$ are trainable parameters. In practice, the residual connection in Eq. (5.20) is also employed to improve the performance of MHSA.

Compared with the convolutional layer, the MHSA has a great advantage in directly encoding long-range correlation through $\mathbf{Q}\mathbf{K}^T$, making transformers the most popular architecture in deep learning. However, the transformer also suffers from the $\mathcal{O}(N^2)$ complexity of $\mathbf{Q}\mathbf{K}^T$. Compared with the $\mathcal{O}(N)$ complexity of convolutional layers, MHSA requires much more computing resources which becomes a main bottleneck in its applications.

5.5.2 Vision transformer NQS with factored attention

The vision transformer (ViT) [116] is a transformer architecture proposed for computer vision tasks with the same MHSA layer as shown in Eq. (5.31). To reduce the computational complexity of MHSA to $\mathcal{O}(N)$, the factored attention is proposed [118] in which

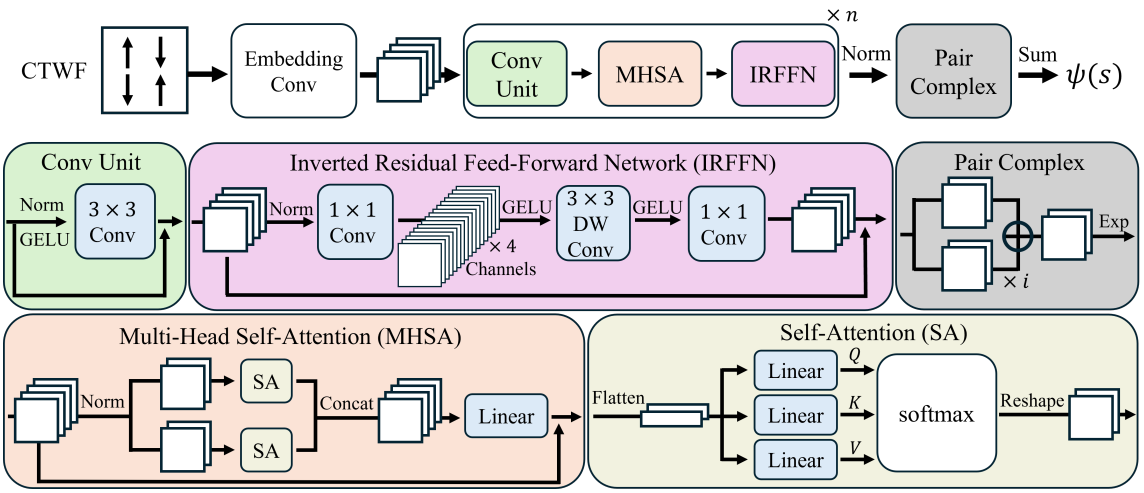


Figure 5.10: Convolutional transformer wavefunction (taken from Ref. [125]). The full network is shown on the top and its ingredient units are further illustrated by colored squares. Each block of CTWF contains three parts, namely Conv Unit, MHSA, and IRFFN. The main part of CTWF consists of n repetitions of these three blocks. The pair-complex function at the end of the network will be explained in Eq. (5.39).

one computes only $\mathbf{V}$ but not $\mathbf{Q}$ and $\mathbf{K}$ in Eq. (5.26), and the attention $\mathbf{A}$ is given by

$$\mathbf{A} = \mathbf{PV}, \tag{5.32}$$

where $\mathbf{P} \in \mathbb{R}^{N\times N}$ is the relative positional encoding [119] with elements given by

$$P_{ij} = p(\mathbf{r}_i - \mathbf{r}_j). \tag{5.33}$$

In 2D systems with side length L_x and L_y, the possible amount of $\mathbf{r}_i - \mathbf{r}_j$ is $N = L_x \times L_y$ considering the periodic boundary, leading to N trainable values in $p(\mathbf{r}_i - \mathbf{r}_j)$. Then the factored attention can be expressed as

$$A_{in} = \sum_{j=1}^{N} p(\mathbf{r}_i - \mathbf{r}_j) V_{jn}, \tag{5.34}$$

which is equivalent to the convolutional kernel shown in Eq. (5.14) with full kernel size $L_x \times L_y$ and no bias [62]. The architecture of ViT NQS with factored attention is illustrated in Fig. 5.9. This architecture has been utilized in several NQS studies [117, 118, 120–124].

5.5.3 Convolutional transformer wavefunction

Another architecture named convolutional transformer wavefunction (CTWF) [125] is also proposed as a combination of CNN and transformer inspired by similar attempts in computer vision [126, 127]. Based on the original MHSA in Eq. (5.31), the MHSA layer in CTWF has an additional relative positional encoding $\mathbf{P}$ as shown in Eq. (5.33) to respect the spatial features, and the attention is given by

$$\mathbf{A} = \text{softmax}\left(\frac{\mathbf{QK}^T + \mathbf{P}}{\sqrt{d_K}}\right)\mathbf{V}. \tag{5.35}$$

Table 5.1: Performance comparison of various NQS architectures in the Heisenberg J_1-J_2 model on the 6×6 square lattice. The tested networks include complex-valued RBM with translation symmetry, CNN in Fig. 5.6, ViT with factored attention in Fig. 5.9, and CTWF in Fig. 5.10. The number of real parameters N_p and the number of multiply-accumulate operations (MACs) are also shown to indicate the complexity of architecture, which we attempt to keep at the same level for different networks.

NQS	N_p	MACs	ϵ_{rel}	σ^2/N	$\mathcal{I}$
RBM	9472	331776	0.0153	0.0228	0.1376
CNN	7120	254016	0.0024	**0.0047**	0.0227
ViT (factored)	7992	317920	0.0040	0.0083	0.0283
CTWF	7884	309420	**0.0023**	0.0055	**0.0137**

After the self-attention, results from different heads are concatenated and linearly transformed according to Eq. (5.30). This structure is also utilized in Ref. [128].

The whole network is designed according to the convolutional transformer architecture in computer vision [127] with some modifications. The attention layer is sandwiched by two convolutional blocks, namely a convolutional unit and an inverted residual feed-forward network (IRFFN), to enhance its ability in representing local features and keep the translation symmetry across the whole network. The combination of MHSA and convolutional layers is expected to encode both long-range and short-range correlations. An illustration of CTWF is shown in Fig. 5.10.

In Table 5.1, we provide a comparison of different network architectures in the 6×6 square J_1-J_2 model at $J_2/J_1 = 0.5$. For a convenient comparison, symmetries other than translation are not applied. Apart from the relative error of energy in Eq. (2.16) and the energy variance in Eq. (2.27), the infidelity between the variational state $|\Psi\rangle$ and the exact ground state $|\Psi_0\rangle$ from ED is provided. The deep networks including CNN, ViT with factored attention, and CTWF all outperform RBM significantly, indicating the importance of deep ANN architecture in solving complex frustrated magnets.

5.6 How to express sign structures?

NQS achieves an outstanding accuracy in the Heisenberg model as shown in Fig. 5.3(b). However, the Heisenberg model with only nearest-neighbor coupling exhibits no sign problem and is hence exactly solvable by QMC. In VMC, the Marshall sign rule (MSR) also provides the exact sign structure of the Heisenberg model, so NQS only needs to express the probability. To enter the next level of complexity, it is necessary to consider models with sign problems and non-trivial sign structures, most of which cannot be solved by existing numerical methods. These quantum many-body problems are then fundamentally different from the CV and Go tasks discussed in Sec. 1.2 because the sign problem is a fundamental difficulty of quantum systems.

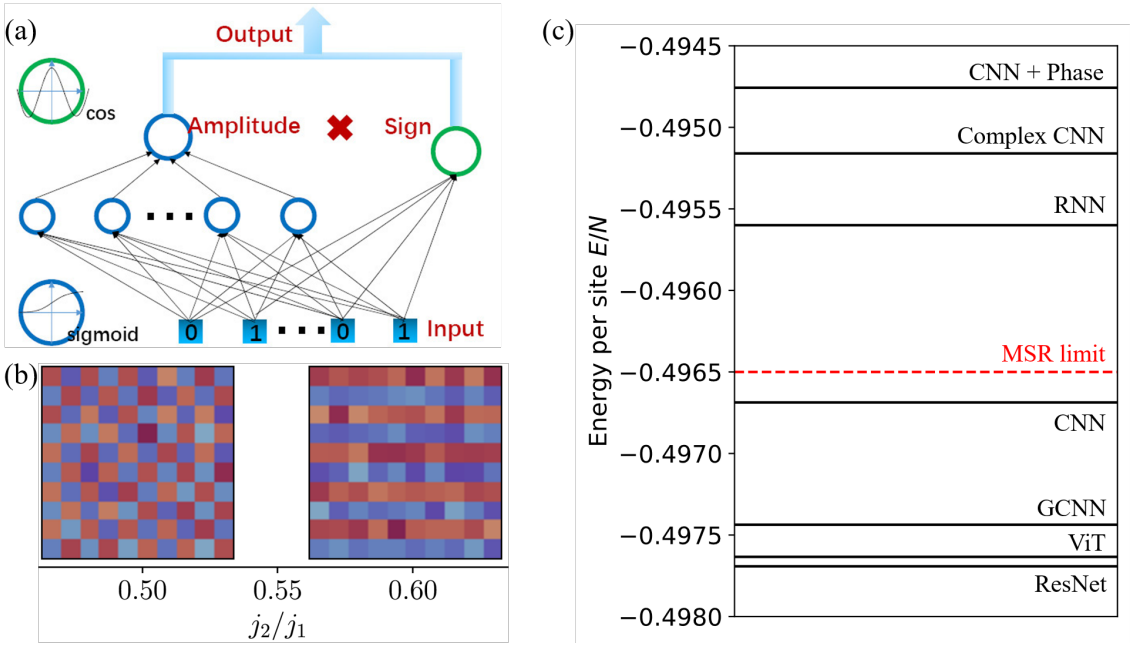


Figure 5.11: Sign network. (a) Combination of amplitude and sign networks (taken from Ref. [86]). (b) Sign structure pattern in the frustrated 10×10 J_1-J_2 model (taken from Ref. [129]). (c) Variational per site E/N at $J_2/J_1 = 0.5$. The data shown here, from top to bottom, includes CNN+Phase [130], complex-valued CNN [104], RNN [45], MSR limit [58], real-valued CNN [107], GCNN [112], ViT [62], and ResNet [58].

5.6.1 Sign network

A direct approach widely adopted in the early years of NQS research is to employ a sign network, as shown in panel (a) of Fig. 5.11. The full wavefunction $\psi(s)$ is decomposed into

$$\psi(s) = \psi^{\text{amp}}(s)\, \psi^{\text{sgn}}(s), \tag{5.36}$$

where $\psi^{\text{amp}}(s)$ is given by an amplitude network only for a positive amplitude, and $\psi^{\text{sgn}}(s)$ is given by a sign network only for a sign or phase [86, 95, 129–131]. Although this design choice appears natural, it does not provide sign structures beyond trivial ones like MSR.

In panels (b) and (c) of Fig. 5.11, we show the performance of sign networks in the 10×10 frustrated J_1-J_2 Heisenberg model. As shown in panel (b), the sign structure pattern obtained by sign networks is either a checkerboard pattern or a stripe pattern [129], both of which can be expressed by simple sign rules like MSR and do not correspond to the correct sign structure of the frustrated system. In panel (c), we show the MSR limit as the best variational energy one can obtain with the sign structure fixed by MSR [58]. The combination of a CNN and a phase network (CNN+Phase) [130], unfortunately, does not outperform the MSR limit, indicating that the variational sign structure given by the phase network does not outperform a trivial one. The recurrent neural network (RNN) [132] or the similar autoregressive network [133] also separates the amplitude and sign parts and therefore does not outperform the MSR limit [45]. These results from numerical experiments suggest that the independent sign network is only suitable for simple sign structures like MSR, while the remaining frustrated sign structures cannot be separated from amplitudes. In Ref. [131], it is also shown that the generalization ability is limited in an independent sign network.

5.6.2 Product output

Another design choice is to utilize a complex-valued ANN and, similar to RBM, have the wavefunction given by a product output

$$\psi = \prod_i z_i, \tag{5.37}$$

where z_i is the neurons at the last layer of the ANN. The complex-valued ANN allows a continuous change of phase without encountering the singular point $\psi = 0$ [67, 104, 118]. As also shown in Fig. 5.11, however, the complex-valued CNN result [104] does not outperform the MSR limit, which seems to disprove the validity of this design choice. Although the utilization of complex-valued ANN appears in a few machine learning applications [134], it is in general not popular for deep networks like CNN because the common ReLU and GELU activation functions cannot be applied to complex-valued neurons. The lack of numerical experience from the machine learning community might be a possible reason for the unsatisfactory performance of the complex-valued CNN.

To avoid complex-valued activation functions in deep networks, Ref. [81] utilizes a complex-valued RBM with a projection of translation and point-group symmetries as shown in Eq. 4.25, which outperforms the MSR limit in the J_1-J_2 model with system sizes up to 8×8. Recently, the ViT architecture shown in Ref. [62, 117] utilizes a real-valued ViT and only a complex-valued RBM-like layer at the end to generate complex-valued wavefunctions, which also avoids complex-valued activation functions and obtains great accuracy as shown in Fig. 5.11. In Ref. [107], it is further demonstrated that one can directly utilize a real-valued CNN for sign structures beyond MSR, and the singular point $\psi = 0$ does not always lead to instability.

5.6.3 Summation output

The great improvement brought by the symmetry projection in Ref. [81] can be understood partially as a benefit of superposition. The symmetry projection creates a symmetrized state as a superposition of component symmetry-broken states as shown in Eq. (4.25). The superposition allows the NQS to adjust the sign structure by adjusting each component state, and the sign change within each component state does not directly impact the sign structure of the full state. On the other hand, the change of signs in components leads to an abrupt change in the full state if it is only a direct product as shown in Eq. (5.37), which might result in instability or worse expressive power of NQS.

As discussed in Eq. (5.16) and Eq. (5.23), the summation output can be employed to create a superposition of more component states. In the usual case in which we search for the ground state in a trivial representation $\mathbf{k} = \mathbf{0}$ or $\chi^{(r)}(g) = 1$, the summation can be simplified as

$$\psi = \frac{1}{|G|} \sum_i z_i, \tag{5.38}$$

where $|G|$ is the order of the symmetry group G.

The many-body wavefunction should scale exponentially with the system size, which is directly implemented in the product output Eq. (5.37) but not in the summation output

Eq. (5.38). To overcome this problem, one has to apply an exponential activation function before applying Eq. (5.38). In Ref. [112], the neural network output z_i is separated into two groups, $z_{i,R}$ and $z_{i,I}$, and a pair-complex function is applied to obtain

$$z'_i = \exp(z_{i,R} + i\, z_{i,I}). \tag{5.39}$$

Then the summation is performed over z'_i. This layer helps the neural network to express exponential scales and complex phases. Inspired by this idea, a shifted sinh layer is proposed in Ref. [58], in which the full network including the output wavefunction is real-valued and the summation is performed over

$$z'_i = \sinh(z_i) + 1, \tag{5.40}$$

where exponential scale is implemented by sinh and the shift $+1$ ensures the wavefunctions to be initialized around $\psi = 1$ instead of $\psi = 0$. As shown in Fig. 5.11, the variational accuracy of NQS with summation outputs significantly outperforms the MSR limit in both Ref. [112] and Ref. [58].

In summary, one can utilize a product or a summation in the output layer to express sign structures together with amplitudes while an independent sign network does not help. The summation shows better performance according to current numerical attempts but further experiments are still needed for a clearer conclusion.

Chapter 6

Neural quantum states for quantum spin liquids

In the previous chapter, we introduced several mainstream NQS architectures for studying quantum spin systems. It is a natural next step to apply these NQSs for solving spin models with theoretical and experimental importance, one of which is the famous quantum spin liquid (QSL). In this chapter, we will introduce the basic idea of QSLs, present NQS benchmarks on QSL models, and then provide recent NQS progress in solving these models.

This chapter is partially related to my following publication and preprint.

- Ao Chen and Markus Heyl
  "Empowering deep neural quantum states through efficient optimization"
  Nat. Phys. 20, 1476 (2024)
- AO Scheie, Minseong Lee, Kevin Wang, P Laurell, ES Choi, D Pajerowski, Qingming Zhang, Jie Ma, HD Zhou, Sangyun Lee, SM Thomas, MO Ajeesh, PFS Rosa, Ao Chen, Vivien S Zapf, M Heyl, CD Batista, E Dagotto, JE Moore, and D Alan Tennant
  "Spectrum and low-energy gap in triangular quantum spin liquid $NaYbSe_2$"
  arXiv:2406.17773 (2024)

6.1 Quantum spin liquid

Physically, a "gas" state means that particles are free to move around due to weak interactions, and in a "solid" state the motion of particles is constrained around fixed positions by strong interactions leading to spontaneous symmetry breaking. The routine to study these systems is more or less clear in theoretical physics. The property of a "gas" can be studied by perturbation of interactions on free particles, and the property of a "solid" is similarly given by perturbation of interactions on collective harmonic modes like phonons. A "liquid" state is between "gas" and "solid" – particles are still relatively free to move around, but the interactions are too strong to be treated perturbatively. The liquid state has been notoriously difficult for theoretical studies. For example, Richard

Feynman claimed turbulence to be "the most important unsolved problem of classical physics". Although fixed spins do not really move around in space, the notion of liquids is generalized to the case in which the interaction does not orient spins toward a specific direction, hence no spontaneous symmetry breaking, while the correlation between spins is still strong.

The notion of quantum spin liquids (QSLs) was proposed by Philip W. Anderson in 1973 [135]. He suggested that the system could remain in a disordered liquid state due to strong quantum fluctuations. This idea was revolutionary at the time, as it challenged the prevailing view that all magnetic systems should order at low temperatures. To understand QSLs, we can consider a J_1-J_2 Heisenberg model

$$\hat{\mathcal{H}} = J_1 \sum_{\langle i,j \rangle} \hat{\mathbf{S}}_i \cdot \hat{\mathbf{S}}_j + J_2 \sum_{\langle\langle i,j \rangle\rangle} \hat{\mathbf{S}}_i \cdot \hat{\mathbf{S}}_j, \tag{6.1}$$

where $\langle i,j \rangle$ and $\langle\langle i,j \rangle\rangle$ represent pairs of nearest neighbors and next-nearest neighbors, respectively.

When the temperature T is much higher than interactions, i.e. $T \gg J_1, J_2$, spins are free to point to arbitrary directions, making the system a paramagnetic "spin gas". On the other hand, when $T \ll J_1, J_2$ and one of J_1 and J_2 is dominant, the system shows an antiferromagnetic order due to the Heisenberg interaction and becomes a "spin solid". However, when $T \ll J_1, J_2$ and $J_1 \sim J_2$, the competition between J_1 and J_2 dominates the behavior of the system.

Although J_1 and J_2 each lead to an antiferromagnetic order, these two orders are often different and incompatible with each other as we will show in Fig. 6.6. Then it is impossible to arrange the orientation of spins to fulfill all antiferromagnetic constraints, which is called spin frustration. Spin frustration leads to highly degenerate and disordered ground states in the classical case. More interestingly, the quantum ground state of frustrated spins becomes a vast quantum superposition of degenerate classical ground states and strong quantum fluctuations might suppress any magnetic order. This state hence becomes a quantum spin liquid, in which quantum effects dominate the formation of spin liquids.

The first example of QSL, namely the resonating valence bond (RVB) state, was proposed by Anderson for an intuitive description of high-temperature d-wave superconductivity in cuprates [137]. Similar to conventional s-wave superconductivity with Cooper pairs, d-wave superconductivity is widely believed to be formed by singlet pairs

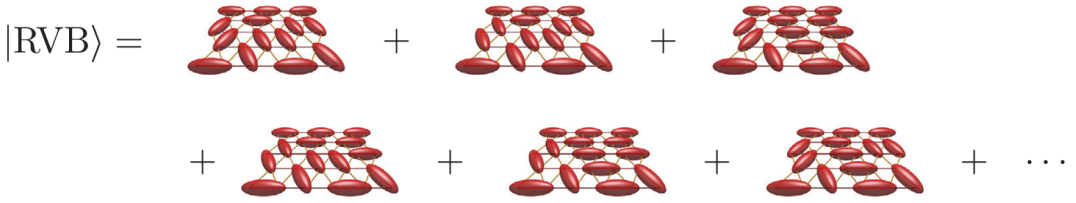
$$|(ab)\rangle = \frac{|\uparrow_a\downarrow_b\rangle - |\downarrow_a\uparrow_b\rangle}{\sqrt{2}}, \tag{6.2}$$

where a and b denote the indices of spins, and a covering of a whole lattice can be constructed by

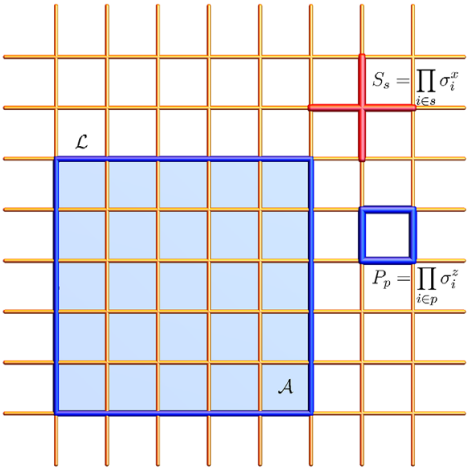
$$|C\rangle = |(ab)(cd)...\rangle\,, \tag{6.3}$$

where the pairs such as (ab) or (cd) are neighbor bonds. Due to the existence of strong interaction and correlation in high-temperature superconductivity, Anderson proposed the RVB state to be a superposition as shown in Fig. 6.1(a),

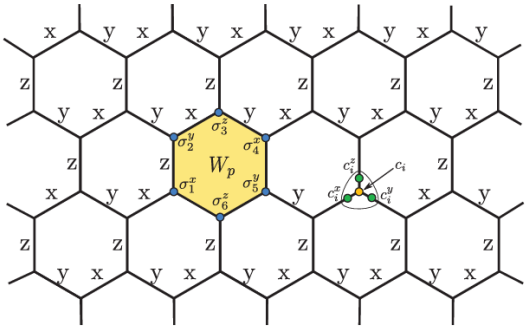
$$|\mathrm{RVB}\rangle = \sum_C |C\rangle\,, \tag{6.4}$$



(a) Resonating valence bond



(b) Toric code



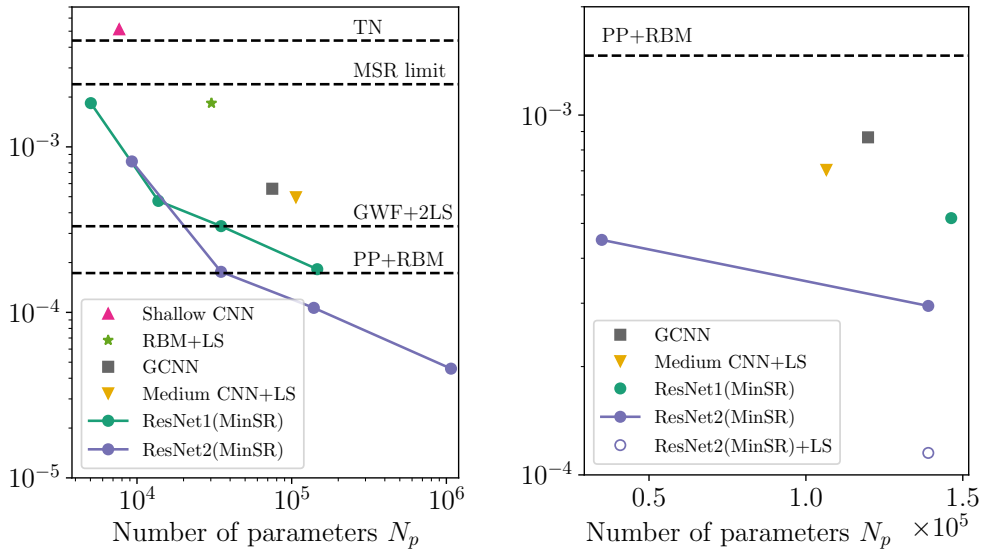
(c) Kitaev honeycomb model

Figure 6.1: Examplary quantum spin liquids (all taken from Ref. [136])

where the summation covers all possible simple states $|C\rangle$ formed by neighbor singlet dimers. It is believed that the RVB state with suitable doping can serve as a qualitative description of high-temperature superconductivity. The RVB state also becomes an instance of QSL, in which the magnetic order is fully suppressed by quantum superposition.

After that, several impressive works show that the QSL can be constructed as the ground state of exactly-solvable Hamiltonians, including the gapped QSL in the toric code model [138] in Fig. 6.1(b) and the gapless QSL in the Kitaev honeycomb model [139] in Fig. 6.1(c). These theoretical models show that the most important property of QSLs is the high entanglement. The RVB state, for instance, is a superposition of a large number of simple states, in which each simple state can also contain long-range pairs enhancing the entanglement. The exceptionally high entanglement can even serve as a more fundamental property of QSLs than the lack of orders [136]. Another defining characteristic of QSLs is the presence of fractionalized excitations. In conventional magnetic phases, excitations take the form of spin waves or magnons, which are collective modes of the ordered spins. In contrast, QSLs can host excitations that carry only a fraction of the spin quantum number, such as spinons, which are spin-1/2 quasiparticles. These fractionalized excitations are a hallmark of the highly entangled nature of QSLs and are closely related to the concept of emergent gauge fields in these systems.

Although QSLs have been found in several designed Hamiltonians, it is extremely difficult to prove their existence in a Hamiltonian closer to real materials like the J_1-J_2 model because these Hamiltonians are in general not exactly solvable. Therefore, great numerical efforts including NQS simulations have been made to study these frustrated quantum magnets to search for evidence supporting the existence of QSLs.



(a) 10×10 (b) 16×16

Figure 6.2: Relative error of variational energy given by various variational methods in the square J_1-J_2 model with PBC at $J_2/J_1 = 0.5$ (taken from Ref. [58]). We include the NQS results from shallow CNN [104], RBM with a Lanczos step (LS) [50], GCNN [112], Medium-scale CNN with a LS [107], and ResNet [58]. Other results not given by pure NQS wavefunctions are shown in dashed lines because their number of parameters is not directly comparable. These results include the tensor network (TN), to be specific, DMRG [140], the Gutzwiller projected wavefunction with 2 LS [141], and the pair product (PP) state with RBM [142].

6.2 Benchmark

Due to the importance and difficulty of solving QSL systems, they are often utilized as benchmarks for numerical methods. In NQS studies, the common benchmark system is the J_1-J_2 model with the Hamiltonian given in Eq. (6.1) on the square lattice. At zero temperature, the system has antiferromagnetic orders when J_1 or J_2 is dominant. There is widely believed to be a QSL phase at $J_2/J_1 \approx 0.5$ in this model [141–148], although some controversies still exist [140, 149–151]. The benchmark is usually performed at $J_2/J_1 = 0.5$ within the expected QSL phase regime.

Before employing NQS, multiple methods have been applied to the square J_1-J_2 model to study its phase diagram and properties. Among these methods, an outstanding one is the Gutzwiller-projected fermionic wavefunction (GWF). As we will discuss in Chapter 7, GWF projects fermionic mean-field wavefunctions into spin degrees of freedom to obtain results beyond uncorrelated particles. Nevertheless, GWF only expresses limited correlation due to its mean-field nature and cannot fully capture the large entanglement in QSLs. Furthermore, tensor network (TN) methods, such as MPS and PEPS, are also applied in the study of QSLs. In large 2D systems, however, TNs are generally limited either by the quasi-1D structure of MPS or the huge computational cost of PEPS. Therefore, these traditional methods provide good benchmarks while leaving many QSL problems unsettled for NQS to explore.

The benchmark has been shown in Fig. 5.11 to illustrate the suitable architecture

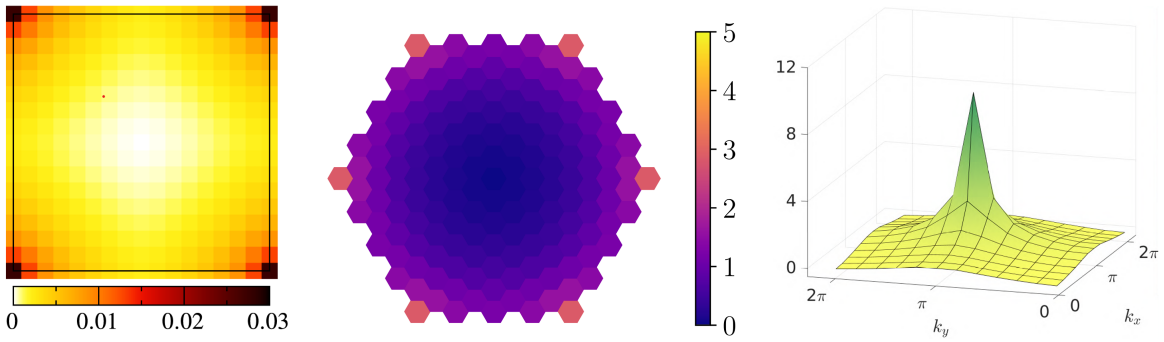
Table 6.1: Variational energy and energy variance in the 10×10 square J_1-J_2 Heisenberg model with PBC at $J_2/J_1 = 0.5$. The ground-state energy is estimated by the zero-variance extrapolation [58].

Wavefunction	Parameters	Ref	E/N	σ^2/N
Ground state		[58]	$\approx -0.497715(9)$	0
TN				
DMRG	8192 SU(2) or 32000 U(1) states	[140]	-0.495530	
PEPS + CNN	$D = 4$ in PEPS + 3531 real in CNN	[152]	-0.495502	
GWF				
Det	5 real	[141]	$-0.49521(1)$	$\approx 7 \times 10^{-3}$
Det + 2LS	5 real	[141]	$-0.49755(1)$	$\approx 2 \times 10^{-3}$
Det + RBM	810 cpl	[153]	$-0.49575(3)$	
PP + RBM	10000 real in PP + 1616 cpl in RBM	[142]	$-0.497629(1)$	$4.9(2) \times 10^{-4}$
Pure NQS				
CNN	3838 cpl	[104]	$-0.49516(1)$	
RNN	Unknown	[45, 132]	$-0.49560(4)$	2.9×10^{-3}
RBM	30000 cpl	[50]	$-0.49580(2)$	
RBM + LS	30000 cpl	[50]	$-0.4968(4)$	
CNN + LS	106529 real	[107]	-0.497468	$\approx 1.4 \times 10^{-3}$
GCNN	≈ 75000 real	[112]	$-0.497437(7)$	
ViT	267720 real	[62]	$-0.497634(1)$	
CTWF	255440 real	[125]	$-0.497679(0)$	$1.59(6) \times 10^{-4}$
ResNet	1071488 real	[58]	$-0.4976921(4)$	$1.80(4) \times 10^{-4}$

for frustrated sign structures. In Fig. 6.2 we present a similar benchmark including the variational energy of various methods in the J_1-J_2 Heisenberg model on the 10×10 and 16×16 square lattices with PBC. Since the exact ground-state energy is not available in these large frustrated systems, we utilize the zero-variance extrapolation result [58] explained in Sec. 2.4 as an estimation, which is energy per site $E/N = -0.497715(9)$ and $-0.4969(4)$ on the 10×10 and 16×16 lattices.

The NQS results provide a clear tendency that more parameters lead to better variational accuracy, which is consistent with our argument in Sec. 1.2, hence proving the necessity of utilizing efficient optimization methods like MinSR in Sec. 3.2. The ResNet reaches an unprecedented size and provides the variational energy $E/N = -0.4976921(4)$, which outperforms all existing numerical results. The extraordinary variational outcomes allow one to accurately estimate the ground-state energy. Compared with the previous best result, the variational error of ResNet is around 4 times lower, suggesting that the deep NQS result is substantially more accurate.

Other variational results not given by pure NQSs are also shown in Fig. 6.2 for comparison. The tensor network (TN), to be specific, DMRG [140], is only suitable for 1D or quasi-1D systems with OBC and hence does not provide good accuracy in this model on a large 10×10 lattice with PBC. The Gutzwiller-projected fermionic wavefunction GWF with the help of 2 Lanczos steps (LS) outperforms previous NQS results but is worse



(a) Square J_1-J_2 [112] (b) Triangular J_1-J_2 [58] (c) Shastry-Sutherland [117]

Figure 6.3: Spin structure factor $S(\mathbf{k})$ in the 16×16 square J_1-J_2 model at $J_2/J_1 = 0.5$, the 12×12 triangular J_1-J_2 model at $J_2/J_1 = 0.125$, and the 12×12 Shastry-Sutherland model at $J/J' = 0.80$. The structure factor in panel (a) is divided by N compared to Eq. (6.3) due to the choice of convention.

than deep ResNet. The pair-product (PP) state is a GWF with pfaffian mean-field wavefunction as we will discuss in Sec. 7.2. The combination of PP and RBM, together with a symmetry projection of translation and point-group symmetries as shown in Eq. 4.25, provides accurate variational energy on the 10×10 lattice but is still outperformed by the deep NQS result. Furthermore, the complexity of the utilized PP+RBM wavefunction on the 10×10 lattice reaches $\mathcal{O}(N^4)$ due to the cost of VMC, pfaffian, and translation symmetry, hence not scalable to larger system size N. As a comparison, the complexity of most NQSs is only $\mathcal{O}(N^2)$ if the network size is kept unchanged. Therefore, a sublattice structure is utilized on the 16×16 lattice to reduce the complexity of PP+RBM to $\mathcal{O}(N^3)$ but in the meantime causes worse variational accuracy. It is then not hard for NQS with $N_p \approx 10^5$ to outperform PP+RBM.

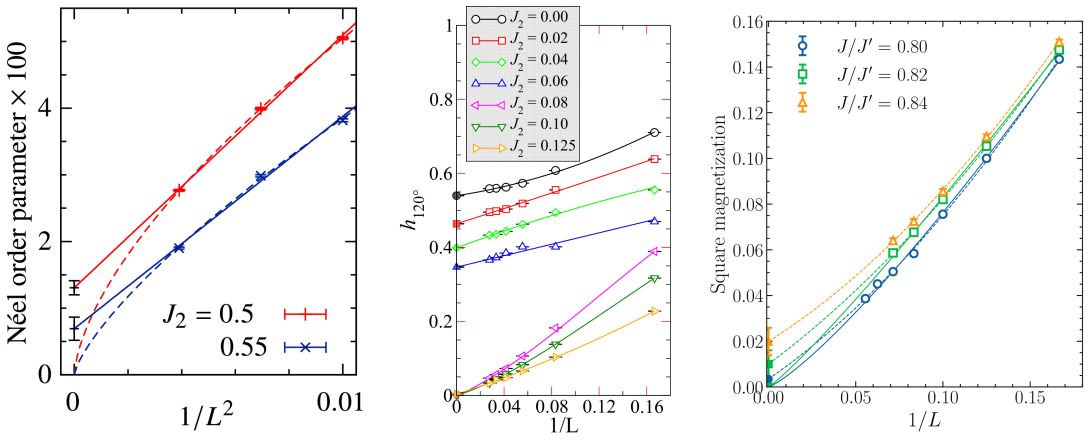
In Table 6.1, we summarize the variational energy and variance values from different variational methods in the 10×10 square J_1-J_2 model at $J_2/J_1 = 0.5$ for a convenient reference.

6.3 Phase diagram

The zero-temperature phase diagram is a fundamental property of a quantum model, by determining which we can verify the existence of suspected QSLs in frustrated magnets. Here we present the numerical methods and results mostly provided by NQSs for the phase diagrams in several QSL candidate models, including the square J_1-J_2 model [112, 142], the triangular J_1-J_2 model [58, 112] and the Shastry-Sutherland model [117].

6.3.1 Structure factor

The square and triangular J_1-J_2 models have similar phase diagrams. The system shows an antiferromagnetic order when J_1 or J_2 is dominant, while the competition between J_1 and J_2 gives rise to the QSL phase at an intermediate J_2/J_1 regime. Several approaches can be employed to determine the phase diagram numerically, and many of them rely



(a) Square J_1-J_2 [112] (b) Triangular J_1-J_2 [154] (c) Shastry-Sutherland [117]

Figure 6.4: Extrapolation of the order parameter $S(\mathbf{K})/N$

on the spin and dimer structures of the variational state. A direct evaluation of the periodic antiferromagnetic order often reveals 0 due to the translation symmetry in NQS, so a correct measure of the order is the spatial correlation [106, 142]. The spin-spin correlation is

$$C_{i,j} = \langle \hat{\mathbf{S}}_i \cdot \hat{\mathbf{S}}_j \rangle \,, \tag{6.5}$$

and the dimer-dimer correlation is

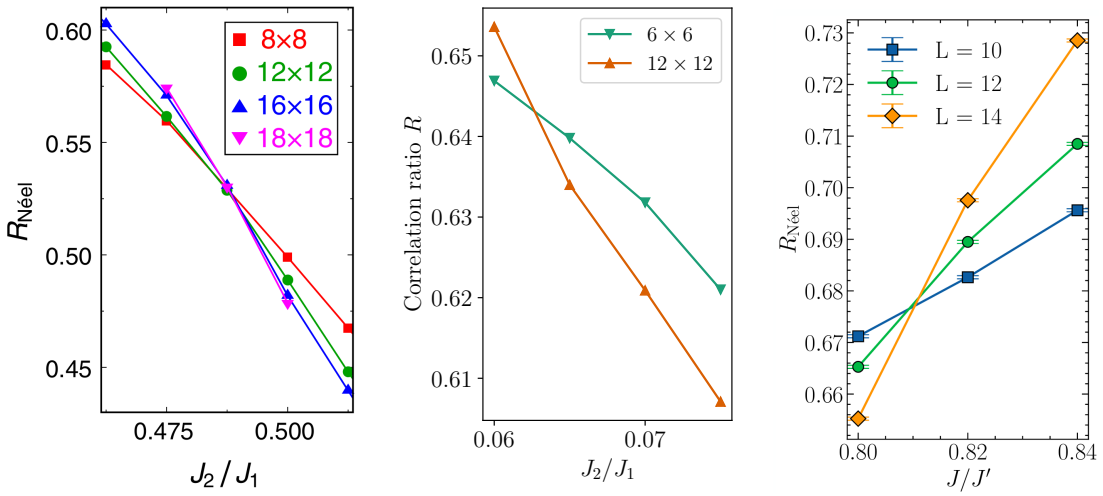
$$C_{i,j} = \langle (\hat{\mathbf{S}}_i \cdot \hat{\mathbf{S}}_{i+\alpha})(\hat{\mathbf{S}}_j \cdot \hat{\mathbf{S}}_{j+\beta}) \rangle - \langle \hat{\mathbf{S}}_i \cdot \hat{\mathbf{S}}_{i+\alpha} \rangle \langle \hat{\mathbf{S}}_j \cdot \hat{\mathbf{S}}_{j+\beta} \rangle \,, \tag{6.6}$$

where α and β are unit vectors in the lattice, and $i+\alpha$ or $j+\beta$ represent the neighbor site of i or j. Quantities such as $\langle \hat{\mathbf{S}}_i \cdot \hat{\mathbf{S}}_j \rangle$ can be computed directly as the expectation of observables in VMC as shown in Eq. (2.22). On the other hand, it is more convenient to compute $\langle (\hat{\mathbf{S}}_i \cdot \hat{\mathbf{S}}_{i+\alpha})(\hat{\mathbf{S}}_j \cdot \hat{\mathbf{S}}_{j+\beta}) \rangle$ from the two-operator expectation discussed in Eq. (2.26) [112].

The periodic spin or dimer order can be detected if the spatial correlation is transformed into the structure factor in momentum space. The structure factor is defined as

$$S(\mathbf{k}) = \frac{1}{N} \sum_{i,j} C_{i,j} e^{i\mathbf{k}\cdot(\mathbf{r}_i - \mathbf{r}_j)} = \sum_i C_{i,0} e^{i\mathbf{k}\cdot\mathbf{r}_i}, \tag{6.7}$$

where N is the system size, $C_{i,j}$ can be either the spin-spin correlation in Eq. (6.5) or the dimer-dimer correlation in Eq. (6.6), and the last equation utilizes the translation symmetry of the system to fix $j = 0$ and $\mathbf{r}_j = 0$. In Fig. 6.3, we show the spin structure factor at the most frustrated regime of the square J_1-J_2 model, the triangular J_1-J_2 model, and the Shastry-Sutherland model [58, 112, 117]. In these finite systems, one can observe spin structure peaks at $\mathbf{K} = (\pi, \pi)$ in the square J_1-J_2 model, $\mathbf{K} = (4\pi/3, 0)$ in the triangular J_1-J_2 model, and $\mathbf{K} = (\pi, \pi)$ in the Shastry-Sutherland model, corresponding to the antiferromagnetic Néel order in respective cases.



(a) Square J_1-J_2 [142] (b) Triangular J_1-J_2 [58] (c) Shastry-Sutherland [117]

Figure 6.5: Correlation ratio of the antiferromagnetic Néel order

6.3.2 Phase diagram

The phase diagram of frustrated magnets can be determined by extrapolating the order parameter $S(\mathbf{K})/N$ to the thermodynamic limit. Ideally, the structure factor remains finite in the ordered phase but reduces to 0 in the QSL phase in an extrapolation to system size $N \to \infty$ or side length $L \to \infty$. The extrapolation should be performed differently in different phases. In an ordered phase, the peak of the structure factor scales like [89, 156]

$$\frac{S(\mathbf{K})}{N} \approx A_0 + \frac{A_1}{L} + \frac{A_2}{L^2}, \tag{6.8}$$

whereas in a disordered phase it scales like [142]

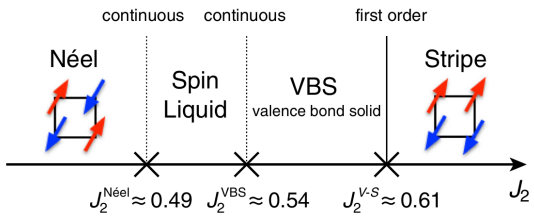
$$\frac{S(\mathbf{K})}{N} \approx L^{-z}. \tag{6.9}$$

Fig. 6.4 shows the extrapolation in different candidate QSL models, including the square and triangular J_1-J_2 models and the Shastry-Sutherland model. In panel (a), the solid and dashed lines show the extrapolation according to Eq. (6.8) with $A_1 = 0$ and Eq. (6.9), respectively [112]. In panel (b), the fitting is performed with Eq. (6.8) [154]. In panel (c), the solid and dashed lines represent the extrapolation with Eq. (6.9) and Eq. (6.8), contrary to panel (a) [117]. The extrapolation to $L \to \infty$ with Eq. (6.8) gives finite order parameters in the ordered phase and vanishing order parameters in the disordered phase, providing a numerical estimation of the phase transition point.

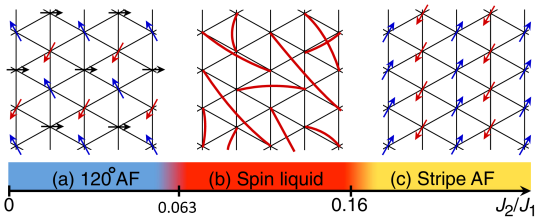
The correlation ratio is an alternative method to obtain the phase transition point from the structure factor but it does not rely on extrapolation. The correlation ratio is defined as [157, 158]

$$R = 1 - \frac{S(\mathbf{K} + \delta\mathbf{k})}{S(\mathbf{K})}, \tag{6.10}$$

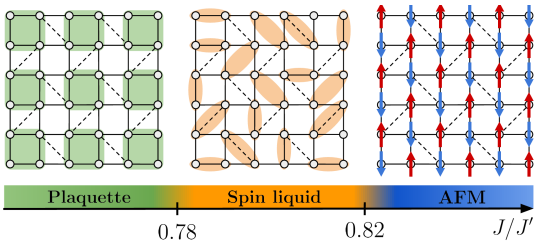
where $S(\mathbf{K})$ is the structure factor peak and $\mathbf{K} + \delta\mathbf{k}$ shifts the peak momentum to a neighbor momentum. The correlation ratio approaches 1 for a strong structure factor



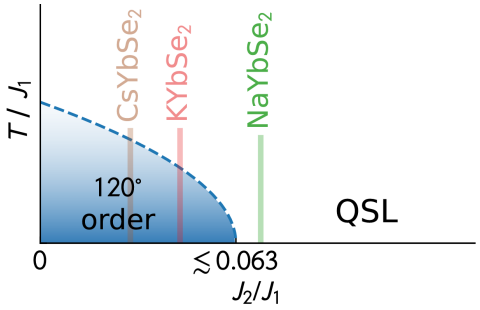
(a) Square J_1-J_2 model [142]



(b) Triangular J_1-J_2 model [58, 154]



(c) Shastry-Sutherland model [117]



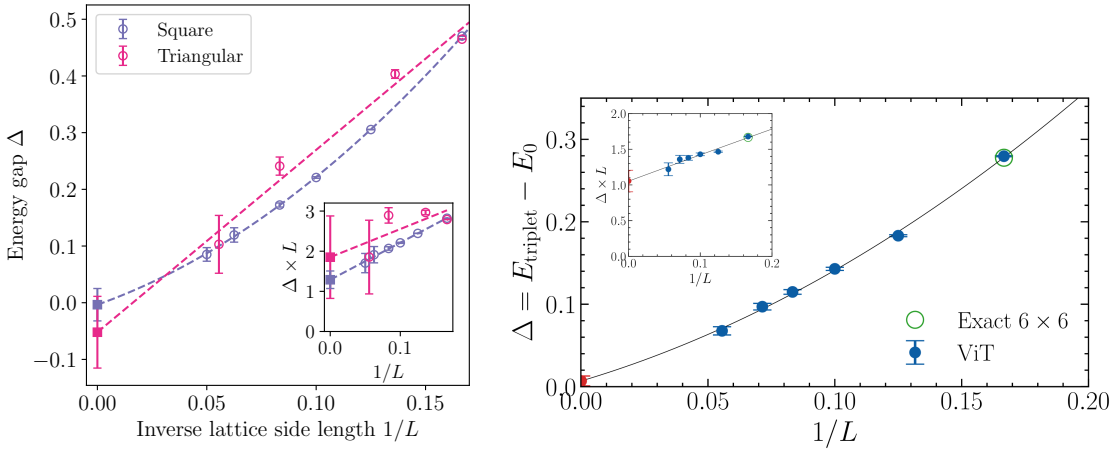
(d) Materials with triangular J_1-J_2 interactions [155]

Figure 6.6: Phase diagrams of QSL candidate models. J_1 is fixed to 1 in panel (a). In panel (b), the phase transition point between the 120° antiferromagnetic phase and the QSL phase is revised to 0.063 according to the result shown in Fig. 6.5 [58].

peak and 0 for a vanishing peak. As shown in Fig. 6.5, the correlation ratio exhibits a crossing point for different system sizes. The peak of the structure factor becomes stronger on one side of the crossing point as the system size increases and weaker on the other side. Therefore, the crossing point indicates a quantum phase transition point and helps us to determine the phase diagram of the studied model.

The phase diagrams obtained by the extrapolation of the structure factor and the crossing point of the correlation ratio are shown in Fig. 6.6. There are QSL phases sandwiched by ordered phases in all three models. The accurate phase diagrams from NQS simulations provide strong evidence supporting the existence of QSLs in frustrated magnets.

One can further compare the numerical phase diagram with real materials. As shown in Fig. 6.6(d), there are several materials with Heisenberg J_1-J_2 interactions at different J_2/J_1. Two materials, $CsYbSe_2$ and $KYbSe_2$, fall into the 120° antiferromagnetic phase and exhibit the predicted order in experiments. The controversial one is $NaYbSe_2$, which has $J_2/J_1 \approx 0.63$ and it is unclear from previous numerical results whether it is in the antiferromagnetic phase or the QSL phase [154, 159–164]. In our work, the NQS simulations show that the phase transition point is at $J_2/J_1 \leq 0.063$, hence suggesting $NaYbSe_2$ to be within the QSL phase, which matches the phenomenon observed in the collaborated experiment [155].



(a) J_1-J_2 model [58]

(b) Shastry-Sutherland model [117]

Figure 6.7: Extrapolation of the triplet gap in QSLs. Both insets show the fitting of $\Delta \times L$ against $1/L$ to further confirm the vanishing gap.

6.4 Quantum spin liquid properties

6.4.1 Energy gap

With the phase diagram settled, one can also study the properties of QSLs in the target models, typically deep in the QSL phase. Here we present energy gaps at $J_2/J_1 = 0.5$ in the square J_1-J_2 model, $J_2/J_1 = 0.125$ in the triangular J_1-J_2 model, and $J/J' = 0.80$ in the Shastry-Sutherland model.

The energy gap is a fundamental property of QSLs and hints at possible topological excitations. As we have shown in Fig. 6.2 and Table 6.1, NQS can provide accurate estimations of the ground-state energy E_0 in QSLs, which can be further enhanced by the zero-variance extrapolation in Sec. 2.4 and the Lanczos step in Sec. 2.5. For the excited-state energy E_1, one can utilize Eq. (4.25) to project wavefunctions to a sector of excited states and search for the lowest energy state in this sector still through VMC and NQS. The ground state is a singlet state with an even total spin S, and a common choice of the excited state is in the triplet sector with an odd S. Then the singlet and triplet states can be distinguished by a symmetry projection

$$\psi^{\pm}(s) = \frac{1}{2}(\psi(s) \pm \psi(-s)), \tag{6.11}$$

where $\psi^{+}(s)$ and $\psi^{-}(s)$ represent the singlet and triplet states, respectively. The lowest energy state in the triplet sector has momentum $\mathbf{k} = (\pi, \pi)$ in the square J_1-J_2 model, $\mathbf{k} = (4\pi/3, 0)$ in the triangular J_1-J_2 model, and $\mathbf{k} = (0, 0)$ in the Shastry-Sutherland model, which should be taken care of in the search of excited states. Then the energy gap can be simply computed as

$$\Delta = E_1 - E_0. \tag{6.12}$$

The finite-size extrapolation of the singlet-triplet gap is usually performed with [165–167]

$$\Delta \approx \Delta_\infty + \frac{a}{L} + \frac{b}{L^2}, \tag{6.13}$$

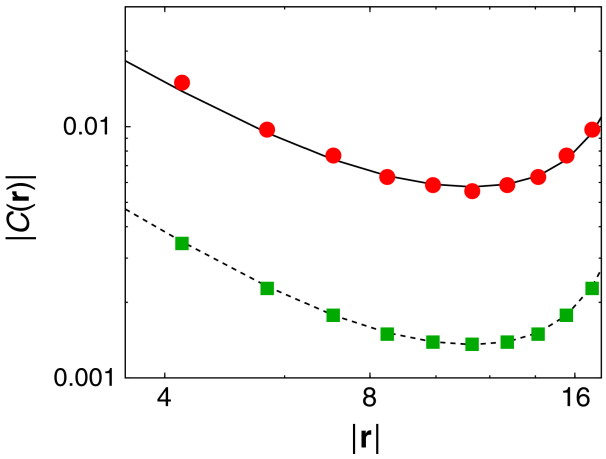


Figure 6.8: Real-space spin-spin correlation (red) and dimer-dimer correlation (green) in the square J_1-J_2 model [142]. The lines are given by the fitting according to the power-law decay in Eq. (6.15).

where Δ_∞ is the energy gap in the thermodynamic limit, or equivalently

$$\Delta \times L \approx a + \frac{b}{L} \tag{6.14}$$

when $\Delta_\infty = 0$. As shown in Fig. 6.7, the energy gaps vanish in the thermodynamic limit in the square and triangular J_1-J_2 models and the Shastry-Sutherland model, indicating the gapless nature of QSLs in all these models.

6.4.2 Real-space correlation

The real-space spin-spin correlation in Eq. (6.5) and dimer-dimer correlation in Eq. (6.6) reflect the microscopic local behavior of spins in QSLs. The power-law decay of the real-space correlation in QSLs is given by [142]

$$C(\mathbf{r}) \propto |\mathbf{r}|^{-(z+\eta)}, \tag{6.15}$$

where $C(\mathbf{r}) = C_{i,j}$ with $\mathbf{r} = \mathbf{r}_i - \mathbf{r}_j$, and z and η are critical exponents. As the real-space correlation can also be measured in experiments, a comparison between numerical simulations and experimental results might provide solid evidence for the existence of QSLs in real materials. In Fig. 6.8, we show the real-space correlation in the 16×16 square J_1-J_2 model at $J_2/J_1 = 0.5125$ [142]. The correlation is measured in the diagonal direction with $r_x = r_y$ to reduce the finite-size effect. The correlation goes up at long distances due to the periodic boundary, which is also fitted well by the power-law decay with PBC as shown by the curves. The great match between the numerical correlation data and the power-law decay further verifies the correctness of NQS wavefunctions.

Part III

Fermion System

Chapter 7

Mean-field fermionic wavefunction

In the subsequent chapters of the thesis, we will be interested in the application of the NQS method to fermion systems. As we will discuss in Sec. 8.1, it is difficult to apply NQSs directly to fermion problems, and it is conventional to combine NQSs to mean-field fermionic wavefunctions including determinants and pfaffians. These wavefunctions are simply derived from the solutions of quadratic Hamiltonians and have been employed as variational wavefunctions in VMC for several decades. In the present chapter, we will take the chance to introduce mean-field fermionic wave functions, which will also be the starting point for the fermionic NQS discussed in the later chapters.

We will focus on spinful fermion systems with N sites ($2N$ orbitals due to spin-1/2) and N_e electrons. For convenience, we define fermion operators in the following way,

$$\hat{c}_i = \begin{cases} \hat{c}_{i,\uparrow}, & i \leq N, \\ \hat{c}_{i-N,\downarrow}, & i > N, \end{cases} \tag{7.1}$$

$$\hat{n}_i = \hat{c}_i^\dagger \hat{c}_i = \begin{cases} \hat{c}_{i,\uparrow}^\dagger \hat{c}_{i,\uparrow}, & i \leq N, \\ \hat{c}_{i-N,\downarrow}^\dagger \hat{c}_{i-N,\downarrow}, & i > N, \end{cases} \tag{7.2}$$

so that an index i can cover both spatial and spinful degrees of freedom.

7.1 Mean-field determinant state

The mean-field determinant state is the ground state of a non-interacting Hamiltonian. The variational method based on the mean-field determinant state is sometimes known as the Hartree-Fock (HF) approximation. In this section, we will introduce how to construct this state and discuss some important properties of the state.

7.1.1 Determinant wavefunction

Consider a non-interacting mean-field Hamiltonian

$$\hat{\mathcal{H}} = \sum_{i,j=1}^{2N} t_{i,j} \hat{c}_i^\dagger \hat{c}_j = \hat{\mathbf{c}}^\dagger \mathbf{T} \hat{\mathbf{c}}, \tag{7.3}$$

where

$$\hat{\mathbf{c}}^\dagger = (\hat{c}_1^\dagger, \hat{c}_2^\dagger, ..., \hat{c}_{2N}^\dagger), \tag{7.4}$$

and

$$\mathbf{T} = \begin{pmatrix} t_{1,1} & ... & t_{1,2N} \\ \vdots & & \vdots \\ t_{2N,1} & ... & t_{2N,2N} \end{pmatrix} \tag{7.5}$$

By matrix diagonalization $\mathbf{T} = \mathbf{U}\mathbf{E}\mathbf{U}^\dagger$ with $\mathbf{E} = \mathrm{diag}(E_0, E_1, ...)$, one can define the quasi-particle operator $\hat{\boldsymbol{\gamma}} = \mathbf{U}^\dagger \hat{\mathbf{c}}$ such that

$$\hat{\mathcal{H}} = \hat{\mathbf{c}}^\dagger \mathbf{U}\mathbf{E}\mathbf{U}^\dagger \hat{\mathbf{c}} = \boldsymbol{\gamma}^\dagger \mathbf{E} \boldsymbol{\gamma} = \sum_\alpha E_\alpha \hat{\gamma}_\alpha^\dagger \hat{\gamma}_\alpha, \tag{7.6}$$

where

$$\hat{\gamma}_\alpha^\dagger = \sum_{i=1}^{2N} U_{i,\alpha}\, \hat{c}_i^\dagger. \tag{7.7}$$

The ground state of $\hat{\mathcal{H}}$ can be constructed by filling non-interacting orbitals as

$$|\Psi_0\rangle = \prod_{\alpha=1}^{N_e} \hat{\gamma}_\alpha^\dagger |0\rangle = \prod_{\alpha=1}^{N_e} \left(\sum_{i=1}^{2N} U_{i,\alpha}\, \hat{c}_i^\dagger \right) |0\rangle \,. \tag{7.8}$$

Since only the lowest N_e orbitals are used, from now on we define $\mathbf{U}$ as a $2N \times N_e$ matrix.

In VMC, one needs to compute the wavefunction of given Fock states. Considering a fermion configuration with fermion occupations at $r_1, r_2, ..., r_{N_e}$ and $r_1 < r_2 < ... < r_{N_e}$, its corresponding Fock state is

$$|n\rangle = \prod_{i=1}^{N_e} \hat{c}_{r_i}^\dagger |0\rangle \,, \tag{7.9}$$

whose wavefunction component in the state $|\Psi_0\rangle$ is given by

$$\psi(n) = \langle n|\Psi_0\rangle = \langle 0|\, \hat{c}_{r_{N_e}} ... \hat{c}_{r_1} \left(\sum_i U_{i,1} \hat{c}_i^\dagger \right) ... \left(\sum_i U_{i,N_e} \hat{c}_i^\dagger \right) |0\rangle = \det(U_{r_i,\alpha}). \tag{7.10}$$

For future convenience, we introduce an operator $n \star \mathbf{U}$ that takes the rows of $\mathbf{U}$ according to the occupation in n, for instance,

$$(1,0,1,0) \star \begin{pmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ 7 & 8 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 5 & 6 \end{pmatrix}. \tag{7.11}$$

Therefore, the wavefunction can be simply expressed as

$$\psi(n) = \det(n \star \mathbf{U}), \tag{7.12}$$

where $\mathbf{U}$ is a $2N \times N_e$ matrix, and $\mathbf{M} = n \star \mathbf{U}$ is an $N_e \times N_e$ matrix. According to the Jacobi's formula proved in Eq. (B.19), the gradient of Eq. (7.12) is

$$\frac{\partial \psi(n)}{\partial M_{ij}} = \frac{\partial \det(\mathbf{M})}{\partial M_{ij}} = \det(\mathbf{M})\, (\mathbf{M}^{-1})_{ji}, \tag{7.13}$$

which can be used to compute the matrix $\mathbf{O}$ and $\bar{\mathbf{O}}$ in VMC, as shown in Eq. (3.7).

7.1.2 Rank-1 update

The $\mathcal{O}(N_e^3)$ complexity of $\det(n \star \mathbf{U})$ is usually too expensive in the simulation of large systems. Fortunately, in VMC the typical situation is one has $\psi(n)$ stored in memory and needs to compute $\psi(n')$, where n' and n are only different by a few particle hoppings. In this case, $n \star \mathbf{U}$ and $n' \star \mathbf{U}$ are only different by a few rows (up to a permutation), and one does not have to recompute the full determinant.

Assume two similar configurations n and n' are only different by one fermion hopping, and the two matrices $\mathbf{M} = n \star \mathbf{U}$ and $\mathbf{M}' = n' \star \mathbf{U}$ are only different at the j'th row. We will show how to compute $\det(\mathbf{M}')$ if $\det(\mathbf{M})$ and $\mathbf{M}^{-1}$ are stored in memory.

The matrix update can be written as a low-rank form

$$\mathbf{M}' - \mathbf{M} = \begin{pmatrix} 0 & ... & 0 \\ \vdots & & \vdots \\ 0 & ... & 0 \\ u_1 & ... & u_{N_e} \\ 0 & ... & 0 \\ \vdots & & \vdots \\ 0 & ... & 0 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} (u_1, ..., u_{N_e}) = \mathbf{e}\mathbf{u}^T, \tag{7.14}$$

where $\mathbf{u}^T = (u_1, ..., u_{N_e})$ is the j'th row (non-zero row) of $\mathbf{M}' - \mathbf{M}$, and $\mathbf{e}$ is the one-hot vector with $e_i = \delta_{i,j}$. Utilizing the matrix determinant lemma in Eq. (B.14), one obtains

$$\det(\mathbf{M}') = \det(\mathbf{M} + \mathbf{e}\mathbf{u}^T) = \det(\mathbf{M})(1 + \mathbf{u}^T\mathbf{M}^{-1}\mathbf{e}), \tag{7.15}$$

and the ratio

$$r = \frac{\det \mathbf{M}'}{\det \mathbf{M}} = 1 + \mathbf{u}^T\mathbf{M}^{-1}\mathbf{e}. \tag{7.16}$$

Therefore, $\det(\mathbf{M}')$ can be computed with $\mathcal{O}(N_e)$ time complexity if $\det(\mathbf{M})$ and $\mathbf{M}^{-1}$ have been computed and stored in memory, which is the typical case in the computation of local energy. The memory complexity is then $\mathcal{O}(N_e^2)$ for storing $\mathbf{M}^{-1}$. In the Markov-chain Monte Carlo sampling, however, one also needs to update $\mathbf{M}^{-1}$ to $\mathbf{M}'^{-1}$ when the new sample n' is accepted, which is given by the Sherman–Morrison formula in Eq. (B.42),

$$\mathbf{M}'^{-1} = (\mathbf{M} + \mathbf{e}\mathbf{u}^T)^{-1} = \mathbf{M}^{-1} - \frac{\mathbf{M}^{-1}\mathbf{e}\mathbf{u}^T\mathbf{M}^{-1}}{1 + \mathbf{u}^T\mathbf{M}^{-1}\mathbf{e}} = \mathbf{M}^{-1} - \frac{\mathbf{M}^{-1}\mathbf{e}\mathbf{u}^T\mathbf{M}^{-1}}{r} \tag{7.17}$$

with $\mathcal{O}(N_e^2)$ complexity.

The time bottleneck of this method comes from the outer product $(\mathbf{M^{-1}e})(\mathbf{u}^T\mathbf{M}^{-1})$, which is bounded by memory bandwidth instead of float-point operations. The delayed update strategy can be utilized to accelerate the computation. At each time step t, one does not have to update $\mathbf{M}_{t-1}^{-1}$ to $\mathbf{M}_t^{-1}$, but instead stores additional quantities $\mathbf{a}_t = \mathbf{M}_{t-1}^{-1}\mathbf{e}_t$, $\mathbf{b}_t^T = \mathbf{u}_t^T\mathbf{M}_{t-1}^{-1}$, and r_t such that

$$\mathbf{M}_\tau^{-1} = \mathbf{M}_0^{-1} - \sum_{t=1}^{\tau} \mathbf{a}_t\mathbf{b}_t^T. \tag{7.18}$$

Without explicitly constructing $\mathbf{M}_\tau^{-1}$, one can update the following quantities

$$\mathbf{a}_\tau = \left(\mathbf{M}_0^{-1} - \sum_{t=1}^{\tau-1} \mathbf{a}_t \mathbf{b}_t^T\right) \mathbf{e}_\tau = \mathbf{M}_0^{-1}\mathbf{e}_\tau - \sum_{t=1}^{\tau-1} \mathbf{a}_t (\mathbf{b}_t^T \mathbf{e}_\tau), \tag{7.19}$$

$$\mathbf{b}_\tau = \left((\mathbf{M}_0^{-1})^T - \sum_{t=1}^{\tau-1} \mathbf{b}_t \mathbf{a}_t^T\right) \mathbf{u}_\tau = (\mathbf{M}_0^{-1})^T\mathbf{u}_\tau - \sum_{t=1}^{\tau-1} \mathbf{b}_t (\mathbf{a}_t^T \mathbf{u}_\tau), \tag{7.20}$$

$$r_\tau = 1 + \mathbf{u}_\tau^T \left(\mathbf{M}_0^{-1} - \sum_{t=1}^{\tau-1} \mathbf{a}_t \mathbf{b}_t^T\right) \mathbf{e}_\tau = 1 + \mathbf{u}_\tau^T\mathbf{M}_0^{-1}\mathbf{e}_\tau - \sum_{t=1}^{\tau-1} (\mathbf{u}_\tau^T \mathbf{a}_t)(\mathbf{b}_t^T \mathbf{e}_\tau). \tag{7.21}$$

Delayed updates avoid the slow outer product by changing both time and memory complexity from $\mathcal{O}(N_e^2)$ to $\mathcal{O}(N_e^2 + \tau N_e)$. Therefore, Eq. (7.18) should be utilized to explicitly construct matrix $\mathbf{M}$ after some update steps to constrain the maximum value of τ and limit the complexity within $\mathcal{O}(N_e^2)$.

7.1.3 Rank-k update

In a more general case, $\mathbf{M}'$ and $\mathbf{M}$ are different at $j_1, j_2, ..., j_k$ rows. One can generalize $\mathbf{u}^T$ to a $k \times N_e$ matrix containing the non-zero rows of $\mathbf{M}' - \mathbf{M}$, and $\mathbf{e}$ to an $N_e \times k$ matrix with $e_{i,n} = \delta_{i,j_n}$. The matrix update can still be expressed as

$$\mathbf{M}' - \mathbf{M} = \mathbf{e}\mathbf{u}^T, \tag{7.22}$$

and

$$\mathbf{R} = \mathbf{1}_k + \mathbf{u}^T\mathbf{M}^{-1}\mathbf{e}, \tag{7.23}$$

$$\det \mathbf{M}' = \det \mathbf{M} \det \mathbf{R}, \tag{7.24}$$

$$\mathbf{M}'^{-1} = \mathbf{M}^{-1} - \mathbf{M}^{-1}\mathbf{e}\mathbf{R}^{-1}\mathbf{u}^T\mathbf{M}^{-1}, \tag{7.25}$$

where $\mathbf{R}$ is a small $k \times k$ matrix. The delayed update can be implemented similarly by storing $\mathbf{a}_t = \mathbf{M}_{t-1}^{-1}\mathbf{e}_t$ and $\mathbf{b}_t^T = \mathbf{R}^{-1}\mathbf{u}_t^T\mathbf{M}_{t-1}^{-1}$. The overall time complexity is $\mathcal{O}(kN_e^2 + k^3)$ or $\mathcal{O}(kN_e^2)$ in the typical $k \ll N_e$ case, and the memory complexity is still $\mathcal{O}(N_e^2)$.

7.2 Mean-field pfaffian state

7.2.1 Pfaffian wavefunction

The pfaffian wavefunction is introduced to consider not only quasi-particles as described by determinants but also the pairing between particles [79, 168]. Therefore, it has a great advantage in describing the conventional superconductivity in which electrons form Cooper pairs, and is potentially beneficial in the study of unconventional superconductivity.

Consider a quadratic mean-field Hamiltonian

$$\hat{\mathcal{H}} = \sum_{i,j=1}^{2N} (t_{i,j}\hat{c}_i^\dagger \hat{c}_j + \Delta_{i,j}\hat{c}_i^\dagger \hat{c}_j^\dagger + \Delta_{i,j}^* \hat{c}_i \hat{c}_j), \tag{7.26}$$

where $t_{i,j} = t_{j,i}^*$ and $\Delta_{i,j} = -\Delta_{j,i}$. This mean-field Hamiltonian can be solved by a generalized Bogoliubov transformation [169]

$$\hat{\gamma}_\alpha = \sum_{i=1}^{2N} (u_{i,\alpha}\hat{c}_i + v_{i,\alpha}\hat{c}_i^\dagger). \tag{7.27}$$

To ensure $\hat{\gamma}_\alpha$ satisfies $\{\hat{\gamma}_\alpha, \hat{\gamma}_\beta\} = 0$, $\{\hat{\gamma}_\alpha^\dagger, \hat{\gamma}_\beta^\dagger\} = 0$, and $\{\hat{\gamma}_\alpha^\dagger, \hat{\gamma}_\beta\} = \delta_{\alpha,\beta}$, the matrices $\mathbf{u}$ and $\mathbf{v}$ should satisfy the following conditions,

$$\begin{aligned} \mathbf{u}^\dagger\mathbf{u} + \mathbf{v}^\dagger\mathbf{v} &= \mathbf{1}, \\ \mathbf{u}^T\mathbf{v} + \mathbf{v}^T\mathbf{u} &= \mathbf{0}. \end{aligned} \tag{7.28}$$

The ground-state solution takes the form

$$|\Psi\rangle = \exp\left(\frac{1}{2}\sum_{i,j=1}^{2N} F_{i,j}\hat{c}_i^\dagger\hat{c}_j^\dagger\right)|0\rangle\,, \tag{7.29}$$

where $F_{i,j}$ is determined by the condition $\hat{\gamma}_\alpha|\Psi\rangle = 0$, which leads to

$$\mathbf{F}\mathbf{u} = \mathbf{v}. \tag{7.30}$$

In practice, one usually does not start from $t_{i,j}$ and $\Delta_{i,j}$ but directly takes $F_{i,j}$ as variational parameters in VMC. There is redundancy in this parametrization because $F_{i,j}\hat{c}_i^\dagger\hat{c}_j^\dagger + F_{j,i}\hat{c}_j^\dagger\hat{c}_i^\dagger = (F_{i,j} - F_{j,i})\hat{c}_i^\dagger\hat{c}_j^\dagger$ shows that only $F_{i,j} - F_{j,i}$ is an independent parameter. By convention, we choose $F_{i,j} = -F_{j,i}$ and the matrix $\mathbf{F}$ is skew-symmetric. Then one can also express $\sum_{i,j} F_{i,j}\hat{c}_i^\dagger\hat{c}_j^\dagger/2$ as $\sum_{i<j} F_{i,j}\hat{c}_i^\dagger\hat{c}_j^\dagger$.

The number of particles is not conserved in Eq. (7.29) while it should be conserved in many systems. Then it is often helpful to project $|\Psi\rangle$ into the sector with N_e electrons as

$$|\Psi\rangle = \frac{1}{(N_e/2)!}\left(\frac{1}{2}\sum_{i,j=1}^{2N} F_{i,j}\hat{c}_i^\dagger\hat{c}_j^\dagger\right)^{N_e/2}|0\rangle\,, \tag{7.31}$$

which is the most popular form utilized in VMC.

The wavefunction $\psi(n)$ is obtained by applying a Fock state $|n\rangle$ given in Eq. (7.9) on the left, i.e.

$$\psi(n) = \langle n|\Psi_0\rangle = \langle 0|\,\hat{c}_{r_{N_e}}...\hat{c}_{r_1}\frac{1}{(N_e/2)!}\left(\frac{1}{2}\sum_{i,j} F_{i,j}\hat{c}_i^\dagger\hat{c}_j^\dagger\right)^{N_e/2}|0\rangle = \mathrm{pf}(F_{r_i,r_j}), \tag{7.32}$$

where pf is a pfaffian function that takes a matrix as input and outputs a number similar to the determinant, as explained in Eq. (B.26). To simplify the expression of the pfaffian wavefunction, we extend the definition of the "$\star$" operator in Eq. (7.11) to express row slicing as $n \star \mathbf{F}$ and column slicing as $\mathbf{F} \star n$. For instance,

$$(1,0,1,0)\star\begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{pmatrix}\star(0,1,0,1) = \begin{pmatrix} 2 & 4 \\ 10 & 12 \end{pmatrix}. \tag{7.33}$$

Therefore, the wavefunction can be simplified as

$$\psi(n) = \mathrm{pf}(n \star \mathbf{F} \star n), \tag{7.34}$$

where $\mathbf{F}$ is a $2N \times 2N$ matrix, and $\mathbf{M} = n \star \mathbf{F} \star n$ is an $N_e \times N_e$ matrix. The gradient of the pfaffian wavefunction, as proved in Eq. (B.38), is given by [170]

$$\frac{\partial \psi(n)}{\partial M_{ij}} = \frac{\mathrm{pf}(\mathbf{M})}{\partial M_{ij}} = \mathrm{pf}(\mathbf{M})\,(\mathbf{M}^{-1})_{ji}. \tag{7.35}$$

7.2.2 Rank-1 update

The pfaffian complexity $\mathcal{O}(N_e^3)$ can also be reduced by low-rank updates on the pfaffian wavefunction in Eq. (7.34). Consider n and n_0 only different by one fermion hopping, and $\mathbf{M} = n \star \mathbf{F} \star n$ and $\mathbf{M}_0 = n_0 \star \mathbf{F} \star n_0$ only different at the j'th row and j'th column, then

$$\begin{aligned}
\mathbf{M} - \mathbf{M}_0 &= \begin{pmatrix}
0 & \dots & 0 & -u_1 & 0 & \dots & 0 \\
\vdots & & \vdots & \vdots & \vdots & & \vdots \\
0 & \dots & 0 & -u_{j-1} & 0 & \dots & 0 \\
u_1 & \dots & u_{j-1} & 0 & u_{j+1} & \dots & u_{N_e} \\
0 & \dots & 0 & -u_{j+1} & 0 & \dots & 0 \\
\vdots & & \vdots & \vdots & \vdots & & \vdots \\
0 & \dots & 0 & -u_{N_e} & 0 & \dots & 0
\end{pmatrix} \\
&= -\begin{pmatrix}
u_1 & 0 \\
\vdots & \vdots \\
u_{j-1} & 0 \\
u_j & 1 \\
u_{j+1} & 0 \\
\vdots & \vdots \\
u_{N_e} & 0
\end{pmatrix}
\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}
\begin{pmatrix}
u_1 & \dots & u_{j-1} & u_j & u_{j+1} & \dots & u_{N_e} \\
0 & \dots & 0 & 1 & 0 & \dots & 1
\end{pmatrix} \\
&= -\mathbf{v}\mathbf{A}_1\mathbf{v}^T,
\end{aligned} \tag{7.36}$$

where $\mathbf{v} = (\mathbf{u}, \mathbf{e})$ is an $N_e \times 2$ matrix with $\mathbf{u}^T$ the non-zero row of $\mathbf{M} - \mathbf{M}_0$ and $\mathbf{e}$ the one-hot vector with $e_i = \delta_{i,j}$, and

$$\mathbf{A}_k = \begin{pmatrix} \mathbf{0} & \mathbf{1}_k \\ -\mathbf{1}_k & \mathbf{0} \end{pmatrix} \tag{7.37}$$

is a $2k \times 2k$ anti-symmetric identity matrix. A useful property of $\mathbf{A}_k$ is $\mathbf{A}_k^{-1} = -\mathbf{A}_k$. According to the formula proved in Eq. (B.33),

$$\frac{\mathrm{pf}(\mathbf{M} + \mathbf{B}\mathbf{C}\mathbf{B}^T)}{\mathrm{pf}(\mathbf{M})} = \frac{\mathrm{pf}(\mathbf{C}^{-1} + \mathbf{B}^T\mathbf{M}^{-1}\mathbf{B})}{\mathrm{pf}(\mathbf{C}^{-1})}, \tag{7.38}$$

we have

$$\begin{aligned}
\mathrm{pf}\mathbf{M} &= \mathrm{pf}(\mathbf{M}_0 - \mathbf{v}\mathbf{A}_1\mathbf{v}^T) \\
&= \frac{\mathrm{pf}\mathbf{M}_0}{\mathrm{pf}(-\mathbf{A}_1^{-1})}\mathrm{pf}(-\mathbf{A}_1^{-1} + \mathbf{v}^T\mathbf{M}_0^{-1}\mathbf{v}) \\
&= \mathrm{pf}\mathbf{M}_0\,\mathrm{pf}\mathbf{R},
\end{aligned} \tag{7.39}$$

and the ratio

$$r = \frac{\mathrm{pf}\mathbf{M}}{\mathrm{pf}\mathbf{M}_0} = \mathrm{pf}\mathbf{R}, \tag{7.40}$$

where

$$\mathbf{R} = \mathbf{A}_1 + \mathbf{v}^T\mathbf{M}_0^{-1}\mathbf{v}. \tag{7.41}$$

Therefore, pf$\mathbf{M}$ can be computed with $\mathcal{O}(N_e^2)$ time complexity if pf$\mathbf{M}_0$ and $\mathbf{M}_0^{-1}$ have been computed and stored in memory. The memory complexity is also $\mathcal{O}(N_e^2)$ for storing $\mathbf{M}_0^{-1}$.

To update $\mathbf{M}_0^{-1}$ to $\mathbf{M}^{-1}$, one needs to utilize the Woodbury matrix identity in Eq. (B.44),

$$\begin{aligned}\mathbf{M}^{-1} &= (\mathbf{M}_0 - \mathbf{v}\mathbf{A}_1\mathbf{v}^T)^{-1} \\ &= \mathbf{M}_0^{-1} - \mathbf{M}_0^{-1}\mathbf{v}(-\mathbf{A}_1^{-1} + \mathbf{v}^T\mathbf{M}_0^{-1}\mathbf{v})^{-1}\mathbf{v}^T\mathbf{M}_0^{-1} \\ &= \mathbf{M}_0^{-1} + (\mathbf{M}_0^{-1}\mathbf{v})\mathbf{R}^{-1}(\mathbf{M}_0^{-1}\mathbf{v})^T,\end{aligned} \tag{7.42}$$

where we have used $(\mathbf{M}_0^{-1})^T = -\mathbf{M}_0^{-1}$. This update of $\mathbf{M}^{-1}$ has $\mathcal{O}(N_e^2)$ complexity. For faster computation in the $k = 1$ case, Eq. (7.42) can be further simplified as

$$\begin{aligned}\mathbf{M}^{-1} &= \mathbf{M}_0^{-1} + (\mathbf{M_0^{-1}u}, \mathbf{M_0^{-1}e}) \begin{pmatrix} 0 & r \\ -r & 0 \end{pmatrix}^{-1} (\mathbf{M_0^{-1}u}, \mathbf{M_0^{-1}e})^T \\ &= \mathbf{M}_0^{-1} + \frac{1}{r}\left[(\mathbf{M_0^{-1}e})(\mathbf{M_0^{-1}u})^T - (\mathbf{M_0^{-1}u})(\mathbf{M_0^{-1}e})^T\right].\end{aligned} \tag{7.43}$$

Similar to the low-rank update of the determinant state, the time bottleneck of the pfaffian state also comes from the outer product $(\mathbf{M}_0^{-1}\mathbf{v}\mathbf{R}^{-1})(\mathbf{M}_0^{-1}\mathbf{v})^T$, and the delayed update is also helpful. At each time step t, one stores $\mathbf{a}_t = \mathbf{M}_{t-1}^{-1}\mathbf{v}_t$ and $\mathbf{R}_t^{-1}$ such that

$$\mathbf{M}_\tau^{-1} = \mathbf{M}_0^{-1} + \sum_{t=1}^{\tau} \mathbf{a}_t\mathbf{R}_t^{-1}\mathbf{a}_t^T. \tag{7.44}$$

Without explicitly constructing $\mathbf{M}_\tau^{-1}$, one can update the following quantities

$$\mathbf{a}_\tau = \left(\mathbf{M}_0^{-1} + \sum_{t=1}^{\tau-1} \mathbf{a}\mathbf{w}_t\mathbf{R}_t^{-1}\mathbf{a}_t^T\right)\mathbf{v}_\tau = \mathbf{M}_0^{-1}\mathbf{v}_t + \sum_{t=1}^{\tau-1} \mathbf{a}_t\mathbf{R}_t^{-1}(\mathbf{a}_t^T\mathbf{v}_\tau), \tag{7.45}$$

$$\mathbf{R}_\tau = \mathbf{A}_1 + \mathbf{v}_\tau^T\left(\mathbf{M}_0^{-1} + \sum_{t=1}^{\tau-1} \mathbf{a}_t\mathbf{R}_t^{-1}\mathbf{a}_t^T\right)\mathbf{v}_\tau = \mathbf{A}_1 + \mathbf{v}_\tau^T\mathbf{M}_0^{-1}\mathbf{v}_\tau + \sum_{t=1}^{\tau-1}(\mathbf{v}_\tau^T\mathbf{a}_t)\mathbf{R}_t^{-1}(\mathbf{a}_t^T\mathbf{v}_\tau). \tag{7.46}$$

7.2.3 Rank-k update

When $\mathbf{M}$ and $\mathbf{M_0}$ are different at $j_1, j_2, ..., j_k$ rows and columns, one can generalize $\mathbf{v}$ to an $N_e \times 2k$ matrix with the first k columns given by the non-zero rows of $\mathbf{M} - \mathbf{M}_0$, and the last k rows given by $\mathbf{v}_{i,k+n} = \delta_{i,j_n}$. Then we still have

$$\mathbf{M} - \mathbf{M}_0 = -\mathbf{v}\mathbf{A}_k\mathbf{v}^T, \tag{7.47}$$

and

$$\mathbf{R} = \mathbf{A}_k + \mathbf{v}^T \mathbf{M}_0^{-1} \mathbf{v}, \tag{7.48}$$

$$\mathrm{pf}\mathbf{M} = \mathrm{pf}\mathbf{M}_0 \, \mathrm{pf}\mathbf{R}, \tag{7.49}$$

$$\mathbf{M}^{-1} = \mathbf{M}_0^{-1} + \mathbf{M}_0^{-1} \mathbf{v} \mathbf{R}^{-1} (\mathbf{M}_0^{-1} \mathbf{v})^T, \tag{7.50}$$

where $\mathbf{R}$ is a $2k \times 2k$ matrix. The delayed update can be implemented similarly by storing $\mathbf{a}_t = \mathbf{M}_{t-1}^{-1} \mathbf{v}_t$ and $\mathbf{R}_t^{-1}$. The overall time complexity is $\mathcal{O}(kN_e^2 + k^3)$ or $\mathcal{O}(kN_e^2)$ in the typical $k \ll N_e$ case, and the memory complexity is still $\mathcal{O}(N_e^2)$.

7.3 Relation between determinant and pfaffian states

The mean-field pfaffian state can be viewed as a generalization of the mean-field determinant state. Therefore, any determinant state can be written as a pfaffian state, while a general pfaffian state cannot be expressed as a determinant state.

7.3.1 From determinant to pfaffian

Consider a general determinant state with orbitals $\mathbf{U}$ and wavefunctions $\psi(n) = \det(n \star \mathbf{U})$. One can construct the pfaffian matrix $\mathbf{F}$ by

$$\mathbf{F} = \mathbf{U} \mathbf{A}_N \mathbf{U}^T, \tag{7.51}$$

where $\mathbf{A}_N$ is the antisymmetric identity matrix defined in Eq. (7.37) and N is the number of sites. The matrix $\mathbf{F}$ constructed in this way has $2N \times 2N$ elements, while the matrix $\mathbf{U}$ only has $2N \times N_e$ elements, so one cannot express all possible $\mathbf{F}$ with the transformation Eq. (7.51). This verifies that the pfaffian state is a generalization of the determinant state. Combining Eq. (7.12), Eq. (7.34), Eq. (7.51), and Eq. (B.29), we have

$$\begin{aligned} \psi^{\mathrm{pf}}(n) &= \mathrm{pf}(n \star (\mathbf{U} \mathbf{A}_N \mathbf{U}^T) \star n) \\ &= \mathrm{pf}((n \star \mathbf{U}) \mathbf{A}_N (n \star \mathbf{U})^T) \\ &= \det(n \star \mathbf{U}) \mathrm{pf}(\mathbf{A}_N) \\ &= \det(n \star \mathbf{U}) \\ &= \psi^{\det}(n), \end{aligned} \tag{7.52}$$

where we have utilized $\mathrm{pf}(\mathbf{A}_N) = 1$. Therefore, the determinant state with $\mathbf{U}$ and the pfaffian state with $\mathbf{F} = \mathbf{U} \mathbf{A}_N \mathbf{U}^T$ are equivalent.

7.3.2 From pfaffian to determinant

According to the spectral theory of skew-symmetric matrices, one can decompose a pfaffian pairing matrix $\mathbf{F}$ into

$$\mathbf{F} = \mathbf{U} \mathbf{D} \mathbf{U}^T, \tag{7.53}$$

where $\mathbf{U}$ is a unitary matrix and $\mathbf{D}$ is a real block-diagonal matrix given by

$$\mathbf{D} = \begin{pmatrix} 0 & f_1 & & & & & \\ -f_1 & 0 & & & & & \\ & & 0 & f_2 & & & \\ & & -f_2 & 0 & & & \\ & & & & \ddots & & \\ & & & & & 0 & f_N \\ & & & & & -f_N & 0 \end{pmatrix}. \tag{7.54}$$

Denoting $\hat{\mathbf{c}}^\dagger = (\hat{c}_1^\dagger, \hat{c}_2^\dagger, ..., \hat{c}_{2N}^\dagger)$, one can express the paired creation operators in Eq. (7.31) as

$$\frac{1}{2}\sum_{i,j=1}^{2N} F_{i,j}\hat{c}_i^\dagger \hat{c}_j^\dagger = \frac{1}{2}\hat{\mathbf{c}}^\dagger \mathbf{F}(\hat{\mathbf{c}}^\dagger)^T = \frac{1}{2}\hat{\mathbf{c}}^\dagger \mathbf{U}\mathbf{D}\mathbf{U}^T(\hat{\mathbf{c}}^\dagger)^T = \frac{1}{2}\hat{\boldsymbol{\gamma}}^\dagger \mathbf{D}(\hat{\boldsymbol{\gamma}}^\dagger)^T = \sum_{\alpha=1}^{N} f_\alpha \hat{\gamma}_{2\alpha-1}^\dagger \hat{\gamma}_{2\alpha}^\dagger, \tag{7.55}$$

where $\hat{\boldsymbol{\gamma}}^\dagger = \hat{\mathbf{c}}^\dagger \mathbf{U}$ is the single-particle orbital similar to the one defined in the determinant state as shown in Eq. (7.7). Then the pfaffian state can also be written as

$$|\Psi\rangle = \frac{1}{(N_e/2)!}\left(\sum_{\alpha=1}^{N} f_\alpha \hat{\gamma}_{2\alpha-1}^\dagger \hat{\gamma}_{2\alpha}^\dagger\right)^{N_e/2} |0\rangle\,. \tag{7.56}$$

This shows that the pfaffian wavefunction cannot be reformulated into independent quasi-particles as in determinant wavefunctions but can be viewed as dimerized quasi-particles.

It is not possible to express a general pfaffian state, either Eq. (7.31) or Eq. (7.56), as a determinant state. However, some specific pfaffian states with suitable constraints can be written into determinants. A famous example is the Bardeen–Cooper–Schrieffer (BCS) state constructed from the mean-field Hamiltonian

$$\hat{\mathcal{H}} = \sum_{i,j,\sigma} t_{i,j}\hat{c}_{i,\sigma}^\dagger \hat{c}_{j,\sigma} + \sum_{i,j} \Delta_{i,j}\hat{c}_{i,\uparrow}^\dagger \hat{c}_{j,\downarrow}^\dagger + h.c. \tag{7.57}$$

At first sight, it is a pfaffian mean-field Hamiltonian in Eq. (7.26) instead of a determinant one in Eq. (7.3). However, under a particle-hole transformation on spin-down electrons,

$$\begin{aligned} \hat{d}_{i,\uparrow} &= \hat{c}_{i,\uparrow}, \\ \hat{d}_{i,\downarrow} &= \hat{c}_{i,\downarrow}^\dagger, \end{aligned} \tag{7.58}$$

the BCS mean-field Hamiltonian becomes

$$\hat{\mathcal{H}} = \sum_{i,j} t_{i,j}(\hat{d}_{i,\uparrow}^\dagger \hat{d}_{j,\uparrow} - \hat{d}_{i,\downarrow}^\dagger \hat{d}_{j,\downarrow}) + \sum_{i,j} \Delta_{i,j}\hat{d}_{i,\uparrow}^\dagger \hat{d}_{j,\downarrow} + h.c. \tag{7.59}$$

Then it can be formulated into a determinant mean-field Hamiltonian in Eq. (7.3).

The same conclusion can be obtained if we look at the wavefunction. A general pfaffian matrix for spinful fermions can be written as

$$\mathbf{F} = \begin{pmatrix} \mathbf{F}_{\uparrow\uparrow} & \mathbf{F}_{\uparrow\downarrow} \\ -\mathbf{F}_{\uparrow\downarrow}^T & \mathbf{F}_{\downarrow\downarrow} \end{pmatrix}, \tag{7.60}$$

while the pfaffian matrix of a BCS state is

$$\mathbf{F} = \begin{pmatrix} \mathbf{0} & \mathbf{F}_{\uparrow\downarrow} \\ -\mathbf{F}_{\uparrow\downarrow}^T & \mathbf{0} \end{pmatrix}, \tag{7.61}$$

because the pairing only happens between different spins. Therefore, the wavefunction is

$$\begin{aligned} \psi(n) &= \mathrm{pf}(n \star \mathbf{F} \star n) \\ &= \mathrm{pf} \begin{pmatrix} \mathbf{0} & n_\uparrow \star \mathbf{F}_{\uparrow\downarrow} \star n_\downarrow \\ -(n_\uparrow \star \mathbf{F}_{\uparrow\downarrow} \star n_\downarrow)^T & \mathbf{0} \end{pmatrix} \\ &= (-1)^m \det(n_\uparrow \star \mathbf{F}_{\uparrow\downarrow} \star n_\downarrow), \end{aligned} \tag{7.62}$$

where $m = N_e(N_e/2-1)/4$, and $(-1)^m$ only contributes to a constant sign to the wavefunction. In conclusion, the BCS wavefunction as a special pfaffian state can be computed by determinants [168].

Chapter 8

Fermionic neural quantum state

Given the outstanding performance of NQSs in spin systems, it is natural to also consider its application in fermion systems. Nevertheless, we will explain in this chapter that NQSs cannot be directly applied to fermions due to the fermion sign structure. To utilize the power of ANNs in fermion systems, we will discuss how to combine them with mean-field fermionic wavefunctions in Chapter 7.

8.1 What's the difficulty?

When NQS is applied to spin systems as discussed in Chapter 5 and 6, one takes a basis state $|s\rangle$ as the input and uses a neural network to directly generate a wavefunction $\psi(s)$. In fermionic systems, a natural idea is to similarly employ a neural network to construct a map from a Fock state $|n\rangle$ to a fermionic wavefunction $\psi(n)$ under second quantization. Although this method is in principle feasible, it encounters serious difficulties in practice.

Consider a simple example of free fermions with Hamiltonian

$$\hat{\mathcal{H}}_0 = -t \sum_{\langle i,j\rangle,\sigma} \hat{c}^\dagger_{i,\sigma} \hat{c}_{j,\sigma}, \tag{8.1}$$

where $\langle i, j\rangle$ represents nearest neighbors. To find a ground state of $\hat{\mathcal{H}}_0$ variationally, one

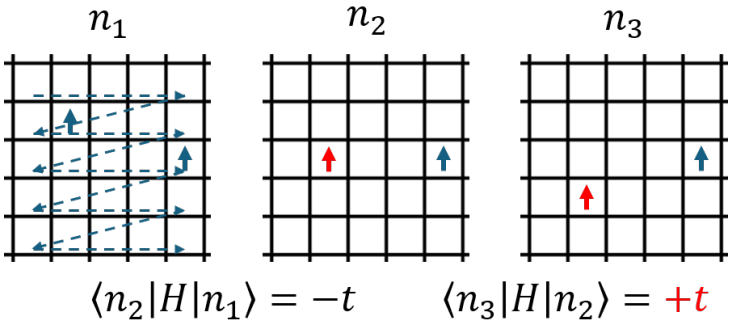


Figure 8.1: Difficulty of fermionic NQS. The fermion ordering is shown by the dashed arrows in n_1.

needs to minimize the variational energy

$$E = \langle\Psi|\hat{\mathcal{H}}_0|\Psi\rangle = \sum_{n,n'} \psi^*(n)\,\langle n|\hat{\mathcal{H}}_0|n'\rangle\,\psi(n'), \tag{8.2}$$

where we assume the variational wavefunction is normalized. If the wavefunction of a Fock state $|n_1\rangle$ is $\psi(n_1) > 0$ and another Fock state n_2 generated by a single particle hopping of n_1 as shown in Fig. 8.1 has $\langle n_1|\hat{\mathcal{H}}_0|n_2\rangle = -t$, then one should let $\psi(n_2) > 0$ for the minimization of variational energy in Eq. (8.2). The $|n_3\rangle$ generated by a subsequent hopping of $|n_2\rangle$, however, has $\langle n_2|\hat{\mathcal{H}}_0|n_3\rangle = +t$ due to the additional minus sign in the fermion exchange, and hence one should let $\psi(n_3) < 0$. Under this second quantization situation, the signs of $\psi(n_2)$ and $\psi(n_3)$ are different due to the existence of another non-interacting electron.

In large 2D and 3D systems, the additional fermion signs become a severe issue for NQS simulations. For any local hopping or interaction, far-apart fermions not involved in the interaction can lead to sign changes in the Hamiltonian matrix elements, which is difficult for a usual ANN to adapt. Therefore, although the direct ANN map from n to $\psi(n)$ has also been attempted [107, 171, 172], major fermionic NQSs mostly utilize mean-field wavefunctions to account for fermion signs as we will discuss in the following sections.

8.2 Neural Jastrow wavefunction

8.2.1 Jastrow factor

We have discussed the mean-field determinant and pfaffian states. A direct approach to obtaining wavefunctions beyond the mean field is by multiplying the Jastrow factor defined as

$$J(n) = \exp\left(\sum_{i<j} n_i W_{i,j} n_j\right), \tag{8.3}$$

where $W_{i,j}$ are trainable parameters. Therefore, the wavefunction gets an additional factor $W_{i,j}$ for the correlation of particles at i and j. The Jastrow factor is combined with the mean-field wavefunction $\psi^{\mathrm{mf}}(n)$ to construct the full variational wavefunction

$$\psi(n) = J(n)\,\psi^{\mathrm{mf}}(n). \tag{8.4}$$

One might use the determinant or pfaffian state as the mean-field wavefunction, so the full wavefunction can be

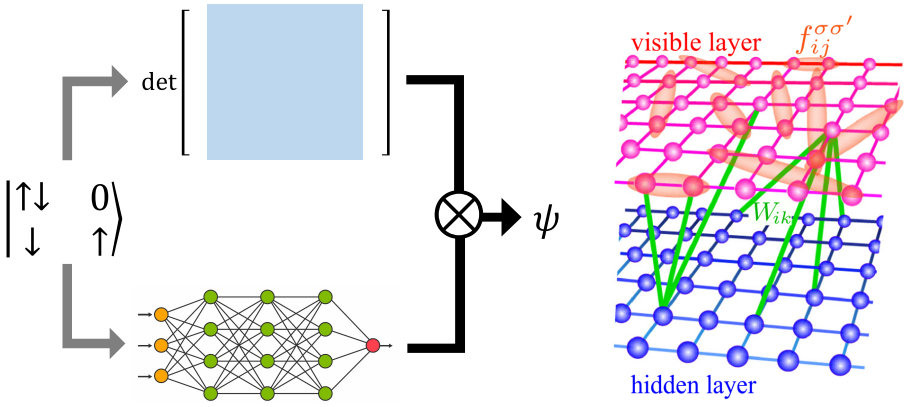
$$\psi(n) = J(n)\,\det(n \star \mathbf{U}), \tag{8.5}$$

or

$$\psi(n) = J(n)\,\mathrm{pf}(n \star \mathbf{F} \star n). \tag{8.6}$$

8.2.2 Neural Jastrow

The Jastrow factor is a direct way to encode correlations into wavefunctions, but it is usually too simple to express strong correlations. Alternatively, one can use neural



(a) Neural Jastrow (b) RBM+PP

Figure 8.2: Neural Jastrow wavefunction. Panel (b) is taken from Ref. [173].

networks to replace the simple Jastrow factor $J(n)$, as shown in Fig. 8.2(a). A direct benefit of this form is that the low-rank update can still be applied as

$$\psi(n') = J(n')\, r\psi^{\mathrm{mf}}(n), \tag{8.7}$$

where $r = \psi^{\mathrm{mf}}(n')/\psi^{\mathrm{mf}}(n)$ is the ratio of mean-field wavefunctions in Eq. (7.16) and Eq. (7.40). Therefore, the complexity of the mean-field part can still be reduced from $\mathcal{O}(N_e^3)$ to $\mathcal{O}(N_e^2)$ in the neural Jastrow wavefunction.

One popular neural Jastrow wavefunction is the RBM+PP wavefunction [173], which is given by a combination of the RBM Jastrow factor and the pfaffian mean-field state. To utilize RBM in the fermion case, one needs to rewrite fermion degrees of freedom $(n_{i,\uparrow}, n_{i,\downarrow})$ into the spin form as $s = (2n_{i,\uparrow} - 1, 2n_{i,\downarrow} - 1)$ so that each input element is ± 1. The RBM+PP formula combining Eq. (5.12) and Eq. (7.34) is given by

$$\psi(n) = \prod_{i=1}^{M} \cosh\left(\sum_{j=1}^{2N} W_{i,j}(2n_j - 1) + b_i\right) \mathrm{pf}(n \star \mathbf{F} \star n). \tag{8.8}$$

As shown in Fig. 8.2(b), $W_{i,j}$ encodes the correlation between the visible layer and the hidden layer, and $\mathbf{F}$ encodes the two-body correlation within the visible particles.

The neural Jastrow wavefunction can also be employed in spin systems by applying the following Gutzwiller projection [153, 173]

$$|\Psi^{\mathrm{spin}}\rangle = \hat{\mathcal{P}}_G^{\infty} |\Psi\rangle = \prod_{i=1}^{N} (1 - n_{i,\uparrow} n_{i,\downarrow}) |\Psi\rangle\,, \tag{8.9}$$

which projects the system to spin degrees of freedom by removing doubly occupied states, as shown in Fig. 8.3. The performance of Gutzwiller-projected fermionic wavefunction (GWF) combined with NQS has been shown in Sec. 6.2.

8.3 Neural network backflow

The neural Jastrow factor can be viewed as an envelope on top of mean-field wavefunctions. Although it helps to encode correlations, the neural Jastrow method cannot fully

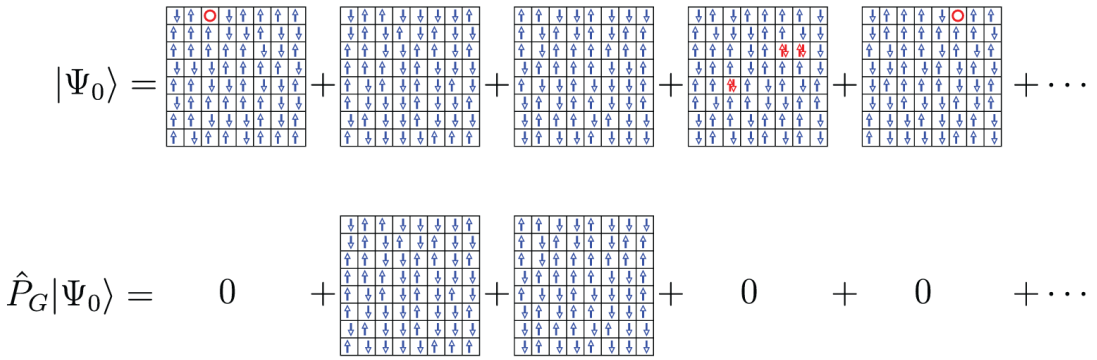


Figure 8.3: Gutzwiller projection of a fermion state (taken from Ref. [136])

change the distribution given by mean-field wavefunctions, and hence its expressive power is limited. The neural network backflow method can be used to solve this problem [174].

8.3.1 Backflow correlation

The mean-field wavefunctions are constructed from non-interacting particles moving freely in space. In real quantum systems, however, electrons interact with each other and generate a "backflow" that pushes other electrons away from free particle positions.

In the first quantization formalism, one can write the backflow correlation as

$$\psi(\mathbf{r}) = \psi^{\mathrm{mf}}(\mathbf{r}^b), \tag{8.10}$$

where ψ^{mf} is the mean-field wavefunction, and $\mathbf{r}^b$ is the coordinate after backflow transformation. A common backflow coordinate is given by

$$\mathbf{r}_i^b = \mathbf{r}_i + \sum_j \eta_{i,j}(\mathbf{r})(\mathbf{r}_j - \mathbf{r}_i), \tag{8.11}$$

where η is a suitable function describing the backflow correlation.

With second quantization and discrete coordinates, one cannot perform continuous displacement on electron positions. Alternatively, the backflow can be imposed directly on mean-field orbitals as

$$\psi(n) = \det(n \star \mathbf{U}^b(n)), \tag{8.12}$$

where the backflow orbital is

$$\mathbf{U}^b(n) = \mathbf{U} + \boldsymbol{\eta}(n)\mathbf{U}, \tag{8.13}$$

and $\boldsymbol{\eta}(n)$ is the backflow transformation on the mean-field orbital $\mathbf{U}$. Equivalently,

$$U^b_{i,\alpha}(n) = U_{i,\alpha} + \sum_j \eta_{i,j}(n) U_{j,\alpha}. \tag{8.14}$$

A common choice of $\eta_{i,j}$ is

$$\eta_{i,j}(n) = \begin{cases} D_i H_j \theta(|\mathbf{r}_i - \mathbf{r}_j|), & \sigma_i = \sigma_j, \\ 0, & \sigma_i \neq \sigma_j, \end{cases} \tag{8.15}$$

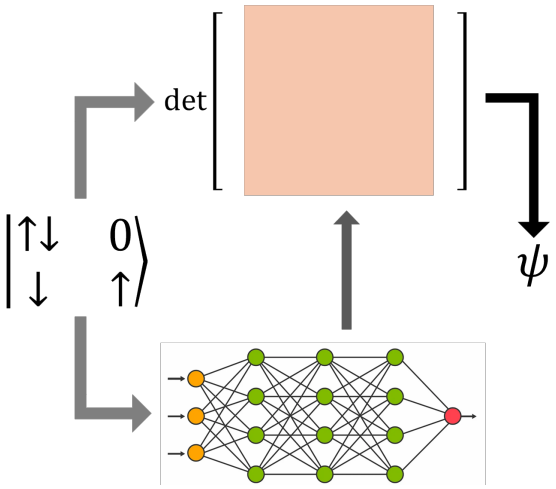


Figure 8.4: Neural network backflow (NNBF)

where $D_i = n_{i,\uparrow}n_{i,\downarrow}$, $H_j = (1 - n_{j,\uparrow})(1 - n_{j,\downarrow})$, and $\{\theta(|\mathbf{r}_i - \mathbf{r}_j|)\}$ is a set of variational parameters. One can also use a big tensor for $\boldsymbol{\eta}$ to gain more expressive power. In this case,

$$U^b_{i,\alpha}(n) = U_{i,\alpha} + \sum_j \eta_{i,j,\alpha,n_i,n_j} U_{j,\alpha}. \tag{8.16}$$

8.3.2 Neural network backflow

The neural network backflow (NNBF) [174] instead uses neural networks to generate the backflow transformation, as shown in Fig. 8.4. The NNBF is usually combined with a neural Jastrow factor to enhance its expressive power, i.e.,

$$\psi(n) = J(n) \det(n \star \mathbf{U}^b(n)), \tag{8.17}$$

where $\mathbf{U}^b(n)$ is the backflow orbital generated by the neural network, either as a shift on mean-field orbitals

$$\mathbf{U}^b(n) = \mathbf{U} + \mathbf{U}^{\mathrm{NN}}(n), \tag{8.18}$$

or as an element-wise product

$$\mathbf{U}^b(n) = \mathbf{U} \circ \mathbf{U}^{\mathrm{NN}}(n). \tag{8.19}$$

Since the full mean-field orbital is transformed by a neural network in NNBF, the low-rank update is usually not applicable. To obtain the NNBF wavefunction, one has to compute the full determinant with $\mathcal{O}(N_e^3)$ complexity instead of $\mathcal{O}(N_e^2)$. Therefore, the computation of NNBF is restricted to small systems in most cases.

8.3.3 NNBF with CNN

The backflow $\mathbf{U}^{\mathrm{NN}}(n)$ generated by the neural network should have the same shape $2N \times N_e$ as the mean-field orbital $\mathbf{U}$. One is free to use an arbitrary neural network to take n as the input and generate a $2N \times N_e$ matrix as the output. For instance, in the

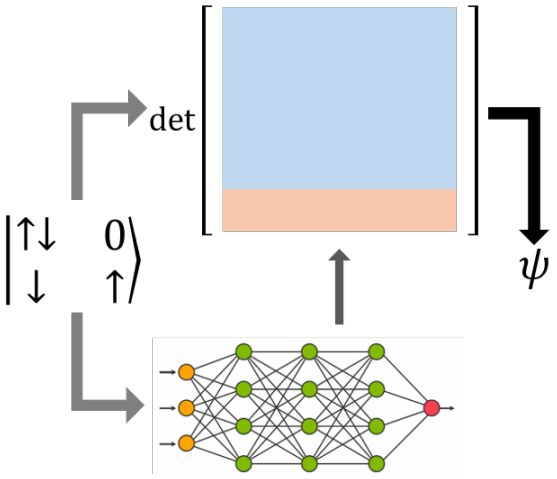


Figure 8.5: Hidden fermion determinant state (HFDS). The blue part of the determinant is given by mean-field parameters, and the orange part is given by the neural network.

first NNBF practice the multi-layer perception introduced in Sec. 5.1 is used. To encode lattice geometry, the CNN architecture introduced in Sec. 5.3 can be utilized for better accuracy. The matrix shape produced by CNN is $C \times N$, so one can choose $C = 2N_e$ to generate $\mathbf{U}^{\mathrm{NN}}(n)$ after suitable reshape and transpose. An advantage of this choice is that when one performs a translation $|n'\rangle = \hat{T}\,|n\rangle$,

$$n' \star \mathbf{U}^{\mathrm{NN}}(n') = n \star \mathbf{U}^{\mathrm{NN}}(n) \tag{8.20}$$

because the slicing vector n' and the backflow orbitals $\mathbf{U}^{\mathrm{NN}}(n')$ are both translated. Therefore, this choice naturally encodes the translation invariance into the wavefunction [175].

8.4 Hidden fermion determinant state

The hidden fermion method is another approach to encoding strong correlations into mean-field states [176]. For any Fock state $|n\rangle$ with N sites and N_e electrons, the hidden fermion method extends it to $|n, n'\rangle$ with additional N_h hidden electrons. In the original Ref. [176], n' depends implicitly on n. In this thesis, however, we interpret the hidden part as additional N_h electrons and N_h fully occupied orbitals, which means n' is simply $(1, ..., 1)$. The two interpretations are equivalent in HFDS.

In this extended space, the quasi-particle in Eq. (7.7) should be modified to

$$\hat{\gamma}_\alpha^\dagger = \sum_{i=1}^{2N} V_{i,\alpha}\,\hat{c}_i^\dagger + \sum_{j=1}^{N_h} H_{j,\alpha}\hat{d}_j^\dagger, \tag{8.21}$$

where the $2N \times (N_e + N_h)$ matrix $\mathbf{V}$ and the $N_h \times (N_e + N_h)$ matrix $\mathbf{H}$ are orbitals of visible and hidden fermions, and $\hat{c}_i^\dagger$ and $\hat{d}_j^\dagger$ are the fermion operators of visible and hidden fermions. The fermionic mean-field wavefunction in Eq. (7.8) in this extended

space becomes

$$|\Psi\rangle = \prod_{\alpha=1}^{N_e+N_h} \hat{\gamma}_\alpha^\dagger |0\rangle = \prod_{\alpha=1}^{N_e+N_h} \left(\sum_{i=1}^{2N} V_{i,\alpha}\, \hat{c}_i^\dagger + \sum_{j=1}^{N_h} H_{j,\alpha} \hat{d}_j^\dagger \right) |0\rangle \,. \tag{8.22}$$

For an input Fock state $|n\rangle$, we define its wavefunction as

$$\psi(n) = \langle n, n' | \Psi \rangle = \det \left((n, n') \star \begin{pmatrix} \mathbf{V} \\ \mathbf{H} \end{pmatrix} \right) = \det \begin{pmatrix} n \star \mathbf{V} \\ \mathbf{H} \end{pmatrix}, \tag{8.23}$$

where we have utilized our interpretation that n' is fully occupied. In HFDS, we assume $\mathbf{V}$ is the mean-field orbital independent of n, while $\mathbf{H}$ is generated by a neural network like the case of NNBF, so

$$\psi(n) = \det \begin{pmatrix} n \star \mathbf{V} \\ \mathbf{H}_\theta(n) \end{pmatrix}, \tag{8.24}$$

where the mean-field orbital $\mathbf{V}$ and the neural network parameter $\boldsymbol{\theta}$ are trainable parameters. The HFDS can also be understood as backflow acting only on hidden fermions.

The low-rank update can be partially utilized on the $n \star \mathbf{V}$ part of HFDS for accelerated updates. The rank-k update becomes rank-$(k + N_h)$ in HFDS, so the computational cost, as discussed in Sec. 7.1, is $\mathcal{O}((k + N_h)(N_e + N_h)^2 + (k + N_h)^3)$. Typically, we have $N_e > N_h \gg k$, and the computational complexity of HFDS becomes $\mathcal{O}(N_e^2 N_h)$.

8.4.1 HFDS as configuration interaction

Configuration interaction (CI) is a popular method in quantum chemistry expressing correlation by a superposition of multiple mean-field states. CI considers N_{CI} quasi-particles $\hat{\gamma}_\alpha^\dagger$ with $N_{\mathrm{CI}} > N_e$, and each mean-field state $|\Psi_I\rangle$ is constructed by selecting N_e quasi-particles from the total number N_{CI}. The variational wavefunction is given by the superposition of these mean-field states as

$$|\Psi^{\mathrm{CI}}\rangle = \sum_I c_I \, |\Psi_I\rangle \,, \tag{8.25}$$

where I iterates over all possible combinations of N_e quasi-particles.

The HFDS can be directly interpreted as a CI variational state. To show this, we first utilize the following relation

$$\psi(n) = \langle n, n' | \Psi \rangle = \langle n | \prod_{j=N_h}^{1} \hat{d}_j | \Psi \rangle \,. \tag{8.26}$$

Therefore, the HFDS is

$$|\Psi^{\mathrm{HFDS}}\rangle = \prod_{j=N_h}^{1} \hat{d}_j \, |\Psi\rangle = \prod_{j=N_h}^{1} \hat{d}_j \prod_{\alpha=1}^{N_e+N_h} \left(\sum_{i=1}^{2N} V_{i,\alpha}\, \hat{c}_i^\dagger + \sum_{j=1}^{N_h} H_{j,\alpha} \hat{d}_j^\dagger \right) |0\rangle \,, \tag{8.27}$$

where we have utilized Eq. (8.22). Consider the simplest case with only one hidden particle $N_h = 1$. The HFDS becomes

$$|\Psi^{\mathrm{HFDS}}\rangle = \hat{d}_1 \prod_{\alpha=1}^{N_e+1} \left(\sum_{i=1}^{2N} V_{i,\alpha}\, \hat{c}_i^\dagger + H_{1,\alpha} \hat{d}_1^\dagger \right) |0\rangle = \sum_{\alpha=1}^{N_e+1} H_{1,\alpha} \sum_{\substack{i=1 \\ i\neq\alpha}}^{2N} V_{i,\alpha} \hat{c}_i^\dagger |0\rangle = \sum_{\alpha=1}^{N_e+1} H_{1,\alpha} |\Psi_\alpha\rangle , \tag{8.28}$$

where $|\Psi_\alpha\rangle = \sum_{i\neq\alpha} V_{i,\alpha} \hat{c}_i^\dagger |0\rangle$ is a mean-field state. Comparing Eq. (8.25) and Eq. (8.28), one can directly see that $H_{1,\alpha}$ is the superposition coefficient c_I and $|\Psi_\alpha\rangle$ is the mean-field state $|\Psi_I\rangle$. For $N_h > 1$, it can also be similarly verified that $|\Psi^{\mathrm{HFDS}}\rangle$ still express a CI state with $N_{\mathrm{CI}} = N_e + N_h$.

8.4.2 HFDS as other states

It is also common to have $\mathbf{V} = (\mathbf{V}_1, \mathbf{V}_2)$ where $\mathbf{V}_1$ is a $2N \times N_e$ matrix and $\mathbf{V}_2$ is a $2N \times N_h$ matrix, and similarly $\mathbf{H} = (\mathbf{H}_1, \mathbf{H}_2)$. The wavefunction in Eq. (8.24) is therefore expressed as

$$\psi(n) = \det \begin{pmatrix} n \star \mathbf{V}_1 & n \star \mathbf{V}_2 \\ \mathbf{H}_1(n) & \mathbf{H}_2(n) \end{pmatrix}. \tag{8.29}$$

The HFDS can be used to express other variational states as listed below.

1. Mean-field fermion: Let $\mathbf{V}_2 = \mathbf{0}$, $\mathbf{H}_1 = \mathbf{0}$, and $\mathbf{H}_2 = \mathbf{1}$, then

$$\psi(n) = \det \begin{pmatrix} n \star \mathbf{V}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} = \det(n \star \mathbf{V}_1). \tag{8.30}$$

2. Neural network Jastrow factor: Let $\mathbf{V}_2 = \mathbf{0}$ and $\mathbf{H}_1 = \mathbf{0}$, then

$$\psi(n) = \det \begin{pmatrix} n \star \mathbf{V}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_2(n) \end{pmatrix} = \det \mathbf{H}_2(n)\ \det(n \star \mathbf{V}_1) = J(n) \det(n \star \mathbf{V}_1), \tag{8.31}$$

   where $J(n) = \det \mathbf{H}_2(n)$ is the Jastrow factor.

3. Neural network backflow: The block formula of determinant in Eq. (B.12) gives

$$\begin{aligned} \psi(n) &= \det \mathbf{H}_2(n)\ \det[(n \star \mathbf{V}_1) - (n \star \mathbf{V_2}) \mathbf{H}_2^{-1}(n) \mathbf{H}_1(n)] \\ &= J(n) \det(n \star \mathbf{U}^b(n)), \end{aligned} \tag{8.32}$$

   where $J(n) = \det \mathbf{H}_2(n)$ is the Jastrow factor, and $\mathbf{U}^b(n) = \mathbf{V}_1 - \mathbf{V}_2 \mathbf{H}_2^{-1}(n) \mathbf{H}_1(n)$ is the backflow-transformed orbital [177].

8.5 Hidden fermion pfaffian state

In previous sections, we reviewed the main design choices of fermionic NQS, including neural Jastrow, NNBF, and HFDS. However, these methods have respective problems in numerical simulations.

1. Neural Jastrow cannot change the nodal structure of mean-field wavefunctions, and hence does not exhibit enough expressive power in numerical experiments.

2. The computational complexity of NNBF is $\mathcal{O}(N_e^3)$ because low-rank updates of mean-field wavefunctions cannot be utilized, which makes it hard to simulate large systems.

3. HFDS requires ANNs to provide the hidden orbital $\mathbf{H}$ with a shape $N_h \times (N_e + N_h)$, which is not compatible with the usual CNN or transformer output shape $C \times N$.

4. NNBF and HFDS are based on the mean-field determinant instead of the more general mean-field pfaffian.

We propose the hidden fermion pfaffian state (HFPS) to solve these problems. HFPS employs the idea of hidden fermions in HFDS and generalizes it to pfaffian wavefunctions. Starting from the mean-field pfaffian state in Eq. (7.31), we generalize it to the case with hidden fermions as

$$|\Psi\rangle = \frac{1}{((N_e+N_h)/2)!}\left(\sum_{i<j} F_{i,j} c_i^\dagger c_j^\dagger + \sum_{k<l} \tilde{F}_{k,l} d_k^\dagger d_l^\dagger + \sum_{i,k} H_{i,k} c_i^\dagger d_k^\dagger\right)^{(N_e+N_h)/2} |0\rangle\,, \quad (8.33)$$

where $c^\dagger$ and $d^\dagger$ are fermion operators on visible and hidden fermions. Similar to Eq. (8.23), the wavefunction of HFPS is defined as

$$\psi(n) = \langle n, n'|\Psi\rangle = \mathrm{pf}\left[(n,n') \star \begin{pmatrix} \mathbf{F} & \mathbf{H} \\ -\mathbf{H}^T & \tilde{\mathbf{F}} \end{pmatrix} \star (n,n')\right] = \mathrm{pf}\begin{pmatrix} n\star\mathbf{F}\star n & n\star\mathbf{H} \\ -(n\star\mathbf{H})^T & \tilde{\mathbf{F}} \end{pmatrix}, \quad (8.34)$$

where we also utilized our interpretation that n' is fully occupied. We assume $\mathbf{F}$ and $\tilde{\mathbf{F}}$ are given by directly tunable mean-field parameters, and $\mathbf{H}$ is given by a neural network. A neural Jastrow factor is also added to enhance the expressive power. Therefore, the full HFPS wavefunction can be written as

$$\psi(n) = J_\theta(n)\,\mathrm{pf}\begin{pmatrix} n\star\mathbf{F}\star n & n\star\mathbf{H}_\theta(n) \\ -(n\star\mathbf{H}_\theta(n))^T & \tilde{\mathbf{F}} \end{pmatrix}, \quad (8.35)$$

where $\mathbf{F}$, $\tilde{\mathbf{F}}$, and $\boldsymbol{\theta}$ are trainable parameters. This structure is illustrated in Fig. 8.6. One can understand HFPS as a combination of a visible system $\mathbf{F}$ and a hidden system $\tilde{\mathbf{F}}$ with their correlation encoded by neural network outputs $\mathbf{H}$.

The previous problems in neural Jastrow, NNBF, and HFDS can be fixed by HFPS.

1. HFPS can change mean-field wavefunctions directly similar to NNBF and HFDS.

2. The low-rank update can be applied in HFPS similar to HFDS. It is a rank-$(k+N_h)$ update typically with $N_e > N_h \gg k$, and the complexity is $\mathcal{O}(N_e^2 N_h)$ instead of $\mathcal{O}(N_e^3)$ in NNBF.

3. HFPS requires the hidden orbital $\mathbf{H}$ with shape $2N \times N_h$, where the factor 2 comes from spins. One can utilize a popular architecture like CNN or transformer whose output shape is $C \times N$ with $C = 2N_h$, and perform a transpose and reshape it into the required shape of $\mathbf{H}$.

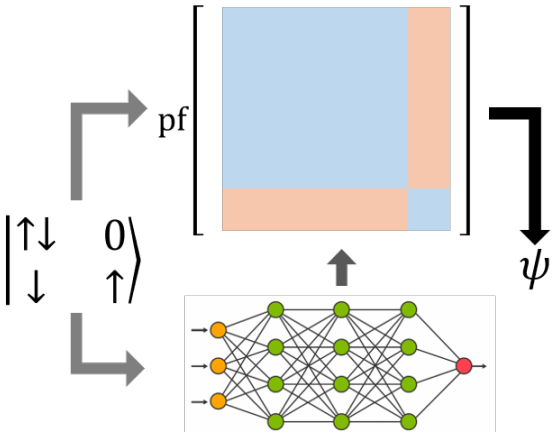


Figure 8.6: Hidden fermion pfaffian state (HFPS). The blue part of the pfaffian is given by mean-field parameters, and the orange part is given by the neural network.

4. HFPS directly utilizes the pfaffian wavefunction as a generalization of the determinant.

The HFPS can also be viewed as a low-rank backflow. According to the block pfaffian in Eq. (B.36), one can rewrite Eq. (8.35) into

$$\psi(n) = \mathrm{pf}(\tilde{\mathbf{F}})\mathrm{pf}[n \star (\mathbf{F} + \mathbf{H}(n)\tilde{\mathbf{F}}^{-1}\mathbf{H}^T(n)) \star n], \tag{8.36}$$

where the red part is essentially a backflow correction on the mean-field pfaffian matrix $\mathbf{F}$. The rank of the correction matrix $\mathbf{H}(n)\tilde{\mathbf{F}}^{-1}\mathbf{H}^T(n)$ is only N_h, so it becomes a low-rank backflow whose rank is controlled by the number of hidden fermions, which explains why the fast low-rank update can be applied in HFPS.

As discussed in Sec. 7.3, the pfaffian is a generalization of the determinant wavefunction. Similarly, HFPS is also a generalization of HFDS. Combining the HFDS wavefunction in Eq. (8.23) and converting it to a pfaffian according to Eq. (7.52), one obtains

$$\psi^{\mathrm{HFDS}}(n) = \mathrm{pf}\left[\begin{pmatrix} n \star \mathbf{V} \\ \mathbf{H}(n) \end{pmatrix} \mathbf{A}(\mathbf{V}^T \star n, \mathbf{H}^T(n))\right] = \mathrm{pf}\begin{pmatrix} n \star \mathbf{V}\mathbf{A}\mathbf{V}^T \star n & n \star \mathbf{V}\mathbf{A}\mathbf{H}^T(n) \\ \mathbf{H}(n)\mathbf{A}\mathbf{V}^T \star n & \mathbf{H}(n)\mathbf{A}\mathbf{H}^T(n) \end{pmatrix}. \tag{8.37}$$

Compared with Eq. (8.35), a map from HFDS to HFPS can be achieved by $\mathbf{V}\mathbf{A}\mathbf{V}^T \to \mathbf{F}$, $\mathbf{V}\mathbf{A}\mathbf{H}^T(n) \to \mathbf{H}(n)$, and $\mathbf{H}(n)\mathbf{A}\mathbf{H}^T(n) \to \tilde{\mathbf{F}}$. Although $\tilde{\mathbf{F}}$ is mean-field orbitals independent of n, the dependence can be encoded indirectly in the Jastrow factor $J(n)$.

Chapter 9

Neural quantum states for Fermi-Hubbard models

In the previous chapter, we introduced several mainstream fermionic NQS architectures for studying fermion systems. It is a natural next step to apply these NQSs for solving fermion models important for theoretical and experimental research, one of which is the famous Fermi-Hubbard model. In this chapter, we will introduce the motivation of the Fermi-Hubbard model, present NQS benchmarks, and then provide recent NQS progress in solving these models.

9.1 Fermi-Hubbard model

The Fermi-Hubbard model is one of the most fundamental and widely studied models in condensed matter physics. It provides a simplified yet powerful framework for understanding strongly correlated electron systems, which are at the heart of many intriguing phenomena in quantum materials, especially high-temperature superconductivity. Despite its conceptual simplicity, the model exhibits a rich phase diagram and poses significant challenges for both analytical and numerical approaches. This has made it a cornerstone of modern theoretical and computational physics, as well as a testing ground for new methodologies in quantum many-body theory.

The Hubbard model was introduced in the 1960s by John Hubbard [179], Martin Gutzwiller [180], and Junjiro Kanamori [181] as a minimal model to describe the behavior of electrons in narrow energy bands, such as those found in transition metal oxides. At its core, the model captures the competition between two key physical processes, the kinetic energy of electrons, which favors delocalization, and the Coulomb repulsion between electrons, which favors localization. This interplay gives rise to a wealth of emergent phenomena that cannot be understood by considering either effect in isolation. The Hamiltonian of the model is given by

$$\hat{\mathcal{H}} = -t \sum_{\langle i,j\rangle,\sigma} (\hat{c}^{\dagger}_{i,\sigma}\hat{c}_{j,\sigma} + h.c.) + U \sum_{i} \hat{n}_{i,\uparrow}\hat{n}_{i,\downarrow}, \tag{9.1}$$

where t is the hopping amplitude between neighboring sites, and U is the on-site interaction energy. The first term represents the kinetic energy of the fermions, while the second

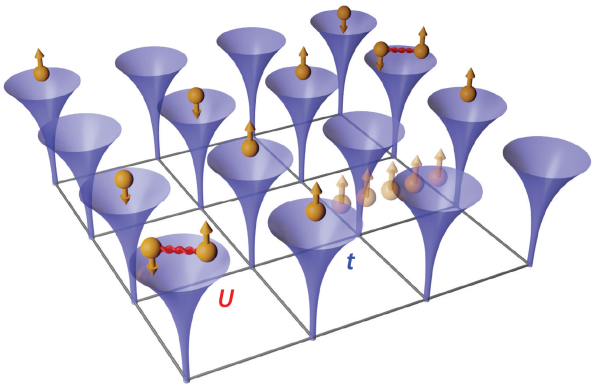
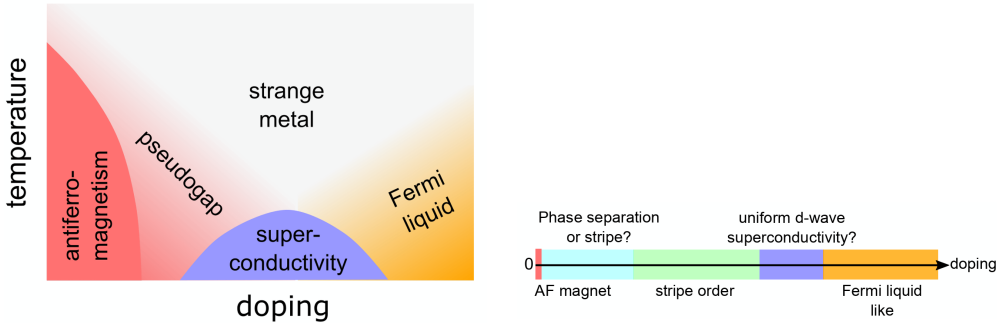


Figure 9.1: Illustration of Fermi-Hubbard model (taken from Ref. [178]).

term describes the repulsive interaction between fermions of opposite spins on the same site. Despite the simplicity of the Fermi-Hubbard model, its emergent complexity has made it one of the most prominent models in condensed matter physics.

One of its most notable successes is the description of the Mott metal-insulator transition. In the regime of weak interaction, the system becomes a conducting metal with nearly free electrons described by the Landau Fermi-liquid theory. In the limit of strong interactions ($U \gg t$), however, the model becomes an insulator due to electron-electron correlations although it is predicted to be still a metal in the band theory, demonstrating the inadequacy of the band theory in describing strongly-interacting electrons. This Mott transition into an insulating phase is characterized by the localization of electrons on individual lattice sites, with charge fluctuations suppressed by the large energy cost U of double occupancy.

Another key application of the Fermi-Hubbard model is in the study of unconventional superconductivity with high critical temperature T_c [182]. The discovery of cuprate superconductors in the 1980s [183, 184] revealed a class of materials that exhibit superconductivity at temperatures far above those predicted by conventional BCS theory. The Fermi-Hubbard model, particularly in its two-dimensional form, has been widely used to explore the possibility of unconventional superconductivity driven by strong electron correlations. While a complete theoretical understanding of high-temperature superconductivity remains elusive, the model has provided valuable insights into the role of



(a) Finite temperature (b) Zero temperature

Figure 9.2: Possible phase diagrams of the Hubbard model at intermediate interaction strength U (taken from Ref. [178]).

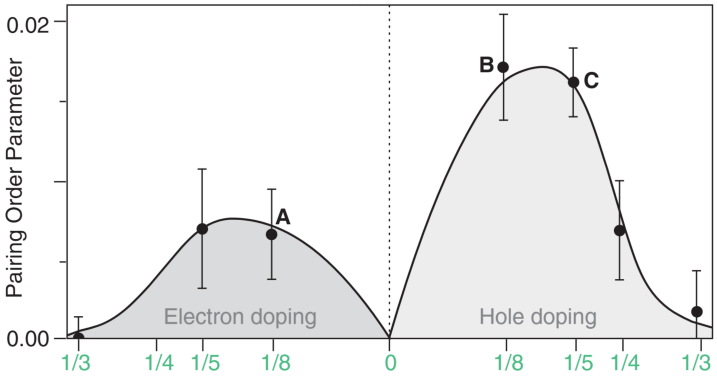


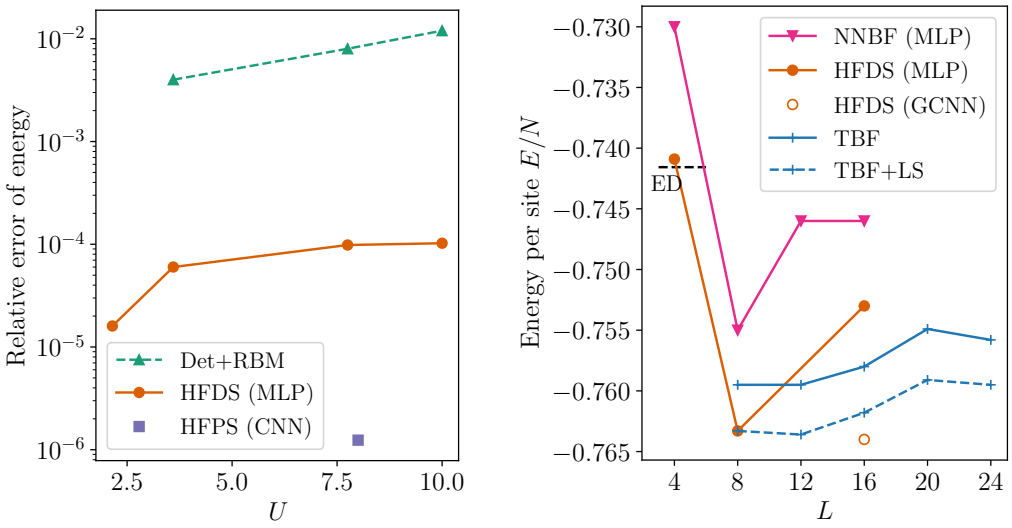
Figure 9.3: The d-wave pairing order parameter in the t-t' Hubbard model (taken from Ref. [185])

antiferromagnetic fluctuations, doped Mott insulators, and the emergence of pseudogap phenomena.

A deeper understanding of the Hubbard model relies on the progress of numerical methods. A cross-verification of constrained-path auxiliary field quantum Monte Carleo (CP-AFQMC) and DMRG shows that the superconductivity phase probably does not exist in the pure Hubbard model [186]. The focus then switches to the t-t' Hubbard model with Hamiltonian

$$\hat{\mathcal{H}} = -t \sum_{\langle i,j\rangle,\sigma} (\hat{c}^{\dagger}_{i,\sigma}\hat{c}_{j,\sigma} + h.c.) - t' \sum_{\langle\langle i,j\rangle\rangle,\sigma} (\hat{c}^{\dagger}_{i,\sigma}\hat{c}_{j,\sigma} + h.c.) + U \sum_{i} \hat{n}_{i,\uparrow}\hat{n}_{i,\downarrow}, \tag{9.2}$$

where $\langle ...\rangle$ and $\langle\langle ...\rangle\rangle$ represent the nearest neighbor and the next-nearest neighbor, respectively. The d-wave superconductivity is observed in this model as shown in Fig. 9.3, suggesting that the Hubbard model is a correct qualitative description of the high-T_c superconductivity in cuprates [185].



(a) 4×4 with $n = 5/8$ at different U

(b) $4 \times L$ with $n = 7/8$ and $U = 8$

Figure 9.4: Relative error of variational energy given by various variational methods in the Hubbard model on the square lattice with PBC. We include the results from ED [176], determinant mean-field with RBM as neural Jastrow (Det+RBM) [176], NNBF [174], HFDS [50], HFDS (unpublished, provided by Christopher Roth), HFPS (unpublished), tensor backflow (TBF) [187], and TBF with a Lanczos step [187].

9.2 Benchmark

The NQS simulation of the Hubbard model is still at an early stage with limited benchmark data available. Here we present the variational energy of $4 \times L$ Hubbard models in Fig. 9.4 to illustrate the accuracy of different variational methods.

We can first have a straightforward comparison between fermionic NQSs. The neural Jastrow method, as shown by Det+RBM in Fig. 9.4(a), does not provide sufficient accuracy in the Hubbard model, especially the strongly-correlated regime with large U. The NNBF in Fig. 9.4(b) employs a shallow MLP discussed in Sec. 5.1 for backflow corrections and achieves better accuracy than the simple Jastrow method but the relative error of energy is still 10% to 20% [174], which is far from enough in strongly-correlated systems. The HFDS in the original literature [176] also utilizes MLPs for hidden fermions, while the unpublished result provided by C. Roth utilizes the GCNN architecture introduced in Sec. 5.4. The GCNN provides a great improvement compared with MLP, demonstrating the significance of modern deep neural architecture, including CNN, GCNN, and transformers, in the complex fermion case. The shallow MLP network without symmetries can also explain the insufficient accuracy in NNBF. The HFPS also utilizes a deep CNN architecture, providing 100 times better accuracy than HFDS with MLP.

The variational results discussed above are also listed in Table 9.1 for convenient reference. The Hubbard model is specified by the square lattice shape and boundary conditions, average site occupancy n, and interaction U, while we fix $t = 1$. Three series of benchmarks are provided, including 4×4 PBC lattice with $n = 5/8$ (3/8 hole doping) and different U, $4 \times L$ PBC lattice with $n = 7/8$ (1/8 hole doping) and $U = 8$, and $L \times L$ lattice with $n = 1$ (half-filling), $U = 8$, and different boundary conditions.

Table 9.1: Variational energy and energy variance in the Hubbard model on the square lattice. By default, the boundary condition is assumed to be PBC. When the boundary condition is specified, P-P means both sides are PBC and P-AP means one side is PBC and the other is APBC. The numerically exact results are marked red, including the ground state wavefunction from ED, and the auxiliary-field quantum Monte Carlo (AFQMC) [188] at half-filling ($n = 1$).

Lattice	n	U	Wavefunction	Ref	E/N	σ^2/N
4×4	5/8	2.1544	Ground state		-1.32576474	0
			HFDS (MLP)	[176]	$-1.3257435(1)$	0.0017(1)
		3.5981	Ground state		-1.24322732	0
			HFDS (MLP)	[176]	$-1.2431527(2)$	0.0019(6)
		7.74264	Ground state		-1.10023322	0
			HFDS (MLP)	[176]	$-1.10012(5)$	0.0023(4)
		8	Ground state		-1.09439792	0
			HFPS (CNN)		-1.0943966	1.7×10^{-5}
		10	Ground state		-1.05647250	0
			HFDS (MLP)	[176]	$-1.056364(5)$	
4×4	7/8	8	Ground state		-0.74180222	0
			Det+Jastrow	[174]	-0.702	
			NNBF (MLP)	[174]	-0.730	
			HFDS (MLP)	[176]	$-0.7409(1)$	
4×8			Det+Jastrow	[174]	-0.719	
			NNBF (MLP)	[174]	-0.755	
			HFDS (MLP)	[176]	$-0.7633(7)$	
			TBF	[187]	-0.7595	
			TBF+LS	[187]	-0.7633	
4×12			Det+Jastrow	[174]	-0.722	
			NNBF (MLP)	[174]	-0.746	
			TBF	[187]	-0.7595	
			TBF+LS	[187]	-0.7636	
4×16			Det+Jastrow	[174]	-0.722	
			NNBF (MLP)	[174]	-0.746	
			HFDS (MLP)	[176]	$-0.753(2)$	
			HFDS (GCNN)		-0.764	
			TBF	[187]	-0.7580	
			TBF+LS	[187]	-0.7618	
4×20			TBF	[187]	-0.7549	
			TBF+LS	[187]	-0.7591	
4×24			TBF	[187]	-0.7558	
			TBF+LS	[187]	-0.7595	
8×8 P-P	1	8	AFQMC	[188]	$-0.5256(6)$	
			PP+RBM+LS	[45, 173]	$-0.52460(0)$	0.0097(3)
			TBF+LS	[187]	-0.5241	
8×8 P-AP			AFQMC	[188]	$-0.5259(3)$	
			PP+RBM	[173]	$-0.5244(6)$	
			HFDS (MLP)	[176]	$-0.5244(6)$	0.0125(5)
16×16 P-AP			AFQMC	[188]	$-0.524(4)$	
			PP+RBM+LS	[45, 173]	$-0.5189(9)$	0.031(8)

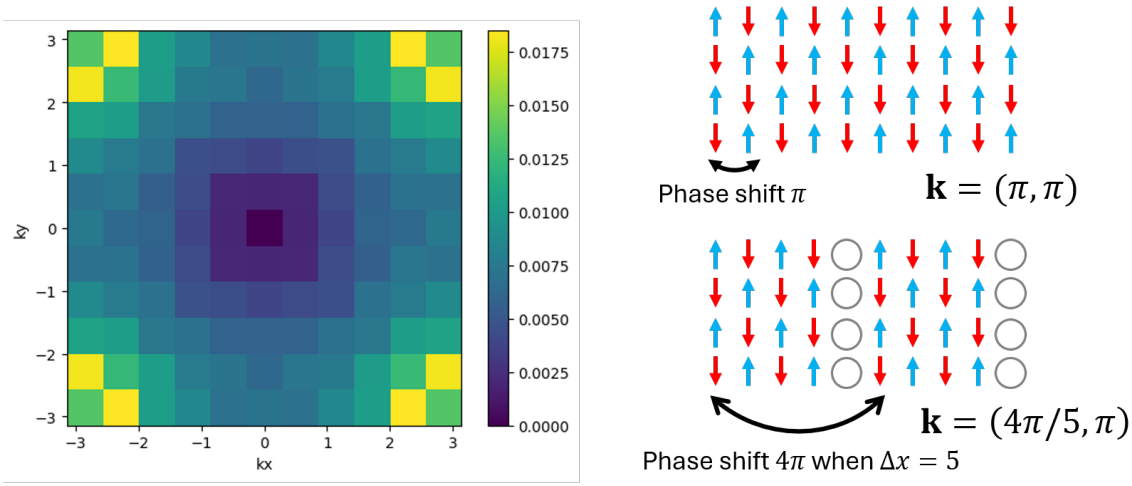


Figure 9.5: The spin structure factor $S(\mathbf{k})$ and the spin density wave in the 10×10 Hubbard model with 1/5 doping.

9.3 Neural quantum state simulations

9.3.1 Nearest-neighbor Hubbard model

Although CP-AFQMC and DMRG suggest an absence of superconductivity in the pure Hubbard model in Eq. (9.1) with only nearest-neighbor hoppings [186], this problem is not fully settled and requires more careful studies. As CP-AFQMC contains bias from the constrained path and DMRG is limited by the geometry of 2D systems, the conclusion obtained by these methods might contain systematic error. Therefore, we utilize the HFPS in Sec. 8.5 for cross-verification with existing numerical results. The result is from an unpublished collaborated work and the data shown in this subsection is contributed by C. Roth.

The system we study is the 10×10 Hubbard model with 1/5 doping and $U = 8$. We begin by verifying that the spin density wave (SDW) exists in the wavefunction expressed by NQS. As translation and rotation symmetries are imposed in the variational, it is difficult to see SDW directly in the real space. To detect SDW, we compute the spin structure factor

$$S(\mathbf{k}) = \frac{1}{N} \sum_{i,j} \langle \hat{S}_i^z \hat{S}_j^z \rangle \, e^{i\mathbf{k}\cdot(\mathbf{r}_i - \mathbf{r}_j)} \tag{9.3}$$

As shown in Fig. 9.5, the peak of $S(\mathbf{k})$ is not at the (π, π) point as one usually observes in the half-filling Hubbard model, but at $(4\pi/5, \pi)$ and all its symmetric points. As also illustrated in the figure, it corresponds to the stripe pattern in the ground-state wavefunction. Therefore, the structure factor reflects the existence of SDW in the 1/5 doping situation, which is also observed by HFDS [176].

For the superconductivity order, we define the d-wave pairing operator

$$\hat{\Delta}(\mathbf{r}) = (\hat{c}_{\mathbf{r}-\hat{\mathbf{x}}} + \hat{c}_{\mathbf{r}+\hat{\mathbf{x}}} - \hat{c}_{\mathbf{r}-\hat{\mathbf{y}}} - \hat{c}_{\mathbf{r}+\hat{\mathbf{y}}})\hat{c}_{\mathbf{r}}, \tag{9.4}$$

where $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ represents the lattice unit vector in the x and y directions. The pair-pair correlation is given by

$$P(\mathbf{r}) = \langle \hat{\Delta}^\dagger(\mathbf{r}) \hat{\Delta}(\mathbf{0}) \rangle \, . \tag{9.5}$$

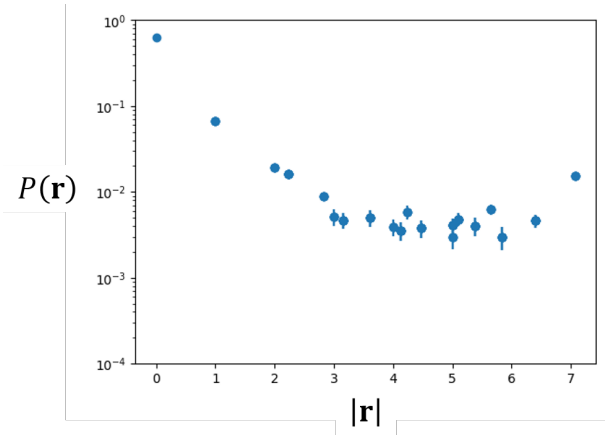


Figure 9.6: D-wave pair-pair correlation in real space

In Fig. 9.6, we show the pair-pair correlation in the real space as a function of distance $|\mathbf{r}|$. Although the correlation exhibits exponential decay at a short distance, a small residual value still exists at a long distance, suggesting a possible existence of long-range pairing order and d-wave superconductivity. This result contradicts the conclusion from Ref. [186], while a more solid conclusion requires larger-scale simulations and more analysis.

9.3.2 Fully-connected Hubbard model

As the NQS is capable of capturing long-range entanglement, it makes the variational approach available in the fully-connected Hubbard model

$$\hat{\mathcal{H}} = -\frac{1}{\sqrt{N}} \sum_{i,j=1}^{N} \sum_{\sigma} t_{i,j} \hat{c}_{i,\sigma}^{\dagger} \hat{c}_{j,\sigma} + U \sum_{i=1}^{N} \hat{n}_{i,\uparrow} \hat{n}_{i,\downarrow}, \tag{9.6}$$

where $t_{i,j}$ are random hopping amplitudes drawn from the Gaussian normal distribution. This model is exactly solvable by the dynamical mean-field theory (DMFT) and is known to undergo a phase transition from a metallic Fermi liquid to a Mott insulator as U increases. Although DMFT provides the exact value of most observables in the fully-connected Hubbard model in the thermodynamics limit [189], the wavefunction is still not directly available from DMFT. Furthermore, the traditional DMRG method is not applicable due to the volume-law entanglement of the system induced by long-range interactions. Therefore, the fermionic NQS can be utilized to obtain wavefunctions in this strongly-correlated fermion system. In Ref. [29], the HFDS discussed in Sec. 8.4 is utilized to approximate the ground state of this model.

The phase diagram of this model can be obtained by analyzing the many-body wavefunction provided by the NQS. For any Fock state $|n\rangle$, one can define the number of double occupancies as

$$D = \sum_{i} \langle s | \hat{n}_{i,\uparrow} \hat{n}_{i,\downarrow} | s \rangle \,. \tag{9.7}$$

It is expected that the system tends to have more doubly occupied sites in the Fermi liquid phase while less in the Mott insulator phase. One can further define the weights

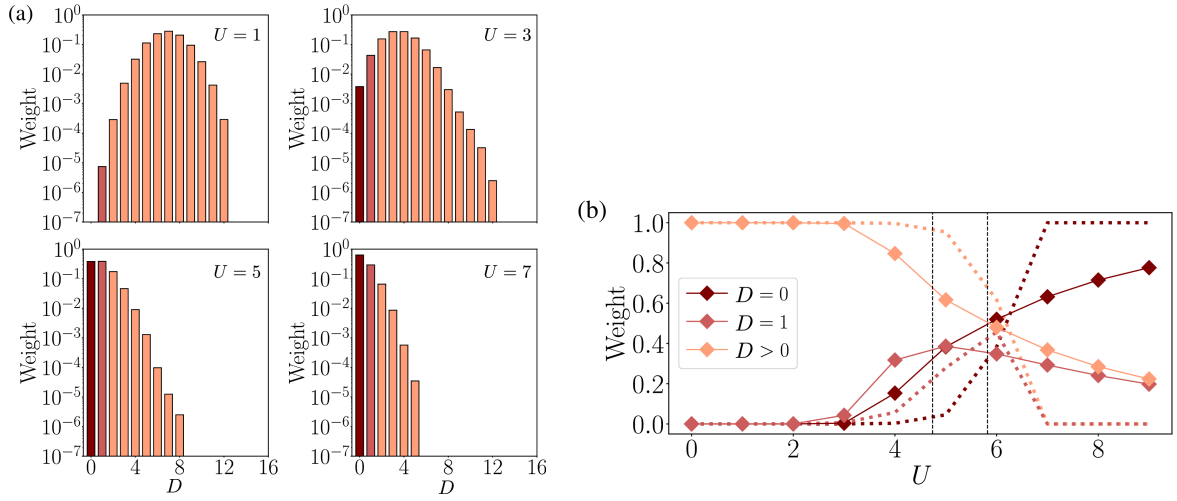


Figure 9.7: Double occupancy weights in the fully-connected Hubbard model with $N = 32$ and half-filling. The dashed vertical lines show the phase transition points $U_{c1} \approx 4.74$ and $U_{c2} \approx 5.82$ at $N \to \infty$ from DMFT.

of double occupancy

$$\mathcal{W}(D) = \sum_{n_D} |\psi(n_D)|^2, \tag{9.8}$$

where the summation is performed over all Fock states $|n_D\rangle$ with D double occupancies. As shown in Fig. 9.7, $\mathcal{W}(0)$ and $\mathcal{W}(1)$ are indeed smaller in the Fermi liquid phase and larger in the Mott insulator phase. Furthermore, the crossing points of $\mathcal{W}(D)$ reproduce the phase transition point predicted by DMFT [189, 190]. Therefore, the NQS successfully produces reliable wavefunctions in the long-range Hubbard model, indicating the great power of the fermionic NQS in solving systems with strongly-correlated electrons and volume-law entanglement.

Chapter 10

Conclusion and outlook

10.1 Conclusion

Computing the ground state of interacting quantum matter is a long-standing challenge in quantum physics. Although many numerical methods have been developed, including ED, QMC, mean-field wavefunctions, and tensor networks, they all face respective limitations in large 2D or 3D systems with strong interactions, where most of the intriguing quantum many-body phenomena emerge.

Inspired by successful practices of ANNs on machine learning, NQS was proposed as a novel variational wavefunction with non-linear structures and great expressive power [19] (Chapter 1). Benefiting from decades of historical experience in VMC (Chapter 2) and SR (Chapter 3), NQS soon showed its outstanding performance in quantum many-body problems. It outperformed the accuracy of several traditional methods in the non-frustrated Heisenberg model at that time. Apart from the ground-state search tasks we strengthen in the whole thesis, it also provides new solutions to other quantum many-body problems including quantum dynamics [19, 67–71], open quantum systems [191–193], and electronic structures [194–198].

Nevertheless, the NQS faced severe challenges when it encountered quantum systems with non-trivial sign structures, which is a fundamental difficulty of quantum mechanics compared to classical physics or computer science. The first challenge comes from the choice of network architecture. With a suitable combination of symmetry projection [80, 81] (Chapter 4) and deep neural networks [58, 107, 112, 118] (Chapter 5), NQS finally gains enough expressive power for frustrated sign structures, as demonstrated also in our recent work [125]. Another key challenge for the NQS approach, until recently, has been the optimization performed by SR. As a method designed for traditional small-scale wavefunctions, SR has a huge computational complexity when it comes to deep NQSs with a large number of parameters. This problem was resolved by the MinSR method proposed in my research [58] (Chapter 3). With suitable network architecture and MinSR optimization, it has now finally become possible to apply NQSs in frustrated quantum spin systems to achieve unprecedented accuracy.

The NQS was then applied to the study of frustrated magnets, prominently QSLs (Chapter 6). The evidence and properties of QSLs were obtained through NQS simula-

tions in various quantum spin systems, including the square J_1-J_2 model [58, 112, 142], the triangular J_1-J_2 model [58, 112], and the Shastry-Sutherland model [117]. These recent advances demonstrate that NQS has become a competitive numerical method for solving cutting-edge quantum many-body problems. The toolbox of computational quantum physics, containing the ED, QMC, mean-field wavefunctions, and tensor networks, now has NQS as another powerful member for frustrated magnets and interacting fermions beyond 1D.

As outlined in this thesis, the NQS can also be applied to fermion systems. Based on mean-field fermionic wavefunctions (Chapter 7), several fermionic NQSs have been proposed, including neural Jastrow [173], NNBF [174], HFDS [176], and HFPS (Chapter 8). Although the fermionic NQSs is still at an early stage, it has already made impressive progress in the study of the famous Fermi-Hubbard model. The fermionic NQS, to be specific, HFDS and HFPS, demonstrates the existence of the spin density wave and d-wave pairing order in the pure Fermi-Hubbard model, providing new insights into the description of high-temperature superconductivity (Chapter 9).

In summary, the NQS research has made great progress in recent years and is moving toward a deep era. In many-body quantum systems, especially QSLs and Fermi-Hubbard models, NQS starts to provide new insights beyond the reach of traditional numerical methods. After many years of NQS benchmarks against traditional methods, we are now finally at a stage to exploit the great potential of NQS to push forward the study of complex emergent behavior in quantum many-body systems.

10.2 Outlook

Although impressive progress has been achieved in the NQS research, this novel method is still facing multiple challenges. During the past years of NQS research, the three outstanding limiting factors listed below have attracted considerable attention.

- **Expressive power:** The NQS cannot approximate the target quantum state due to the limited amount of variational parameters.
- **Training dynamics:** The NQS can express the target state, but it is not properly optimized to convergence.
- **Generalization ability:** The NQS can express the target state, but it cannot generalize the information learned from samples to the full Hilbert space.

In order to push the NQS approach further, it will be key to achieve progress on all three challenges. In the following, we will provide ideas and outlooks to make potential progress for solving these issues. In the end, we will provide an overview on the potential future application of NQSs in strongly-correlated systems.

10.2.1 Expressive power: ANN scalability for new physics

The expressive power of NQS depends on the available number of parameters N_p and the architecture of the ANN. Then two directions are available to improve the expressive

power, utilizing large-scale GPU clusters for more trainable parameters or adjusting the network architecture for higher efficiency. These improvements are often technical but will be the key step to producing even more accurate numerical results. As the NQS has been a powerful tool in solving many QSL systems, the expressive power is probably the main limiting factor in this case. By exploiting the expressive power of ANNs, we can potentially make further progress in the study of QSLs.

Although clear routines are available for the study of spin systems, in fermion systems it is still an open problem whether the expressive power is the main limit of current fermionic NQSs, including NNBF, HFDS, and HFPS. Although these states often promise the exact expression of any fermion state in a limit of $N_p \to \infty$, the efficiency of extending network sizes under the current framework is still unknown. Therefore, it is more important to adopt a suitable architecture of ANNs and combine it properly with fermion sign structures. Then careful design choices are necessary to exploit the full power of ANNs in fermion systems, which also affects the generalization ability as we will discuss later.

10.2.2 Training dynamics: Search of better algorithm

The MinSR method I proposed in Ref. [58] changes the cubic complexity on the number of variational parameters $\mathcal{O}(N_p^3)$ to a cubic complexity on the number of samples $\mathcal{O}(N_s^3)$. Although MinSR provides great acceleration for large-scale NQSs and has pushed NQS research to a new level, it also limits the number of samples available in VMC. Therefore, the optimization tends to be noisier, making the training dynamics less reliable. Therefore, the training could stop when the VMC noise dominates the optimization step while the NQS still has not reached its limiting expressive power.

There are several possible directions to establish better training dynamics. First, in SR and MinSR the local energy E_θ^{loc} often has a large variance, making the optimization dominated by noise. It will be possible to obtain more reliable training steps if the noisy part can be suppressed [67]. Second, the second-order optimization can be employed in VMC to improve the training dynamics [199], while it also requires a careful design of the algorithm to reduce the cost of computing the Hessian matrix.

10.2.3 Generalization ability: A "myopia" test

Consider VMC optimization with N_s samples in each iteration and in total N_{iter} iterations. Then the total amount of samples seen by the NQS in the whole training process is $N_s \times N_{\text{iter}}$. Although one can perform large-scale simulations to increase N_s and N_{iter}, the total number of samples is still much smaller than the Hilbert space dimension in most cases. Therefore, it is important for the NQS to generalize the information learned from the limited amount of samples to unseen samples to ensure reliable performance in the full Hilbert space.

The generalization ability of NQSs has been initially discussed in Ref. [131], in which the authors claim that it is difficult for NQSs to generalize the frustrated sign structure learned from a finite portion of the Hilbert space. In this case, the NQS has "myopia", limiting it to only obtain information local to given samples instead of the full Hilbert space. As discussed in Sec. 5.6, this problem is caused by separating the amplitudes and

signs into independent networks and can be resolved by a suitable choice of network architecture. Therefore, the NQS has "myopia" if it has to fit a rugged distribution like sign structures, but the difficulty can be alleviated by suitable design choices.

A similar problem also happens to the fermionic NQS if one does not encode fermion statistics into the wavefunction, as discussed in Sec. 8.1. The additional minus signs have to be included in fermionic wavefunctions according to the fermion ordering, which is difficult for neural networks to learn directly and will also cause "myopia". Therefore, mean-field wavefunctions have been combined with ANNs to create better fermionic NQSs, but it is still unclear whether it fully solves the "myopia" problem. The problem of generalization ability has not attracted much attention so far although it might be a significant limiting factor in fermionic NQSs. It will be key in the future to achieve control over this problem in order to make NQS an efficient and superior method to solve fermionic quantum matter.

10.2.4 NQS for strongly-correlated quantum matter

As a novel method proposed in 2016, the NQS research is still at an early stage. Despite the limitations listed above, it has begun to revolutionize the study of strongly-correlated quantum matters recently. With the improvement of expressive power, training dynamics, and generalization ability, the NQS is expected to develop into an even more powerful tool that helps to understand more models beyond the reach of traditional methods.

In the study of QSLs, it has been a long-lasting question whether QSLs can be realized in real materials. Clear evidence of QSLs, including entanglement entropy and fractional excitations, is not directly measurable from experiments. Although the vanishing order parameter can be viewed as a feature of QSL, it cannot distinguish the QSL phase from the trivial paramagnetic phase and hence does not directly verify the existence of QSLs. The computational approach, especially the NQS, can be utilized to solve this problem. Numerically, it is straightforward to determine the existence of the QSL regime by measuring order parameters at zero temperature. The Rényi-2 entropy can also be obtained in NQSs to provide direct evidence of QSLs. Then the key step is to prove the qualitative similarity between numerical models and real materials. There will be strong evidence if the phase boundary in simulations and experiments match each other, as we have done for $NaYbSe_2$ [155]. Furthermore, one can also compare the spectral function or real-space correlation measured from NQS simulations and neutron scattering experiments. With efforts from both the numerical and experimental sides, we are getting closer to verifying the existence of QSLs in real materials.

On the other hand, the Fermi-Hubbard model is more controversial. Although the high-T_c superconductivity is widely believed to exist in this model or its variants, it is difficult to obtain decisive evidence. In most numerical methods, including the NQS, charge and spin density waves can be found and the d-wave pairing order sometimes coexists with these stripe patterns [185]. Then it is the key next step to observe how stripe patterns interact with d-wave pairing in large 2D systems. Since the CP-AFQMC contains unavoidable bias and DMRG is limited to quasi-1D systems, the future contribution of NQS can be indispensable for understanding this model. This will insert new momentum into this field or even help us to reveal the mechanism of high-T_c superconductivity, which has been a central problem of condensed matter physics for many decades.

Appendix A

Complex derivatives

This appendix explains the concept of complex derivatives and discusses the correct derivative to obtain complex gradients. Then we introduce the quantum state gradient as a direct way to obtain the gradient direction in the Hilbert space.

A.1 Complex scalar gradient

A.1.1 $\mathbb{CR}$-derivatives

Consider a complex-valued function $f : \mathbb{C} \to \mathbb{C}$. The complex derivative of f is defined as

$$f'(z) = \lim_{\Delta z \to 0} \frac{f(z + \Delta z) - f(z)}{\Delta z}. \tag{A.1}$$

Let $z = x + iy$ and $f(z) = u(x, y) + iv(x, y)$. To have a well-defined value of $f'(z)$ independent of the phase of Δz, f should be a holomorphic function satisfying the Cauchy-Riemann equations

$$\begin{aligned} \frac{\partial u}{\partial x} &= \frac{\partial v}{\partial y}, \\ \frac{\partial v}{\partial x} &= -\frac{\partial u}{\partial y}. \end{aligned} \tag{A.2}$$

In this case, the complex derivative is performed similarly to real-valued functions. For instance, if $f(z) = z^2$, then $f'(z) = 2z$. We call it complex-derivative, or $\mathbb{C}$-derivative [200].

The derivative of a non-holomorphic function f is usually ill-defined since $f'(z)$ in Eq. (A.1) depends on the phase of Δz. Nevertheless, one can still generalize the definition of derivatives to non-holomorphic cases.

Definition A.1.1 (Real-derivative ($\mathbb{R}$-derivative) [200])**.** The $\mathbb{R}$-derivative of a function $f : \mathbb{C} \to \mathbb{C}$ or $\mathbb{C} \to \mathbb{R}$ is

$$\frac{\partial f(z, z^*)}{\partial z}, \tag{A.3}$$

where z and z^* are formally considered as independent parameters in partial derivatives.

Definition A.1.2 (Conjugate real-derivative ($\bar{\mathbb{R}}$-derivative) [200])**.** The $\bar{\mathbb{R}}$-derivative of a function $f : \mathbb{C} \to \mathbb{C}$ or $\mathbb{C} \to \mathbb{R}$ is

$$\frac{\partial f(z, z^*)}{\partial z}. \tag{A.4}$$

As any holomorphic function $f(z)$ does not explicitly depend on z^*, the $\mathbb{R}$-derivative is equivalent to the $\mathbb{C}$-derivative for holomorphic functions. Here we further state several theorems of $\mathbb{R}$-derivative and $\bar{\mathbb{R}}$-derivative without proof.

Theorem A.1.1. *For a function $f : \mathbb{C} \to \mathbb{C}$ or $f : \mathbb{C} \to \mathbb{R}$,*

$$\left(\frac{\partial f}{\partial z}\right)^* = \frac{\partial f^*}{\partial z^*}, \quad \left(\frac{\partial f^*}{\partial z}\right)^* = \frac{\partial f}{\partial z^*}. \tag{A.5}$$

Theorem A.1.2 (Chain rule)**.** *Consider functions $f_1 : \mathbb{C} \to \mathbb{C}$ with $z_2 = f_1(z_1)$ and $f_2 : \mathbb{C} \to \mathbb{C}$ with $z_3 = f_2(z_2)$. Then*

$$\frac{\partial z_3}{\partial z_1} = \frac{\partial z_3}{\partial z_2}\frac{\partial z_2}{\partial z_1} + \frac{\partial z_3}{\partial z_2^*}\frac{\partial z_2^*}{\partial z_1}, \tag{A.6}$$

$$\frac{\partial z_3}{\partial z_1^*} = \frac{\partial z_3}{\partial z_2}\frac{\partial z_2}{\partial z_1^*} + \frac{\partial z_3}{\partial z_2^*}\frac{\partial z_2^*}{\partial z_1^*}. \tag{A.7}$$

A.1.2 Gradient

For a real-valued scalar function $f : \mathbb{R} \to \mathbb{R}$, the gradient is defined as

$$g = f'(x) = \frac{\mathrm{d}f}{\mathrm{d}x}. \tag{A.8}$$

For a real-valued vector function $f : \mathbb{R}^N \to \mathbb{R}$ with inputs $\mathbf{x} = (x_1, x_2, ..., x_N)^T$, the gradient is defined as

$$\mathbf{g} = \sum_n \frac{\partial f}{\partial x_n}\hat{\mathbf{e}}_n, \tag{A.9}$$

where $\hat{\mathbf{e}}_n$ is a unit vector at n'th axis. The gradient is sometimes also denoted as

$$\mathbf{g} = \frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}^T}. \tag{A.10}$$

Consider a non-holomorphic function $f : \mathbb{C} \to \mathbb{R}$. One can also rewrite it as a real-valued vector function $\tilde{f} : \mathbb{R}^2 \to \mathbb{R}$ with $\tilde{f}(x, y) = f(z, z^*) = f(x + iy, x - iy)$. The gradient of $\tilde{f}$ is given by

$$\mathbf{g}^{(r)} = \left(\frac{\partial \tilde{f}}{\partial x}, \frac{\partial \tilde{f}}{\partial y}\right)^T = \left(\frac{\partial f}{\partial z} + \frac{\partial f}{\partial z^*}, i\frac{\partial f}{\partial z} - i\frac{\partial f}{\partial z^*}\right)^T. \tag{A.11}$$

Then the gradient on the complex plane can be obtained as

$$g = \frac{1}{2}(\mathbf{g}_0^{(r)} + i\mathbf{g}_1^{(r)}) = \frac{\partial f}{\partial z^*}, \tag{A.12}$$

where $1/2$ is a conventional prefactor. Therefore, we have the following theorem.

Theorem A.1.3. *The gradient of a function $f : \mathbb{C} \to \mathbb{R}$ is given by its $\bar{\mathbb{R}}$-derivative.*

It might appear unnatural that the gradient is given by the $\bar{\mathbb{R}}$-derivative instead of $\mathbb{R}$-derivative. To provide an example to understand it, we can consider a simple function $f(z, z^*) = zz^* = |z|^2$. The gradient given by the $\bar{\mathbb{R}}$-derivative is $g = z$, which is a correct result indicating the ascending direction of f.

A.2 Quantum state gradient

For a complex-valued vector function $f : \mathbb{C}^n \to \mathbb{R}$, we can combine Eq. (A.10) and Eq. (A.12) to obtain the gradient

$$\mathbf{g} = \sum_n \frac{\partial f}{\partial z_n^*} \hat{\mathbf{e}}_n = \frac{\partial f}{\partial \mathbf{z}^\dagger}. \tag{A.13}$$

With Dirac's bra-ket notation, it can be rewritten as

$$|g\rangle = \frac{\partial f}{\partial \langle z|}, \tag{A.14}$$

where $|g\rangle$ is a vector in the Hilbert space and $\langle z|$ is in the dual space. In quantum physics, we often need to minimize a loss function L as a function of a variational quantum state $|\Psi\rangle$. Then it is convenient to define a quantum state gradient as

$$|g_\Psi\rangle = ||\Psi|| \frac{\partial f}{\partial \langle \Psi|}, \tag{A.15}$$

where $||\Psi|| = \sqrt{\langle \Psi|\Psi\rangle}$ is a normalization factor to make the gradient invariant under a change of norm.

Here we provide several useful quantum state gradients. The loss function utilized in VMC is the energy

$$L = E = \frac{\langle \Psi|\hat{\mathcal{H}}|\Psi\rangle}{\langle \Psi|\Psi\rangle}, \tag{A.16}$$

whose quantum state gradient is

$$|g_\Psi\rangle = ||\Psi|| \left(\frac{\hat{\mathcal{H}}|\Psi\rangle}{\langle \Psi|\Psi\rangle} - \frac{E|\Psi\rangle}{\langle \Psi|\Psi\rangle} \right) = (\hat{\mathcal{H}} - E) \frac{|\Psi\rangle}{||\Psi||}. \tag{A.17}$$

In supervised learning, it is common to define the loss function as minus-log-fidelity

$$L = -\log \mathcal{F} = -\log \frac{\langle \Psi|\Phi\rangle \langle \Phi|\Psi\rangle}{\langle \Psi|\Psi\rangle \langle \Phi|\Phi\rangle}, \tag{A.18}$$

where $|\Psi\rangle$ is the variational state and $|\Phi\rangle$ is the target state. Then the quantum state gradient is

$$|g_\Psi\rangle = ||\Psi|| \frac{\partial}{\partial \langle \Psi|} (\log \langle \Psi|\Psi\rangle - \log \langle \Psi|\Phi\rangle) = \frac{|\Psi\rangle}{||\Psi||} - ||\Psi|| \frac{|\Phi\rangle}{\langle \Psi|\Phi\rangle}. \tag{A.19}$$

Appendix B

Linear algebra

B.1 Linear equation

Definition B.1.1 (Least-squares minimum-norm solution)**.** For a given linear equation $\mathbf{Ax} = \mathbf{b}$, the least-squares solution is the solution minimizing $||\mathbf{Ax} - \mathbf{b}||^2$. Among all least-squares solutions, the least-squares minimum-norm solution is the one minimizing $||\mathbf{x}||^2$.

Theorem B.1.1. *The least-squares minimum-norm solution of* $\mathbf{Ax} = \mathbf{b}$ *is*

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b} = (\mathbf{A}^\dagger\mathbf{A})^{-1}\mathbf{A}^\dagger\mathbf{b} = \mathbf{A}^\dagger(\mathbf{A}\mathbf{A}^\dagger)^{-1}\mathbf{b}, \tag{B.1}$$

where the matrix inverse is the Moore–Penrose inverse (pseudo-inverse).

Proof. Firstly, we prove $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$ is the least-squares minimum-norm solution. The singular value decomposition of $\mathbf{A}$ gives

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\dagger, \tag{B.2}$$

where $\mathbf{U}$ and $\mathbf{V}$ are unitary matrices, and $\mathbf{\Sigma}$ is a diagonal matrix with $\sigma_i = \Sigma_{ii} = 0$ if and only if $i > r$ with r the rank of $\mathbf{A}$. The least-squares solution is given by minimizing

$$\begin{aligned} ||\mathbf{Ax} - \mathbf{b}||^2 &= ||\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\dagger\mathbf{x} - \mathbf{b}||^2 = ||\mathbf{\Sigma}\mathbf{x}' - \mathbf{b}'||^2 \\ &= \sum_{i=1}^{r}(\sigma_i x_i' - b_i')^2 + \sum_{i=r+1}^{N} b_i'^2, \end{aligned} \tag{B.3}$$

where $\mathbf{x}' = \mathbf{V}^\dagger\mathbf{x}$, $\mathbf{b}' = \mathbf{U}^\dagger\mathbf{b}$, N is the dimension of $\mathbf{b}$, and the second step is because applying a unitary matrix does not change the norm of a vector. Therefore, all the least-squares solutions take the form

$$x_i' = \begin{cases} b_i'/\sigma_i & i \le r, \\ \text{any value} & i > r. \end{cases} \tag{B.4}$$

Among all these possible solutions, the one minimizes $||\mathbf{x}|| = ||\mathbf{x}'||$ is

$$x_i' = \begin{cases} b_i'/\sigma_i & i \le r, \\ 0 & i > r. \end{cases} \tag{B.5}$$

With the following definition of pseudo-inverse

$$\begin{aligned} \mathbf{A}^{-1} &= \mathbf{V}\boldsymbol{\Sigma}^{+}\mathbf{U}^{\dagger}, \\ \Sigma^{+}_{ij} &= \delta_{ij} \times \begin{cases} 1/\sigma_i & \sigma_i > 0, \\ 0 & \sigma_i = 0, \end{cases} \end{aligned} \tag{B.6}$$

we have $\mathbf{x}' = \boldsymbol{\Sigma}^{+}\mathbf{b}'$, so the final solution is

$$\mathbf{x} = \mathbf{V}\mathbf{x}' = \mathbf{V}\boldsymbol{\Sigma}^{+}\mathbf{U}^{\dagger}\mathbf{b} = \mathbf{A}^{-1}\mathbf{b}. \tag{B.7}$$

Furthermore, we show the following equality

$$\mathbf{A}^{-1} = (\mathbf{A}^{\dagger}\mathbf{A})^{-1}\mathbf{A}^{\dagger} = \mathbf{A}^{\dagger}(\mathbf{A}\mathbf{A}^{\dagger})^{-1}. \tag{B.8}$$

With the singular value decomposition of $\mathbf{A}$ in Eq. (B.2), Eq. (B.8) can be directly proved by

$$\begin{aligned} (\mathbf{A}^{\dagger}\mathbf{A})^{-1}\mathbf{A}^{\dagger} &= (\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^{\dagger}\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\dagger})^{-1}\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^{\dagger} \\ &= \mathbf{V}(\boldsymbol{\Sigma}^{+})^{2}\mathbf{V}^{\dagger}\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^{\dagger} \\ &= \mathbf{V}\boldsymbol{\Sigma}^{+}\mathbf{U}^{\dagger} = \mathbf{A}^{-1}, \end{aligned} \tag{B.9}$$

and

$$\begin{aligned} \mathbf{A}^{\dagger}(\mathbf{A}\mathbf{A}^{\dagger})^{-1} &= \mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^{\dagger}(\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\dagger}\mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^{\dagger})^{-1} \\ &= \mathbf{V}\boldsymbol{\Sigma}\mathbf{U}^{\dagger}\mathbf{U}(\boldsymbol{\Sigma}^{+})^{2}\mathbf{U}^{\dagger} \\ &= \mathbf{V}\boldsymbol{\Sigma}^{+}\mathbf{U}^{\dagger} = \mathbf{A}^{-1}. \end{aligned} \tag{B.10}$$

In the derivation, the shapes of diagonal matrices $\boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}^{+}$ are not fixed but assumed to match their neighbor matrices to make the matrix multiplication valid.

□

B.2 Determinant

Theorem B.2.1. *Suppose*

$$\mathbf{M} = \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{pmatrix}, \tag{B.11}$$

where $\mathbf{A}$ *and* $\mathbf{D}$ *are invertible square matrices. Then*

$$\det(\mathbf{M}) = \det(\mathbf{A})\det(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B}) = \det(\mathbf{D})\det(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}). \tag{B.12}$$

Proof.

$$\begin{aligned} \det(\mathbf{M}) &= \det\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{pmatrix} \\ &= \det(\mathbf{D})\det\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{pmatrix}\det\begin{pmatrix} \mathbf{1} & \mathbf{0} \\ -\mathbf{D}^{-1}\mathbf{C} & \mathbf{D}^{-1} \end{pmatrix} \\ &= \det(\mathbf{D})\det\begin{pmatrix} \mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C} & \mathbf{B}\mathbf{D}^{-1} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \\ &= \det(\mathbf{D})\det(\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C}). \end{aligned} \tag{B.13}$$

Similarly, one can also prove that $\det(\mathbf{M}) = \det(\mathbf{A})\det(\mathbf{D} - \mathbf{C}\mathbf{A}^{-1}\mathbf{B})$. □

Theorem B.2.2 (Matrix determinant lemma)**.** *Suppose* $\mathbf{A}$ *is an* $n \times n$ *invertible matrix and* $\mathbf{u}, \mathbf{v}$ *are column vectors with length* n*. Then*

$$\det(\mathbf{A} + \mathbf{u}\mathbf{v}^T) = \det(\mathbf{A})(1 + \mathbf{v}^T\mathbf{A}^{-1}\mathbf{u}). \tag{B.14}$$

Proof. Firstly we prove the special case $\mathbf{A} = \mathbf{1}_n$.

$$\begin{pmatrix} \mathbf{1}_n & \mathbf{0} \\ \mathbf{v}^T & 1 \end{pmatrix} \begin{pmatrix} \mathbf{1}_n + \mathbf{u}\mathbf{v}^T & \mathbf{u} \\ \mathbf{0} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{1}_n & \mathbf{0} \\ -\mathbf{v}^T & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{1}_n & \mathbf{u} \\ 0 & 1 + \mathbf{v}^T\mathbf{u} \end{pmatrix}. \tag{B.15}$$

The determinant of both sides gives

$$\det(\mathbf{1}_n + \mathbf{u}\mathbf{v}^T) = 1 + \mathbf{v}^T\mathbf{1}_n\mathbf{u}. \tag{B.16}$$

Then

$$\det(\mathbf{A} + \mathbf{u}\mathbf{v}^T) = \det(\mathbf{A}) \det(\mathbf{1}_n + (\mathbf{A}^{-1}\mathbf{u})\mathbf{v}^T) = \det(\mathbf{A})(1 + \mathbf{v}^T\mathbf{A}^{-1}\mathbf{u}). \tag{B.17}$$

□

Theorem B.2.3. *Suppose* $\mathbf{A}$ *is an* $n \times n$ *invertible matrix and* $\mathbf{U}, \mathbf{V}$ *are* $n \times m$ *matrices. Then*

$$\det(\mathbf{A} + \mathbf{U}\mathbf{V}^T) = \det(\mathbf{A}) \det(\mathbf{1}_m + \mathbf{V}^T\mathbf{A}^{-1}\mathbf{U}). \tag{B.18}$$

Theorem B.2.4 (Jacobi's formula)**.** *Suppose* $\mathbf{A}$ *is an* $n \times n$ *invertible matrix with elements* A_{ij}*. Then*

$$\frac{\partial \det(\mathbf{A})}{\partial A_{ij}} = \det(\mathbf{A})(\mathbf{A}^{-1})_{ji}. \tag{B.19}$$

Proof. Assume a variation is imposed at the i'th row and j'th column of $\mathbf{A}$, i.e.

$$(\mathrm{d}\mathbf{A})_{kl} = \delta_{ik}\delta_{jl}\mathrm{d}x, \tag{B.20}$$

where $\mathrm{d}x$ is an infinitesimal scalar. Then

$$\det(\mathbf{A} + \mathrm{d}\mathbf{A}) = \det(\mathbf{A}) \det(\mathbf{1}_n + \mathbf{A}^{-1}\mathrm{d}\mathbf{A}) = \det(\mathbf{A})(1 + \mathrm{tr}(\mathbf{A}^{-1}\mathrm{d}\mathbf{A})), \tag{B.21}$$

where we have utilized

$$\det(\mathbf{1} + \mathrm{d}\mathbf{X}) = 1 + \mathrm{tr}(\mathrm{d}\mathbf{X}) \tag{B.22}$$

The trace can be simplified as

$$\mathrm{tr}(\mathbf{A}^{-1}\mathrm{d}\mathbf{A}) = (\mathbf{A}^{-1})_{lk}(\mathrm{d}\mathbf{A})_{kl} = (\mathbf{A}^{-1})_{lk}\delta_{ik}\delta_{jl}\mathrm{d}x = (\mathbf{A}^{-1})_{ji}\mathrm{d}x, \tag{B.23}$$

given the Einstein summation rule. Then

$$\det(\mathbf{A} + \mathrm{d}\mathbf{A}) = \det(\mathbf{A})(1 + (\mathbf{A}^{-1})_{ji}\mathrm{d}x). \tag{B.24}$$

The partial derivative is then given by

$$\frac{\partial \det(\mathbf{A})}{\partial A_{ij}} = \frac{\det(\mathbf{A} + \mathrm{d}\mathbf{A}) - \det(\mathbf{A})}{\mathrm{d}x} = \det(\mathbf{A})(\mathbf{A}^{-1})_{ji}. \tag{B.25}$$

□

B.3 Pfaffian

B.3.1 Definition

Pfaffian maps a $2n \times 2n$ skew-symmetric matrix $\mathbf{A}$ to a number $\mathrm{pf}\,\mathbf{A}$. The pfaffian can be formally defined as follows. Partition all numbers $\{1, ..., 2n\}$ into n pairs $\alpha = \{(i_1, j_1), ..., (i_n, j_n)\}$ with $i_k < j_k$ and $i_1 < i_2 < ... < i_n$, in total $(2n-1)!!$ possible partitions. Then the pfaffian of matrix $\mathbf{A}$ with elements $A_{i,j}$ is

$$\mathrm{pf}\mathbf{A} = \sum_{\alpha} \mathrm{sign}(\alpha) \prod_{k=1}^{n} A_{i_k, j_k}, \tag{B.26}$$

where $\mathrm{sign}(\alpha)$ is given by the parity of the permutation $(i_1, j_1, i_2, j_2, ..., i_n, j_n)$. For example,

$$\mathrm{pf} \begin{pmatrix} 0 & a \\ -a & 0 \end{pmatrix} = a, \tag{B.27}$$

$$\mathrm{pf} \begin{pmatrix} 0 & a & b & c \\ -a & 0 & d & e \\ -b & -d & 0 & f \\ -c & -e & -f & 0 \end{pmatrix} = af - be + cd. \tag{B.28}$$

B.3.2 Relation with determinant

One can check Ref. [201] for the proof of the following theorems.

Theorem B.3.1. *Suppose* $\mathbf{A}$ *is an* $2n \times 2n$ *skew-symmetric matrix and* $\mathbf{B}$ *is an* $2n \times 2n$ *matrix. Then*

$$\mathrm{pf}(\mathbf{B}\mathbf{A}\mathbf{B}^T) = \det(\mathbf{B})\mathrm{pf}(\mathbf{A}). \tag{B.29}$$

Theorem B.3.2. *Suppose* $\mathbf{A}$ *is an* $2n \times 2n$ *skew-symmetric matrix. Then*

$$b\mathrm{pf}^2(\mathbf{A}) = \det(\mathbf{A}). \tag{B.30}$$

Theorem B.3.3. *Suppose* $\mathbf{A}$ *is an* $n \times n$ *matrix. Then*

$$\mathrm{pf} \begin{pmatrix} \mathbf{0} & \mathbf{A} \\ -\mathbf{A}^T & \mathbf{0} \end{pmatrix} = (-1)^{n(n-1)/2} \det(\mathbf{A}) \tag{B.31}$$

B.3.3 Other theorems

We have the next theorem from the definition of pfaffian in Eq. (B.26).

Theorem B.3.4. *Suppose* $\mathbf{A}$ *and* $\mathbf{A}'$ *are two* $n \times n$ *matrices. Then*

$$\mathrm{pf} \begin{pmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}' \end{pmatrix} = \mathrm{pf}(\mathbf{A})\mathrm{pf}(\mathbf{A}') \tag{B.32}$$

Theorem B.3.5. *Suppose* $\mathbf{A}$ *and* $\mathbf{C}$ *are two invertible skew-symmetric matrices with shape* $2n \times 2n$ *and* $2m \times 2m$*, and* $\mathbf{B}$ *is a* $2n \times 2m$ *matrix. Then*

$$\frac{\mathrm{pf}(\mathbf{A}+\mathbf{B}\mathbf{C}\mathbf{B}^T)}{\mathrm{pf}(\mathbf{A})} = \frac{\mathrm{pf}(\mathbf{C}^{-1}+\mathbf{B}^T\mathbf{A}^{-1}\mathbf{B})}{\mathrm{pf}(\mathbf{C}^{-1})} \tag{B.33}$$

Proof. One can directly verify that

$$\begin{aligned} &\begin{pmatrix} \mathbf{1} & \mathbf{B}\mathbf{C} \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \begin{pmatrix} \mathbf{A}+\mathbf{B}\mathbf{C}\mathbf{B}^T & \\ & \mathbf{C}^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ (\mathbf{B}\mathbf{C})^T & \mathbf{1} \end{pmatrix} \\ =&\begin{pmatrix} \mathbf{1} & \mathbf{0} \\ (\mathbf{A}^{-1}\mathbf{B})^T & \mathbf{1} \end{pmatrix} \begin{pmatrix} \mathbf{A} & \\ & \mathbf{C}^{-1}+\mathbf{B}^T\mathbf{A}^{-1}\mathbf{B} \end{pmatrix} \begin{pmatrix} \mathbf{1} & \mathbf{A}^{-1}\mathbf{B} \\ \mathbf{0} & \mathbf{1} \end{pmatrix}. \end{aligned} \tag{B.34}$$

Applying pfaffian on both sides and utilizing Eq. (B.29) and Eq. (B.32), we obtain Eq. (B.33). □

Theorem B.3.6. *Suppose*

$$\mathbf{A} = \begin{pmatrix} \mathbf{M} & \mathbf{Q} \\ -\mathbf{Q}^T & \mathbf{N} \end{pmatrix}, \tag{B.35}$$

where $\mathbf{M}$ *and* $\mathbf{N}$ *are invertible skew-symmetric matrices. Then*

$$\mathrm{pf}(\mathbf{A}) = \mathrm{pf}(\mathbf{M})\mathrm{pf}(\mathbf{N}+\mathbf{Q}^T\mathbf{M}^{-1}\mathbf{Q}) = \mathrm{pf}(\mathbf{N})\mathrm{pf}(\mathbf{M}+\mathbf{Q}\mathbf{N}^{-1}\mathbf{Q}^T). \tag{B.36}$$

Proof. It is direct to verify that

$$\begin{pmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{N}+\mathbf{Q}^T\mathbf{M}^{-1}\mathbf{Q} \end{pmatrix} = \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{Q}^T\mathbf{M}^{-1} & \mathbf{1} \end{pmatrix} \begin{pmatrix} \mathbf{M} & \mathbf{Q} \\ -\mathbf{Q}^T & \mathbf{N} \end{pmatrix} \begin{pmatrix} \mathbf{1} & -\mathbf{M}^{-1}\mathbf{Q} \\ \mathbf{0} & \mathbf{1} \end{pmatrix}. \tag{B.37}$$

Applying pfaffian on both sides and utilizing Eq. (B.29) and Eq. (B.32), we obtain $\mathrm{pf}(\mathbf{A}) = \mathrm{pf}(\mathbf{M})\mathrm{pf}(\mathbf{N}+\mathbf{Q}^T\mathbf{M}^{-1}\mathbf{Q})$. Similarly, $\mathrm{pf}(\mathbf{A}) = \mathrm{pf}(\mathbf{N})\mathrm{pf}(\mathbf{M}+\mathbf{Q}\mathbf{N}^{-1}\mathbf{Q}^T)$. □

Theorem B.3.7. *Suppose* $\mathbf{A}$ *is an* $n \times n$ *invertible skew-symmetric matrix with elements* A_{ij}*. Then*

$$\frac{\partial \mathrm{pf}(\mathbf{A})}{\partial A_{ij}} = \mathrm{pf}(\mathbf{A})(\mathbf{A}^{-1})_{ji}, \tag{B.38}$$

where we assume the variation happens at both A_{ij} *and* A_{ji} *due to the skew-symmetric constraint.*

Proof. Assume a variation is imposed at the i'th row and j'th column of $\mathbf{A}$, i.e.

$$(\mathrm{d}\mathbf{A})_{kl} = \delta_{ik}\delta jl\mathrm{d}x, \tag{B.39}$$

where $\mathrm{d}x$ is an infinitesimal scalar. Due to the skew-symmetric constraint, the variation of $\mathrm{pf}(\mathbf{A})$ is given by

$$\begin{aligned} \mathrm{pf}(\mathbf{A}+\mathrm{d}\mathbf{A}-\mathrm{d}\mathbf{A}^T) &= \mathrm{pf}[(\mathbf{1}_n+\mathbf{A}^{-1}\mathrm{d}\mathbf{A})^T\mathbf{A}(\mathbf{1}_n+\mathbf{A}^{-1}\mathrm{d}\mathbf{A})] \\ &= \mathrm{pf}(\mathbf{A})\det(\mathbf{1}_n+\mathbf{A}^{-1}\mathrm{d}\mathbf{A}) \\ &= \mathrm{pf}(\mathbf{A})(1+(\mathbf{A}^{-1})_{ji}\mathrm{d}x), \end{aligned} \tag{B.40}$$

where we have utilized Eq. (B.29), Eq. (B.22), and Eq. (B.23). Then the partial derivative is given by

$$\frac{\partial \mathrm{pf}(\mathbf{A})}{\partial A_{ij}} = \frac{\mathrm{pf}(\mathbf{A}+\mathrm{d}\mathbf{A}-\mathrm{d}\mathbf{A}^T)-\mathrm{pf}(\mathbf{A})}{\mathrm{d}x} = \mathrm{pf}(\mathbf{A})(\mathbf{A}^{-1})_{ji}. \tag{B.41}$$

□

B.4 Matrix inverse

Theorem B.4.1 (Shermann-Morrison formula)**.** *Suppose* $\mathbf{A}$ *is an* $n \times n$ *invertible matrix and* $\mathbf{u}, \mathbf{v}$ *are column vectors with length* n*. Then* $\mathbf{A} + \mathbf{u}\mathbf{v}^T$ *is invertible if and only if* $1 + \mathbf{v}^T\mathbf{A}\mathbf{u} \neq 0$*, in which case*

$$(\mathbf{A} + \mathbf{u}\mathbf{v}^T)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{u}\mathbf{v}^T\mathbf{A}^{-1}}{1 + \mathbf{v}^T\mathbf{A}\mathbf{u}}. \tag{B.42}$$

Proof. According to Eq. (B.14), $\det(\mathbf{A} + \mathbf{u}\mathbf{v}^T) = \det(\mathbf{A})(1 + \mathbf{v}^T\mathbf{A}\mathbf{u})$, so $\mathbf{A} + \mathbf{u}\mathbf{v}^T$ invertible $\Leftrightarrow \det(\mathbf{A} + \mathbf{u}\mathbf{v}^T) \neq 0 \Leftrightarrow 1 + \mathbf{v}^T\mathbf{A}\mathbf{u} \neq 0$, given $\det(\mathbf{A}) \neq 0$.

Eq. (B.42) can be proved by denoting its left side and right side as $\mathbf{X}$ and $\mathbf{Y}$ respectively. Then one can directly show that

$$\mathbf{X}\mathbf{Y} = \mathbf{Y}\mathbf{X} = \mathbf{1}_n. \tag{B.43}$$

□

As a generalization,

Theorem B.4.2 (Woodbury matrix identity)**.** *Suppose* $\mathbf{A}$ *is an* $n \times n$ *invertible matrix and* $\mathbf{U}, \mathbf{V}$ *are* $n \times k$ *matrices. Then* $\mathbf{A} + \mathbf{U}\mathbf{V}^T$ *is invertible if and only if* $\mathbf{1}_k + \mathbf{V}^T\mathbf{A}\mathbf{U}$ *is invertible, in which case*

$$(\mathbf{A} + \mathbf{U}\mathbf{V}^T)^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{1}_k + \mathbf{V}^T\mathbf{A}\mathbf{U})^{-1}\mathbf{V}^T\mathbf{A}^{-1}. \tag{B.44}$$

References

[1] L. Landau, "On the theory of the fermi liquid", Sov. Phys. JETP **8**, 70 (1959).

[2] L. D. Landau and E. M. Lifshitz, *Quantum mechanics: non-relativistic theory*, Vol. 3 (Elsevier, 2013).

[3] J. Bardeen, L. N. Cooper, and J. R. Schrieffer, "Microscopic theory of superconductivity", Phys. Rev. **106**, 162–164 (1957).

[4] P. W. Anderson, "More is different", Science **177**, 393–396 (1972).

[5] R. B. Laughlin, "Anomalous quantum hall effect: an incompressible quantum fluid with fractionally charged excitations", Phys. Rev. Lett. **50**, 1395–1398 (1983).

[6] P. Hohenberg and W. Kohn, "Inhomogeneous electron gas", Phys. Rev. **136**, B864–B871 (1964).

[7] D. Ceperley and B. Alder, "Quantum monte carlo", Science **231**, 555–560 (1986).

[8] M. Troyer and U.-J. Wiese, "Computational complexity and fundamental limitations to fermionic quantum monte carlo simulations", Phys. Rev. Lett. **94**, 170201 (2005).

[9] U. Schollwöck, "The density-matrix renormalization group in the age of matrix product states", Annals of Physics **326**, 96–192 (2011).

[10] F. Verstraete and J. I. Cirac, *Renormalization algorithms for quantum-many body systems in two and higher dimensions*, 2004.

[11] F. Mezzacapo, N. Schuch, M. Boninsegni, and J. I. Cirac, "Ground-state properties of quantum many-body systems: entangled-plaquette states and variational monte carlo", New Journal of Physics **11**, 083026 (2009).

[12] N. Schuch, M. M. Wolf, F. Verstraete, and J. I. Cirac, "Computational complexity of projected entangled pair states", Phys. Rev. Lett. **98**, 140506 (2007).

[13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", nature **521**, 436–444 (2015).

[14] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: a large-scale hierarchical image database", in 2009 ieee conference on computer vision and pattern recognition (2009), pp. 248–255.

[15] J. Sanchez and F. Perronnin, "High-dimensional signature compression for large-scale image classification", in Cvpr 2011 (2011), pp. 1665–1672.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", in Advances in neural information processing systems, Vol. 25, edited by F. Pereira, C. Burges, L. Bottou, and K. Weinberger (2012).

[17]D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search", Nature **529**, 484–489 (2016).

[18]D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge", Nature **550**, 354–359 (2017).

[19]G. Carleo and M. Troyer, "Solving the quantum many-body problem with artificial neural networks", Science **355**, 602–606 (2017).

[20]D.-L. Deng, X. Li, and S. Das Sarma, "Quantum entanglement in neural network states", Phys. Rev. X **7**, 021021 (2017).

[21]X. Gao and L.-M. Duan, "Efficient representation of quantum many-body states with deep neural networks", Nature Communications **8**, 662 (2017).

[22]G. Carleo, Y. Nomura, and M. Imada, "Constructing exact representations of quantum many-body systems with deep neural networks", Nature Communications **9**, 5322 (2018).

[23]I. Glasser, N. Pancotti, M. August, I. D. Rodriguez, and J. I. Cirac, "Neural-network quantum states, string-bond states, and chiral topological states", Phys. Rev. X **8**, 011006 (2018).

[24]J. Chen, S. Cheng, H. Xie, L. Wang, and T. Xiang, "Equivalence of restricted boltzmann machines and tensor network states", Phys. Rev. B **97**, 085104 (2018).

[25]Y. Levine, O. Sharir, N. Cohen, and A. Shashua, "Quantum entanglement in deep learning architectures", Phys. Rev. Lett. **122**, 065301 (2019).

[26]O. Sharir, A. Shashua, and G. Carleo, "Neural tensor contractions and the expressive power of deep neural quantum states", Phys. Rev. B **106**, 205136 (2022).

[27]D. Wu, R. Rossi, F. Vicentini, and G. Carleo, "From tensor-network quantum states to tensorial recurrent neural networks", Phys. Rev. Res. **5**, L032001 (2023).

[28]F. B. Trigueros, T. Mendes-Santos, and M. Heyl, "Simplicity of mean-field theories in neural quantum states", Phys. Rev. Res. **6**, 023261 (2024).

[29]C. Gauvin-Ndiaye, J. Tindall, J. R. Moreno, and A. Georges, "Mott transition and volume law entanglement with neural quantum states", Phys. Rev. Lett. **134**, 076502 (2025).

[30]Z. Denis, A. Sinibaldi, and G. Carleo, *Comment on "can neural quantum states learn volume-law ground states?"*, 2024.

[31]T.-H. Yang, M. Soleimanifar, T. Bergamaschi, and J. Preskill, *When can classical neural networks represent quantum states?*, 2024.

[32]Z.-A. Jia, B. Yi, R. Zhai, Y.-C. Wu, G.-C. Guo, and G.-P. Guo, "Quantum neural network states: a brief review of methods and applications", Advanced Quantum Technologies **2**, 1800077 (2019).

[33]J. Carrasquilla and G. Torlai, "How to use neural networks to investigate quantum many-body physics", PRX Quantum **2**, 040201 (2021).

[34]M. Medvidović and J. R. Moreno, "Neural-network quantum states for many-body physics", The European Physical Journal Plus **139**, 631 (2024).

[35]H. Lange, A. Van de Walle, A. Abedinnia, and A. Bohrdt, "From architectures to applications: a review of neural quantum states", Quantum Science and Technology **9**, 040501 (2024).

[36]J. Sevilla, L. Heim, A. Ho, T. Besiroglu, M. Hobbhahn, and P. Villalobos, "Compute trends across three eras of machine learning", in 2022 international joint conference on neural networks (ijcnn) (July 2022), pp. 1–8.

[37]W. L. McMillan, "Ground state of liquid He^4", Phys. Rev. **138**, A442–A451 (1965).

[38]P. A. Gagniuc, *Markov chains: from theory to implementation and experimentation* (John Wiley & Sons, 2017).

[39]N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines", The Journal of Chemical Physics **21**, 1087–1092 (1953).

[40]M. Schmitt and M. Reh, "jVMC: Versatile and performant variational Monte Carlo leveraging automated differentiation and GPU acceleration", SciPost Phys. Codebases, 2 (2022).

[41]R. Frostig, M. Johnson, and C. Leary, "Compiling machine learning programs via high-level tracing", in Systems for machine learning (2018).

[42]F. Vicentini, D. Hofmann, A. Szabó, D. Wu, C. Roth, C. Giuliani, G. Pescia, J. Nys, V. Vargas-Calderón, N. Astrakhantsev, and G. Carleo, "NetKet 3: Machine Learning Toolbox for Many-Body Quantum Systems", SciPost Phys. Codebases, 7 (2022).

[43]*Quantax: flexible neural quantum states based on quspin, jax, and equinox*, `https://github.com/ChenAo-Phys/quantax/tree/main`.

[44]C. Gros, "Criterion for a good variational wave function", Phys. Rev. B **42**, 6835–6838 (1990).

[45]D. Wu, R. Rossi, F. Vicentini, N. Astrakhantsev, F. Becca, X. Cao, J. Carrasquilla, F. Ferrari, A. Georges, M. Hibat-Allah, M. Imada, A. M. Läuchli, G. Mazzola, A. Mezzacapo, A. Millis, J. R. Moreno, T. Neupert, Y. Nomura, J. Nys, O. Parcollet, R. Pohle, I. Romero, M. Schmid, J. M. Silvester, S. Sorella, L. F. Tocchio, L. Wang, S. R. White, A. Wietek, Q. Yang, Y. Yang, S. Zhang, and G. Carleo, "Variational benchmarks for quantum many-body problems", Science **386**, 296–301 (2024).

[46]S. Sorella, "Generalized lanczos algorithm for variational quantum monte carlo", Phys. Rev. B **64**, 024512 (2001).

[47]W. Fu, W. Ren, and J. Chen, "Variance extrapolation method for neural-network variational monte carlo", Machine Learning: Science and Technology **5**, 015016 (2024).

[48]E. R. Gagliano, E. Dagotto, A. Moreo, and F. C. Alcaraz, "Correlation functions of the antiferromagnetic heisenberg model using a modified lanczos method", Phys. Rev. B **34**, 1677–1682 (1986).

[49]F. Becca, W.-J. Hu, Y. Iqbal, A. Parola, D. Poilblanc, and S. Sorella, "Lanczos steps to improve variational wave functions", Journal of Physics: Conference Series **640**, 012039 (2015).

[50]H. Chen, D. Hendry, P. Weinberg, and A. Feiguin, "Systematic improvement of neural network quantum states using lanczos", in Advances in neural information processing systems, Vol. 35, edited by S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (2022), pp. 7490–7503.

[51]J.-Q. Wang, R.-Q. He, and Z.-Y. Lu, *Generalized lanczos method for systematic optimization of neural-network quantum states*, 2025.

[52]S. Sorella, "Green function monte carlo with stochastic reconfiguration", Phys. Rev. Lett. **80**, 4558–4561 (1998).

[53]S. Sorella and L. Capriotti, "Green function monte carlo with stochastic reconfiguration: an effective remedy for the sign problem", Phys. Rev. B **61**, 2599–2612 (2000).

[54]S. Sorella, "Wave function optimization in the variational monte carlo method", Phys. Rev. B **71**, 241103 (2005).

[55]S. Sorella, M. Casula, and D. Rocca, "Weak binding between two aromatic rings: feeling the van der waals attraction by quantum monte carlo methods", The Journal of Chemical Physics **127**, 014105 (2007).

[56]S.-i. Amari, "Natural gradient works efficiently in learning", Neural Computation **10**, 251–276 (1998).

[57]J. Stokes, J. Izaac, N. Killoran, and G. Carleo, "Quantum Natural Gradient", Quantum **4**, 269 (2020).

[58]A. Chen and M. Heyl, "Empowering deep neural quantum states through efficient optimization", Nature Physics **20**, 1476–1481 (2024).

[59]A. Jacot, F. Gabriel, and C. Hongler, "Neural tangent kernel: convergence and generalization in neural networks", in Advances in neural information processing systems, Vol. 31, edited by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (2018).

[60]M. R. Hestenes, E. Stiefel, et al., *Methods of conjugate gradients for solving linear systems*, Vol. 49, 1 (NBS Washington, DC, 1952).

[61]G. Goldshlager, N. Abrahamsen, and L. Lin, "A kaczmarz-inspired approach to accelerate the optimization of neural network wavefunctions", Journal of Computational Physics **516**, 113351 (2024).

[62]R. Rende, L. L. Viteritti, L. Bardone, F. Becca, and S. Goldt, "A simple linear algebra identity to optimize large-scale neural network quantum states", Communications Physics **7**, 260 (2024).

[63]S. Karczmarz, "Angenaherte auflosung von systemen linearer glei-chungen", Bull. Int. Acad. Pol. Sic. Let., Cl. Sci. Math. Nat., 355–357 (1937).

[64]P. Kramer and M. Saraceno, *Geometry of the time-dependent variational principle in quantum mechanics* (Springer, 1981).

[65]J. Haegeman, J. I. Cirac, T. J. Osborne, I. Pi žorn, H. Verschelde, and F. Verstraete, "Time-dependent variational principle for quantum lattices", Phys. Rev. Lett. **107**, 070601 (2011).

[66]J. Broeckhove, L. Lathouwers, E. Kesteloot, and P. Van Leuven, "On the equivalence of time-dependent variational principles", Chemical Physics Letters **149**, 547–550 (1988).

[67]M. Schmitt and M. Heyl, "Quantum many-body dynamics in two dimensions with artificial neural networks", Phys. Rev. Lett. **125**, 100503 (2020).

[68]M. Schmitt, M. M. Rams, J. Dziarmaga, M. Heyl, and W. H. Zurek, "Quantum phase transition dynamics in the two-dimensional transverse-field ising model", Science Advances **8**, eabl6850 (2022).

[69]T. Mendes-Santos, M. Schmitt, and M. Heyl, "Highly resolved spectral functions of two-dimensional systems with neural quantum states", Phys. Rev. Lett. **131**, 046501 (2023).

[70]M. Medvidovi ć and D. Sels, "Variational quantum dynamics of two-dimensional rotor models", PRX Quantum **4**, 040302 (2023).

[71]J. Nys, G. Pescia, A. Sinibaldi, and G. Carleo, "Ab-initio variational wave functions for the time-dependent many-electron schrödinger equation", Nature Communications **15**, 9404 (2024).

[72]R. Novak, J. Sohl-Dickstein, and S. S. Schoenholz, "Fast finite width neural tangent kernel", in Proceedings of the 39th international conference on machine learning, Vol. 162, edited by K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Proceedings of Machine Learning Research (July 2022), pp. 17018–17044.

[73]P. Weinberg and M. Bukov, "QuSpin: a Python package for dynamics and exact diagonalisation of quantum many body systems part I: spin chains", SciPost Phys. **2**, 003 (2017).

[74]P. Weinberg and M. Bukov, "QuSpin: a Python package for dynamics and exact diagonalisation of quantum many body systems. Part II: bosons, fermions and higher spins", SciPost Phys. **7**, 020 (2019).

[75]T. Vieijra, C. Casert, J. Nys, W. De Neve, J. Haegeman, J. Ryckebusch, and F. Verstraete, "Restricted boltzmann machines for quantum states with non-abelian or anyonic symmetries", Phys. Rev. Lett. **124**, 097201 (2020).

[76]T. Vieijra and J. Nys, "Many-body quantum states with exact conservation of non-abelian and lattice symmetries through variational monte carlo", Phys. Rev. B **104**, 045123 (2021).

[77]D. Luo, G. Carleo, B. K. Clark, and J. Stokes, "Gauge equivariant neural networks for quantum lattice gauge theories", Phys. Rev. Lett. **127**, 276402 (2021).

[78]D. Luo, Z. Chen, K. Hu, Z. Zhao, V. M. Hur, and B. K. Clark, "Gauge-invariant and anyonic-symmetric autoregressive neural network for quantum lattice models", Phys. Rev. Res. **5**, 013216 (2023).

[79]D. Tahara and M. Imada, "Variational monte carlo method combined with quantum-number projection and multi-variable optimization", Journal of the Physical Society of Japan **77**, 114701 (2008).

[80]K. Choo, G. Carleo, N. Regnault, and T. Neupert, "Symmetries and many-body excitations with neural-network quantum states", Phys. Rev. Lett. **121**, 167204 (2018).

[81]Y. Nomura, "Helping restricted boltzmann machines with quantum-state representation by restoring symmetry", Journal of Physics: Condensed Matter **33**, 174003 (2021).

[82]M. Reh, M. Schmitt, and M. Gärttner, "Optimizing design choices for neural quantum states", Phys. Rev. B **107**, 195115 (2023).

[83]X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks", in Proceedings of the fourteenth international conference on artificial intelligence and statistics (2011), pp. 315–323.

[84]D. Hendrycks and K. Gimpel, *Gaussian error linear units (gelus)*, 2023.

[85]C. B. Csáji, "Approximation with artificial neural networks", MA thesis (Eötvös Loránd University, 2001).

[86]Z. Cai and J. Liu, "Approximating quantum many-body wave functions using artificial neural networks", Phys. Rev. B **97**, 035116 (2018).

[87]K. Çeven, M. Ö. Oktel, and A. Kele ş, "Neural-network quantum states for a two-leg bose-hubbard ladder under magnetic flux", Phys. Rev. A **106**, 063320 (2022).

[88]Z. Zhu, M. Mattheakis, W. Pan, and E. Kaxiras, "Hubbardnet: efficient predictions of the bose-hubbard model spectrum with deep neural networks", Phys. Rev. Res. **5**, 043084 (2023).

[89]A. W. Sandvik, "Finite-size scaling of the ground-state parameters of the two-dimensional heisenberg model", Phys. Rev. B **56**, 11678–11690 (1997).

[90]P. Smolensky, *Information processing in dynamical systems: foundations of harmony theory*, tech. rep. (Colorado Univ at Boulder Dept of Computer Science, 1986).

[91]R. G. Melko, G. Carleo, J. Carrasquilla, and J. I. Cirac, "Restricted boltzmann machines in quantum physics", Nature Physics **15**, 887–892 (2019).

[92]D.-L. Deng, X. Li, and S. Das Sarma, "Machine learning topological states", Phys. Rev. B **96**, 195145 (2017).

[93]H. Saito, "Solving the bose–hubbard model with machine learning", Journal of the Physical Society of Japan **86**, 093001 (2017).

[94]R. Kaubruegger, L. Pastori, and J. C. Budich, "Chiral topological phases from artificial neural networks", Phys. Rev. B **97**, 195136 (2018).

[95]G. Torlai, G. Mazzola, J. Carrasquilla, M. Troyer, R. Melko, and G. Carleo, "Neural-network quantum state tomography", Nature Physics **14**, 447–450 (2018).

[96]S. Lu, X. Gao, and L.-M. Duan, "Efficient representation of topologically ordered states with restricted boltzmann machines", Phys. Rev. B **99**, 155136 (2019).

[97]K. McBrian, G. Carleo, and E. Khatami, "Ground state phase diagram of the one-dimensional bose-hubbard model from restricted boltzmann machines", Journal of Physics: Conference Series **1290**, 012005 (2019).

[98]V. Vargas-Calderón, H. Vinck-Posada, and F. A. González, "Phase diagram reconstruction of the bose–hubbard model with a restricted boltzmann machine wavefunction", Journal of the Physical Society of Japan **89**, 094002 (2020).

[99]C.-X. Li, S. Yang, and J.-B. Xu, "Learning spin liquids on a honeycomb lattice with artificial neural networks", Scientific Reports **11**, 16667 (2021).

[100]A. Valenti, E. Greplova, N. H. Lindner, and S. D. Huber, "Correlation-enhanced neural networks as interpretable variational quantum states", Phys. Rev. Res. **4**, L012010 (2022).

[101]Y. Nomura, "Investigating network parameters in neural-network quantum states", Journal of the Physical Society of Japan **91**, 054709 (2022).

[102]M. Lubasch, J. I. Cirac, and M.-C. Bañuls, "Algorithms for finite projected entangled pair states", Phys. Rev. B **90**, 064425 (2014).

[103]V. Dumoulin and F. Visin, *A guide to convolution arithmetic for deep learning*, 2016.

[104]K. Choo, T. Neupert, and G. Carleo, "Two-dimensional frustrated J_1-J_2 model studied with neural network quantum states", Phys. Rev. B **100**, 125124 (2019).

[105]X. Liang, W.-Y. Liu, P.-Z. Lin, G.-C. Guo, Y.-S. Zhang, and L. He, "Solving frustrated quantum many-particle models with convolutional neural networks", Phys. Rev. B **98**, 104426 (2018).

[106]N. Astrakhantsev, T. Westerhout, A. Tiwari, K. Choo, A. Chen, M. H. Fischer, G. Carleo, and T. Neupert, "Broken-symmetry ground states of the heisenberg model on the pyrochlore lattice", Phys. Rev. X **11**, 041021 (2021).

[107]X. Liang, M. Li, Q. Xiao, J. Chen, C. Yang, H. An, and L. He, "Deep learning representations for quantum many-body systems on heterogeneous hardware", Machine Learning: Science and Technology **4**, 015035 (2023).

[108]C. Fu, X. Zhang, H. Zhang, H. Ling, S. Xu, and S. Ji, "Lattice convolutional networks for learning ground states of quantum many-body systems", in *Proceedings of the 2024 siam international conference on data mining (sdm)* (), pp. 490–498.

[109]J.-Q. Wang, H.-Q. Wu, R.-Q. He, and Z.-Y. Lu, "Variational optimization of the amplitude of neural-network quantum many-body ground states", Phys. Rev. B **109**, 245120 (2024).

[110]K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition", in Proceedings of the ieee conference on computer vision and pattern recognition (cvpr) (June 2016).

[111]K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks", in Computer vision – eccv 2016, edited by B. Leibe, J. Matas, N. Sebe, and M. Welling (2016), pp. 630–645.

[112]C. Roth, A. Szabó, and A. H. MacDonald, "High-accuracy variational monte carlo for frustrated magnets with deep neural networks", Phys. Rev. B **108**, 054410 (2023).

[113]J. Beck, J. Bodky, J. Motruk, T. Müller, R. Thomale, and P. Ghosh, "Phase diagram of the $J-J_d$ heisenberg model on the maple leaf lattice: neural networks and density matrix renormalization group", Phys. Rev. B **109**, 184422 (2024).

[114]T. Duric, J. H. Chung, B. Yang, and P. Sengupta, *Spin-1/2 kagome heisenberg antiferromagnet: machine learning discovery of the spinon pair density wave ground state*, 2024.

[115]A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need", in Advances in neural information processing systems, Vol. 30, edited by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (2017).

[116]A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: transformers for image recognition at scale", in International conference on learning representations (2021).

[117]L. L. Viteritti, R. Rende, A. Parola, S. Goldt, and F. Becca, *Transformer wave function for two dimensional frustrated magnets: emergence of a spin-liquid phase in the shastry-sutherland model*, 2024.

[118]L. L. Viteritti, R. Rende, and F. Becca, "Transformer variational wave functions for frustrated quantum spin systems", Phys. Rev. Lett. **130**, 236401 (2023).

[119]K. Wu, H. Peng, M. Chen, J. Fu, and H. Chao, "Rethinking and improving relative position encoding for vision transformer", in Proceedings of the ieee/cvf international conference on computer vision (iccv) (Oct. 2021), pp. 10033–10041.

[120]R. Rende, F. Gerace, A. Laio, and S. Goldt, "Mapping of attention mechanisms to a generalized potts model", Phys. Rev. Res. **6**, 023057 (2024).

[121]R. Rende, S. Goldt, F. Becca, and L. L. Viteritti, "Fine-tuning neural network quantum states", Phys. Rev. Res. **6**, 043280 (2024).

[122]S. Roca-Jerat, M. Gallego, F. Luis, J. Carrete, and D. Zueco, "Transformer wave function for quantum long-range models", Phys. Rev. B **110**, 205147 (2024).

[123]R. Rende and L. Loris Viteritti, "Are queries and keys always relevant? a case study on transformer wave functions", Machine Learning: Science and Technology **6**, 010501 (2025).

[124]V. Herráiz-López, S. Roca-Jerat, M. Gallego, R. Ferrández, J. Carrete, D. Zueco, and J. Román-Roche, *First- and second-order quantum phase transitions in the long-range unfrustrated antiferromagnetic ising chain*, 2024.

[125]A. Chen, V. D. Naik, and M. Heyl, *Convolutional transformer wave functions*, 2025.

[126]H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang, "Cvt: introducing convolutions to vision transformers", in Proceedings of the ieee/cvf international conference on computer vision (iccv) (Oct. 2021), pp. 22–31.

[127]J. Guo, K. Han, H. Wu, Y. Tang, X. Chen, Y. Wang, and C. Xu, "Cmt: convolutional neural networks meet vision transformers", in Proceedings of the ieee/cvf conference on computer vision and pattern recognition (cvpr) (June 2022), pp. 12175–12185.

[128]X. Cao, Z. Zhong, and Y. Lu, *Vision transformer neural quantum states for impurity models*, 2024.

[129]A. Chen, K. Choo, N. Astrakhantsev, and T. Neupert, "Neural network evolution strategy for solving quantum sign structures", Phys. Rev. Res. **4**, L022026 (2022).

[130]A. Szabó and C. Castelnovo, "Neural network wave functions and the sign problem", Phys. Rev. Res. **2**, 033075 (2020).

[131]T. Westerhout, N. Astrakhantsev, K. S. Tikhonov, M. I. Katsnelson, and A. A. Bagrov, "Generalization properties of neural network approximations to frustrated magnet ground states", Nature Communications **11**, 1593 (2020).

[132]M. Hibat-Allah, M. Ganahl, L. E. Hayward, R. G. Melko, and J. Carrasquilla, "Recurrent neural network wave functions", Phys. Rev. Res. **2**, 023358 (2020).

[133]O. Sharir, Y. Levine, N. Wies, G. Carleo, and A. Shashua, "Deep autoregressive models for the efficient variational simulation of many-body quantum systems", Phys. Rev. Lett. **124**, 020503 (2020).

[134]C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal, *Deep complex networks*, 2018.

[135]P. Anderson, "Resonating valence bonds: a new kind of insulator?", Materials Research Bulletin **8**, 153–160 (1973).

[136]L. Savary and L. Balents, "Quantum spin liquids: a review", Reports on Progress in Physics **80**, 016502 (2016).

[137]P. W. Anderson, "The resonating valence bond state in la2cuo4 and superconductivity", science **235**, 1196–1198 (1987).

[138]A. Y. Kitaev, "Quantum error correction with imperfect gates", in *Quantum communication, computing, and measurement*, edited by O. Hirota, A. S. Holevo, and C. M. Caves (Springer US, Boston, MA, 1997), pp. 181–188.

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

[140]S.-S. Gong, W. Zhu, D. N. Sheng, O. I. Motrunich, and M. P. A. Fisher, "Plaquette ordered phase and quantum phase diagram in the spin-$\frac{1}{2}$ J_1-J_2 square heisenberg model", Phys. Rev. Lett. **113**, 027201 (2014).

[141]W.-J. Hu, F. Becca, A. Parola, and S. Sorella, "Direct evidence for a gapless Z_2 spin liquid by frustrating néel antiferromagnetism", Phys. Rev. B **88**, 060402 (2013).

[142]Y. Nomura and M. Imada, "Dirac-type nodal spin liquid revealed by refined quantum many-body solver using neural-network wave function, correlation ratio, and level spectroscopy", Phys. Rev. X **11**, 031034 (2021).

[143]H.-C. Jiang, H. Yao, and L. Balents, "Spin liquid ground state of the spin-$\frac{1}{2}$ square J_1-J_2 heisenberg model", Phys. Rev. B **86**, 024424 (2012).

[144]L. Wang, D. Poilblanc, Z.-C. Gu, X.-G. Wen, and F. Verstraete, "Constructing a gapless spin-liquid state for the spin-1/2 $J_1 - J_2$ heisenberg model on a square lattice", Phys. Rev. Lett. **111**, 037202 (2013).

[145]L. Wang and A. W. Sandvik, "Critical level crossings and gapless spin liquid in the square-lattice spin-1/2 $J_1 - J_2$ heisenberg antiferromagnet", Phys. Rev. Lett. **121**, 107202 (2018).

[146]W.-Y. Liu, S. Dong, C. Wang, Y. Han, H. An, G.-C. Guo, and L. He, "Gapless spin liquid ground state of the spin-$\frac{1}{2}$ $J_1 - J_2$ heisenberg model on square lattices", Phys. Rev. B **98**, 241109 (2018).

[147]F. Ferrari and F. Becca, "Gapless spin liquid and valence-bond solid in the J_1-J_2 heisenberg model on the square lattice: insights from singlet and triplet excitations", Phys. Rev. B **102**, 014417 (2020).

[148]W.-Y. Liu, S.-S. Gong, Y.-B. Li, D. Poilblanc, W.-Q. Chen, and Z.-C. Gu, "Gapless quantum spin liquid and global phase diagram of the spin-1/2 j1-j2 square antiferromagnetic heisenberg model", Science Bulletin **67**, 1034–1041 (2022).

[149]L. Wang, Z.-C. Gu, F. Verstraete, and X.-G. Wen, "Tensor-product state approach to spin-$\frac{1}{2}$ square J_1-J_2 antiferromagnetic heisenberg model: evidence for deconfined quantum criticality", Phys. Rev. B **94**, 075143 (2016).

[150]R. Haghshenas and D. N. Sheng, "$U(1)$-symmetric infinite projected entangled-pair states study of the spin-1/2 square J_1-J_2 heisenberg model", Phys. Rev. B **97**, 174408 (2018).

[151]X. Qian and M. Qin, "Absence of spin liquid phase in the $J_1 - J_2$ heisenberg model on the square lattice", Phys. Rev. B **109**, L161103 (2024).

[152]X. Liang, S.-J. Dong, and L. He, "Hybrid convolutional neural network and projected entangled pair states wave functions for quantum many-particle states", Phys. Rev. B **103**, 035138 (2021).

[153]F. Ferrari, F. Becca, and J. Carrasquilla, "Neural gutzwiller-projected variational wave functions", Phys. Rev. B **100**, 125131 (2019).

[154]Y. Iqbal, W.-J. Hu, R. Thomale, D. Poilblanc, and F. Becca, "Spin liquid nature in the heisenberg $J_1 - J_2$ triangular antiferromagnet", Phys. Rev. B **93**, 144411 (2016).

[155]A. O. Scheie, M. Lee, K. Wang, P. Laurell, E. S. Choi, D. Pajerowski, Q. Zhang, J. Ma, H. D. Zhou, S. Lee, S. M. Thomas, M. O. Ajeesh, P. F. S. Rosa, A. Chen, V. S. Zapf, M. Heyl, C. D. Batista, E. Dagotto, J. E. Moore, and D. A. Tennant, *Spectrum and low-energy gap in triangular quantum spin liquid naybse*$_2$, 2024.

[156]M. Calandra Buonaura and S. Sorella, "Numerical study of the two-dimensional heisenberg model using a green function monte carlo technique with a fixed number of walkers", Phys. Rev. B **57**, 11446–11456 (1998).

[157]R. K. Kaul, "Spin nematics, valence-bond solids, and spin liquids in SO(N) quantum spin models on the triangular lattice", Phys. Rev. Lett. **115**, 157202 (2015).

[158]S. Pujari, T. C. Lang, G. Murthy, and R. K. Kaul, "Interaction-induced dirac fermions from quadratic band touching in bilayer graphene", Phys. Rev. Lett. **117**, 086404 (2016).

[159]R. Kaneko, S. Morita, and M. Imada, "Gapless spin-liquid phase in an extended spin 1/2 triangular heisenberg model", Journal of the Physical Society of Japan **83**, 093707 (2014).

[160]W.-J. Hu, S.-S. Gong, W. Zhu, and D. N. Sheng, "Competing spin-liquid states in the spin-$\frac{1}{2}$ heisenberg model on the triangular lattice", Phys. Rev. B **92**, 140403 (2015).

[161]Z. Zhu and S. R. White, "Spin liquid phase of the $S = \frac{1}{2}$ $J_1 - J_2$ heisenberg model on the triangular lattice", Phys. Rev. B **92**, 041105 (2015).

[162]S.-S. Gong, W. Zhu, J.-X. Zhu, D. N. Sheng, and K. Yang, "Global phase diagram and quantum spin liquids in a spin-$\frac{1}{2}$ triangular antiferromagnet", Phys. Rev. B **96**, 075116 (2017).

[163]A. Wietek and A. M. Läuchli, "Chiral spin liquid and quantum criticality in extended $S = \frac{1}{2}$ heisenberg models on the triangular lattice", Phys. Rev. B **95**, 035141 (2017).

[164]H.-Q. Wu, S.-S. Gong, and D. N. Sheng, "Randomness-induced spin-liquid-like phase in the spin-$\frac{1}{2}$ $J_1 - J_2$ triangular heisenberg model", Phys. Rev. B **99**, 085141 (2019).

[165]S. R. White, "Spin gaps in a frustrated heisenberg model for CaV_4O_9", Phys. Rev. Lett. **77**, 3633–3636 (1996).

[166]G. Santoro, S. Sorella, L. Guidoni, A. Parola, and E. Tosatti, "Spin-liquid ground state in a two-dimensional nonfrustrated spin model", Phys. Rev. Lett. **83**, 3065–3068 (1999).

[167]S. Morita, R. Kaneko, and M. Imada, "Quantum spin liquid in spin 1/2 j1–j2 heisenberg model on square lattice: many-variable variational monte carlo study combined with quantum-number projections", Journal of the Physical Society of Japan **84**, 024720 (2015).

[168]Bouchaud, J.P., Georges, A., and Lhuillier, C., "Pair wave functions for strongly correlated fermions and their determinantal representation", J. Phys. France **49**, 553–559 (1988).

[169]F. Becca and S. Sorella, "Variational monte carlo", in *Quantum monte carlo approaches for correlated systems* (Cambridge University Press, 2017), pp. 103–130.

[170]T. Misawa, S. Morita, K. Yoshimi, M. Kawamura, Y. Motoyama, K. Ido, T. Ohgoe, M. Imada, and T. Kato, "Mvmc—open-source software for many-variable variational monte carlo method", Computer Physics Communications **235**, 447–462 (2019).

[171]K. Inui, Y. Kato, and Y. Motome, "Determinant-free fermionic wave function using feed-forward neural networks", Phys. Rev. Res. **3**, 043126 (2021).

[172]E. Ibarra-García-Padilla, H. Lange, R. G. Melko, R. T. Scalettar, J. Carrasquilla, A. Bohrdt, and E. Khatami, "Autoregressive neural quantum states of fermi hubbard models", Phys. Rev. Res. **7**, 013122 (2025).

[173]Y. Nomura, A. S. Darmawan, Y. Yamaji, and M. Imada, "Restricted boltzmann machine learning for solving strongly correlated quantum systems", Phys. Rev. B **96**, 205152 (2017).

[174]D. Luo and B. K. Clark, "Backflow transformations via neural networks for quantum many-body wave functions", Phys. Rev. Lett. **122**, 226401 (2019).

[175]I. Romero, J. Nys, and G. Carleo, "Spectroscopy of two-dimensional interacting lattice electrons using symmetry-aware neural backflow transformations", Communications Physics **8**, 46 (2025).

[176]J. R. Moreno, G. Carleo, A. Georges, and J. Stokes, "Fermionic wave functions from neural-network constrained hidden states", Proceedings of the National Academy of Sciences **119**, e2122059119 (2022).

[177]Z. Liu and B. K. Clark, "Unifying view of fermionic neural network quantum states: from neural network backflow to hidden fermion determinant states", Phys. Rev. B **110**, 115124 (2024).

[178]M. Qin, T. Schäfer, S. Andergassen, P. Corboz, and E. Gull, "The hubbard model: a computational perspective", Annual Review of Condensed Matter Physics **13**, 275–302 (2022).

[179]J. Hubbard and B. H. Flowers, "Electron correlations in narrow energy bands", Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences **276**, 238–257 (1963).

[180]M. C. Gutzwiller, "Effect of correlation on the ferromagnetism of transition metals", Phys. Rev. Lett. **10**, 159–162 (1963).

[181]J. Kanamori, "Electron correlation and ferromagnetism of transition metals", Progress of Theoretical Physics **30**, 275–289 (1963).

[182]P. A. Lee, N. Nagaosa, and X.-G. Wen, "Doping a mott insulator: physics of high-temperature superconductivity", Rev. Mod. Phys. **78**, 17–85 (2006).

[183]J. G. Bednorz and K. A. Müller, "Possible hightc superconductivity in the ba-la-cu-o system", Zeitschrift für Physik B Condensed Matter **64**, 189–193 (1986).

[184]M. K. Wu, J. R. Ashburn, C. J. Torng, P. H. Hor, R. L. Meng, L. Gao, Z. J. Huang, Y. Q. Wang, and C. W. Chu, "Superconductivity at 93 k in a new mixed-phase y-ba-cu-o compound system at ambient pressure", Phys. Rev. Lett. **58**, 908–910 (1987).

[185]H. Xu, C.-M. Chung, M. Qin, U. Schollwöck, S. R. White, and S. Zhang, "Coexistence of superconductivity with partially filled stripes in the hubbard model", Science **384**, eadh7691 (2024).

[186]M. Qin, C.-M. Chung, H. Shi, E. Vitali, C. Hubig, U. Schollwöck, S. R. White, and S. Zhang (Simons Collaboration on the Many-Electron Problem), "Absence of superconductivity in the pure two-dimensional hubbard model", Phys. Rev. X **10**, 031016 (2020).

[187]Y.-T. Zhou, Z.-W. Zhou, and X. Liang, "Solving fermi-hubbard-type models by tensor representations of backflow corrections", Phys. Rev. B **109**, 245107 (2024).

[188]M. Qin, H. Shi, and S. Zhang, "Benchmark study of the two-dimensional hubbard model with auxiliary-field quantum monte carlo method", Phys. Rev. B **94**, 085103 (2016).

[189]A. Georges, G. Kotliar, W. Krauth, and M. J. Rozenberg, "Dynamical mean-field theory of strongly correlated fermion systems and the limit of infinite dimensions", Rev. Mod. Phys. **68**, 13–125 (1996).

[190]S.-S. B. Lee, J. von Delft, and A. Weichselbaum, "Doublon-holon origin of the subpeaks at the hubbard band edges", Phys. Rev. Lett. **119**, 236402 (2017).

[191]A. Nagy and V. Savona, "Variational quantum monte carlo method with a neural-network ansatz for open quantum systems", Phys. Rev. Lett. **122**, 250501 (2019).

[192]M. J. Hartmann and G. Carleo, "Neural-network approach to dissipative quantum many-body dynamics", Phys. Rev. Lett. **122**, 250502 (2019).

[193]F. Vicentini, A. Biella, N. Regnault, and C. Ciuti, "Variational neural-network ansatz for steady states in open quantum systems", Phys. Rev. Lett. **122**, 250503 (2019).

[194]J. Hermann, J. Spencer, K. Choo, A. Mezzacapo, W. M. C. Foulkes, D. Pfau, G. Carleo, and F. Noé, "Ab initio quantum chemistry with neural-network wavefunctions", Nature Reviews Chemistry **7**, 692–709 (2023).

[195]K. Choo, A. Mezzacapo, and G. Carleo, "Fermionic neural-network states for ab-initio electronic structure", Nature Communications **11**, 2368 (2020).

[196]D. Pfau, J. S. Spencer, A. G. D. G. Matthews, and W. M. C. Foulkes, "Ab initio solution of the many-electron schrödinger equation with deep neural networks", Phys. Rev. Res. **2**, 033429 (2020).

[197]J. Hermann, Z. Schätzle, and F. Noé, "Deep-neural-network solution of the electronic schrödinger equation", Nature Chemistry **12**, 891–897 (2020).

[198]D. Pfau, S. Axelrod, H. Sutterud, I. von Glehn, and J. S. Spencer, "Accurate computation of quantum excited states with neural networks", Science **385**, eadn0137 (2024).

[199]R. Peng and G. K.-L. Chan, *An analysis of first- and second-order optimization algorithms in variational monte carlo*, 2025.

[200]K. Kreutz-Delgado, *The complex gradient operator and the cr-calculus*, 2009.

[201]Wikipedia, *Pfaffian*, `https://en.wikipedia.org/wiki/Pfaffian`.