國立台灣大學理學院物理學系

碩士論文

Department of Physics

College of Science

National Taiwan University

Master Thesis

比較不同張量網路演算法應用在二微多體量子物理系統之優劣

Comparison between Tensor Network Algorithms for Two Dimensional Infinite Quantum Many-Body Systems

周昀萱

Yun-Hsuan Chou

指導教授：高英哲博士

Advisor: Ying-Jer Kao, Ph.D.

中華民國 105 年 4 月

April, 2016

誌謝

隨著時光流逝，終於完成了我在物理系的碩班生涯。比起大學在材料系的無聊日子，這三年的經歷讓我感到彌足珍貴。

首先我要感謝高英哲老師的指導與包容。讓比較晚才理解並進入狀況的我也能有機會學習現在十分流行的 Tensor Network 演算法並參與開發 Uni10 的工作，讓我了解在設計 CPU 與 GPU 程式時該有的相關資訊與技巧。並在我研究十分掙扎時給予我研究的方向與建議，也讓我有許多機會與其他學者討論以克服問題，對於我的研究有了長足的幫助。

再來是要感謝組上的同學。其中特別感謝謝昀達學長在他繁忙日程裡，仍撥空指導一個剛開始不怎麼會寫 c/c++ 的菜鳥，並讓我對 Tensor Network 相關的演算法有更進一步的認識。再來我想感謝從未謀面的張學文學長，許許多多研究上的問題, 都可以在神秘的玩貝資料夾中得到答案或起發。還要感謝感謝楊淵棨學長、吳柏寬學長、郭子傑學長、李致遠學長、高文瀚、林育平、易德、吳凱析對於我研究與課業上的種種幫助。其中特別感謝吳柏寬學長與林育平，除了對於我在物理理論、學業和娛樂上的幫忙外，也讓本應因研究卡關而在研究室崩潰的夜晚變得十分熱鬧充滿活力，並授與了我二階張亮黑魔導的頭銜。

最後，十分想感謝我的父母。不論我的選擇結果是好與壞、風險高或低，總是不斷地給與我精神上的鼓勵與物質上的支持，感謝你們的包容，才我能讓我毫無顧慮、充實的過完我碩班的時光，體驗著不一樣的人生。

摘要

如何判斷多體量子系統的相變化，且從微觀系統來得到巨觀上的物理性質，在現代仍為凝態物理學中十分有趣的領域。

從 NRG 的為起點開始，多年來出現了許多突破性的演算法。其中 DMRG 在一維的系統的模擬中得到了相當好的結果。但在二微系統中，因為 Area law 的關係使其表現不如在一微系統中精確，不僅如此，在二維系統中，計算複雜度上升之速度也非一維系統可比擬。為了解決這些問題，因而出現了許許多多不同的建立在張亮網路理論上的演算法。

此篇論文，紀錄了幾個當今較為主流或新穎並用以模疑二維量子系統的張亮網路演算法。一開始將簡單解釋張亮網路的基本理論; 再來會介紹如何實做、優化演算法，以增加精確度和降低計算複雜度。章節中也附上偽代碼，來說明實作中應注意之細節。最後會比較它們計算二維易辛模型與海森堡模型的結果，來說明各演算法之優缺點。

Abstract

Determining the phase transition of many body systems and the physical properties of macroscopic systems from microscopic description are still challenging in condense matter physics.

Since the numerical renormalization group (NRG) came out, various algorithms sprang up like mushrooms for analyzing these problems . Among all, the density matrix renormalization group (DMRG) could be considered as the most remarkable outcome, which analyze accurately in one dimensional systems. However, it perform worse in two dimensional systems. Not only the physical reasons, such as the area law, but also the rapid increment of computational complexity which is much higher than in one dimensional systems.

In order to study the phenomenals in twe-dimensional systems. First of all, we briefly introduce the tensor network theory. Secondly, we recorded some of popular tensor network algorithms which are developed for handling the problems in two-dimensional systems. Furthermore, the network diagrams and pseudo-code are presented, which gives the instruction of how to implement these algorithms.

Key words— matrix product state(MPS), projected entangled pair state(PEPS), projected entangled simplex state, infinite time-evolveing block-decimation, corner transfer matrix, tensor renormalization group, Benchmarks, uni10.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Understanding the phenomena of quantum many-body systems is one of the most challenging problems in condensed matter physics due to the coefficients required for describing entire systems grows exponentially with system size. For instance, If we desired to fully describe a N-site spin chain and each spin has d probable states, the requirement of coefficients is d^N. As a result, it is impossible to simulate the system whose size is larger than 50 for classical computers due to the rapid increment of computational consumptions.

Various numerical methods have been developed to address the problems mentioned above. For instance, density matrix renormalization group (DMRG) [1] [2] acquire the accuracy ground state energy and provide a dominant tool to study properties of one dimensional systems. Furthermore, the theory of DMRG is related with matrix product states (MPS) [3] [4] , which could describe the wave-function of one dimensional system and be explicitly represented by *tensor diagrams*. Therefore, DMRG is recognized as the one of the most successful method in one-dimensional systems. However, it's failure in higher dimensional systems due to the insufficiency of dealing with the entanglement in systems from matrix product states. For two-dimensional lattices, the projected entangled pair state (PEPS) [5] [6] has been applied to deal with that problem and many algorithms sprang up like mushrooms, such as time-evolving block decimation [7] [8] and multiscale entanglement renormalization ansatz [9].

Chapter 2

Tensor Network Theory

This section begins from a fundamental question: How to draw a tensor network diagrams? In tensor network theory [10] [11] [12], we are used to represent tensors as the notations, shown in Fig. 2.1, because *tensor diagrams* can fully map to quantum states of any geometric lattice systems explicitly. Furthermore, base on its clear representation, the implementation of tensor network algorithms become simply.

2.1 Representation of tensors in tensor Networks

In mathematical concept, a tensor is considered as a multi-dimensional array of scalers. The arrangement of the elements in a tensor is dependent on its *indices* and the *rank* of a tensor is equivalent to the number of indices. Thus, a scaler (T), a vector (T_i) and a matrix (T_{ij}) can be recognized as a rank-0, rank-1 and rank-2 tensor, and so on.

Graphically, we usually use a node and few bonds to represent a tensor. Some specified examples are shown in figure 2.1, the number of bond is equal to the rank of tensor, which means that notations without bonds, with a single bond, with two bonds and with three bonds are considered as scalers, vectors, matrices and rank-3 tensors. Moreover, the dimension of a rank-N tensor D_{total} is dependent on the dimensions of the bonds in the corresponded notions except that the dimension of a rank-0 tensor (scalers) is 1,

$$D_{total} = \begin{cases} 1 & \text{, if } N = 0 \\ \chi_1\chi_2\chi_3\ldots\chi_N & \text{, if } N \neq 0 \end{cases} \tag{2.1}$$

For instance, assume that the dimensions of the indices of the rank-3 tensor $T_{\alpha\beta\gamma}$, shown in Fig. 2.1(iv) are χ_α, χ_β and χ_γ. The tensor $T_{\alpha\beta\gamma}$ contains $\chi_\alpha\chi_\beta\chi_\gamma$ components.

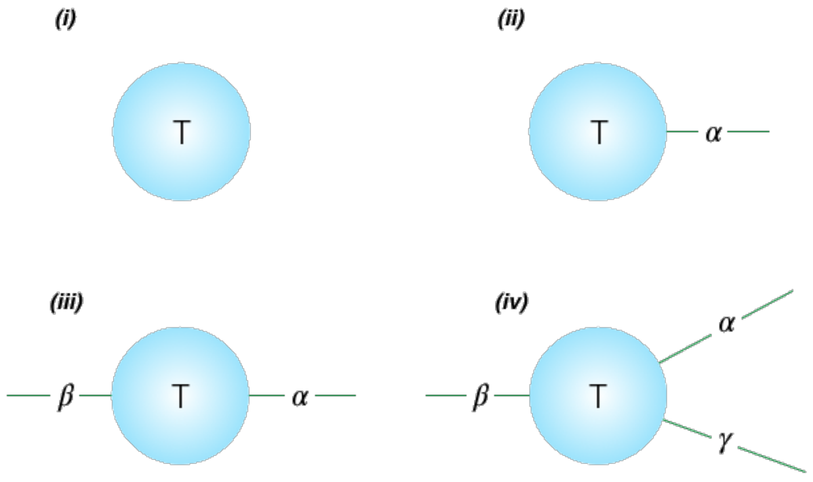


Figure 2.1: Usually we use a note and few bonds to compose a tensor and the numbers of bond depend on the rank of tensor. (i) A tensor without bonds is a scaler T, (ii) A tensor with one bond is a vector T_α, (iii) A tensor with two bonds is a Matrix $T_{\alpha\beta}$, (iv) A tensor with three bonds is a rank-3 tensor $T_{\alpha\beta\gamma}$.

2.2 Tensor operations and tensor network diagrams

In the computational languages, a fold tensor can be regard as just a container of the elements. It is not endowed with meanings until unfolded to a specified shape. In other words, we must transform it to a tensor less than rank-3 before doing any operation. The process is also called *permuteation*.

2.2.1 Permutation

See Fig. 2.2, the goal of this example is to the permute the tensor $A_{\alpha\beta\gamma}$ into $\hat{A}_{\alpha\gamma\beta}$,

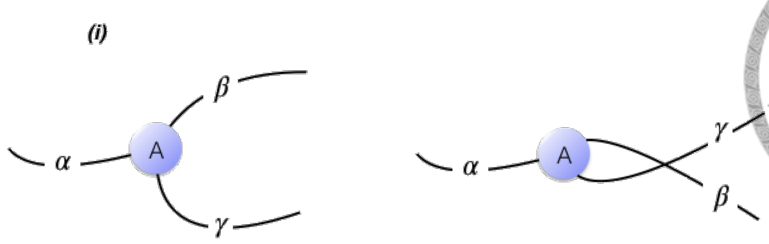


Figure 2.2: Permute tensor $A_{\alpha\beta\gamma}$ to $\hat{A}_{\alpha\gamma\beta}$

$$A_{\alpha\beta\gamma} \rightarrow \hat{A}_{\alpha\gamma\beta} \tag{2.2}$$

Nevertheless, the arrangements of tensor $A_{\alpha\beta\gamma}$ is modified to $\hat{A}_{\alpha\gamma\beta}$. The components of them are exact equivalence.

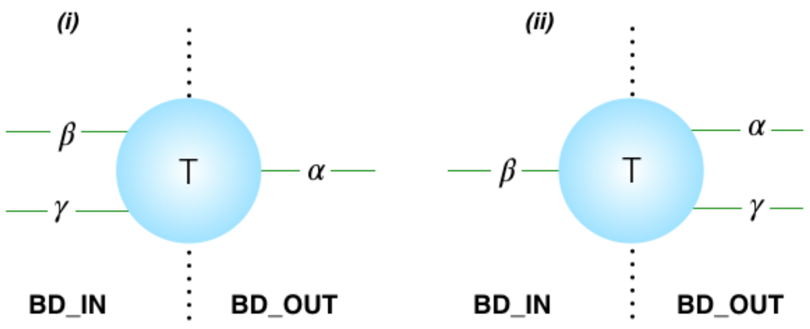


Figure 2.3: (i) Unfold a tensor to a matrix $T_{\chi_\beta\chi_\gamma,\chi_\alpha}$, (ii) Unfold a tensor to a matrix $T_{\chi_\beta,\chi_\alpha\chi_\gamma}$.

In order to explain more clearly, we separate the tensor to two part, incoming (BD_IN) and outgoing (BD_OUT) which are also designed for distinguishing different types of *uni10::Bond* in *Uni10 Library* [], to show what the meaning of reshape is in the linear algebra. The total dimensions of the bonds in the part BD_IN and BD_OUT corresponds to rows and columns of a matrix. For instance, if the indices of $T_{\alpha\beta\gamma}$ ordered like Fig. 2.3(i), $T_{\alpha\beta\gamma}$ is equivalent to a matrix $M_{\chi_\beta\chi_\gamma,\chi_\alpha}$, where χ_α, χ_β and χ_γ are the dimensions of the bonds, α, β and γ. Similarity, The tensor shown as Fig. 2.3(ii) can be recognized as a matrix $M_{\chi_\beta,\chi_\alpha\chi_\gamma}$.

2.2.2 Tensor contraction

Tensor contraction is defined as the sum of all products of the shared indices of tensors. For instance, the tensor diagram of contracting two rank-2 tensors $A_{\alpha\beta}$ and $B_{\beta\gamma}$ is shown as Fig. 2.4 which can be written as,

$$C_{\alpha\gamma} = \sum_{\beta=1}^{\chi_\beta} A_{\alpha\beta} B_{\beta\gamma}, \tag{2.3}$$

and obviously this example corresponds to the inner-product of two matrices $A_{\chi_\alpha,\chi_\beta}$ and $B_{\chi_\beta,\chi_\gamma}$,

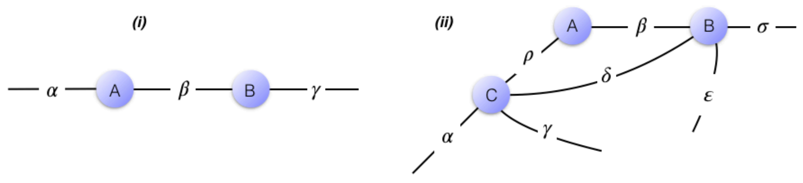


Figure 2.4: (i) Contract rank-2 tensors $A_{\alpha\beta}$ and $B_{\beta\gamma}$ (ii) Contract a rank-2 tensor $A_{\rho\beta}$, and two rank-4 tensors $B_{\sigma\varepsilon\delta}$ and $C_{\delta\rho\alpha}$

Now that we considered a more complicated example shown in Fig2.4(ii). The equation of the tensor diagram is described as,

$$D_{\alpha\gamma\sigma\epsilon} = \sum_{\beta\rho\delta} A_{\rho\beta} B_{\beta\sigma\epsilon\delta} C_{\gamma\delta\rho\alpha}, \tag{2.4}$$

In order to contract the tensors $A_{\rho\beta}$, $B_{\beta\sigma\epsilon\delta}$ and $C_{\gamma\delta\rho\alpha}$ in the diagram, we can follow the steps, see Fig. 2.5

1. Selected the pair of tensors arbitrarily: In this example, we choose the tensor $A_{\rho\beta}$ and $C_{\gamma\alpha\delta\rho}$ firstly.

2. Permute the tensors to specified shape and operate the inner-product: As shown in Fig. 2.5(ii)-(iii) Reshape (unfold) $A_{\rho\beta}$ and $C_{\gamma\alpha\delta}$ to matrices A_{χ_ρ,χ_β} and $C_{\chi_\gamma\chi_\alpha\chi_\delta\chi_\rho}$

and do the inner-product,

$$\theta_{\gamma\alpha\delta\beta} = C_{\chi_\gamma\chi_\alpha\chi_\delta\chi_\rho} \dot{A}_{\chi_\rho,\chi_\beta} \tag{2.5}$$

3. Repeat the step (1) and (2) until all tensors be contracted: See Fig. 2.5(iii)-(iv), repeat the steps again to contract the remained tensors $\theta_{\gamma\alpha\delta\beta}$ and $B_{\beta\sigma\delta\varepsilon}$

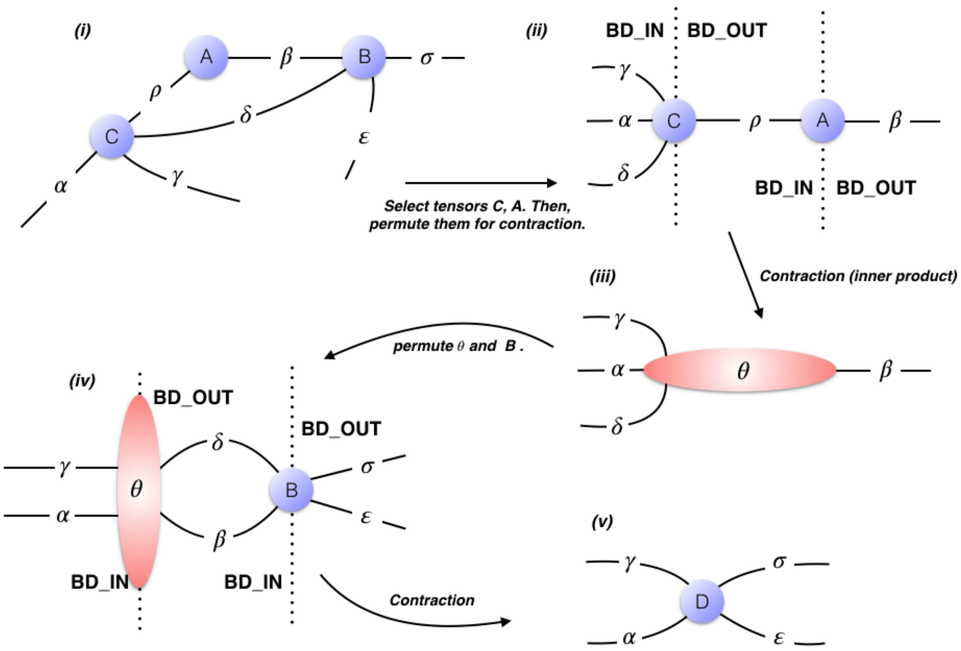


Figure 2.5: The contraction procedures of the network shown in Fig2.4

Actually, the choice in step(1) is restricted in some algorithms, because the order of the network contraction is strongly correlated to the efficiency, such as the corner transfer matrix, the fast full update and so on. For example, if the dimensions of the bonds of the tensors, $A_{\rho\beta}$, $B_{\beta\sigma\epsilon\delta}$ and $C_{\gamma\delta\rho\alpha}$ [Fig. 2.5(i)], are D. The consumption of the contraction of $A_{\rho\beta}$ and $C_{\gamma\delta\rho\alpha}$ is D^4. However, it increase to D^6, if we contract $B_{\beta\sigma\epsilon\delta}$ and $C_{\gamma\delta\rho\alpha}$ at first. Therefor, how to find the cheapest contraction order is a non-trivial issue. Nevertheless, it still hard be determined efficiently when the network contains too many tensors.

2.3 Describe Quantum states with tensor network

Before drawing a many-body system with tensor-network representation, we should discuss how to describe a many-body systems which is a chain of N particles, with each particles having d states. The system can be regard as a congregation of N localized particles and we have studied that a pure state corresponds to a vector in Hilbert space. Hence, the wave-function of many-body systems can be described by N subspace,

$$|\Psi_N\rangle = \sum_{i_1,i_2,\ldots,i_N} C_{i_1,i_2,i_3,\ldots,i_N} |i_1\rangle \otimes |i_2\rangle \otimes \ldots \otimes |i_N\rangle, \tag{2.6}$$

where each individual $|i_1\rangle, |i_2\rangle, \ldots, |i_N\rangle$ spanned by an orthogonal basis which the degree of freedom is d. After writing down the formulation of the wave function, eq.2.6, we are able to build a tensor-network representation for quantum states. The wave function $|Psi_N\rangle$ is shown as Fig. 2.6(a), each bond of the tensor corresponds to the local Hilbert space $|i_n\rangle$ and the dimension of each bond is equivalent to the probable states of the particle on the nth site and the coefficients of the rank-N tensor corresponds to $C_{i_1,i_2,i_3,\ldots,i_N}$.

No matter from eq. 2.6 or Fig. 2.6(a), we can directly notice that the number of coefficients, $C_{i_1,i_2,i_3,\ldots,N}$, is proportional to d^N. Therefor, it is impossible to fully describe a many-body system by a classical machine, if the system size larger then fifty. Fortunately, according to the theory of MPS [], the wave-function can be decomposed to two subsystem by Schmidt decomposition and there are more details in chapter.3.

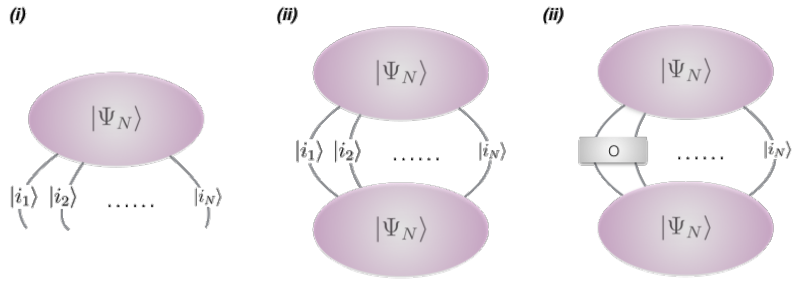


Figure 2.6: (i) Wave-function: $|\Psi_N\rangle$ (ii) Norm of $|\Psi_N\rangle$; $\langle\Psi_N \mid \Psi_N\rangle$ (iii) Expectation value of observable O; $\langle\Psi_N| O |\Psi_N\rangle$

Chapter 3

2-D Imaginary Time Evolving Block Decimation

In one-dimensional many-body systems, the performance of one-dimension imaginary time evolving block decimation (1D-iTEBD) is very well, especially its high efficiency. Naturally, people want to extend the framework to study the two-dimensional systems. However, through many experiments we notice that the accuracy in 2-D is less than 1-D. In order to resolve that the obstacle, optimizations of 2-D algorithms became an essential part in condense matter physics.

In the sections 3.1 and 3.2, we briefly review the idea of imaginary time evolution (TEBD) [7] [8] and explain how to extend it to simulate an infinite two-dimensional system (2D-iTEBD)[13]. However, the method recorded in ref.[13] is unstable because it need multiply too many pseudo-inverse matrices during updating the wave functions. Hence, In the sections 3.3 and 3.4 we record more advance discussions about optimizing 2D-iTEBD, such as the method developed by Hastings [], combined with QR decomposition, and so on. In the final section 3.5, we utilize 2D-iTEBD to simulate the toy models, Heisenberg and transverse Ising, on two-dimensional square lattice and compare the features among 2D-TEBD algorithms which are implemented by different way. Notice that in this chapter we just focus on obtaining good projected entangled pair states [] by 2D-iTEBD. The optimization of measurement (accuracy) will be discussed in chapters 5.

3.1 Imaginary Time Evolution

In the section, we apply the conclusion of the theory of imaginary time evolution and matrix product states (MPS) [] to explain the fundamental concept of TEBD-like algorithms. Theoretically, if existing the imaginary time evolution operator $e^{-\tau H}$, where H is the Hamiltonian of a specified model, we could project any initial random states to the ground state,

$$|\psi_0\rangle = \frac{e^{-\tau H}\,|\Psi\rangle}{\|\; e^{-\tau H}\,|\Psi\rangle \;\|} \tag{3.1}$$

However, according to eq.2.6, we notice that the number of coefficients in an origin evolution operator $e^{-\tau H}$ is proportional to $d^N \times d^N$. In other words, it is impossible to update entire system directly. Therefor, in 1D quantum many-body systems we apply MPS structure to restrict the exponential increment of dimension. Base on the theory of the MPS, the wave function composed by pure states can be decompose to many unit cells by *sigular value decomopostion* and *Schmidt decomposition*. To explain the MPS structure more explicitly, we begin from splitting the wave function $|\Psi_N\rangle$ [Fig. 2.6(a)] between n and $n+1$ sites with Schmidt decomposition,

$$|\Psi_N\rangle = \sum_{\alpha_n} \lambda_{\alpha_n} \left|\psi^{[1\ldots n]}_{\alpha_n}\right\rangle \left|\psi^{[n+1\ldots N]}_{\alpha_n}\right\rangle \tag{3.2}$$

where $\lambda_{\alpha_n} > 0$ and $\sum\limits_{\alpha+n} \lambda^2_{\alpha_n} = 1$. To obtain the one site wave function $\left|\psi^{[n+1]}_{\alpha_n}\right\rangle$, we perform Schmidt decomposition on $\left|\psi^{[n+1\ldots N]}_{\alpha_n}\right\rangle$ between the $n+1$ and $n+2$ sites,

$$\left|\psi^{[n+1\ldots N]}_{\alpha_n}\right\rangle = \sum_{\alpha_{n+1}} \lambda_{\alpha_{n+1}} \left|\psi^{[n+1]}_{\alpha_{n+1}}\right\rangle \left|\psi^{[n+2\ldots N]}_{\alpha_{n+2}}\right\rangle \tag{3.3}$$

then span $\left|\psi^{[n+1]}_{\alpha_{n+1}}\right\rangle$ by the spin basis i_{n+1},

$$\left|\psi^{[n+1]}_{\alpha_{n+1}}\right\rangle = \sum_{i_{n+1}} \Gamma^{[n+1]i_{n+1}}_{\alpha_n \alpha_{n+1}} \left| i_{n+1} \right\rangle \tag{3.4}$$

and the Eq. 3.2 can be re-written as,

$$|\Psi_N\rangle = \sum_{\alpha_n,\alpha_{n+1}} \sum_{i_{n+1}} \lambda_{\alpha_n} \Gamma^{[n+1]i_{n+1}}_{\alpha_n \alpha_{n+1}} \lambda_{\alpha_{n+1}l} \left|\psi^{[1\dots n]}_{\alpha_n}\right\rangle \left|i_{n+1}\right\rangle \left|\psi^{[n+2\dots N]}_{\alpha_{n+2}}\right\rangle \tag{3.5}$$

In the end, we can perform the same process site-by-site in the entire system and obtain the MPS structure,

$$|\Psi_N\rangle = \sum_{\alpha_1,\dots,\alpha_N} \sum_{i_1,\dots,i_N} \Gamma^{[1]i_1}_{\alpha_1} \lambda_{\alpha_1} \Gamma^{[2]i_2}_{\alpha_1\alpha_2} \lambda_{\alpha_2} \dots \lambda_{\alpha_{N-2}} \Gamma^{[N-1]i_{N-1}}_{\alpha_{N-2}\alpha_{N-1}} \lambda_{\alpha_{N-1}} \Gamma^{[N]i_N}_{\alpha_N} \left|i_1 i_2 \dots i_N\right\rangle \tag{3.6}$$

and the tensor network representation is shown as Fig. 3.1,

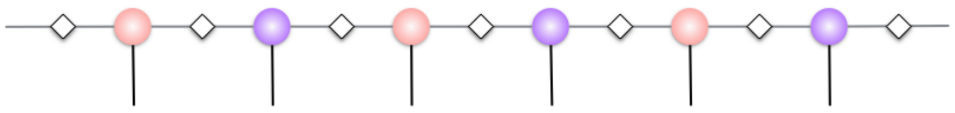


Figure 3.1: The tensor network representation of matrix product states.

Now that we try to expand it to a infinite chain []. Since the translation invariance, the wave function $|\Psi_{N=\infty}\rangle$ can be represent as n-site translational symmetric states, which means that $\Gamma^{[i]}$ and $\lambda^{[i_i]}$ are independent of $\Gamma^{[i+n]}$ and $\lambda^{[i+n]}$. For instance, the tensor diagram can be drawn as Fig.3.2, when $n = 2$, which means that the wave function of a infinite chain can be recognized as a composite of matrix product states $\lambda^{[A]}\Gamma^{[A]}$ (red nodes) and $\lambda^{[B]}\Gamma^{[B]}$ (purple nodes).

Next, in order to update the two states in the unit cell, we utilize the *Suzuki Trotter decomposition* to approximate the entire evolution operator with 2-sites operators, *The*

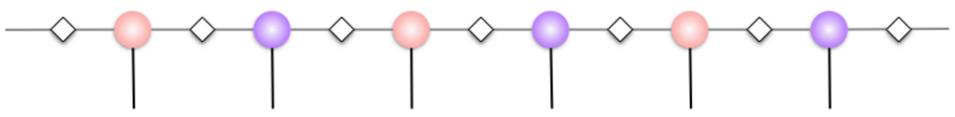


Figure 3.2: The tensor network representation of matrix product states.

first-order Suzuk-Trotter decompostio of operator $e^{\delta(A+B)}$ is,

$$e^{\delta A+B} = e^{\delta A} e^{\delta B} + O(\delta^2) \tag{3.7}$$

where A and B are two non-commutative operators.Therefor, we can approximate the entire evolution operator by grouping the two site operator H_{AB} and H_{BA},

$$e^{-\tau H} = \left(e^{-\delta H}\right)^{\frac{\tau}{\delta}} \approx \left(\prod e^{\delta H_{AB}}\right)\left(\prod e^{\delta H_{BA}}\right) \tag{3.8}$$

Therefor, we can obtain the evolution operator $e^{H_{AB}}$ and $e^{H_{BA}}$ straightly after solving two site H_{AB} and H_{BA}. So far, we have obtained the infinite MPS composed by 2-site unit cells and the 2-site evolution operators $e^{-\tau H_{AB}}$ and $e^{-\tau H_{BA}}$. Hence, the tensor network representation of Eq. 3.38 can be drawn as Fig. 3.3, which means that the ground state $|\psi_0\rangle$ can be regard as contracting all the tensors in the diagram. So the next problem: How can we contract them and preserve the structure like Fig3.1?

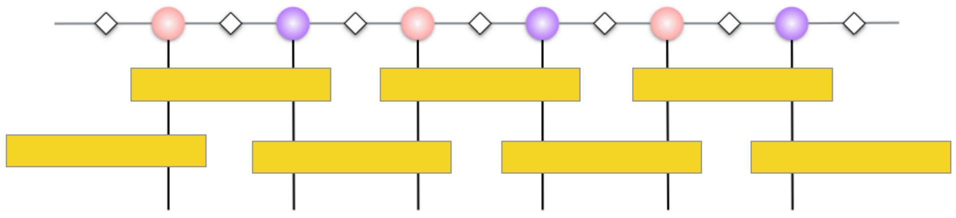


Figure 3.3: The red and blue tensor denotes on *odd* and *even* sites. The yellow one are time evolution operators $e^{-\tau H_{k,k+1}}$ and $e^{-\tau H_{k+1,k}}$

3.2 Simple Infinite Imaginary-Time Evolving Block Decimation for 2-D system

In this section, we focus on how to implement and optimize 2D-iTEBD algorithms and the more theoretical details are derived in the references [14] [15] [16].

3.2.1 One-dimensional iTEBD

To resolve the problem in the end of the section 3.1, the ”Simple Update” scheme was developed and have widely applied to iPEPS [] and PESS ansatz []. In order to explain it clearly, we start from a simple case, one-dimensional iTEBD.

The tensor diagrams shown in the Fig. 3.4 are the procedures of 1D-iTEBD which can be simply build by following steps,

1. Initialization: According to Eq. 3.38, the ground state can be obtain from any random state $|\Psi\rangle$ theoretically. Hence, we provide two rank-3 tensors $\Gamma^{[A]}$ and $\Gamma^{[B]}$ which dimension are dD^2, where d is the dimension of physical basis and D is the virtual bonds dimension, and two random diagonal matrix $\lambda^{[A]}$ and $\lambda^{[B]}$ which represent the entanglement between each sites at first. See Fig. 3.4(i).
2. Obtain a cluster tensor θ: As shown in Fig. 3.4(b),
   (a) Absorb the entangled matrices;

$$\Gamma'^{[A]} = \sum_{ij} \lambda_i^{[A]} \Gamma^{A}_{ij\sigma_i} \lambda_j^{[B]} \tag{3.9}$$

$$\Gamma'^{[B]} = \sum_{k} \Gamma^{B}_{k\sigma_j} \lambda_k^{[A]} \tag{3.10}$$

   (b) Utilize the evolution operate $U(\tau)$: The 2-site evolution operator $U(\tau)$ can be simply obtained from Eq. 3.8, In this case,

$$U(\tau) = e^{-\tau H_{AB}} \tag{3.11}$$

3. Decompose Θ into the general form of the MPS: In this step, we utilize singular value decomposition (SVD) to split the tensor Θ . The SVD decomposition allow to decompose a matrix $A_{m.n}$, with $m \geq n$, into two unitary matrices, U, with $m \times m$, and V^T,with $n \times n$, and an $m \times n$ diagonal matrix Σ. However, in bottom $m-n$ rows of Σ consists entirely of zeros. Hence, the dimension of U and Σ can be reduced,

$$A = U\Sigma V^T = \begin{bmatrix} U_1 U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 \\ 0 \end{bmatrix} V^T = U_1 S_1 V^T \tag{3.12}$$

where U_1 is a $m \times n$ unitary matrix, Σ_1 is a $n \times n$ diagonal matrix. Analogously, when $A_{m,n}$, with $m \leq n$, the matrix Σ_1 and V^T can be reduced,

$$A = U\Sigma V^T = U \begin{bmatrix} \Sigma_1 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} = U\Sigma_1 V_1^T \tag{3.13}$$

where Σ_1 is a $m \times m$ diagonal matrix and V_1^T is a $m \times n$ unitary matrix. There are two significant properties of the singular value term, Assume that,

$$\Sigma_1 = diag\left(\sigma_1, \sigma_2, \ldots, \sigma_{\max[m,n]}\right), \tag{3.14}$$

(a) All the singular value in Σ_1 are real.

(b) The singular values are ordered from large to small,

$$\sigma_1 \geq \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\max[m,n]} \tag{3.15}$$

4. Truncation: See Fig. 3.4(iv), we notice that the dimension of bond j increase to dD. To avoid the exponential increment of the dimension, the dimension of bond j must be resized to D.

5. Absorb the inverse entangled matrix $\lambda^{[A]}$: Remove the entangle influence from $\widetilde{\Gamma}^{[A]}$

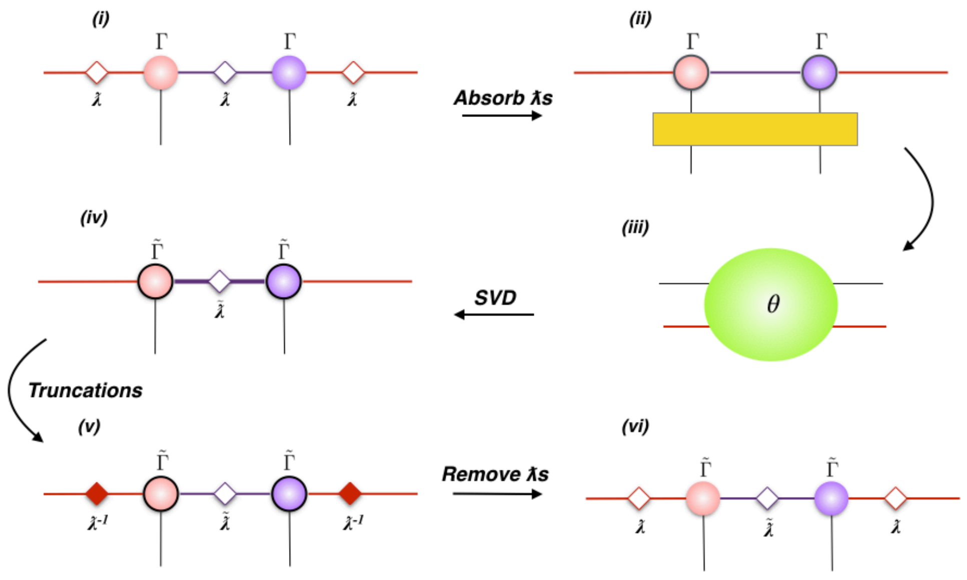
and $\widetilde{\Gamma}^{[B]}$ and return the general form of the MPS, as shown in Fig. 3.4(v).

$$\widetilde{\Gamma}^{[A]} = \sum_i \lambda_i^{[A]^{-1}} \Gamma^{[A]}_{ij,\sigma_i} \tag{3.16}$$

$$\widetilde{\Gamma}^{[B]} = \sum_k \Gamma^{[B]}_{jk,\sigma_j} \lambda_k^{[A]^{-1}} \tag{3.17}$$

6. Repeat the steps (2)-(5) to update the tensors $\widetilde{\Gamma}^{[B]}$, $\widetilde{\Gamma}^{[A]}$ and $\lambda^{[A]}$ with the evolution operator $e^{-\tau H_{BA}}$.

7. Iterate the steps (2)-(6) until the wave function converges.



[b]

Figure 3.4: (i)Absorb all λ to Γ. (ii) Contract an evolution operator $e^{-\delta H}$ for evolving the system. (iii) Decompose the tensor θ by SVD. (iv) Truncate and Update the states and λ on the green bond.(v) Remove λ for obtaining the states. (iv) After updating the states and λ on the purple bond, apply the way to update the red bond and repeat all the steps until the ground state convergence.

In one dimensional many-body systems, the performance of iTEBD is pretty well. Although the accuracy is little less than DMRG [], it is still widely applied to study or test models due to its high efficiency.

3.2.2 Two-dimensional iTEBD

Due to the success in 1-D cases, we desire to extend the framework of iTEBD to simulate 2-D systems . In 2-D systems, we describe the wave functions by the projected entangled pair states (PEPS) rather than MPS. The PEPS structure is an naturally extension of MPS and obey the restriction of MPS. Therefore, it can be expanded to a infinite structure, iPEPS, which are composed by n-site translation invariance states. For instance, we can draw an tensor diagram as shown in Fig. 3.5 to represent the wave function of a 2-D many-body system which is composed by 2-site translational symmetric states.

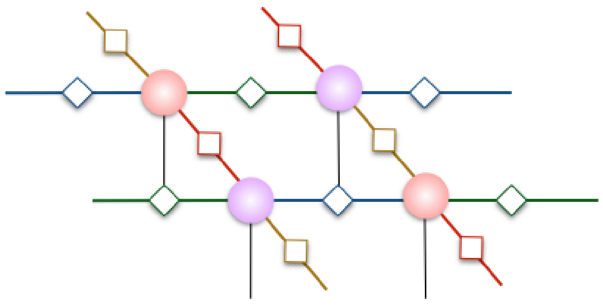


Figure 3.5: Four sites unit cell in iPEPS.

After building the form of iPEPS, we start to deal with the problem of updating states. The most intuitive method is directional simple update, which means that the states of iPEPS must be updated by iTEBD along four directional moves, namely up, right, down and left. The scheme of implementing 2D-iTEBD is shown as follows which start from right move,

1. Initialization: To describe iPEPS states [Fig. 3.6], we need two random rank-5 tensors $\Gamma^{[A]}_{uldr,\sigma_j}$ and $\Gamma^{[B]}_{uldr,\sigma_j}$ which dimension are dD^4, and four random diagonal matrix λ_u, λ_l, λ_d and λ_r.

2. Obtain a cluster tensor Θ: As shown in Fig. 3.5(ii),

(a) Absorb the entangled matrices;

$$\Gamma'^{[A]}_{uldr,\sigma_i} = \sum_{uldr} \lambda_u \lambda_l \Gamma^{[A]}_{ulrd,\sigma_i} \lambda_r \lambda_d \tag{3.18}$$

$$\Gamma'^{[B]}_{uldr,\sigma_j} = \sum_{uld} \lambda_d \Gamma^{B}_{uldr,\sigma_j} \lambda_u \lambda_l \tag{3.19}$$

(b) Utilize the evolution operate $U(\tau)$: See Fig. 3.6(iii)

$$\Theta = \sum_{r,\sigma_i'\sigma_j'} U^{\sigma_i'\sigma_j'}_{\sigma_i\sigma_j} \Gamma'^{\sigma_i[A]}_{uld,r} \Gamma'^{\sigma_j[B]}_{r,u'l'd'} \tag{3.20}$$

the rank of Θ is eight and the dimension is d^2D^6.

3. Decompose θ into the general form of iPEPS:

4. Truncation: See Fig. 3.6(iv), the dimension of bond r increase to dD^3. To reduce the consumption, we must truncate the dimension to a smaller D.

5. Absorb the inverse entangled matrix surrounding $\Gamma'^{[A]}$ and $\Gamma'^{[B]}$: Return to the general form of the iPEPS, as shown in Fig. 3.6(v).

$$\widetilde{\Gamma}^{[A]}_{uldr,\sigma_i} = \sum_{uldr} \lambda_u^{-1} \lambda_l^{-1} \Gamma^{[A]}_{ulrd,\sigma_i} \lambda_d^{-1} \tag{3.21}$$

$$\widetilde{\Gamma}^{[B]}_{uldr,\sigma_j} = \sum_{uld} \lambda_d^{-1} \Gamma^{[B]}_{uldr,\sigma_j} \lambda_u^{-1} \lambda_l^{-1} \tag{3.22}$$

6. Repeat the steps (2)-(5) to update the others directional moves. To explain more explicitly, the procedures of the up move are shown in Fig. 3.7.

7. Iterate the steps (2)-(6) until the wave function converges.

In one dimensional many-body systems, the performance of iTEBD is pretty well. Although the accuracy is little less than DMRG [], it is still widely applied to study or test models due to its high efficiency.

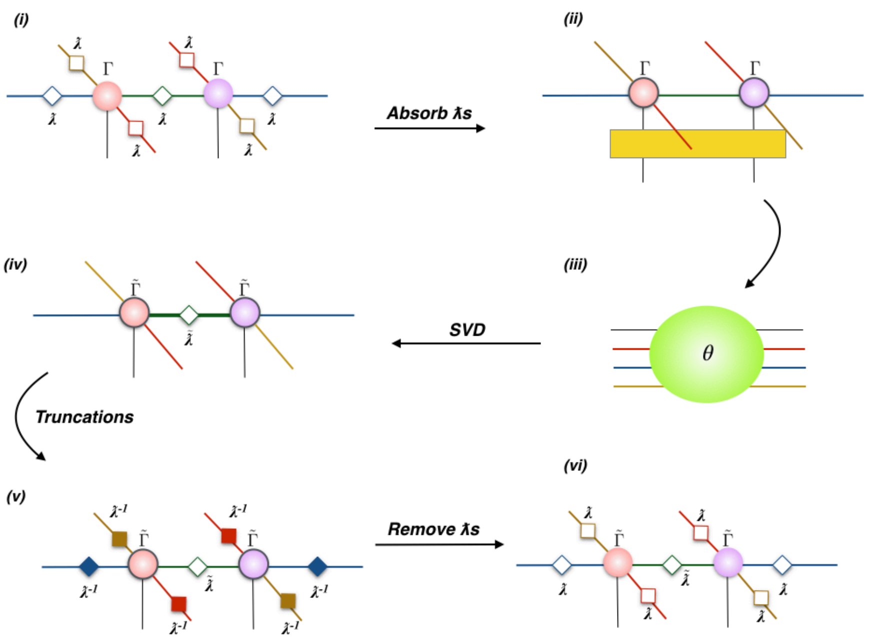


Figure 3.6: Absorb all λ to Γ. (ii) Contract an evolution operator $e^{-\delta H}$ for evolving the system. (iii) Decompose the tensor θ by SVD. (iv) Truncate and Update the states and λ on the green bond.(v) Remove λ for obtaining the states. (iv) Obtain a original form of iPEPS. Repeat all the step to update the other bonds until the ground state energy convergence

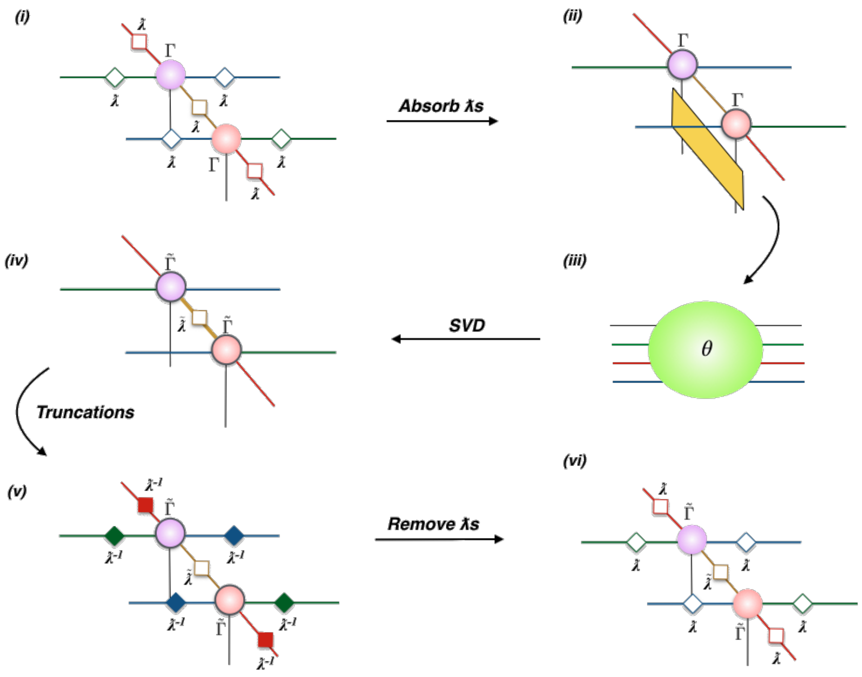


Figure 3.7: Update the yellow bond and the steps are similar to Fig.3.5

3.3 Ameliorate two-dimensional iTEBD

The directional simple update discussed in section.3.2.2 can obtain good iPEPS states in toy models. However, in some complicated models it becomes unstable and inefficient. The reason is that there are too many multiplications of pseudo-inverse λ^{-1} at the step Fig.3.6(v). In numerical methods, it's dangerous to divide a value which is equal or approach to zero. In other words, the more inverse operations, the more probability of destroying algorithms and harder to converge. Hence, *M. B. Hastings* develop another scheme to improve that problem [17]. The procedures from the right is shown as follows,

1. Initialization: As directional simple update, we declare two random $\Gamma^{[A]}$ and $\Gamma^{[B]}$, and four random diagonal matrix λ_u, λ_r, λ_l and λ_d to describe the wave function. 3.10(i). However, the definition of $\Gamma^{[A]}$ and $\Gamma^{[B]}$ are different. In this scheme,

$$\Gamma^{[A]}_{ulrd,\sigma_i} \equiv \sum_{ur} \lambda_u \Gamma^{[A_s]}_{ulrd,\sigma_i} \lambda_r \tag{3.23}$$

$$\Gamma^{[B]}_{ulrd,\sigma_j} \equiv \sum_{dl} \lambda_d \Gamma^{[B_s]}_{ulrd,\sigma_j} \lambda_l \tag{3.24}$$

where $\Gamma^{[A_s]}_{ulrd,\sigma_i}$ and $\Gamma^{[B_s]}_{ulrd,\sigma_j}$ are the definition in directional simple update. [In sec.3.2.2].

2. Obtain the cluster tensor C: As shown in Fig. 3.10(b),

   (a) Absorb the entangled matrices λ_u into $\Gamma^{[B]}_{ulrd,\sigma_j}$:

$$\Gamma'^{[B]}_{uldr,\sigma_j} = \sum_{u} \Gamma^{[B]}_{uldr,\sigma_j} \lambda_u \tag{3.25}$$

   (b) Contract the tensors $\Gamma^{[A]}$, $\Gamma'^{[B]}$ and the evolution operate $U(\tau)$:

$$C = \sum_{r,\sigma_i'\sigma_j'} U^{\sigma_i'\sigma_j'}_{\sigma_i\sigma_j} \Gamma^{\sigma_i[A]}_{uld,r} \Gamma'^{\sigma_j[B]}_{r,u'l'd'} \tag{3.26}$$

3. Obtain the cluster tensor Θ: In the step, the entanglement surrounding $\Gamma^{[A]}$ would be absorbed in the tensor C,

$$\Theta = \sum_{dl} \lambda_d \lambda_l C^{udl,\sigma_i}_{u'd'l',\sigma_j} \tag{3.27}$$

4. Decompose θ into the general form of the iPEPS and truncate the updated bond: This step is same as directional simple update, see Fig. 3.10(v)

5. Update $\Gamma^{[A]}$: Due to the tensor C does not contain the entanglement surrounding $\Gamma^{[A]}$, the updated $\Gamma^{[A]}$ can be obtained by contracting tensors C and $\widetilde{\Gamma}^{[B]}$

$$\Gamma^{[A]} = \sum_{u'l'r'\sigma_j} C^{udl,\sigma_i}_{u'd'l',\sigma_j} \widetilde{\Gamma}^{\sigma_j[B]}_{u'd'l'} \tag{3.28}$$

6. Absorb the inverse λ_u into $\widetilde{\Gamma}^{[B]}_{ulrd,\sigma_j}$:

$$\Gamma^{[B]} = \sum_{u} \lambda_u^{-1} \widetilde{\Gamma}^{[B]}_{ulrd,\sigma_j} \tag{3.29}$$

   , which is shown as Fig. 3.10.

7. Repeat the steps (2)-(5) to update the others directional moves, down, left, up.

8. Iterate the steps (2)-(6) until the wave function converges.

In the procedures of directional simple update, there are six pseudo-inverse λ^{-1} must be absorbed in each iteration. Relatively, in this scheme we just need multiply an pseudo-inverse λ^{-1} in step(6). Nevertheless, the consumption of updating states in step(5) is higher than the update step in directional simple update and cost more time in each iteration. The stability is able to make the wave function converge in less iterations.

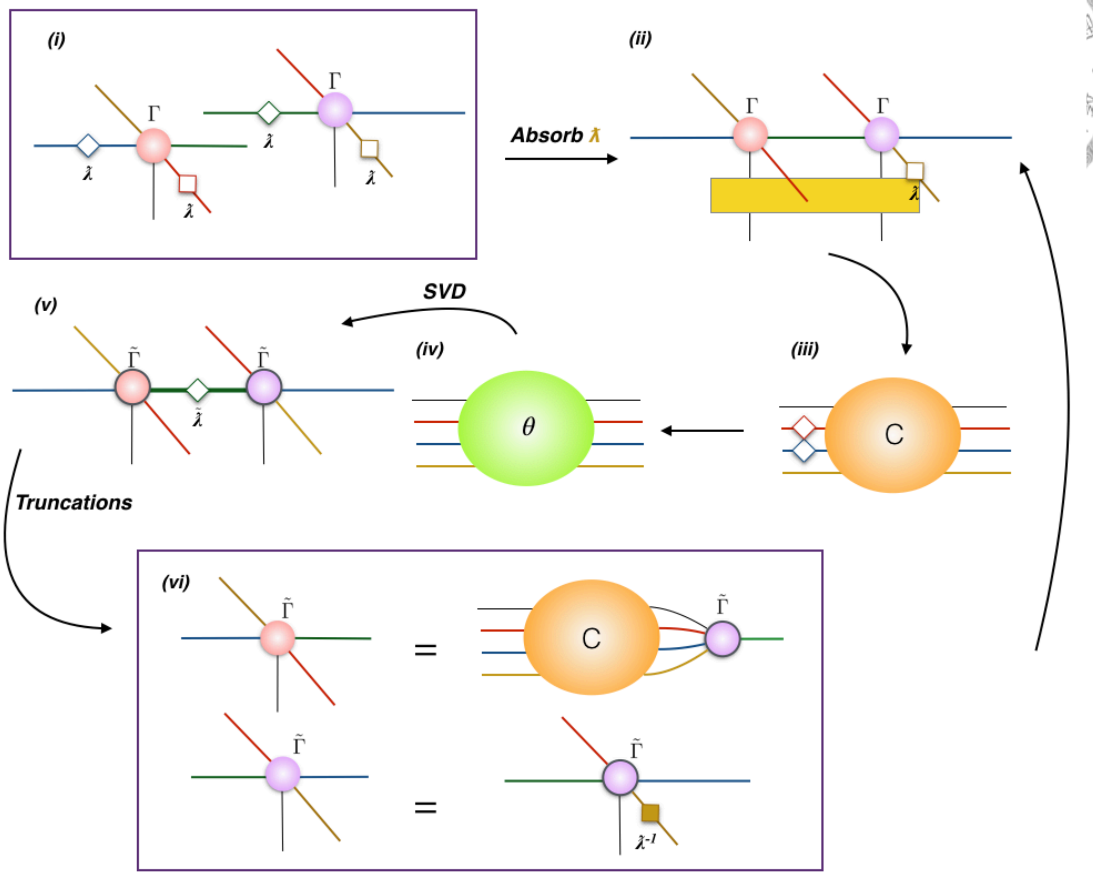


Figure 3.8: The tensor network diagrams for the ameliorated 2-D iTEBD.

3.4 Optimizations

3.4.1 Initialization

Theoretically, no matter update from any random initial state, we can obtain the ground state wave function by iTEBD-like algorithms. However, many experimental results show that stating from a awful initial sates might hardly converge or even break the algorithms. Hence, we need ”guess” a good initial states to protect algorithms.

1. Begin from the Product states.
2. Considered the Translational invariance: For example, theres is a iPEPS shown as Fig. 3.9(i). The arrangement of coefficients in the states A_{hurdl} should be same as B_{hdlur}, see Fig. 3.9(ii).

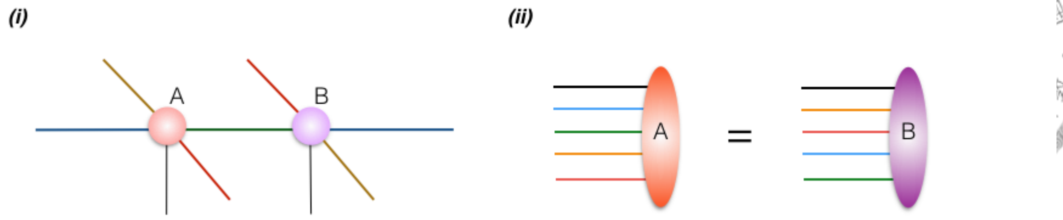


Figure 3.9: (i) The structure of PEPS, (ii) The initialization of states

3.4.2 QR and LQ decomposition

In the section 3.2.2 and 3.3, we have introduced to two different methods to implement 2D-iTEBD. However, it is hard to apply them to study when the dimension of the virtual bonds become larger because of the rapid increment of the dimension of tensor Θ [Fig. 3.6(iii) and Fig. 3.10(iii)] which is a rank-8 tensor with dimension d^2D^6. In addition to the problem of their consumption, the singular value decomposition, which time complexity proportional $O(NM^2)$, is expensive. Therefore, in this section we apply QR and LQ decomposition to reduce the rank of Θ.

The QR decomposition allows to decompose a real or complex matrix $A_{m,n}$, with $m \geq n$, into a $m \times m$ unitary matrix Q and a $m \times n$ upper triangular matrix R. However, in bottom $(m-n)$ rows of R are filled with zeros. Hence, the matrix Q and R can be truncated,

$$A = QR = Q \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = \begin{bmatrix} Q_1, Q_2 \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = Q_1 R_1, \tag{3.30}$$

where Q_1 is a $(m \times n)$ unitary matrix and R_1 is a $(n \times n)$ upper triangular matrix. Analogously, the LQ decomposition can decompose a matrix $A_{m,n}$, with $m \leq m$ into a $m \times n$ lower triangular matrix L_1 and a $m \times n$ unitary matrix Q_1,

$$A = LQ = \begin{bmatrix} L_1, 0 \end{bmatrix} Q = \begin{bmatrix} L_1, 0 \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} = L_1 Q_1, \tag{3.31}$$

In the case of directional simple update, most of the steps of implementing with QR decomposition is similar to original one, see Fig. 3.10. There are just two differences,

1. After the entangled matrices λs are absorbed, we decompose the states $\Gamma^{[A]}$ into a unitary matrix X and a upper triangular matrix a_R,

$$\Gamma^{[A]}_{\chi_d\chi_l\chi_u,\chi_{\sigma_i}\chi_r} = X_{\chi_d\chi_l\chi_u,\chi_{\sigma_i}\chi_r} a_{R_{\chi_{\sigma_i}\chi_r,\chi_{\sigma_i}\chi_r}} \tag{3.32}$$

   and decompose $\Gamma^{[A]}$ into a unitary matrix Y and a lower triangular matrix b_L

$$\Gamma^{[B]}_{\chi_{\sigma_j}\chi_r,\chi_d\chi_l\chi_u} = b_{L_{\chi_{\sigma_j}\chi_r,\chi_{\sigma_j}\chi_r}} Y_{\chi_{\sigma_i}\chi_r,\chi_d\chi_l\chi_u} \tag{3.33}$$

   where χ_n means the dimension of the bond n. See Fig. 3.10(ii).

2. Contract X and a_R into $\Gamma^{[A]}$ and b_L and Y into $\Gamma^{[B]}$ before the inverse matrices are absorbed. See Fig. 3.10(v).

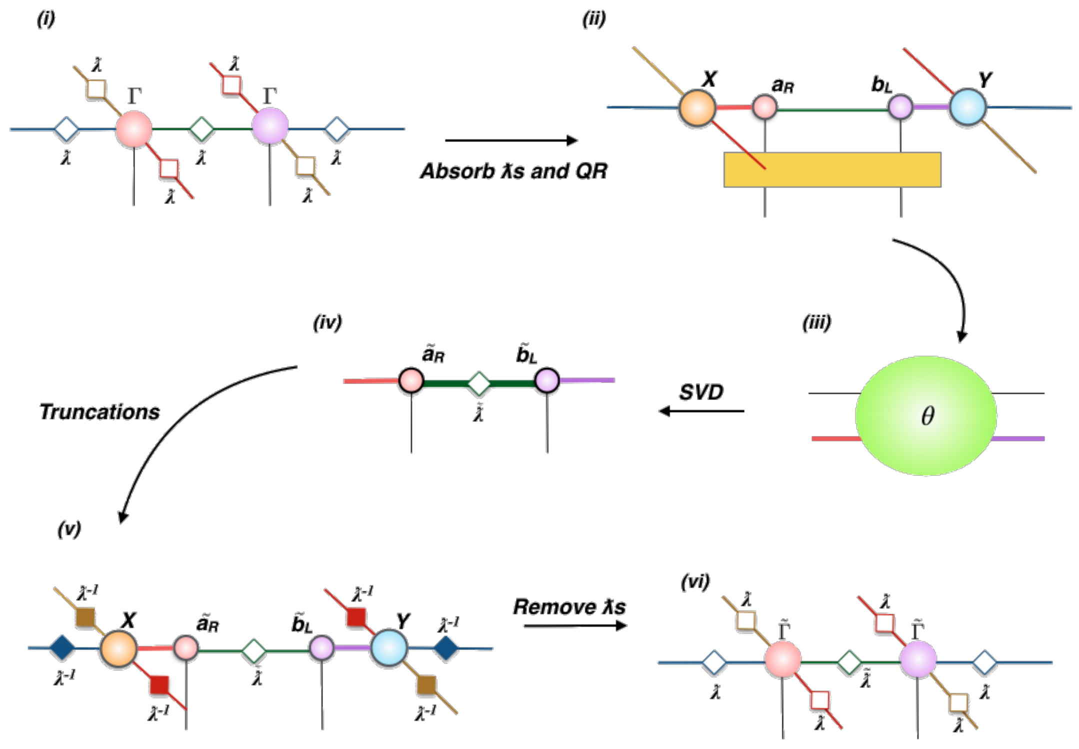


Figure 3.10: The tensor network diagrams for the ameliorated 2-D iTEBD.

Nevertheless, the main idea to reduce the tensor Θ in ameliorate 2D-iTEBD is same as in directional simple update. The scheme becomes more complicated, see Fig. 3.10,

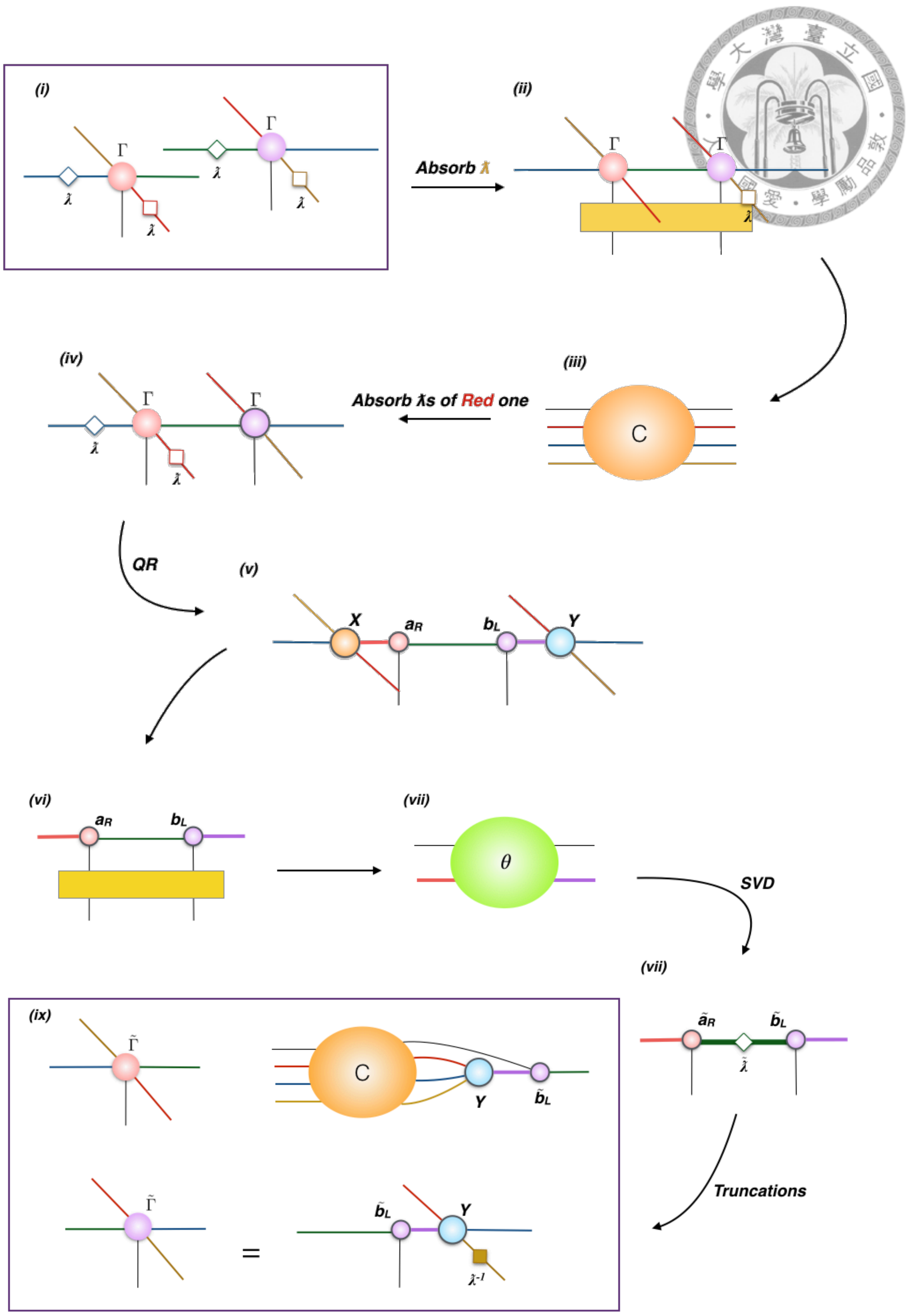


Figure 3.11: The tensor network diagrams for the ameliorated 2-D iTEBD.

1. Instead absorb λ_d and λ_l into the cluster tensor C, we absorb them into $\Gamma^{[A]}$

$$\Gamma'^{[A]}_{uldr,\sigma_i} = \sum_{dl} \lambda_l \Gamma^{[A]}_{uldr,\sigma_i} \lambda_d \tag{3.34}$$

   and apply the QR and LQ decomposition to split $\Gamma'^{[A]}$ and $\Gamma'^{[B]}$ into $X \cdot a_R$ and $b_L \cdot Y$. see Fig. 3.11(iv) and Fig. 3.11(v)

2. Obtain $\widetilde{\Gamma}^{[A]}$ by contracting tensor C, Y and b_L, as shown in Fig. 3.11(ix)

$$\widetilde{\Gamma}^{[A]}_{urld,\sigma_i} = \sum_{u'l'd'\sigma_j,q} C^{dlu\sigma_i}_{d'l'u'\sigma_i} Y_{d'l'u'q} b_{Lqr} \tag{3.35}$$

   and obtain $\widetilde{\Gamma}^{[B]}$ by contracting Y, b_L and λ_u^{-1}

$$\widetilde{\Gamma}^{[B]}_{urld,\sigma_j} = \sum_{q} b_{Lrq} Y_{uld\sigma_j q} \lambda_u^{-1} \tag{3.36}$$

Through these changed, the tensor Θ is reduced to and rank-4 tensor, with dimension d^2D^2.

3.4.3 Truncation Error

In order to control the increment of the dimension of the virtual bonds, we will truncate the updated bond from dD^3 or d^2D to D after decomposing the tensor Θ. However, if the singular values contained in Σ are not concentrated to the leading terms, some influential features and basis would be dropped and the algorithms might be broken. Therefor, we set a cut off ε,

$$\varepsilon \geq 1 - \sum_{i=1}^{\chi} \sigma_i^2 \tag{3.37}$$

to determinate the dimension of the updated bond, where σ_i is the elements at $\Sigma_{[i,i]}$ and χ is the number of the remained basis . However, it not means that the smaller ε, the more accuracy. Setting a small cut off also means that more basis of states and approach zero terms in Σ be remained. Hence, it not only make less efficiency, but also might destroy the algorithms.

3.5 Comparison

In previous sections, we have benchmarked the optimizations of 2D-iTEBD. Now that we will compare the results for the Heisenberg model on the square lattice,

$$|\psi_0\rangle = \frac{e^{-\tau H}\,|\Psi\rangle}{\parallel e^{-\tau H}\,|\Psi\rangle \parallel} \tag{3.38}$$

and show the influences of the various optimizations.

3.5.1 Different Initializations

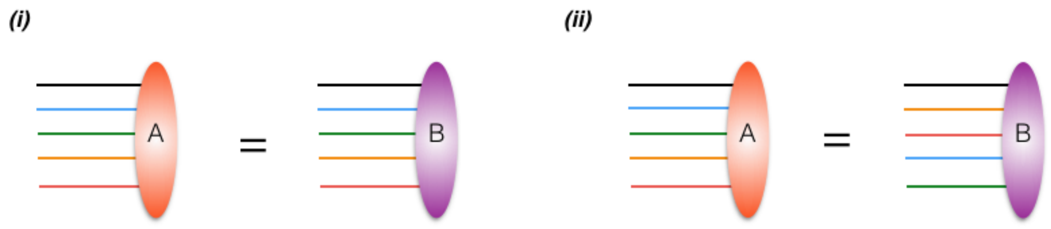


Figure 3.12: (i) Type 1, (ii) Type 2

Assume that the iPEPS states are constructed as Fig. 3.9(i) and the algorithm starts from two different initial states, see Fig. 3.12(i) and Fig. 3.12(ii), Obviously, The Type1 method does not allow translational symmetry. Hence, the algorithm might hardly converge or even be broken. As shown in Fig 3.13, the efficient ameliorate 2D-iTEBD (Hastings+QR) starting from the initial states which are obtained from Type2 [Fig. 3.12(ii)] converge in less 1000 epochs. However, if we begin from the Type1 states [Fig. 3.12(i)], the algorithm can not converge in 7000 epochs and even be broken in the end.

In conclusion, through many experiment we have known that this problem may not only occur in 2D-iTEBD, but also in any algorithm which is build on the theory of imaginary-time evolution, such as fast full update [], PESS []. Therefor, we should initialize the states carefully when applying them to study the models.

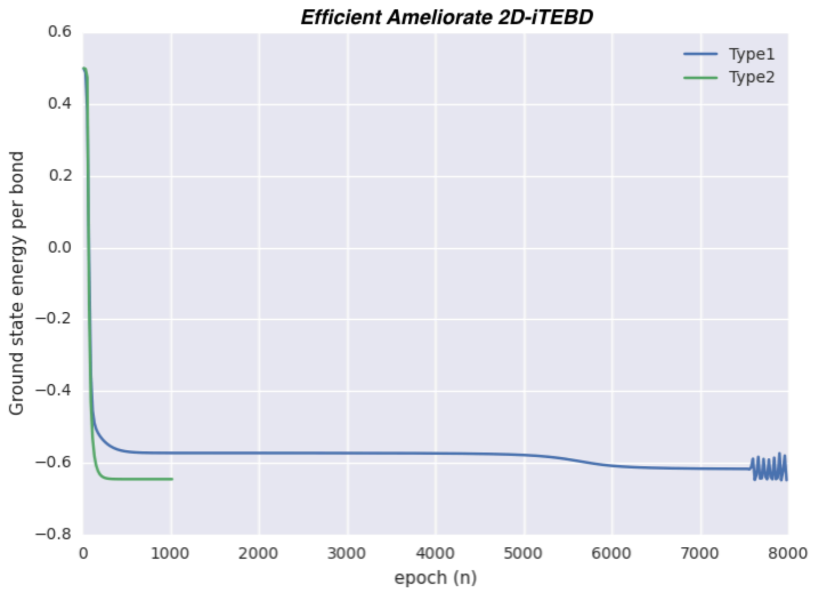


Figure 3.13: The Blue line represents updating tensors from the initial state shown in Fig 3.12 (i) and the green one represents updating from Fig 3.12 (ii)

3.5.2 Different schemes of 2D-iTEBD

In previous sections, we have introduced four different implementations of 2D-iTEBD, *Simple Update*, *Ameliorate Simple Update*, *Simple Update with QR decomposition* and *Ameliorate Simple Update with QR*. In Fig. 3.14, we notice that the methods based on Hastings scheme is more stable than the others which are build on *Simple Update*. The Hastings ways can converge rapidly in 1,000 epochs. However, due to multiplying too many pseudo-inverse matrices, the others methods not only can not converge in 10,000 epochs, but also be broken in the end. Nevertheless, according to the result shown in Fig. 3.14, we recognize that the QR decomposition is independent on the stability but it make the algorithms more efficient. See Fig. 3.15, due to the decrement of the dimension of the tensor Θ, the cost time does not increase exponentially with virtual bond dimensions. Therefor, we can simply study the two dimensional systems with a sufficient large virtual bond dimensions χ, which means that we can obtain the better ground states more quickly and the measurement can be calculated more accurate with larger χ.

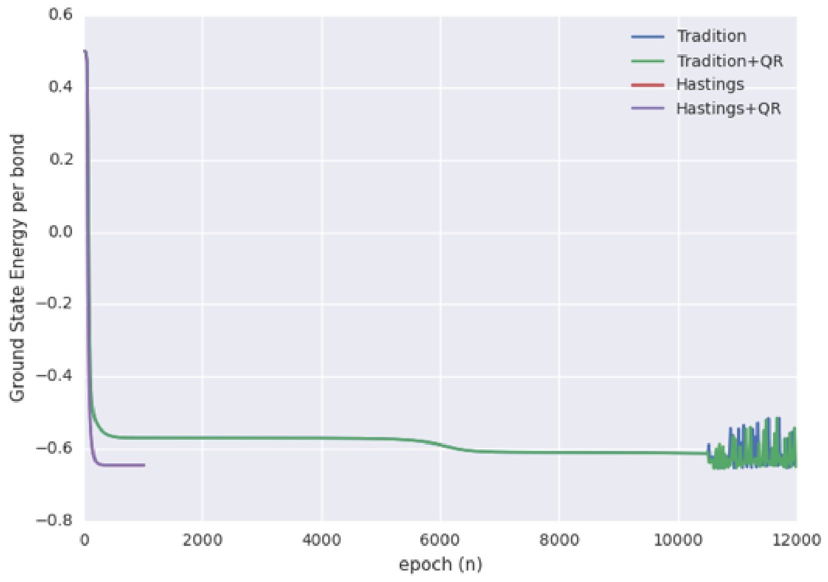


Figure 3.14

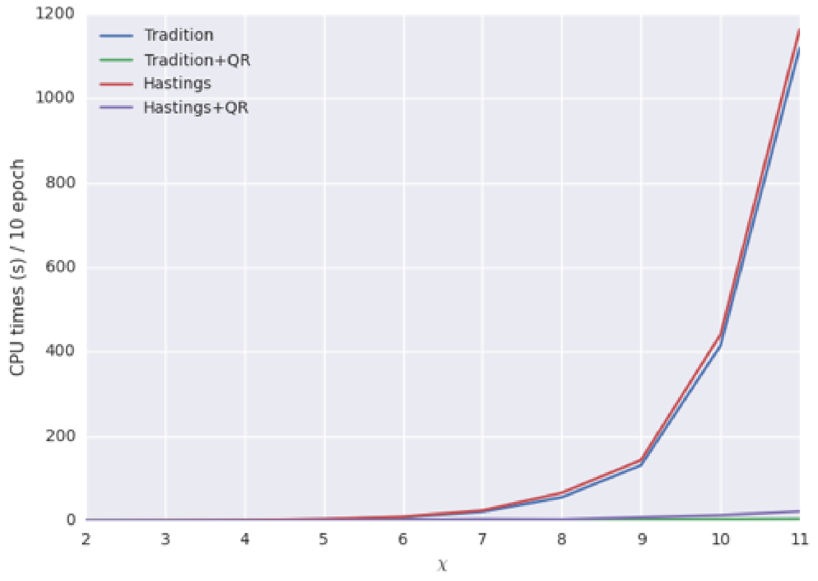


Figure 3.15

3.6 Set the cutoff of the truncation error

To improve the stability of the algorithms, we can set a cutoff ε of the truncation error. In Fig. 3.16, we set $\varepsilon = 10^{-7}$ and the methods base on traditional simple update would not be broken and converge in 2300 epochs. Intuitively, setting the smaller cutoff, obtaining the more accurate ground states. In fact, it is inefficient and unnecessary. As shown in Fig. 3.17, to avoid breaking, the requirement of the virtual bond dimensions increase. It not only waste the consumption and reduce efficiency, but also effects the stability.

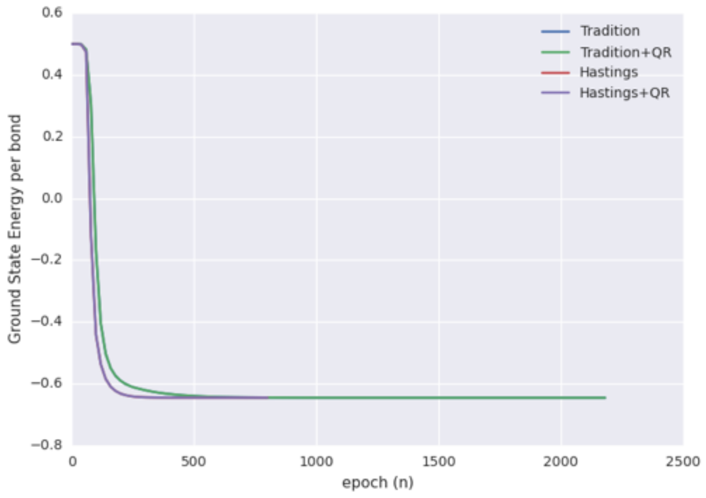


Figure 3.16

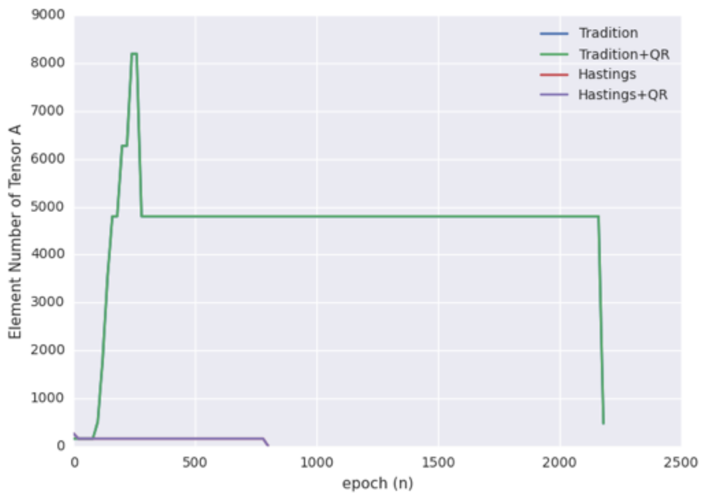


Figure 3.17

Chapter 4

Infinite Projected Entangled Simplex State

4.1 Simplex-solid State

The simplex-solid state of an SU(N) quantum anti-ferromagnet was derived by Arovas [] and the most significant conclusion is that any simplex states could be described by a natural generalization of the AKLT [] valence bond solid state, which means that the bond signlets of the AKLT could be extended to N-site simplices.

The concept of simplex-solid states ansatz were introduce by Xie et al []. The tensor-network representation of simplex states is called projected entangled simplex state (PESS) which is also considered as an extension of PEPS [] obeying the area law of entanglement entropy and characterizing any states if the dimension of the virtual bonds are large enough. The difference from the PEPS is that the entanglement among the virtual particles is described by entangled simplex tensors which depend on the structure of PESS. For example, in Fig. 4.1(b) there are three virtual particles within the simplex state, so the entangled simplex tensor S_{mnl} is a rank-3 tensor.

For illustrating how to write down the formulation of the PESS wave function, we

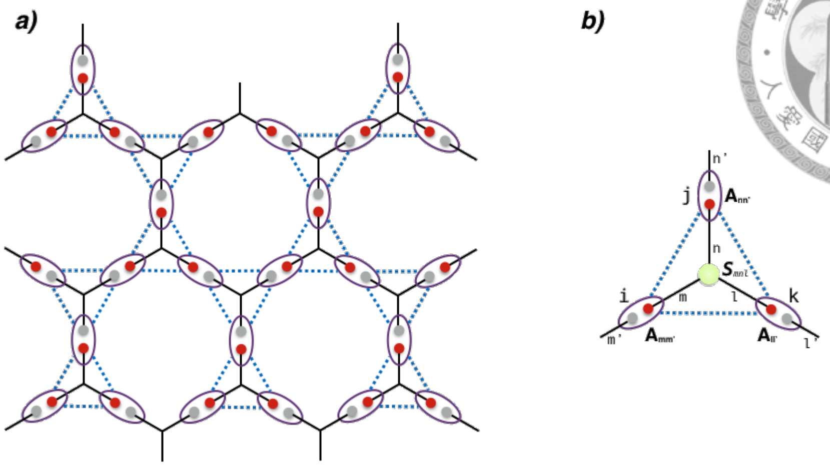


Figure 4.1: The red and blue tensor denotes on *odd* and *even* sites. The yellow one are time evolution operators $e^{-\tau H_{k,k+1}}$, $e^{-\tau H_{k+1,k}}$

begin from a simple system, the spin-2 simplex state on an infinite Kagome lattice []. See Fig. 4.1(a) each physical $S = 2$ states on the lattices could be treated as a symmetric superposition of two virtual $S = 1$ spins. According to the theory of AKLT, each neighbor simplices (triangles) share a single site symmetrically which also means that the $S = 1$ spins could be assigned to one of the simplices (vertex-sharing). Hence, there are three $S = 1$ spins in each simplex states. From the features of the $SU(2)$ group, the decomposition of the direct-product of three integer spins is written as,

$$n \otimes n \otimes n = [a_0 \times 0] \oplus \cdots \oplus [a_k \times k] \oplus \cdots \oplus [a_{3n} \times 3n] \tag{4.1}$$

$$a_k = \begin{cases} 2k+1 & , k \leq n \\ 3n+1-k & , k > n \end{cases} \quad , k = 0, 1, \ldots, 3n, \tag{4.2}$$

where a_k is a constant and k express the kth irreducible representation. Now that we can write down the product of the spins in the simplex as,

$$\underline{1} \otimes \underline{1} \otimes \underline{1} = \underline{0} \oplus (3 \times \underline{1}) \oplus (2 \times \underline{2}) \oplus \underline{3}, \tag{4.3}$$

and show that there is an unique spin-singlet state. The result encourage us to define a virtual singlet on simplex,

$$|\psi_\alpha\rangle = \frac{1}{\sqrt{6}} \sum_{\{s_i s_j s_k\}} S^{\alpha}_{s_i s_j s_k} \big|s_i, s_j, s_k\big\rangle, \tag{4.4}$$

where s_i, s_j, s_k are $S = 1$ virtual spins located at site i, j and k containing in the simplex α and $S^{\alpha}_{s_i s_j s_k}$ is the Levi-Civita antisymmetric tensor ε_{ijk} [Fig. 4.1(b)]. For mapping the two virtual $S = 1$ spins to the spin-2 subspace, we defined the projection operator P_i on each site,

$$P = \sum_{s_i, s_i'} \sum_{\sigma_i} A^{\sigma_i}_{s_i, s_i'} |\sigma_i\rangle \langle s_i, s_i'| \tag{4.5}$$

where $|\sigma_i\rangle$ is a basis of the $S = 2$ spin at site i. $A^{\sigma_i}_{s_i, s_i'}$ is a projected matrix whose components are filled by the Clebsch-Gordan coefficients,

$$\begin{aligned}
&A^{2}_{11} = A^{-2}_{33} = 1, \\
&A^{1}_{12} = A^{1}_{21} = A^{-1}_{23} = A^{-1}_{32} = \frac{1}{\sqrt{2}}, \\
&A^{0}_{13} = A^{0}_{31} = \frac{1}{\sqrt{6}}, \\
&A^{0}_{22} = \frac{2}{\sqrt{6}},
\end{aligned}$$

Now that we can write down the wave function of the simplex-solid state,

$$\begin{aligned}
|\Phi\rangle &= \bigoplus_i P_i \prod_\alpha |\psi_\alpha\rangle \qquad (4.6) \\
&= \mathrm{Tr}\Big(\ldots S^{\alpha}_{s_i s_j s_k} A^{\sigma_i}_{s_i, s_i'} A^{\sigma_j}_{s_j, s_j'} A^{\sigma_k}_{s_k, s_k'} \ldots\Big) \big|\ldots \sigma_i \sigma_j \sigma_k \ldots\big\rangle . \qquad (4.7)
\end{aligned}$$

and apply the tensor-network representation to descrbe it, see in Fig. 4.1(b). This structure could be extended to any higher integer spins. In conlusion, a physical $S = 2n$ (even-integer) spin is considered as a symmetric superposition of two virtual $S = n$ spins and a $S = 2n - 1$ (odd-integer) one is regarded as a symmetric superposition of a virtual $S = n - 1$ spin and a virtual $S = n$ spin [Fig. 4.1(a)]. More details are included in reference [] [].

4.2 Variational PESS ansatz

In this section, we will employ the formulation of the PESS wave function as a variational ansatz. The PESS ansatz is similar to the imaginary time evolution discussed in chapter.3. However, unlike the PEPS algorithms, we apply a higher rank tensor S to describe the entanglement entropy among virtual particles in a simplex state. Due to the difference, we use *high-order singular value decomposition* (HOSVD) rather than SVD to decompose wave function.

4.2.1 High-order singular value decomposition

In the section, we will introduce to Nth-order singular value decomposition (HOSVD) which is proposed for decomposing rank-N tensors and show the pseudocode to illustrate the scheme of the implementation.

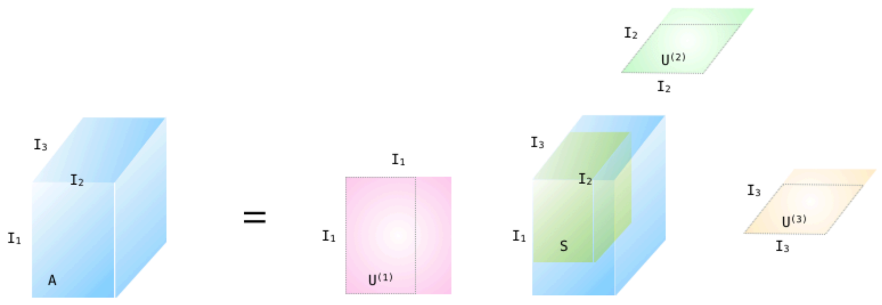


Figure 4.2: The picture of HOSVD for a rank-3 tensor A. $U^{(1)}$,$U^{(2)}$ and $U^{(3)}$ are unitary matrices and S (yellow cuboid) is a core tensor.

According to the theorem of HOSVD [], every *complex* $(I_1 \times I_2 \times \cdots \times I_N)$-tensor A could be decompose to a *core tensor* S and other n-mode unitary matrix $U^{(n)}$, where the maximum of n must be smaller than N,

$$A = S \times U^{(1)} \times U^{(2)} \times \cdots \times U^{(n)} \tag{4.8}$$

As shown in Fig. 4.2, a rank-3 tensor A is decomposed to a core tensor S (yellow cuboid) and three unitary matrices obtaining from three different modes. The core tensor S is a *complex* $(I_1 \times I_2 \times \cdots \times I_N)$-tensor and have some significant properties,

1. Row-orthogonal: Two subtensors $S_\alpha^{(n)}$ and $S_\beta^{(n)}$ are orthogonal when $\alpha \neq \beta$.

   $$\left\langle S_\alpha^{(n)}, S_\beta^{(n)} \right\rangle \quad \text{, if} \quad \alpha = \beta \tag{4.9}$$

   In other words, no matter the shape of S, each rows are orthogonal.

2. Ordering: The Frobenius-norms $\left\| S_i^{(n)} \right\|$ is ordered from large to small.

   $$\left\| S_1^{(n)} \right\| \geq \left\| S_2^{(n)} \right\| \geq \cdots \geq \left\| S_{I_n}^{(n)} \right\| \geq 0, \tag{4.10}$$

   for all possible values of n.

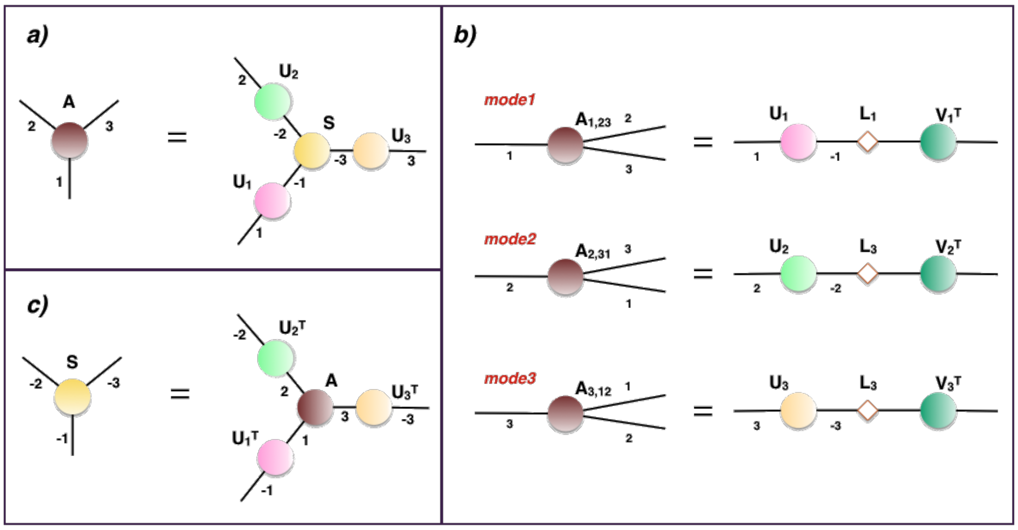


Figure 4.3: (a) Decompose A to a core tensor S and n-mode unitary matrix $U^{(n)}$, (b) Obtain tensors $U^{(n)}$ from decomposing various modes of tensor A by SVD, (c) Obtain the core tensor S from contracting all transpose unitary tensors $U^{(n)T}$

Now that we show the tensor-network representation for explaining more explicitly. If the target is to decompse a rank-3 $(N = 3)$ tensor A by 3 $(n = 3)$ modes, see in Fig .4.3(a), we can implement is by the following steps,

1. Reshaping the tensors to each modes and obtain $U^{(n)}$ by SVD , as shown in Fig. 4.3(b).

2. Contract all $U^{(n)T}$ tensors for getting S express more mathematically,

$$S = A \times U^{(1)T} \times U^{(2)T} \times \cdots \times U^{(n)T} \tag{4.11}$$

, see Fig. 4.3(c).

4.2.2 Simple update for PESS

In chapter.3, we have introduced an efficient approximation, the ”simple update”, for determining the PEPS wave function. In principle, the wave function of PESS can be approached by the same way. In PEPS structure, the ground-state wave function is approximated by iterated application of imaginary-time evolution operators $U(\tau) = e^{-\tau H}$ on an random PEPS state $|\Psi_t\rangle$, where τ is a small constant, see Fig. 3.2. Base on the scheme, we can obtain the ground-state wave function of PESS by following steps,

1. Split Hamiltonian H,

$$H = H_\alpha + H_\beta \tag{4.12}$$

where α and β are dependent on the geometry of the PESS structure.

2. Obtain the evolution operator $U(\tau)$ by Trotter-Suzuki decomposition.

$$e^{-\tau H} = e^{-\tau H_\alpha} e^{-\tau H_\beta} + O(\tau^2). \tag{4.13}$$

3. Absorb the environment bond vectors, which are considered as the entanglement between each simplex states, into projection tensors, and contract all projection tensors in the simplex state with the core tensor to obtain a cluster tensor.

4. Apply the evolution operator U to the cluster tensor for obtaining a new cluster tensor.

5. Decompose the new cluster tensor by HOSVD.

6. Truncate all the projection tensors and the core tensor.

7. Absorb the inverse bond vectors for removing the entanglement on the projection tensors.

which is similar to simple update of PEPS. In the following subsections, we applied various cases to explain it more explicitly.

4.3 Infinite Kagome lattice

In the previous section, we have briefly introduced to the procedures of PESS ansatz. The first step is to choose an suitable n-PESS structure, where n is the number of projection tensors in a simplex, to split the Hamiltonian and describe the system. There are many various graphical representation for the Kagome lattice [Fig. 4.1(a)], such as 3-PESS, 5-PESS and 9-PESS, shown in ref. []. In this section, we will discuss more details about 3-PESS and 5-PESS.

4.3.1 3-PESS

In the 3-PESS case, the PESS state can be drawn as the Fig. 4.4(a), which pictures that the many-body states can be described by composing two different simplices, upward- and downward-oriented triangular as shown in Fig 4.4(b). Each simplices contain three rank-3 projection tensors, $A^{\sigma_i}_{mm'}$ $A^{\sigma_j}_{nn'}$ and $A^{\sigma_k}_{ll'}$ and a rank-3 core tensor, S^{α}_{mnl} or $S^{\beta}_{m'n'l'}$. Therefore, the form of the Hamiltonian is re-written by eq. 4.12,

$$H = H_\alpha + H_\beta \quad ,\text{where} \quad H_\alpha = H_\Delta, \quad H_\beta = H_\nabla \tag{4.14}$$

Next, utilize the imaginary-time evolution operator $e^{-\tau H}$ to an random initial state $|\Psi_0\rangle$ iteratively. According to the theorem of Trotter-Suzuki decomposition, the evolution operator can be decomposed into two product terms, when $\tau \to 0$. Hence, the evolution operator operator of 3-PESS can be written as,

$$e^{-\tau H} = e^{-\tau H_{\triangle}} e^{-\tau H_{\nabla}} + O(\tau^2) \tag{4.15}$$

when the value of τ is small enough.

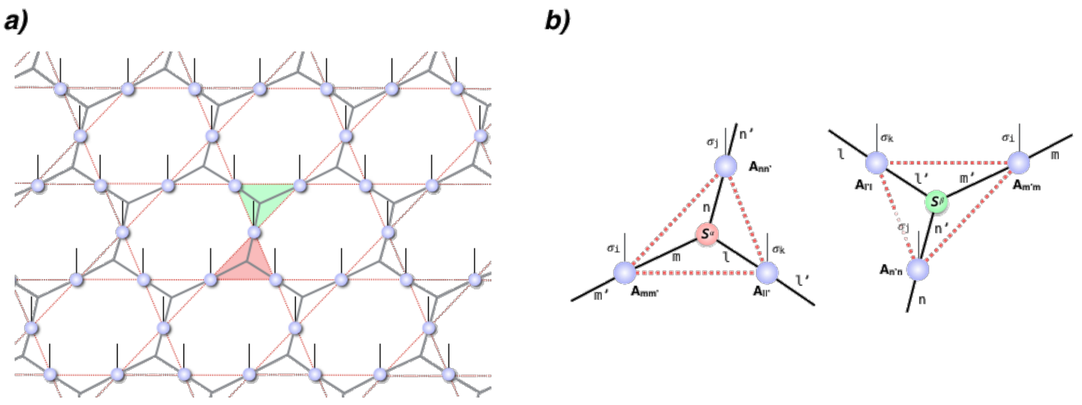


Figure 4.4: The graphical representation of 3-PESS. (a) The PESS state can be considered as composed by two simplices, upward- (green triangular) and downward-triangular (green triangular). The red dash-line represent as the geometry of the kagome lattice. As this figure shown, the projection tensors (Blue circle) are rank-3 with the dimension dD^2, where d is the dimension of the physical basis and D is the dimension of the virtual bonds (gray line), and the entangled simplex tensors, with the dimension D^3, are located at the cross of three virtual bonds (gray line) in each simplex states. (b) Two simplices in 3-PESS structure. These two type simplices, clustered with the sharing projection tensors, $A^{\sigma_i}_{mm'}$, $A^{\sigma_j}_{nn'}$ and $A^{\sigma_k}_{ll'}$. However, their entangled simplex tensor are individual. In (a), the core tensor of the red simplex is S^{α}_{mnl} and the green one is $S^{\beta}_{m'n'l'}$

After determining the evolution operators, $e^{\tau H_{\triangle}}$ and $e^{\tau H_{\nabla}}$, we apply the schem of the simple update to approximate theground state wave function of 3-PESS. As shown in Fig. 4.6, we take an example of updating upward-triangular simplex α to illustrate,

1. Obtain a clustered simplex tensor θ: Firstly, absorb all the environment bond vectors surrounding the simplex state. See Fig. 4.5(a), the environment vectors, $\lambda^{\beta}_{m'm}$, $\lambda^{\beta}_{n'n}$ and $\lambda^{\beta}_{l'l}$, are obtained from the entangled simplex tensor of simplex β, due to the

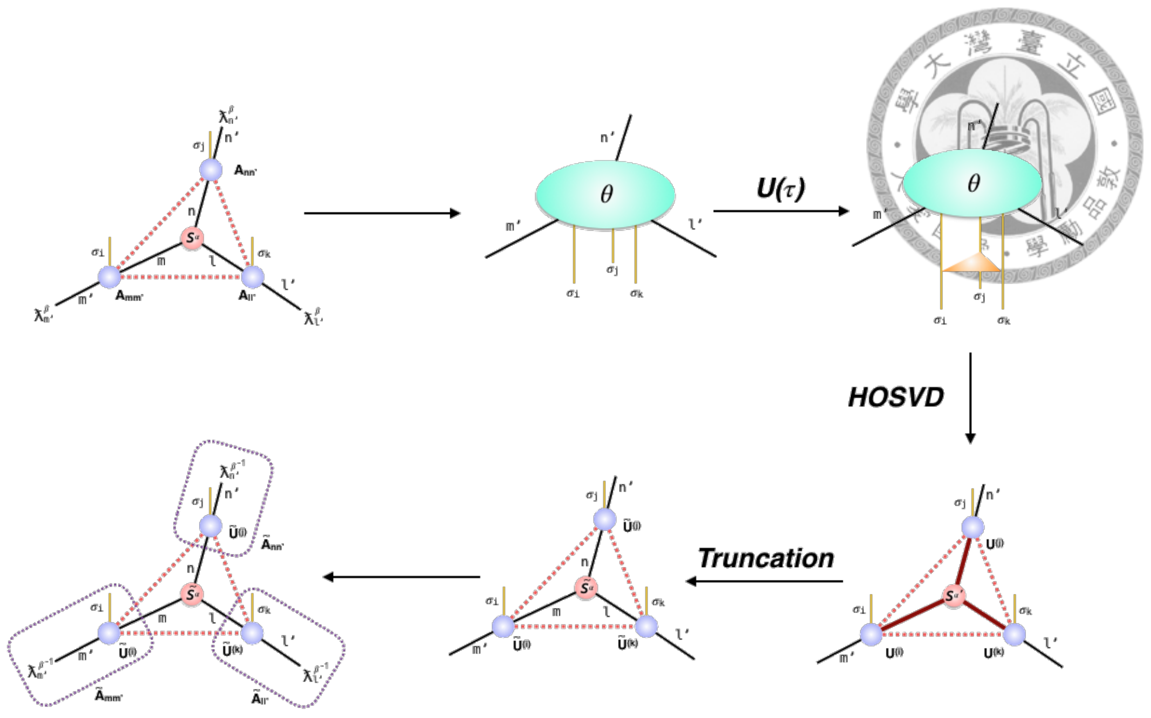


Figure 4.5: The schem of the simple update for 3-PESS. The more detail descriptions are in the paragraph

canonical form of the PESS [],

$$\sum_{n,l} S^{(\beta)}_{mnl}\left(S^{(\beta)}_{m'nl}\right) = \delta_{m,m'}\lambda^{(\beta)^2}_{m} \tag{4.16}$$

$$\sum_{l,m} S^{(\beta)}_{mnl}\left(S^{(\beta)}_{mn'l}\right) = \delta_{n,n'}\lambda^{(\beta)^2}_{n} \tag{4.17}$$

$$\sum_{m,n} S^{(\beta)}_{mnl}\left(S^{(\beta)}_{mnl'}\right) = \delta_{l,l'}\lambda^{(\beta)^2}_{l} \tag{4.18}$$

and base on the features of HOSVD, shown in eq. **??**. The λs can be simply obtained, when we decompose the clustered tensors of the simplex β in the updating process. Secondly, contract all tensors in simplex state to obtain the clustered tensor θ. The mathematical representation of the step is written as,

$$\theta^{\sigma_i\sigma_j\sigma_k}_{m'n'l'} = \sum_{mnl,m'n'l'} S^{(\alpha)}_{mnl}\lambda^{(\beta)}_{m''}A^{\sigma_i}_{mm'}\lambda^{(\beta)}_{n''}A^{\sigma_j}_{nn'}\lambda^{(\beta)}_{l''}A^{\sigma_k}_{ll'} \tag{4.19}$$

2. Apply the evolution operator $U(\tau)$: In eq. 4.15, we have generated the evolution operator for 3-PESS ansatz. Due to updating upward-triangles simplex, $U(\tau)$ is

defined as,

$$U(\tau) = e^{-\tau H_\Delta} \tag{4.20}$$

Now that, we utilize $U(\tau)$ to the cluster tensor θ, see Fig. 4.5(c) and obtain a new cluster tensor θ',

3. Decompose θ' into the general form of the simplex state: Apply the high-order singular value decomposition (HOSVD) to obtain new projection tensors, $U^{(\sigma_i)}$, $U^{(\sigma_j)}$ and $U^{(\sigma_k)}$, and a new core tensor of simplex α, $S^{(\alpha)'}$, see Fig. 4.5(d). During operating HOSVD, we need save the environment bond vectors, $\lambda_m^{(\alpha)}$, $\lambda_n^{(\alpha)}$ and $\lambda_l^{(\alpha)}$, surrounding the simplex β. All $\lambda^{(\alpha)}$ could be obtained from decomposing different modes of θ'. As shown in Fig. 4.3(b).

4. Truncation: In order to avoid the exponential increment of the virtual bonds dimension, we need truncate the brown bonds in Fig. 4.5(d) to specified dimension D' which can be fixed to original dimension D or determined dynamically by setting an truncation error as the discussion in section **??**. The simple way is to truncate projection tensors firstly,

$$U^{\sigma_i} \to \widetilde{U}^{\sigma_i}_{mm'} \tag{4.21}$$

$$U^{\sigma_j} \to \widetilde{U}^{\sigma_j}_{nn'} \tag{4.22}$$

$$U^{\sigma_k} \to \widetilde{U}^{\sigma_k}_{ll'} \tag{4.23}$$

Next, contract the transpose of these tensors with θ', as Fig. 4.3(c), to obtain $\widetilde{S}^{\alpha}$ in Fig. 4.5(e), where

$$\widetilde{S}^{(\alpha)} = \sum_{m'n'l'} \theta'^{\sigma_i\sigma_j\sigma_k}_{m'n'l'} \widetilde{U}^{\sigma_i}_{mm'} \widetilde{U}^{\sigma_j}_{nn'} \widetilde{U}^{\sigma_k}_{ll'} \tag{4.24}$$

5. Absorb the inverse environment bond vectors $\lambda^{(\beta)^{-1}}$ into truncated projection tensors: To obtain the updated projection tensors $\widetilde{A}^{\sigma_i}_{mm'}$, $\widetilde{A}^{\sigma_j}_{nn'}$ and $\widetilde{A}^{\sigma_k}_{ll'}$, we need remove

the influence of environment,

$$\widetilde{A}^{\sigma_i}_{mm'} = \sum_{m''} \lambda^{(\beta)}_{m'm''} \widetilde{U}^{\sigma_i}_{mm''} \tag{4.25}$$

$$\widetilde{A}^{\sigma_j}_{nn'} = \sum_{n''} \lambda^{(\beta)}_{n'n''} \widetilde{U}^{\sigma_j}_{nn''} \tag{4.26}$$

$$\widetilde{A}^{\sigma_k}_{ll'} = \sum_{l''} \lambda^{(\beta)}_{l'l''} \widetilde{U}^{\sigma_i}_{ll''} \tag{4.27}$$

see Fig. 4.5(f),

At here, one updated epoch is completed. Finally, the ground state will be obtained by iterating the procedures for each simplex, upward- and downward- triangular (Δ, ∇), until the wave function of PESS converge,

4.3.2 5-PESS

In the 5-PESS structure [Fig. 4.6(a)], the procedure to approximate the ground state wave function is similar to 3-PESS. However, due to the transformation of the structure, the simplices should be re-defined. In the 5-PESS ansatz, each simplex state is composed by an upward- and a downward-triangular and the most significant change is that the core tensors are located on the physical sites. As shown in Fig. 4.6(b), each simpliices contain 4 projection tensors $A^{\sigma_i}_{ii'}$ $A^{\sigma_j}_{jj'}$ $A^{\sigma_k}_{kk'}$ and $A^{\sigma_l}_{ll'}$, and a core tensor, $S^{\sigma_s(\alpha)}_{ijkl}$ or $S^{\sigma_s(\beta)}_{i'j'k'l'}$. Therefor, we can split the Hamiltonian into

$$H = H_\alpha + H_\beta \quad \text{,where} \quad H_\alpha = H_{{\color{red}⧖}}, H_\beta = H_{{\color{green}⧖}} \tag{4.28}$$

and as previous section, the evolution operator is written as,

$$e^{-\tau H} = e^{-\tau H_{{\color{red}⧖}}} e^{-\tau H_{{\color{green}⧖}}} + O(\tau^2) \tag{4.29}$$

when $\tau \to 0$.

Next, apply the simple update scheme as Fig. 4.7. Most of steps are similar to 3-PESS,

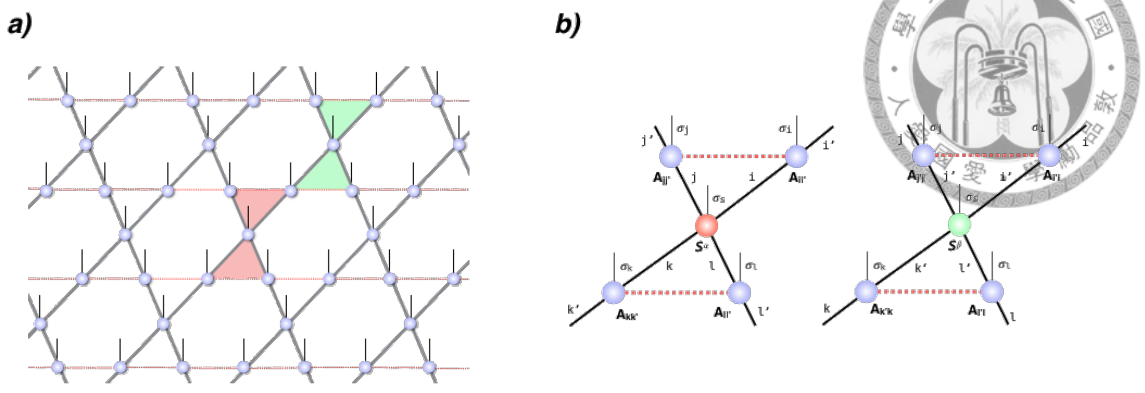


Figure 4.6: The graphical representation of 3-PESS. (a) The PESS state can be considered as composed by two simplices, upward- (green triangular) and downward-triangular (green triangular). The red dash-line represent as the geometry of the kagome lattice. As this figure shown, the projection tensors (Blue circle) are rank-3 with the dimension dD^2, where d is the dimension of the physicla basis and D is the dimension of the virtual bonds (gray line), and the entangled simplex tensors, with the dimension D^3, are located at the cross of three virtual bonds (gray line) in each simplex states. (b) Two simplices in 3-PESS structure. These two type simplices, clustered with the sharing projection tensors, $A^{\sigma_i}_{mm'}$, $A^{\sigma_j}_{nn'}$ and $A^{\sigma_k}_{ll'}$. However, their entangled simplex tensor are individual. In (a), the core tensor of the red simplex is S^{α}_{mnl} and the green one is $S^{\beta}_{m'n'l'}$

shown as Fig. 4.5. However, we should carefully decompose the cluster tensor θ', where

$$\theta' = \theta \times U(\tau) \tag{4.30}$$

see Fig. 4.7(d). In this structure, the core tensors contain physical basis states, which means that it can't be decompose by the simply way shown in Fig. 4.3, due to

$$N \mod n \neq 0 \tag{4.31}$$

where N is the rank of the tensor θ' and n is the mode number of HOSVD. Hence, the bond σ_s, belonging to the core tensor $S^{(\alpha)}_{i'j'k'l'}$ in tensor θ', must be fixed during the decomposition. The more detail and general form of the HOSVD are written in the documentation of *Uni10 Library* [], which is implemented by c/c++ and helpful for calculating the high-rank linear algebra problems.

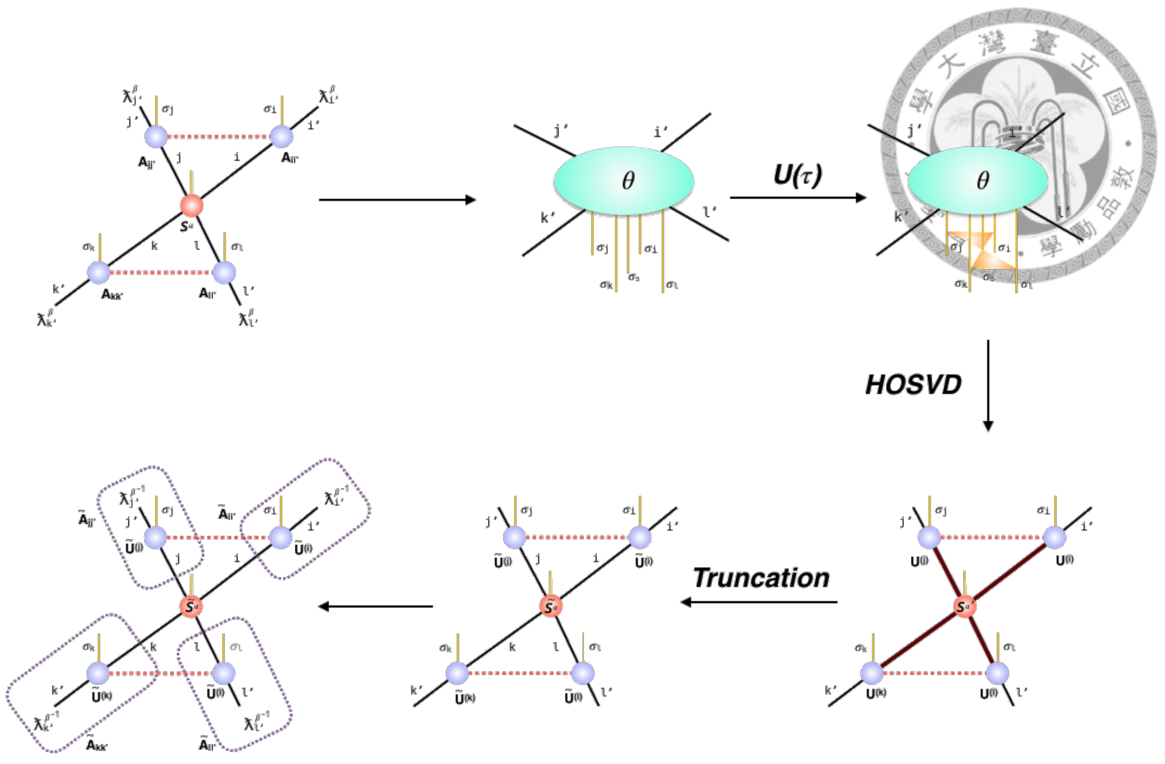


Figure 4.7: The schem of the simple update for 5-PESS. The more detail descriptions are in the paragraph

4.4 Infinite Square lattice

In previous sec. 4.3, we have verified the results in ref. [] []. In order to test and recognize the properties of the PESS ansatz more explicitly, we extend it to simulate infinite square systems and compare it with methods discussed in chap. 3.

4.4.1 4-PESS (Rank-3 projection tensors)

The 4-PESS structure could be build by two different methods, which is drawn in the ref. []. In this section, we use the concept, shown in Fig. 4.8(a), to complete the implementation.

See Fig. 4.8(a), the many-body state can be regarded as composed by two different simplices, shown in Fig. 4.8(b), repeatedly. The components of each simplices are similar to 5-PESS, but the difference is that the core tensors, $S^{(\alpha)}_{ijkl}$ and $S^{(\beta)}_{i'j'k'l'}$, are not located on the lattice sites. Hence, the Hamiltonian is split into,

$$H = H_\alpha + H_\beta \quad \text{, where} \quad H_\alpha = H_{\color{red}\square}, \quad H_\beta = H_{\color{green}\square} \tag{4.32}$$

where ${\color{red}\square}$ and ${\color{green}\square}$ are represented simplex α and β, shown in Fig. 4.8(b). Therefor, the evolution operator can be written as,

$$e^{-\tau H} = e^{-\tau H_{\color{red}\square}} e^{-\tau H_{\color{green}\square}} + O(\tau^2) \tag{4.33}$$

when τ is a small constant $\rightarrow 0$.

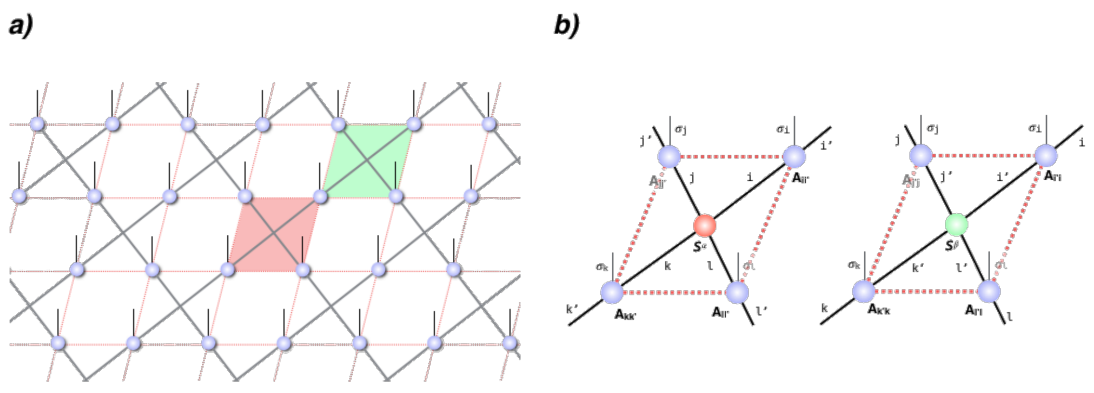


Figure 4.8: The graphical representation of 3-PESS. (a) The PESS state can be considered as composed by two simplices, upward- (green triangular) and downward-triangular (green triangular). The red dash-line represent as the geometry of the kagome lattice. As this figure shown, the projection tensors (Blue circle) are rank-3 with the dimension dD^2, where d is the dimension of the physicla basis and D is the dimension of the virtual bonds (gray line), and the entangled simplex tensors, with the dimension D^3, are located at the cross of three virtual bonds (gray line) in each simplex states. (b) Two simplices in 3-PESS structure. These two type simplices, clustered with the sharing projection tensors, $A^{\sigma_i}_{mm'}$, $A^{\sigma_j}_{nn'}$ and $A^{\sigma_k}_{ll'}$. However, their entangled simplex tensor are individual. In (a), the core tensor of the red simplex is S^{α}_{mnl} and the green one is $S^{\beta}_{m'n'l'}$

Finally, the ground state wave function can be obtained by repeating the following steps, shown in Fig. 4.9, on the simplices, α and β (${\color{red}\square}$, ${\color{green}\square}$), until the wave function of PESS converge.

Before we do more restrict comparisons, it is obvious that the control of dimensional increment in the 4-PESS is better than in the PEPS [Fig. 3.4], In 4-PESS structure, the projection tensors are described by rank-3 tensors whose dimension is dD^2. However, in the PEPS representation, the dimension of local tensors is dD^4. The reduction of tensors dimension is helpful for calculating and studying with a significant larger dimension. Nevertheless, it doesn't also mean that the 4-PESS algorithm is more efficient than PEPS one and actually it encounter in some problems when approximating the environment. We will show mare details in following sections.

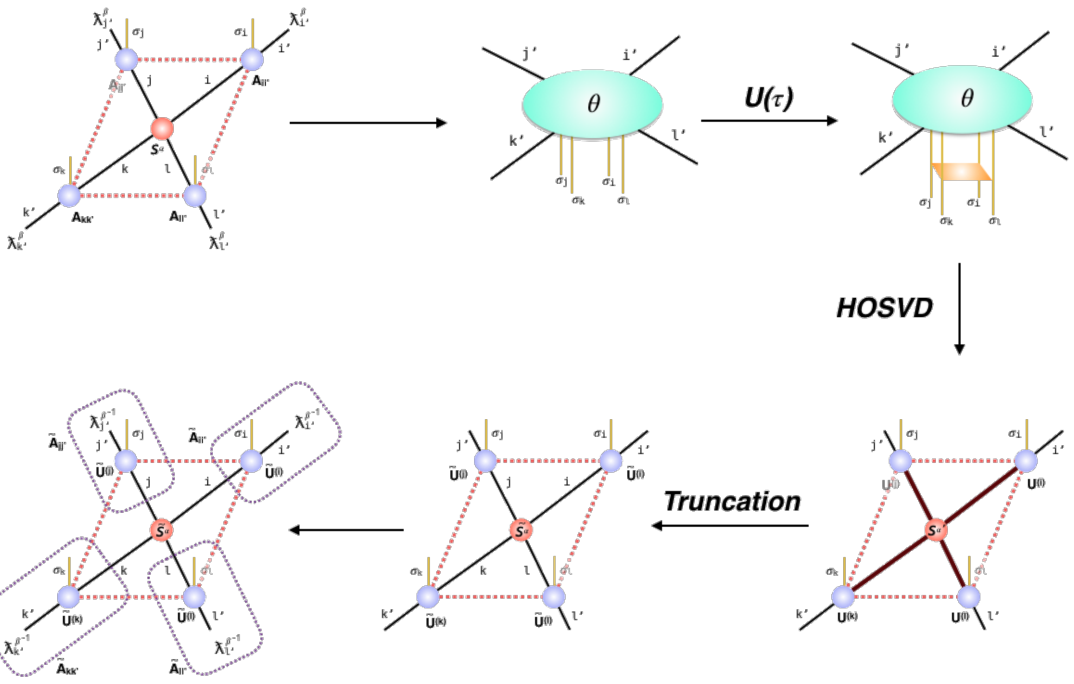


Figure 4.9: The scheme of the simple update for 4-PESS, composed by rank-3 projection tensors. The more detail descriptions are in the paragraph

4.5 Properties of PESS algorithm

4.5.1 3PESS on infinite Kagome lattice

Before applying the PESS algorithms to study square lattice systems, we should know and verify some features. At beginning, we present the results for the Heisenberg model on Husimi lattice [],

4.5.1.1 $S = 1$

In the case of $S = 1$ Heisenberg model on the Husimi lattice, the ground state energy per sites E_0 decrease with virtual dimension D and converge to $E_0 = -1.405$ rapidly, see Fig.4.10. However, according to the theory of the simplex solid states [Sec. 4.1], a physical $S = 2n - 1$ spin can be regarded as a symmetric superposition of a virtual $S = n - 1$ and a virtual $S = n$ spin. We predict that there are two different ground state energy between the two simplices (upward- and downward-triangular). As shown in Fig. 4.11, the energy difference

$$\Delta E = 2\frac{|E_\Delta - E_\nabla|}{3} \tag{4.34}$$

and the spontaneous magnetization

$$M = \frac{1}{N}\sum_i \sqrt{\langle S_x\rangle^2 + \langle S_y\rangle^2 + \langle S_z\rangle^2} \tag{4.35}$$

can be considered as two different order parameters. The energy difference ΔE rapid increase to a constant value and the magnetic order parameter M begin to rapid fall to zero almost at the same time, where $D = 8$. It means that to describe the simplices states, the requirement of virtual bond dimension D_c must be larger than 8. The results is also corresponds to the theory of the simplex solid states.

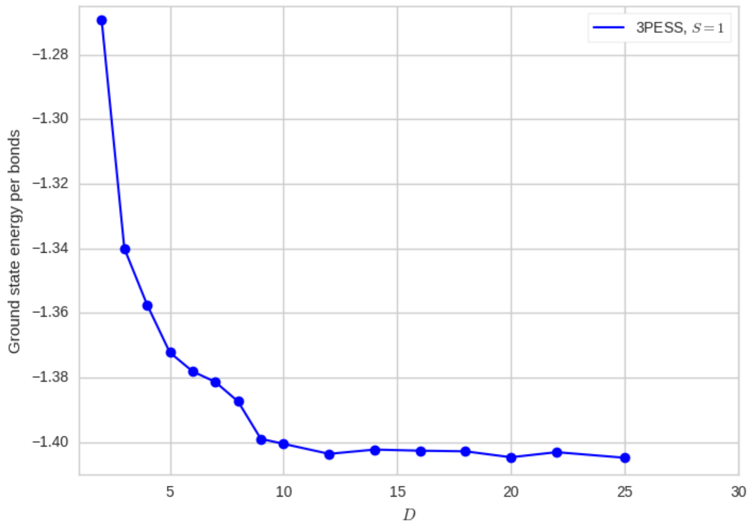


Figure 4.10: The scheme of the simple update for 4-PESS, composed by rank-3 projection tensors. The more detail descriptions are in the paragraph

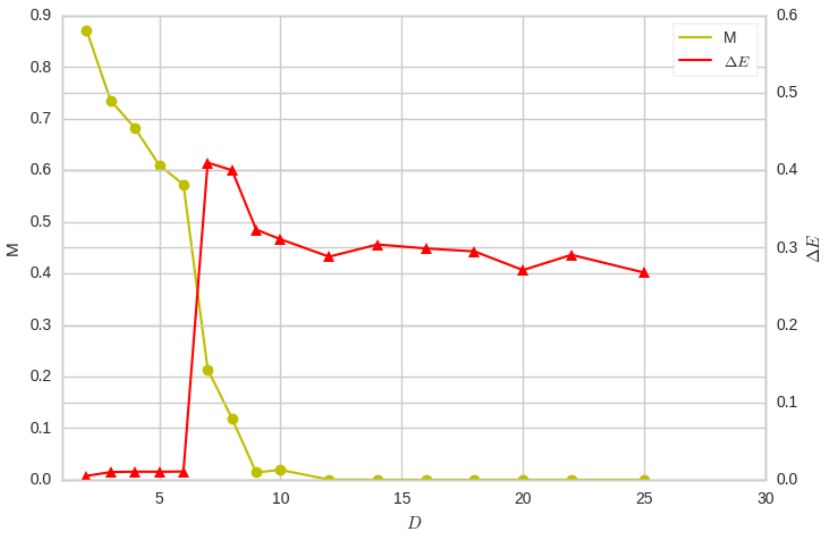


Figure 4.11: The scheme of the simple update for 4-PESS, composed by rank-3 projection tensors. The more detail descriptions are in the paragraph

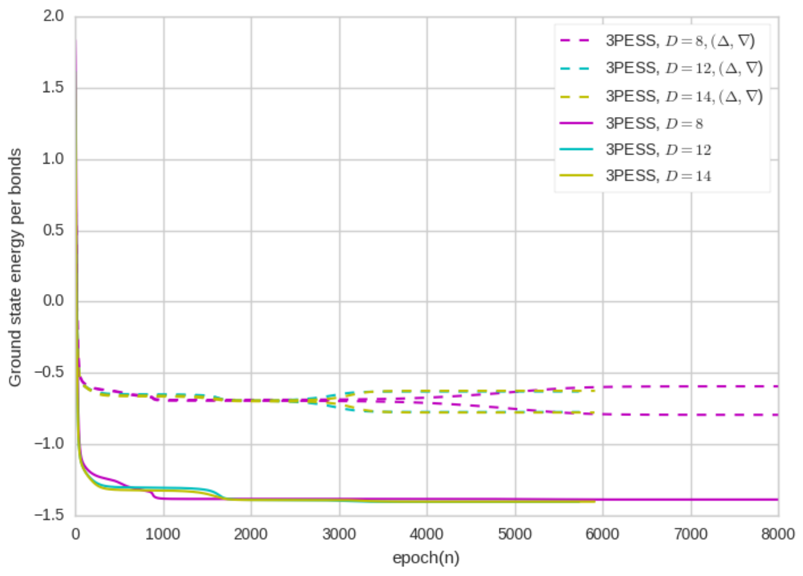


Figure 4.12: The scheme of the simple update for 4-PESS, composed by rank-3 projection tensors. The more detail descriptions are in the paragraph

4.5.1.2 $S = 2$

Turning to the $S = 2$ Heisenberg model, the ground state energy per site again converge to $E_0 = -4.8185$ [Fig. 4.13] when the dimension D is sufficient enough. Unlike the $S = 1$ case, each spins on sites can be considered as a symmetric superposition of two $S = 1$ spins, which means that the ground state is a uniform simplex-solid state. Hence, theoretically there is no energy gap between two different simplices (upward- and downward-triangle). Then see Fig. 4.14, we find again that the local magnetization M vanish suddenly when the virtual bond dimension $D \geq 8$. The result also prove that the ground state in the $S = 2$ case have no magnetic.

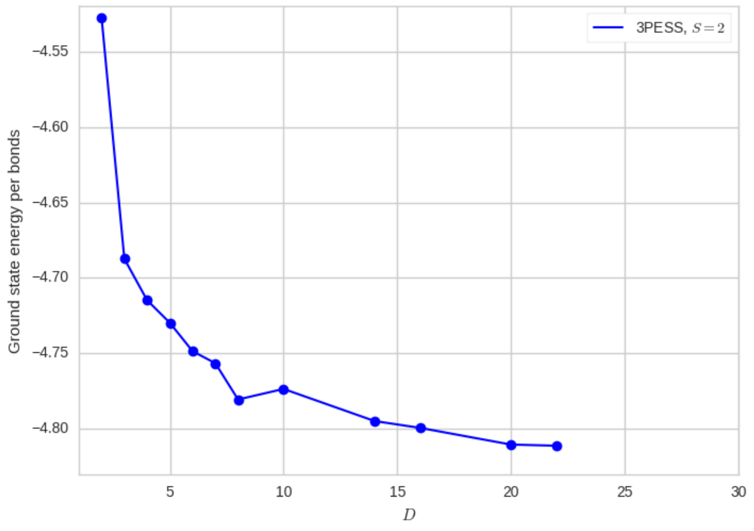


Figure 4.13: The scheme of the simple update for 4-PESS, composed by rank-3 projection tensors. The more detail descriptions are in the paragraph

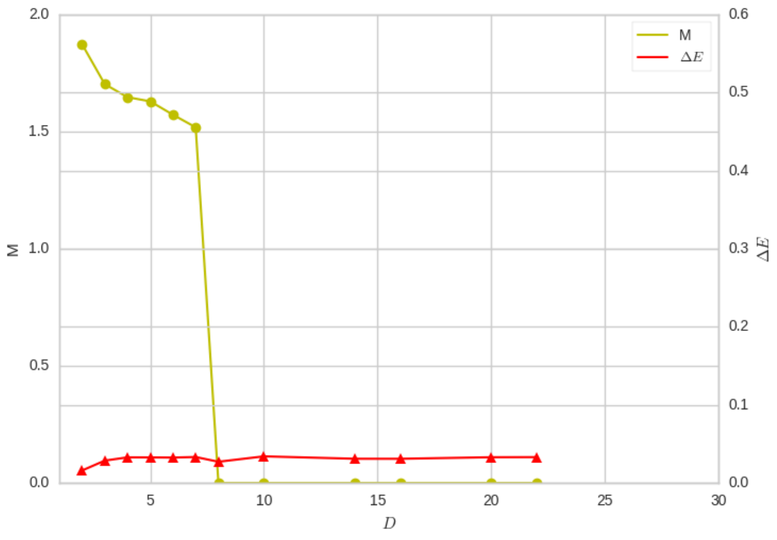


Figure 4.14: The scheme of the simple update for 4-PESS, composed by rank-3 projection tensors. The more detail descriptions are in the paragraph

4.5.2 4-PESS for Heisenberg model on square lattice

For the $spin - \frac{1}{2}$ Heisenberg model on square lattice, we also find that the accuracy is strongly dependent on the virtual bond dimension D, see Fig. 4.15. However, the 4-PESS ansatz is unstable with some specific dimension D, such as, 6,9 and so on. In these cases, the algorithm might hard converge or even be broken.

Our comparison among two dimensional algorithms is shown in Fig. 4.16. The algorithms started from $D = 2$ and setted the cutoff $\varepsilon = 10^{-7}$ to determine how many basis should be truncated. The results shows that the ground states obtain by 4-PESS ansatz is more accuracy then by 2D-iTEBD and the dimension of projection tensors is less than in PEPS.

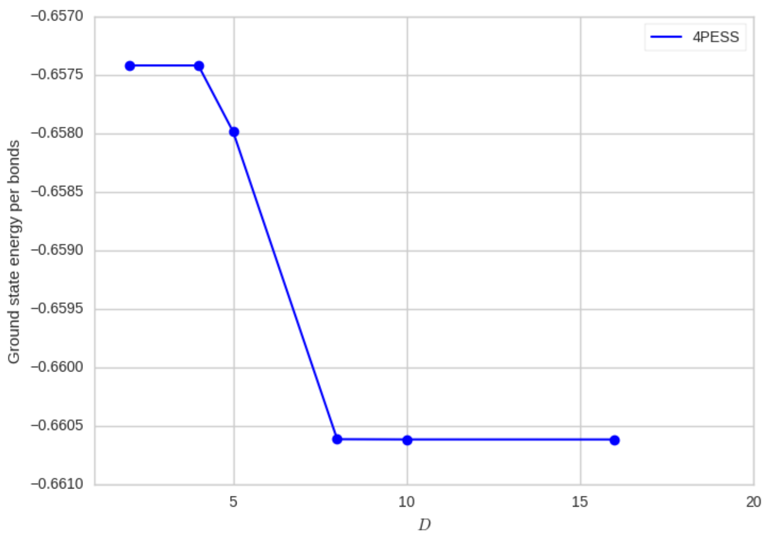


Figure 4.15: The scheme of the simple update for 4-PESS, composed by rank-3 projection tensors. The more detail descriptions are in the paragraph

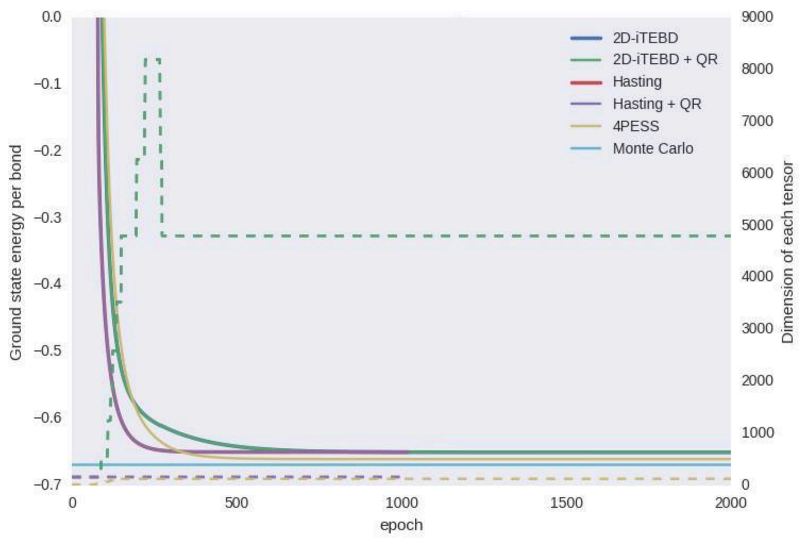


Figure 4.16: The scheme of the simple update for 4-PESS, composed by rank-3 projection tensors. The more detail descriptions are in the paragraph

Chapter 5

Corner Transfer Matrix

The *cerner transfer matrix renormalization group* (CTMRG) [] is an algorithm to numerically compute the *effective environments* which is an approximation of the environment of systems. For example, if the infinite PEPS is composed by a single tensor A^{h}_{uldr} repeatedly, where h express a physical basis of $\mathbb{V}$ with dimension d, and u, l, d, r are virtual bonds with dimension D, see Fig. 5.1(a). Then we can represent the scale norm $\langle \psi \mid \psi \rangle$ by a simple two dimensional tensor network ε which is characterized by reduced tensors a, shown in Fig. 5.1(b). The reduced tensor a is defined as eq.5.2,

$$a \equiv \sum_{h=1}^{d} A_h \otimes A_h^* \tag{5.1}$$

The environment $\varepsilon^{[\vec{r}]}$ of the site $\vec{r}$ could be described by the reduced tensors in the gray rectangles in Fig.5.1(c) and the *effective environments* $G^{[\vec{r}]}$ shown in Fig. 5.1(d) is target of the CTMRG.

In the following subsections, we will show more details of implementation of CTM and compare some features between obtaining the states from iPEPE and PESS.

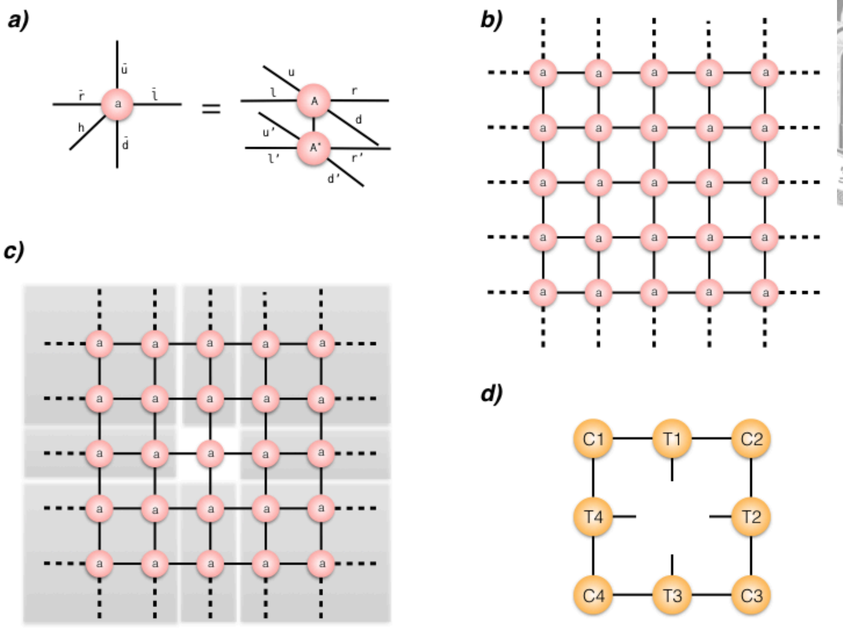


Figure 5.1: The red and blue tensor denotes on *odd* and *even* sites. The yellow one are time evolution operators $e^{-\tau H_{k,k+1}}$, $e^{-\tau H_{k+1,k}}$

5.1 Obtain States from PEPS

In chapter.3, we have discussed about obtaining the infinite PEPS state $|\psi\rangle$ of an infinite 2D square lattice by imaginary time evolution and known that the infinite PEPS could be characterized by two tensors A^h_{uldr} and B^h_{drul} repeatedly (Fig. 5.2(a)). The scaler norm of the iPEPS $\langle\psi\,|\,\psi\rangle$ is composed by reduced tensors $a_{\bar{u}\bar{l}\bar{d}\bar{r}}$ and $b_{\bar{d}\bar{r}\bar{u}\bar{l}}$(Fig. 5.2(b)), where

$$\bar{a} \equiv \sum_{h=1}^{d} A_h \otimes A^*_h \tag{5.2}$$

$$\bar{b} \equiv \sum_{h=1}^{d} B_h \otimes B^*_h \tag{5.3}$$

Then, we can consider the environment $\varepsilon^{[\vec{r_1},\vec{r_2},\vec{r_3},\vec{r_4}]}$ of a four-site structure (Fig. 5.2(c)), and try to approximate it with effective environment $G^{[\vec{r_1},\vec{r_2},\vec{r_3},\vec{r_4}]}$ (Fig. 5.2(d)), which consists of

$$C_1, T_{a1}, T_{b1}, C_2, T_{a2}, T_{b2}, C_3, T_{a3}, T_{b3}, C_4, T_{a4}, T_{b4},$$

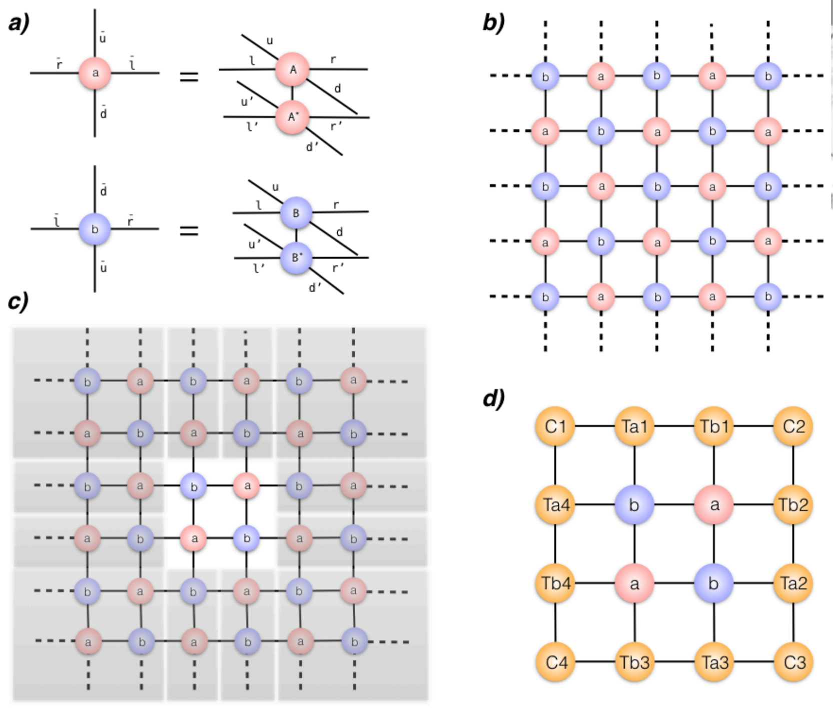


Figure 5.2: The red and blue tensor denotes on *odd* and *even* sites. The yellow one are time evolution operators $e^{-\tau H_{k,k+1}}$, $e^{-\tau H_{k+1,k}}$

For the approximation of environment, the directional variant of the CTMRG was developed. According to *directional coarse-graining moves*, the effective environment could be updated from four different moves , left, right, up and down and iterated until the environments converges.

For instance, the procedures to the left move, shown in the Fig. 5.3 which is derived by Roman and Vidal, is made up of four major steps,

1. Insertion: Insert two new columns which consist of $\{T_{a1}, b, a, T_{b3}\}$ and $\{T_{b1}, a, b, T_{a3}\}$ as in Fig. 5.3(b).

2. Absorption: In order to obtaining two new corner matrices $\tilde{C_1}$ and $\tilde{C_4}$, and two new transfer matrices $\tilde{T_{b4}}$ and $\tilde{T_{a4}}$, we contract tensors C_1 and T_{b1}, tensors C_3 and T_{a3}, tensors T_{a4} and b, and tensors T_{b4} and a. Then, contract tensors $\tilde{C_1}$ and $\tilde{T_{b4}}$, and tensors $\tilde{C_4}$ and $\tilde{T_{a4}}$, obtaining $\tilde{Q_1}$ and $\tilde{Q_4}$ which play significant rules for calculating isometries between $\tilde{T_{b4}}$ and $\tilde{T_{a4}}$ as in Fig. 5.3(c).

3. Renormalization: Truncate the vertical virtual bonds of $\widetilde{C_1}$, $\widetilde{T_{b4}}$, $\widetilde{T_{a4}}$, and $\widetilde{C_4}$ by contracting the isometries Z and W, where

$$Z^{\dagger}Z = I \tag{5.4}$$

$$W^{\dagger}W = I \tag{5.5}$$

and the renormalization of the left CTM, yield as

$$C_1' = Z^{\dagger}\tilde{C}_1 \tag{5.6}$$

$$T_{b4}' = Z\tilde{T}_{b4}W^{\dagger} \tag{5.7}$$

$$T_{a4}' = W\tilde{T}_{b4}Z^{\dagger} \tag{5.8}$$

$$C_4' = Z\widetilde{C_4} \tag{5.9}$$

See the Fig. 5.3(d) and 5.3(f).

4. Truncation: To determinate the isometries Z and W in the *renormalization* steps is the most significant part. In this case, we use the eigenvalue decomposition of

$$\tilde{C}_1\tilde{C}_1^{\dagger} + \tilde{C}_4\tilde{C}_4^{\dagger} = \tilde{Z}D_z\tilde{Z}^{\dagger} \tag{5.10}$$

$$\tilde{Q}_1\tilde{Q}_1^{\dagger} + \tilde{Q}_4\tilde{Q}_4^{\dagger} = \tilde{W}D_w\tilde{W}^{\dagger} \tag{5.11}$$

shown in Fig. 5.3(e). It's not hard to find that the the dimension of bonds of D_z and D_w increase to χ^2. For that reason, we have to truncate $\tilde{Z}$ and $\tilde{W}$ to isometries Z and W which are equivalent to keeping the columns of $\tilde{Z}$ and $\tilde{W}$ corresponding to χ largest eigenvalues of D_z and D_w.

Now, we need repeat the procedures in Fig. 5.3(b)-(d) again for absorbing the other inserted column in Fig. 5.3(d) and obtain a new effective environment $G'^{[\vec{r_1},\vec{r_2},\vec{r_3},\vec{r_4}]}$ for the four-site unit cell. By composing four variant moves of the CTM we build an epoch of CTMRG.

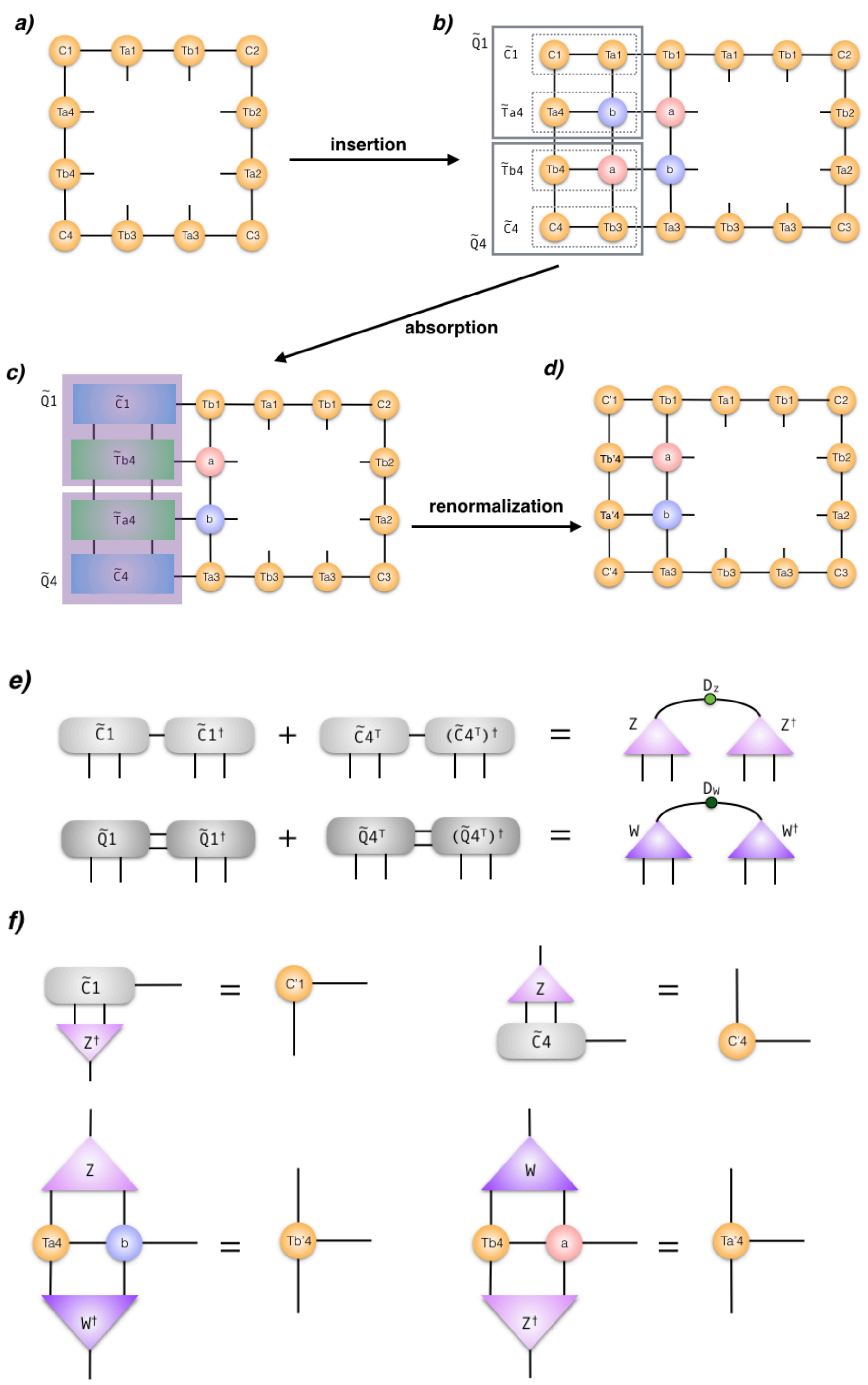


Figure 5.3: The red and blue tensor denotes on *odd* and *even* sites. The yellow one are time evolution operators $e^{-\tau H_{k,k+1}}$, $e^{-\tau H_{k+1,k}}$

5.2 Obtain States from PESS

In this section, we apply the CTM to approximate the effective environment of 4-PESS structure. Firstly, we must transform it to iPEPS structure which is suit for the form of CTM. As shown in Fig. 5.4, the projection tensors, $U^{[0]}$, $U^{[1]}$, $U^{[2]}$ and $U^{[3]}$, and the entangled simplex tensors, $S^{[\alpha]}$ and $S^{[\beta]}$ are obtained from 4-PESS ansatz. To map these states to PEPS-like structure, we group the tensors, $S^{[\alpha]}$, $U^{[0]}$ and $U^{[1]}$, in red rectangles into the tensor A,

$$A^{\sigma_i \sigma_j}_{i'j'kl} = \sum_{ij} U^{[0]}_{ii',\sigma_i} S^{[\alpha]}_{ijkl} U^{[1]}_{jj',\sigma_j} \tag{5.12}$$

and group, $S^{[\beta]}$, $U^{[2]}$ and $U^{[3]}$, in blue rectangles into the tensor B

$$B^{\sigma_k \sigma_l}_{i'j'kl} = \sum_{k'l'} U^{[2]}_{kk',\sigma_i} S^{[\beta]}_{i'j'k'l'} U^{[3]}_{ll',\sigma_j} \tag{5.13}$$

, whee the ranks of tensor A and B are six and there are two physical bonds contained in each of them. Hence, after combined the physical bonds in tensors A and B, the iPEPS structure will be obtained,

$$A^{\sigma_i \sigma_j}_{i'j'kl} \rightarrow A^{\sigma_A}_{i'j'kl} \tag{5.14}$$

$$B^{\sigma_k \sigma_l}_{i'j'kl} \rightarrow B^{\sigma_B}_{i'j'kl} \tag{5.15}$$

Next, in order to make the structure more balance, the entanglement should be well-distributed between each sites,

$$\widetilde{A} = \sum_{i'j'kl} \lambda_{i'}^{[\beta]^{\frac{1}{2}}} \lambda_{j'}^{[\beta]^{\frac{1}{2}}} A^{\sigma_A}_{i'j'kl} \lambda_{l}^{[\alpha]^{-\frac{1}{2}}} \lambda_{k}^{[\alpha]^{-\frac{1}{2}}} \tag{5.16}$$

$$\widetilde{B} = \sum_{i'j'kl} \lambda_{k}^{[\alpha]^{\frac{1}{2}}} \lambda_{l}^{[\alpha]^{\frac{1}{2}}} B^{\sigma_B}_{i'j'kl} \lambda_{i}^{[\beta]^{-\frac{1}{2}}} \lambda_{j'}^{[\beta]^{-\frac{1}{2}}} \tag{5.17}$$

and substitute $\widetilde{A}$ and $\widetilde{B}$ into Eq. 5.2 and Eq. 5.3 to obtain reduced tensors a and b. In the end, we apply these two reduced tensor to build the form of CTM and follow the procedures shown in Fig. **??** to simulate the effective environment tensors.

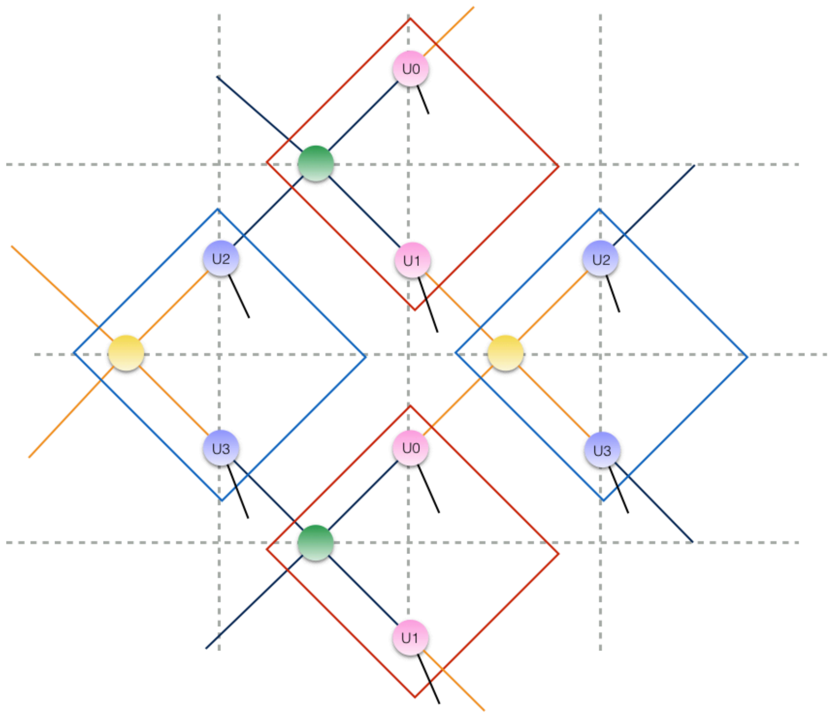


Figure 5.4: The red and blue tensor denotes on *odd* and *even* sites. The yellow one are time evolution operators $e^{-\tau H_{k,k+1}}$, $e^{-\tau H_{k+1,k}}$

5.3 Comparison

To compare the performance of the approximations, we have applied 2D-iTEBD and PESS to approach the ground state of the spin-1/2 quantum transverse Ising model,

$$H = -\sum_{<\vec{r},\vec{r}'>} \sigma_z^{[\vec{r}]}\sigma_z^{[\vec{r}']} - \lambda \sum_{\vec{r}} \sigma_x^{[\vec{r}]} \tag{5.18}$$

, and use directional CTM to obtain the effective environment at each sides. See Fig. 5.5, the order-parameter $m_z \equiv \langle\Psi|\,\sigma_z\,|\Psi\rangle$ as a function of the external magnetic field λ. We find that when measuring the local observable with directional CTM, the better ground states are obtained. However, it have no improvement when approaching to near-critical point. The possible reason is that the original states computed by iPEPS approximation is not accuracy sufficiently. Next, turn to the cases which states are obtained from 4-PESS algorithm.When $D = 2$ and $\chi = 20$, we find that it is hard to converge near the critical point because the virtual bonds dimension too small to describe the systems. After increasing the virtual dimension, we notice that it converge to $\lambda_c \approx 3.220$. In Sec. 4, we have shown that the ground states obtained by 4-PESS has more accuracy. Hence, it is not astonish that the result compute with 4-PESS+CTM is better than 2D-iTEBD+CTM. However, it still can not compare with the quantum Monte Carlo estimation which is $\lambda_c^{MC} \approx 3.044$

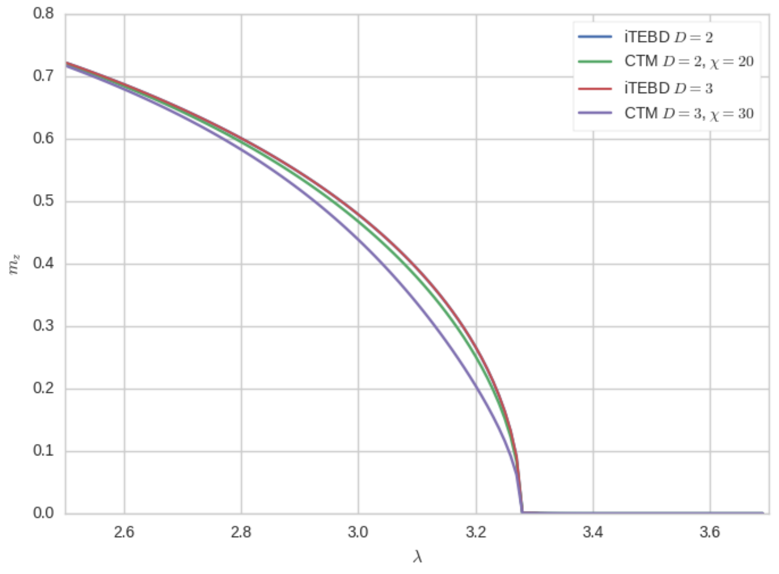


Figure 5.5: The red and blue tensor denotes on *odd* and *even* sites. The yellow one are time evolution operators $e^{-\tau H_{k,k+1}}$, $e^{-\tau H_{k+1,k}}$

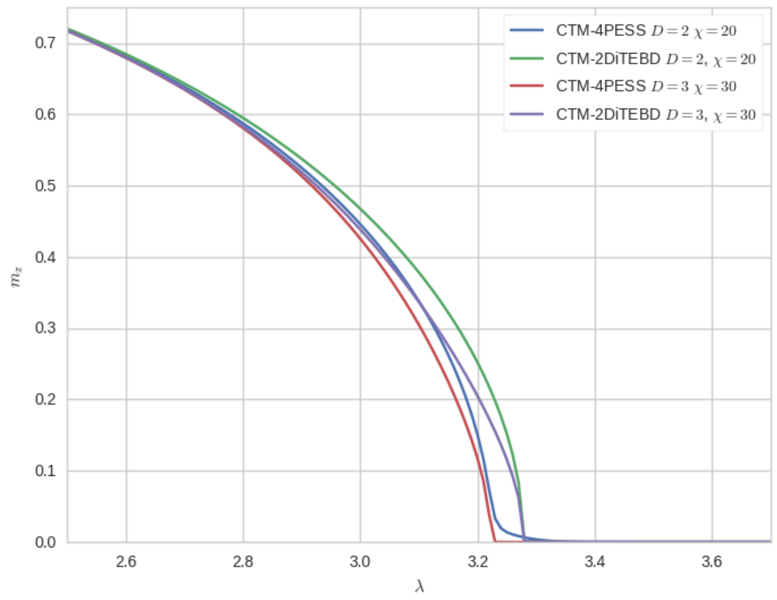


Figure 5.6: The red and blue tensor denotes on *odd* and *even* sites. The yellow one are time evolution operators $e^{-\tau H_{k,k+1}}$, $e^{-\tau H_{k+1,k}}$

Chapter 6

Summary

In this thesis, we reviewed the concept of matrix product states (MPS) and drew the structure with tensor diagrams, where the virtual bond dimension χ between each spins represent that how many the basis and the entanglement information are kept. Next, we introduced the imaginary time-evolving block decimation algorithm (iTEBD), which is the most simple tool to obtain the ground states of the MPS structure. In one dimensional system, the performance of iTEBD have been proved stable and efficient, because it obey the canonical form and have less influences of environment. Then, owing to the success of 1D-iTEBD, we tried to utilize it to simulate the two dimensional systems. However, we encountered in some problems. Firstly, due to the area law, we need consider the environment more restrictively when measuring the local observable. Secondly, the growth of computational consumption to describe project entangled pair states (PEPS) is too large because the dimension of each states is proportional to dD^4, where D is the dimension of virtual bonds in PEPS.

Therefore, optimizing two-dimensional algorithms become an significant work. In the Sec. 3, we started from basic simple update which is unstable due to multiplying to many pseudo-inverse entangled matrices. Next, to improve the stability of 2D-iTEBD, the new simulation proceeds was developed by Hastings. However, these two methods are not useful to study two-dimensional systems with large bond dimensions because the dimension of the projected tensor Θ is d^2D^6 and the cost CPU time will grow exponentially. Hence, we had better applied the decomposition tools, LQ and RQ, to reduce the dimension of the tensor Θ from d^2D^6 to d^4D^2 and it will improve the efficiency effectively. Finally, we have noticed that the ways to initialize the states and setting a suitable cutoff ε to determined how many basis should be truncated have a certain impact on the accuracy and stability of the algorithms.

Since the PEPS structure is hard to describe the interactions between next-neighbor states. In Sec. 4, we have introduced project entangled simplex state (PESS) ansatz to obtain the ground states in two-dimensional systems. Instead of containing the entangled information between each sites, we applied n-rank tensors to describe the entanglement in simplices. In conclusion, the computational consumption is less than PEPS ansatz because the dimension of the states on each sites is reduced to dD^2 and can obtain the ground states more accuracy in strong-correlated and frustrated systems, such as kagome and Husimi lattices. However, in square lattice systems, the PESS ansatz is not only hard to converge but also unstable and even broken in the end.

Finally, we reviewed the corner transfer matrix (CTM) to consider the influences of the environment. In conclusion, the accuracy will be improved when we measure the local observable with effective environment. However, so far we still can not simulate the environment with large virtual bond dimension D simply because the dimension of the reduce tensors is proportional to D^8, which means that the consumption and the cost time would increase exponentially. To deal the obstacle, our lab have developed the open source, Uni10, which not only make the implementation of tensor network algorithms conveniently but also can easily accelerate with GPU.

Bibliography

[1] S. R. White, **69**, 2863 (1992).

[2] S. R. White, **48**, 10345 (1993).

[3] F. Verstraete and J. I. Cirac, **73**, 10.1103/PhysRevB.73.094423.

[4] S. Östlund and S. Rommer, **75**, 3537.

[5] V. Murg, F. Verstraete, and J. I. Cirac, **75**, 10.1103/PhysRevA.75.033605.

[6] F. Verstraete, V. Murg, and J. Cirac, **57**, 143.

[7] G. Vidal, **91** (2003), 10.1103/PhysRevLett.91.147902.

[8] G. Vidal, **93** (2004), 10.1103/PhysRevLett.93.040502.

[9] G. Vidal, **99** (2007), 10.1103/PhysRevLett.99.220405.

[10] J. Jordan, "Studies of infinite two-dimensional quantum lattice systems with projected entangled pair states," .

[11] R. Orús, **349**, 117.

[12] R. B. Bauer, "Tensor network states," .

[13] W. Li, J. von Delft, and T. Xiang, **86**, 10.1103/PhysRevB.86.195137.

[14] G. Vidal, **98** (2007), 10.1103/PhysRevLett.98.070201.

[15] H. C. Jiang, Z. Y. Weng, and T. Xiang, **101**, 10.1103/PhysRevLett.101.090603.

[16] R. Orús and G. Vidal, **78**, 10.1103/PhysRevB.78.155117.

[17] M. B. Hastings, **50**, 095207.