Lecture 4

Tensor networks

Scribe: Weijie Xiong
Reviewer: Xingjian Li

In this lecture we learn the tensor networks (TN) representation – a useful tool for analyzing structured quantum states, Hamiltonians, and circuits. We start with the tensor diagram notation and learn how to use it to denote quantum states, unitaries and circuits. We next introduce TN-based methods for efficient classical simulation of quantum circuits and discuss their complexity. At the end, we review the 1D 2-LH problem and the approach using tensor product state as an Ansatz. Lower and upper bounds of accuracy are given, which will be further discussed in the next lecture.

4.1 Basics

4.1.1 Tensor diagram notation

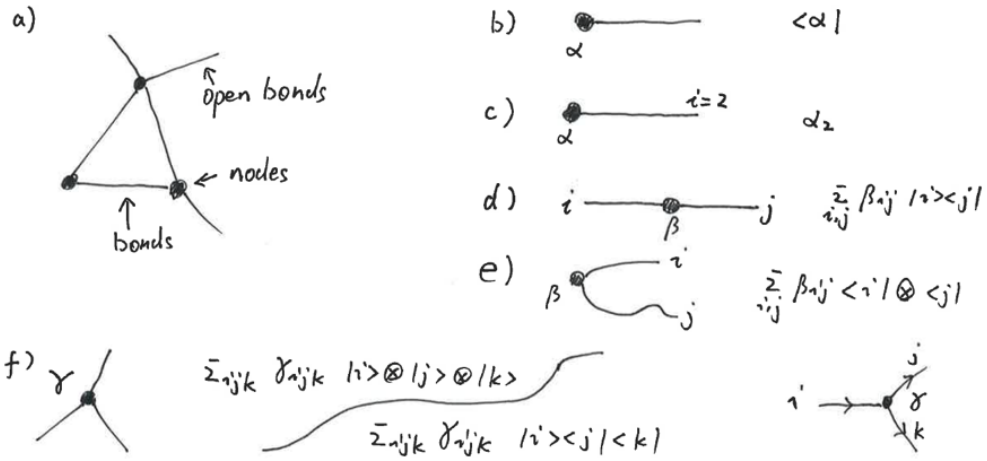

Figure 4.1: **Tensor diagram notations.** a) Terminology; b) Vector; c) Component of a vector; d) Matrix; e) Vectorization of a matrix; f) Three-legs tensor can be viewed as a tripartite state (vector), or as a matrix.

A tensor T is a linear map $T\colon \mathbb{C}^{d_1} \otimes \mathbb{C}^{d_2} \otimes \cdots \otimes \mathbb{C}^{d_t} \to \mathbb{C}^{d_{t+1}} \otimes \cdots \otimes \mathbb{C}^{d_w}$. In this lecture, we will use

tensor diagrams to represent different tensors. A tensor diagram consists of two types of components –nodes and bonds. As shown in Fig. 4.1, nodes are the endpoints of bonds, and bonds with only one endpoint are called "open bonds". Using this notation, a vector can be denoted by a node and an open bond, where a component of the vector can be specified. To denote a matrix, we draw a node connected with two open bonds. By changing the orientation of the bonds, we obtain the vectorization of the matrix. More generally, we can use a three-leg node, i.e. a node connecting three open bonds, to denote a tensor of order 3, as well as its linear transform into matrices.

4.1.2 Tensor diagram of linear operations

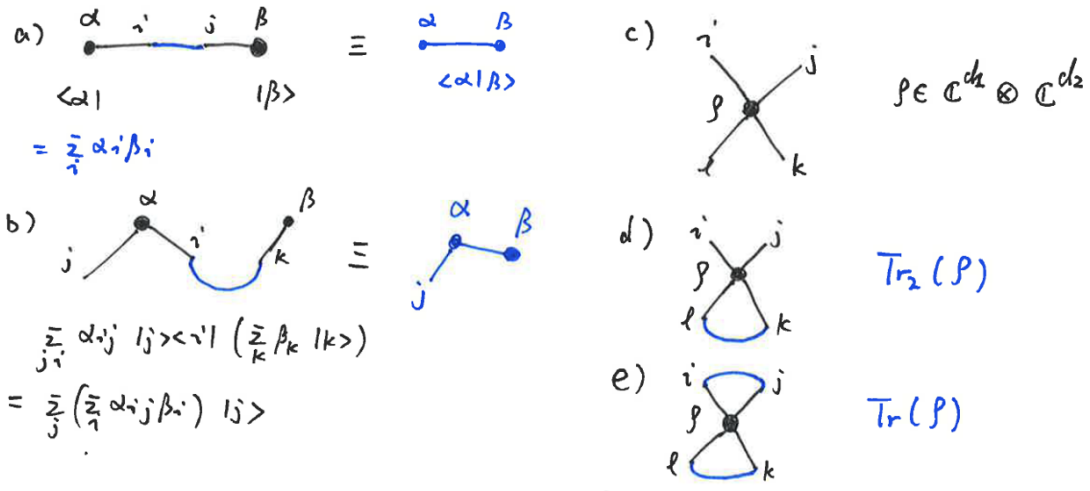

Figure 4.2: **Linear operations written in tensor diagram language.** a) Inner product of vectors; b) Matrix product; c) A separable matrix (density matrix of a bipartite state); d) Partial trace; e) Trace.

After given the tensor diagram notation of basic objects of linear algebra, let us use the tensor diagram to define the contraction operation. For any two open bounds, if we connect them with a line, it represents we are summing over the corresponding indices with the same coordinate. Let us see a few examples.

As illustrated in Fig. 4.2, we use two nodes connected and a bond connecting them to denote the inner product of two vectors. Similarly, a matrix product of a vector can be denoted by adding a node to one of the open bonds of a matrix (compare panel d) of Fig. 4.1). We next consider a matrix defined in a tensor product space, for instance, a density matrix for a quantum state of a bipartite system. This can be represented by a node connecting four open nodes. By connecting the two open bonds of a subspace, we obtain the respective partial trace of the matrix. Further connecting the rest of the bonds, we get the trace.

By connecting different bonds in the tensor diagram, we are able to define more complicated contraction operations. Though the order of contraction will not affect the result, we will soon see that different contraction orders can have significantly different complexity.

4.2 Schmidt decomposition

We now study a more complicated example –using TN to visualize the equivalence of Schmidt decomposition of vectors and singular value decomposition (SVD) of matrices. We review both methods:

Lemma 4.1 (Schmidt decomposition)**.** *Given a vector $|u\rangle$ in the tensor product of Hilbert spaces $H_1 \otimes H_2$ of dimension d_1 and d_2 respectively with $d_1 \leq d_2$, there exists orthonormal bases $\{|v_i\rangle\}_{i=1}^{d_1} \subset H_1$ and $\{|w_i\rangle\}_{i=1}^{d_1} \subset H_2$ such that*

$$|u\rangle = \sum_{i=1}^{d_1} \sqrt{\lambda_i} |v_i\rangle \otimes |w_i\rangle \,. \tag{4.1}$$

And,

Lemma 4.2 (Singular value decomposition)**.** *Given a matrix $U \in \mathbb{C}^{m\times n}$, then there exists unitaries $W^\dagger = \sum_i |i\rangle\langle w_i|$ and $V = \sum_j |v_j\rangle\langle j|$, and a diagonal matrix $\Sigma = \sum_k \sqrt{\lambda_k}|k\rangle\langle k|$, such that*

$$U = V\Sigma W^\dagger \tag{4.2}$$

$$= \sum_k \sqrt{\lambda_k} |v_k\rangle\langle w_k| \,. \tag{4.3}$$

We observe from Eq. (4.1) that the vector $|u\rangle$ can be viewed as the vectorization of matrix U in Eq. (4.3). In TN notation, as illustrated in panel a) and b) of Fig. 4.3, this can be understood as replacing a node (for a matrix) by three nodes connected in series (for matrix product of decomposition).

4.3 Matrix product state

We next extend the previous example of TN decomposition for a vector in generic tensor product space. For instance, let's consider $|u\rangle \in \bigotimes_{i=1}^n \mathbb{C}^2$. Its tensor diagram consists of a node connected with n open bonds, as shown in panel c) of Fig. 4.3. Via decomposition with respect to subspaces $\mathbb{C}^2$ and $\bigotimes_{i=2}^n \mathbb{C}^2$, we can replace the single node for $|u\rangle$ with three connected nodes representing the matrix product in SVD. By further applying the same decomposition on the node for matrix $W \in \mathbb{C}^{2\times 2^{n-1}}$, and the recursively induced nodes, we obtain a matrix product state (MPS) representation of $|u\rangle$, as shown in panel e) of Fig. 4.3.

Although the bond dimension of the tensor diagram of a general MPS increases exponentially, in practice one will neglect part of non-dominant singular values to reduce the complexity. That is, the bond dimensions control the accuracy of the low rank approximation. As discussed in the following sections, for calculations that involve operations on MPS, the n open bonds corresponding to the n qubits can be connected to a quantum circuit that implements the operation. With that quantum circuits with certain structures can be efficiently simulated using TN contraction methods.

Example: the CAT state. We here give an example of states with low bond dimension –the CAT state. The CAT state of n qubits is given by:

$$|\mathrm{CAT}\rangle := \frac{1}{\sqrt{2}} \left(|0\cdots 0\rangle + |1\cdots 1\rangle\right) \,, \tag{4.4}$$

which has the decomposition into matrix products, as shown in Fig. 4.4,

$$|\mathrm{CAT}\rangle = \overrightarrow{\prod_{\mathrm{i}=1}^{n}} \mathrm{A_i} \,, \tag{4.5}$$

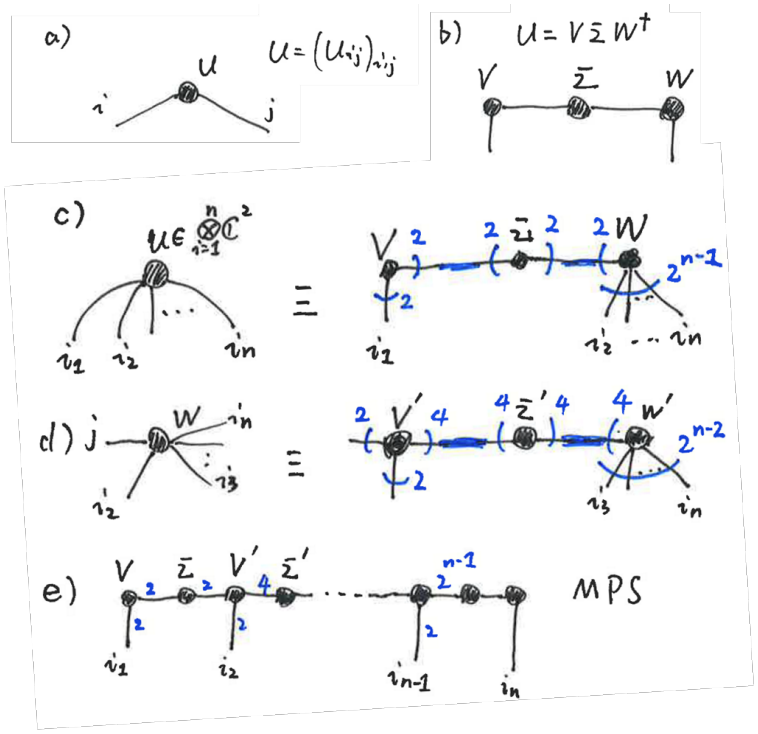

Figure 4.3: **Matrix product state.**

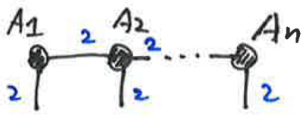

Figure 4.4: **CAT state.**

where $A_1 = \frac{1}{\sqrt{2}}\sum_{i=0}^{1}|i\rangle\langle i|$, $A_n = \sum_{i=0}^{1}|i\rangle|i\rangle$, and $A_j = \sum_{i=0}^{1}|i\rangle|i\rangle\langle i|$, $\forall j \in \{2,3,\cdots n-1\}$.

4.4 Energy of 1D 2-LH

Consider a 1D chain of qubits, given a geometrically 2-local Hamiltonian,

$$H = \sum_{i=1}^{n-1} H_i \tag{4.6}$$

where each H_i acts nontrivially on the neighboring qubits on the chain. Given an MPS $|\psi\rangle$ of bond dimension B, we are interested in its energy

$$\langle\psi|H|\psi\rangle = \sum_i \langle\psi|H_i|\psi\rangle\,. \tag{4.7}$$

Here each term can be evaluated using TN. The tensor diagram notation of $\langle\psi|H_i|\psi\rangle$ is drawn in panel a) of Fig. 4.5. Note that if we first contract the two columns vertically, the computational complexity will be

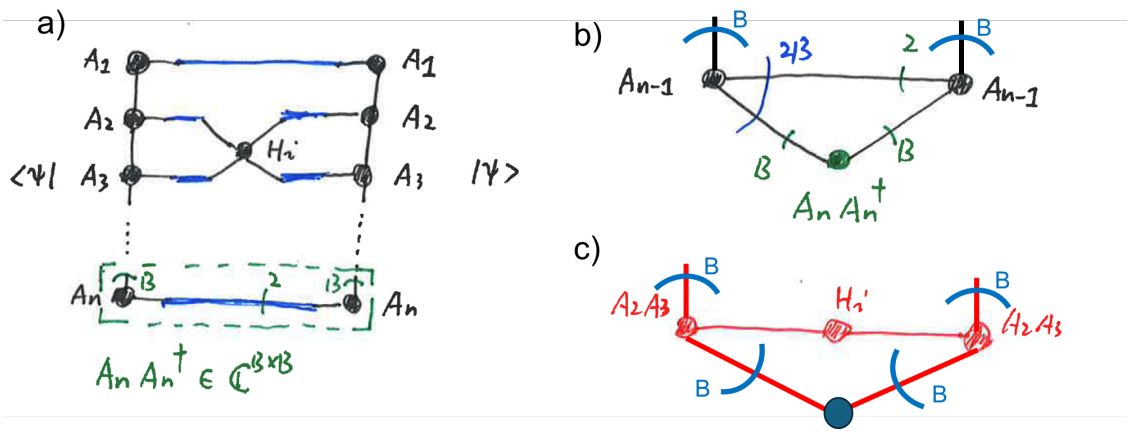

Figure 4.5: **Energy estimation.**

$\mathcal{O}(2^n)$. Instead, we contract horizontally, and start from the nodes and bond at the bottom. As illustrated in the green dashed box, the complexity of calculating $A_n A_n^\dagger$ is in $\mathcal{O}(B^2)$, and the TN components in the green dashed box reduces into the green node ("$A_n A_n^\dagger$") in panel b) connected with two bonds, both have bond dimension B. We further contract the three nodes and the connected bonds in panel b). The complexity of this step is in $\mathcal{O}(B^3) = \mathcal{O}(B \times 2B \times B)$. As the reduced new bottom nodes are connected with bonds that always have bond dimension B, the complexity of the remaining reduction steps is always in $\mathcal{O}(B^3)$. This also includes the step (see panel c)) involving H_i, whose bonds have dimension in $\mathcal{O}(1)$. Consequently, the total complexity of estimating the energy of an MPS with bond dimension B is in $\mathcal{O}(nB^3)$.

4.5 Output amplitude of shallow structured circuits

TN can also be used to simulate quantum circuits. For circuits with a certain structure of symmetry, e.g. alternating-layered 2-local shallow circuits (panel a) of Fig. 4.6), the simulation is efficient. For this example, we provide the following theorem:

Theorem 4.3. *Output amplitudes of log-depth, geometrically local circuits can be computed in poly-time.*

Proof. Formally, we denote the circuit as C, the initial state of qubits as $|0^n\rangle := \bigotimes_{i=1}^n |0\rangle$, and the task is to compute $\langle x|C|0^n\rangle$ where $x \in \{0,1\}^n$. We use TN to compute this quantity and draw TN of this structured circuit. As illustrated in panel b), nodes on both sides with a single leg denote the initial states or measurement projectors, nodes in between with four legs denote the 2-local circuit blocks.

We next reduce the TN and evaluate the complexity. We start from an arbitrary qubit and process the connected path of nodes. The product of two circuit blocks will be first calculated (see the blue bubble in panel b)). Then we proceed to the next node horizontally, i.e. increase the blue bubble to the green one, and then to the red one, until the one neighboring measurement projectors on the right hand side.

We identify all such horizontal paths of nodes and merge the connected bonds. The reduced vertical bond has dimension in $\mathcal{O}(2^d) = \mathcal{O}(\mathrm{poly(n)})$. The number of horizontal paths will be $\mathcal{O}(n)$, thus the total time complexity will be in $\mathcal{O}(\mathrm{poly(n)})$. □

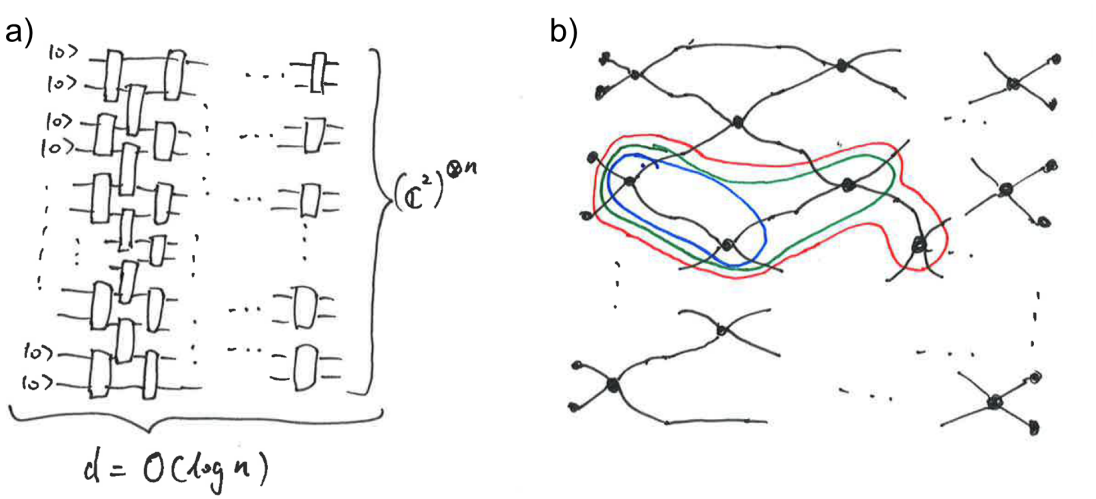

Figure 4.6: **Simulating shallow local circuits using TN.**

Remark: The required size of storage growth exponentially in the number of legs crossing bubble's boundary. By letting bubbles growth horizontally, the number of legs will be limited within $\mathcal{O}(d)$. As $d \in \mathcal{O}(\log(n))$, the storage size required will be also in $\mathcal{O}(\text{poly}(n))$.

4.6 Application of matrix product states

In practice, people often learn the ground state of some local Hamiltonian by assuming the ground state is a matrix product state with bounded bond strength B. Starting from a random MPS $|\psi\rangle$, note that for each Hamiltonian term $\langle\psi|H_i|\psi\rangle$ only depends on the local matrices of $|\psi\rangle$ A_i and A_{i+1}, thus we can optimize the matrices locally, and we repeat the process until we reach a minimal point for $\langle\psi|H|\psi\rangle$. This method is called the density matrix renormalization group (DMRG) method, and it is well applied in practice. However, the DMRG method does not have convergence or global minimality guarantees, and the starting state usually needs to be chosen carefully.

In the next lecture, we are going to show a polynomial time algorithm that can learn the ground state energy of an 1D gapped 2-local Hamiltonian in polynomial time, based on the 1D area law and MPS representations.