Introduction to Tensor Network Notation

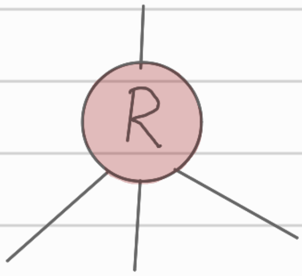
One of the primary reasons that tensors are useful is the straightforward and transparent notation usually used to describe them.

1. TENSORS

They are a generalization of vectors and matrices. A d-dimensional vector can be considered an element of $\mathbb{C}^d$, and a $n \times m$ dim matrix an element of $\mathbb{C}^{d_1 \times \cdots \times d_r}$

A rank-4 tensor R would be represented as

$R^{\rho}_{\sigma\mu\nu}$



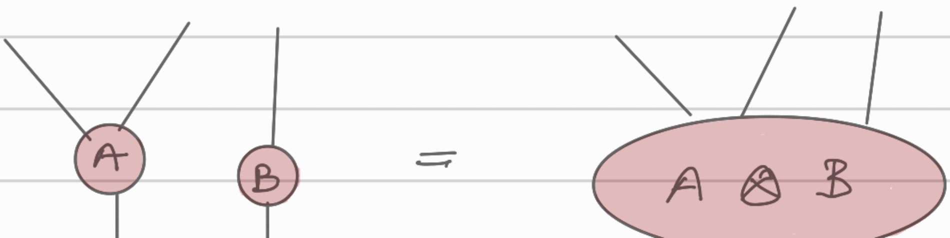
When representing quantum states, it is often convenient to use the directions of the legs to denote whether the corresponding vectors live in the Hilbert space ('kets) or its dual ('bras').

The main advantage in TNN comes in representing tensors that are themselves composed of several other tensors

1.1 TENSOR PRODUCT

The first operation we will consider is the tensor product, a generalisation of the outer product of vectors.

$$[A \otimes B]_{i_1,\dots,i_r,j_1,\dots,j_s} := A_{i_1,\dots,i_r} \cdot B_{j_1,\dots,j_s}$$



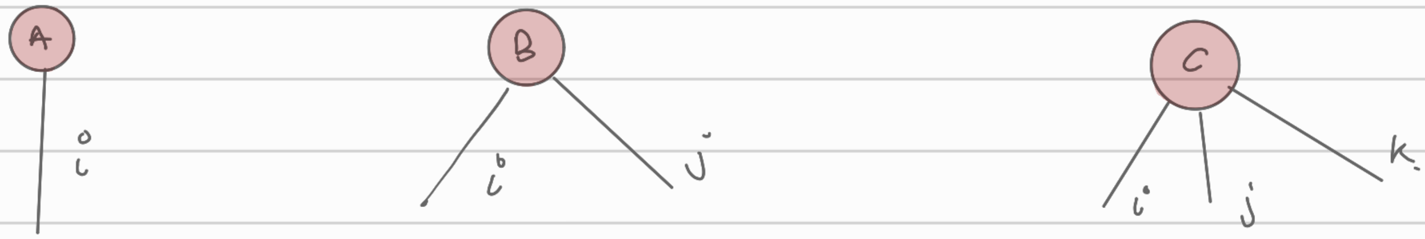
1.2 TRACE

The next operation is partial trace. Given a tensor A, for which the x^{th} and y^{th} indices have identical dimensions, the partial trace

$$[\mathrm{Tr}_{x,y} A]_{i_1 \cdots i_{x-1}, i_{x+1} \cdots, i_{y-1}, i_{y+1} \cdots i_r} = \sum_{\alpha=1}^{d_x} A_{i_1 \cdots i_{x-1} \alpha i_{x+1} \cdots, i_{y-1} \alpha, i_{y+1} \cdots i_r}$$

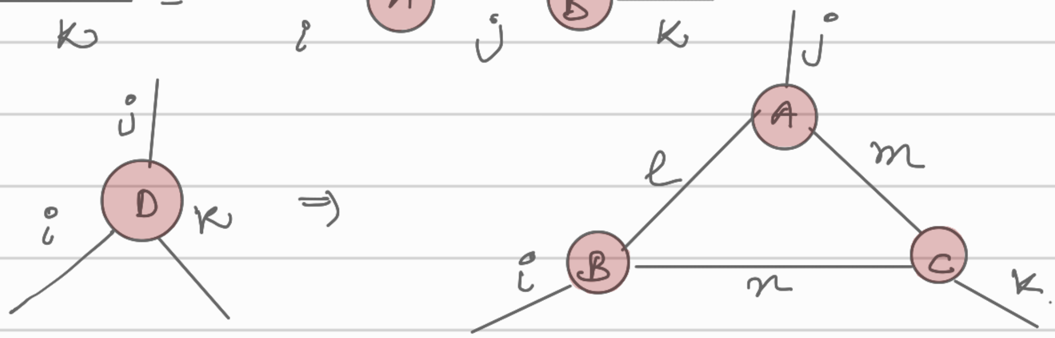
2. FUNDAMENTALS OF TENSOR NETWORKS

$$A = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_n \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & \cdots & B_{1m} \\ \vdots & \ddots & \vdots \\ B_{m1} & \cdots & B_{mn} \end{bmatrix} \qquad C = \left[\begin{bmatrix} C_{111} & \cdots & C_{n11} \\ \vdots & \ddots & \vdots \\ C_{m11} & \cdots & C_{mn1} \end{bmatrix} \cdots \right]$$



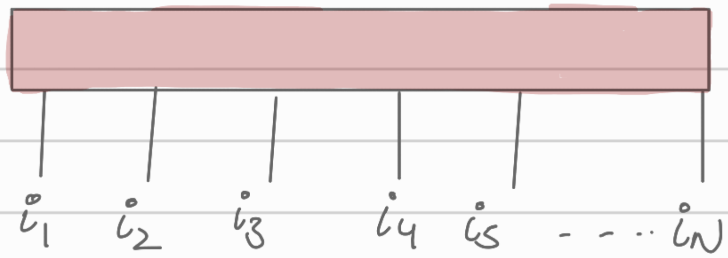
We can form networks comprised of multiple tensors, where an index shared by two tensors denote a contraction over the index.

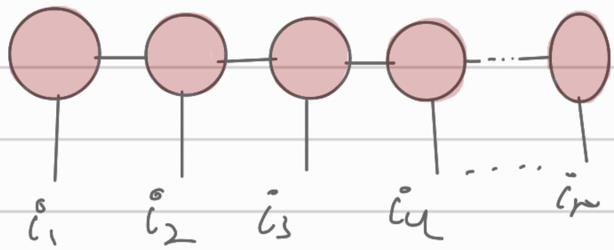
$$C_{ik} = \sum A_{ij} B_{jk}$$



$$D_{ijk} = \sum_{lmn} A_{ljm} B_{lin} C_{nkm}$$

In many applications the goal is to approximate a single high-order tensor as a tensor-network composed of many low-order tensors. Since the total dimension of a tensor grows exponentially with its order (n^d), the latter representation can be vastly more efficient.





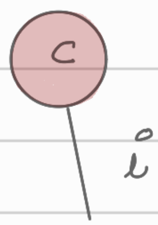
Much of tensor network theory focuses on understanding how these representation work, and in what circumstances are expected to work. In contrast tensor network Algorithm typically focus on methods to efficiently obtain, manipulate and extract information from these.

2.1 TENSOR NETWORKS FOR QUANTUM MANY BODY SYSTEMS

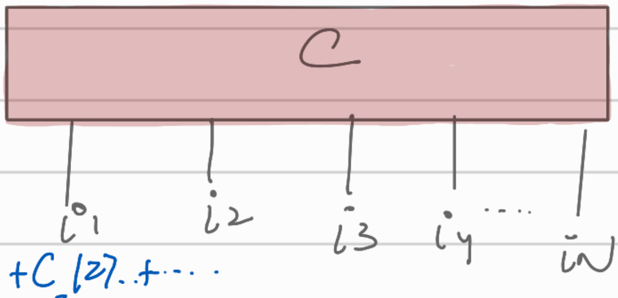
One of the most common use of tensor networks are representing quantum wavefunctions, which characterize the state of a quantum system.

Consider a d-level quantum system: a system where the state space is a d-dimensional $\mathbb{C}$ vector space spanned by d orthogonal basis vectors. A pure state $|\phi\rangle$ of the system is represented as a vector in this space which can be defined as a superposition of basis vectors,

$$|\phi\rangle = c_1|1\rangle + c_2|2\rangle + \dots + c_d|d\rangle = \sum_i c_i|i\rangle$$



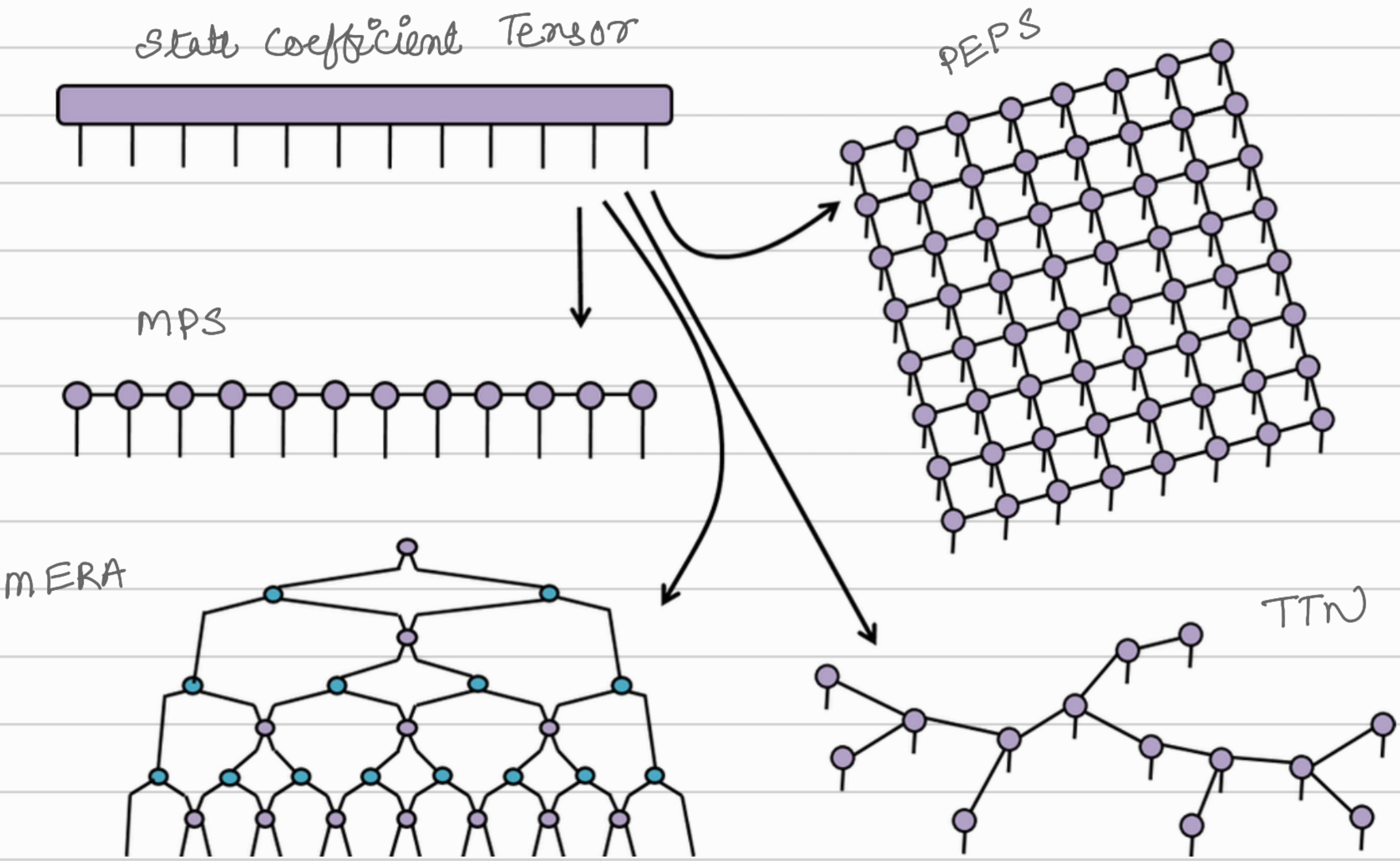
Now consider a quantum-many body system with N individual systems of dimension d



$$|\psi\rangle = \sum_{i_1, i_2 \ldots i_N} C_{i_1, i_2 \ldots i_N} |i_1\rangle |i_2\rangle \cdots |i_n\rangle$$

$$\Rightarrow C_1 |1\rangle + C_2 |2\rangle + \cdots C_d |d\rangle + C_a |a\rangle + C_b |b\rangle + \cdots + C_z |z\rangle \ldots + \cdots$$

In typical many-body problems, we begin with a Hamiltonian which describes how the system interact with each other and the goal is to find the lowest energy eigenstate. In this setting tensor networks are used as an ansatz.



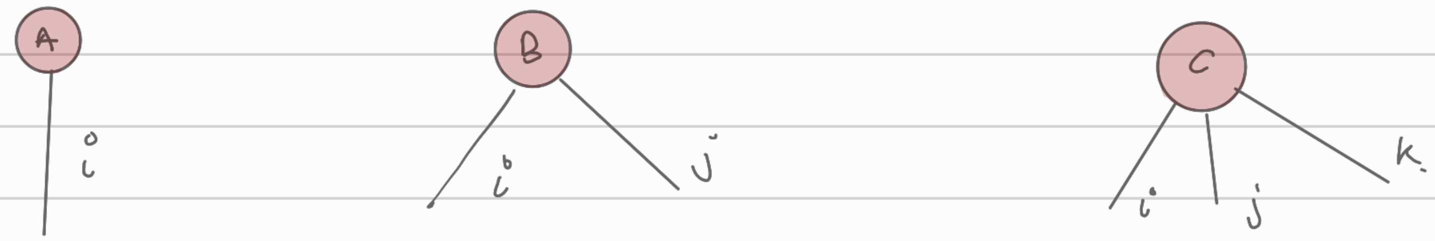
The choice of best tensor network ansatz for a particular problem may depend on the geometry of the problem as well its physical properties.

2.2 TENSOR CONTRACTIONS

It is convenient to represent tensor networks using diagrammatic notation, where individual tensors are represented as a solid shape with a number of legs that correspond to the rank.

$$A = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_m \end{bmatrix} \qquad B = \begin{bmatrix} B_{11} & \cdots & B_{1m} \\ \vdots & \ddots & \vdots \\ B_{m1} & \cdots & B_{mn} \end{bmatrix} \qquad C = \left[\begin{bmatrix} C_{111} & \cdots & C_{1n1} \\ \vdots & \ddots & \vdots \\ C_{m11} & \cdots & C_{mn1} \end{bmatrix}\right]$$

vector　　　matrix　　　order 3 tensor

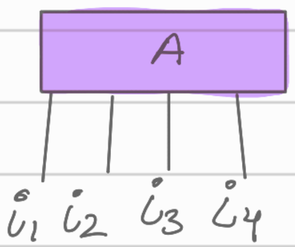


2.2.1 PERMUTE AND RESHAPE OPERATIONS

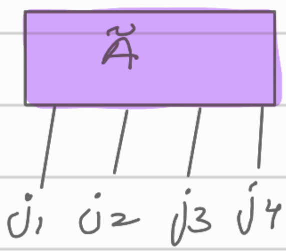
Fundamental to the manipulation of tensors are the permute and reshape functions. Permute allows the index ordering of a tensor to be changed. Reshape allows a collection of tensor indices to be combined in a singular layer.

a) Permute

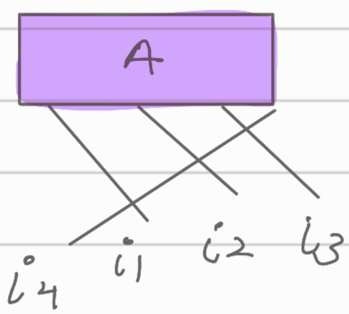
$A_{i_1 i_2 i_3 i_4} =$



$\tilde{A}_{j_1 j_2 j_3 j_4} =$



$\equiv$

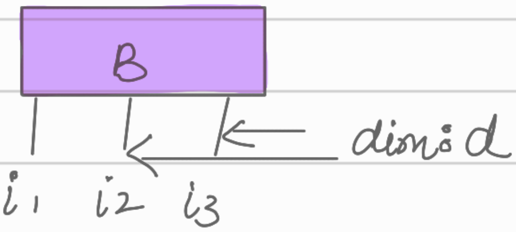


$$\Rightarrow \tilde{A}_{j_1 j_2 j_3 j_4} = A_{i_1 i_2 i_3 i_4}$$

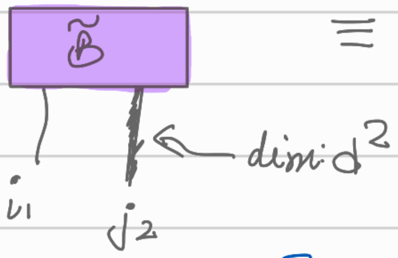
b) Reshape

b) Reshape

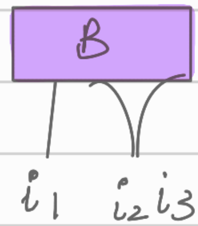
$B_{i_1 i_2 i_3}$ =



$\tilde{B}_{i_1 j_2}$ =



$\equiv$ $B_{i_1 (i_2 i_3)}$ =

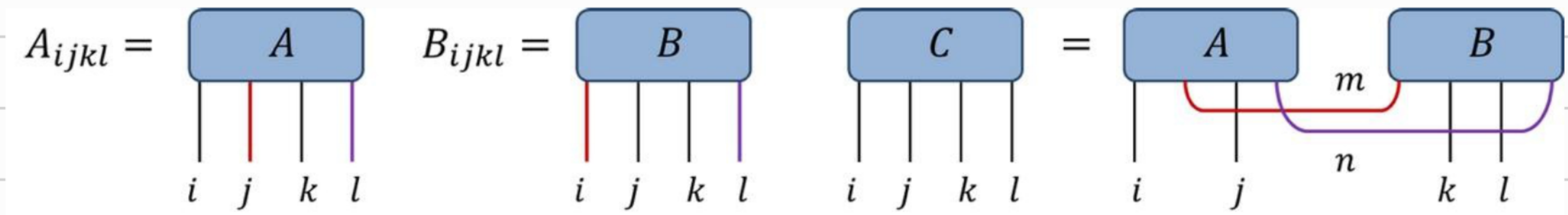


$$\tilde{B}_{i_1 j_2} = B_{i_1 i_2 i_3} \quad \text{if} \quad \left[\begin{array}{l} j_2 = i_2 + (i_3 - 1) d \\ j_2 = (i_2 \times d) + i_3 \end{array}\right.$$

2.2.3 Binary Tensor contractions

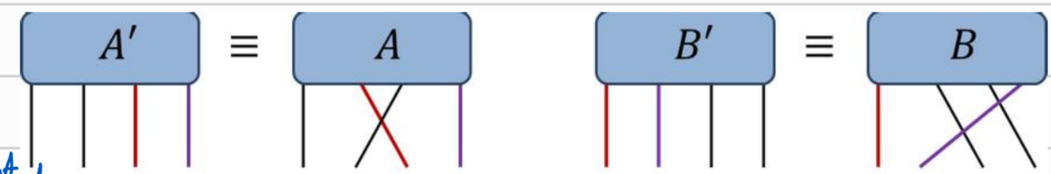
The usefulness of permute and reshape functions is that they allow a contraction between a pair of tensors to be recast as a matrix multiplication. Although the computational cost is the same both ways, it is usually preferable to recast as multiplication as modern hardware performs vectorized operations much faster than when using the equivalent FOR loop.

Evaluate tensor C:

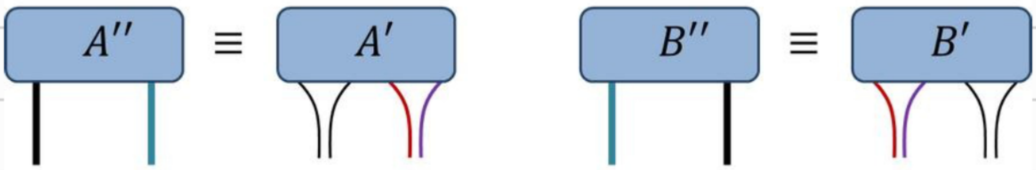


$$C_{ijkl} = \sum_{mn} A_{imjn} B_{mkln}$$

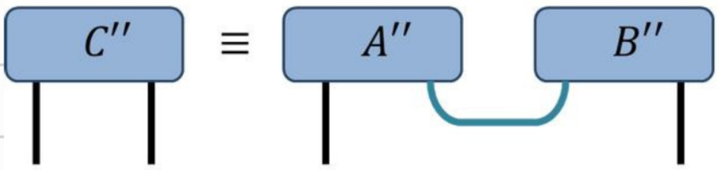
i) Permute the A and B tensor such that the indices to be contracted



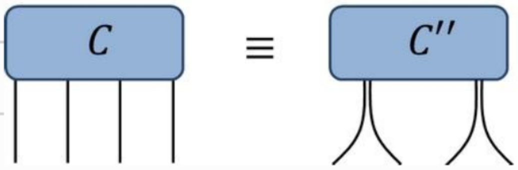
become the trailing and leading indices respectively



ii) Reshape tensors into matrices and do the matrix multiplication



iii) Reshape C back to tensor, perform final permutation.

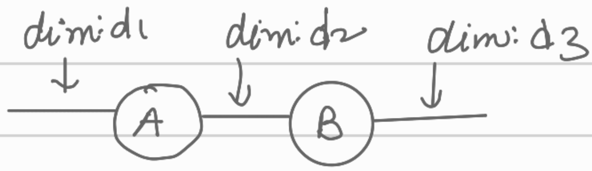


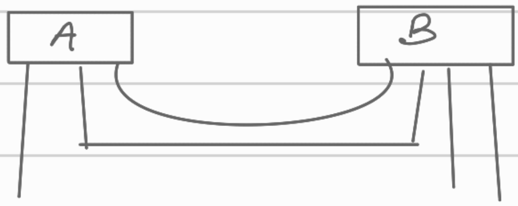
2.2.3 Contraction cost

The computational cost of multiplying a $d_1 \times d_2$ matrix A with $d_2 \times d_3$ matrix B is:

$$\text{cost}(A \times B) = d_1 \cdot d_2 \cdot d_3$$ (number of scalar multiplications)

Given the equivalence with matrix multiplication, this is also the cost of a binary tensor contraction





$$\text{cost}(A \times B) = d_1 \cdot d_2 \cdot d_3$$

$$\text{cost}(A \times B) = \frac{|\dim(A)|\,|\dim(B)|}{|\dim(A \cap B)|}$$

$$\Rightarrow \frac{d^3 \cdot d^4}{d^2} \Rightarrow d^5$$