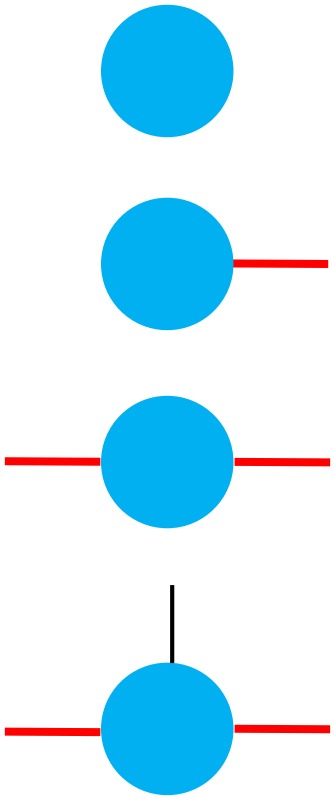
Tensor network

Tensor network is a language in computational science which treat tensors.
Tensor network is introduced to quantum information and then condensed matter physics.
The main reason is that many-body wavefunction is a tensor!
Here we will learn tensor network as a independent language.

Picture representation of tensor



Scalar, no index

Vector, 1 index (one bond)

matrix, 2 index

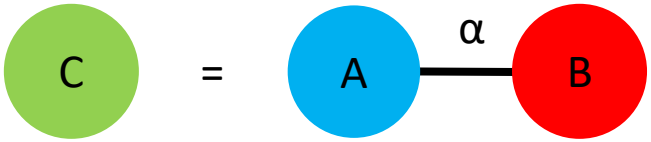
Rank-3 tensor, 3 index

A practical introduction to tensor networks: Matrix product states and projected entangled pair states

Román Orús *

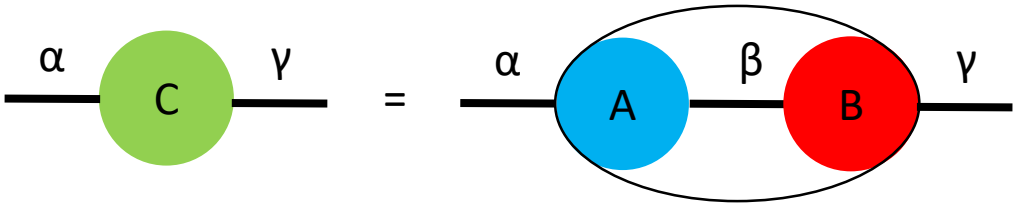
Institute of Physics, Johannes Gutenberg University, 55099 Mainz, Germany

Contraction



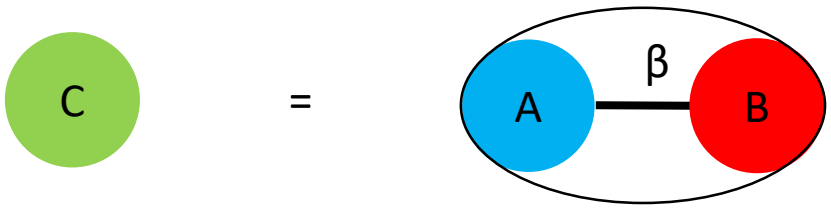
Since all the bonds are contracted, the resulting is just a number.

$$C = \sum_{\alpha} A_{\alpha} B_{\alpha}$$

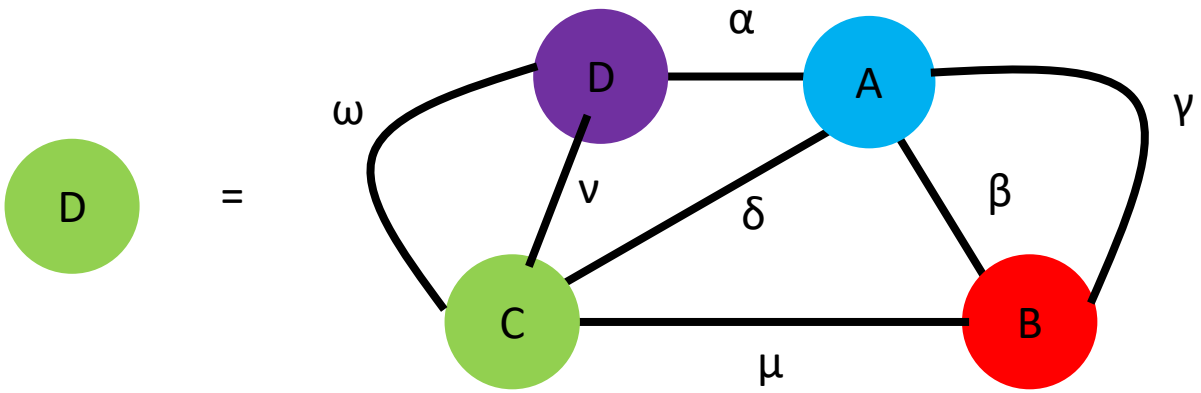


Since there are two open Indices, the resulting is just a matrix.

$$C_{\alpha\gamma} = \sum_{\beta} A_{\alpha\beta} B_{\beta\gamma} = (\mathrm{AB})_{\alpha\gamma}$$



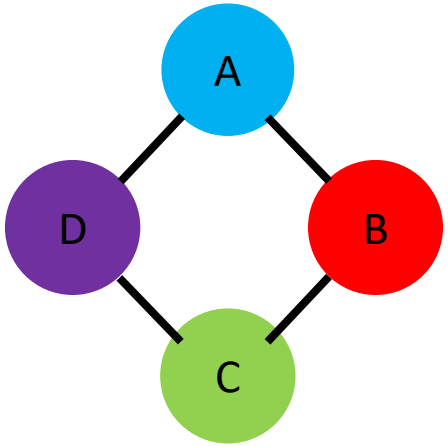
A more complex one



Since all the bonds are contracted, the resulting is just a number.

$$C = \sum_{\alpha} A_{\alpha\beta\gamma\delta} B_{\beta\gamma\mu} C_{\delta\mu\nu\omega} D_{\alpha\nu\omega}$$

Trace



$$tr(\mathrm{ABCD}) = \sum_{\alpha} (\mathrm{ABCD})_{\alpha\alpha}$$

Tensor manipulation

Index fusion and splitting

Tensor are nothing more than a structured way to organize information, that is, an ordered Collection of numbers or variables.

For example, to describe three dimensional relativistic systems, the four space-time variables x,y, z and t, typically are organized in a single vector to improve our representation and manipulation capabilities. However, if it turns out to be more convenient, one could arrange them in a 2*2 matrix.

数学基础是“可数集”

$$X = \begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix} \quad \Rightarrow \quad O = \begin{pmatrix} x & y \\ z & t \end{pmatrix}$$

Index

$$X_i \qquad O_{jk}$$

In general, a rank-n tensor can be rearranged in a tensor of a different rank following some given (invertible) rule. Notice that in transforming a matrix to a vector, we fuse two indexes in a single one.

From now on, we will indicate the fusion operation as

$$\alpha_1, \ldots, \alpha_n \succ i$$

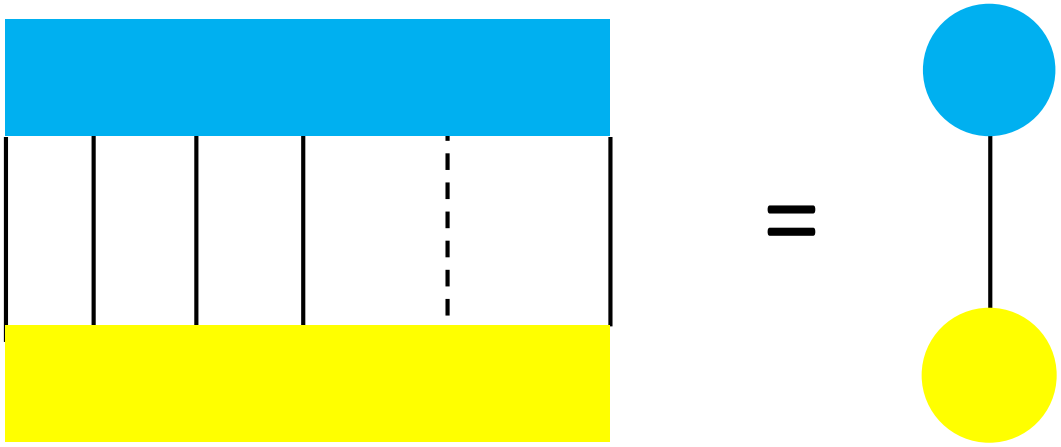
The inverse operation is the splitting of indexes:

$$i \prec \alpha_1, \ldots, \alpha_n$$

Therefore, the product of two rank-N tensor can be written as:

$$\phi^*_{\alpha_1,\ldots,\alpha_N} \psi_{\alpha_1,\ldots,\alpha_N} = \phi^*_i \psi_i$$

Graphic representation



Tensor network differentiation and eigenvalue problem

To know what does this mean, here we take a simple example:

$$H = S_{1z} S_{2z}$$

$$|\psi\rangle = \sum_{\alpha\beta} \psi_{\alpha\beta} |\alpha\beta\rangle = \psi_{\uparrow\uparrow} |\uparrow\uparrow\rangle + \psi_{\uparrow\downarrow} |\uparrow\downarrow\rangle + \psi_{\downarrow\uparrow} |\downarrow\uparrow\rangle + \psi_{\downarrow\downarrow} |\downarrow\downarrow\rangle$$

The energy is given by

$$E(\psi_{\alpha\beta}) = \langle\psi| H |\psi\rangle = \psi_{\uparrow\uparrow}^2 - \psi_{\uparrow\downarrow}^2 - \psi_{\downarrow\uparrow}^2 + \psi_{\downarrow\downarrow}^2$$

With the constraint

$$\langle\psi|\psi\rangle = \psi_{\uparrow\uparrow}^2 + \psi_{\uparrow\downarrow}^2 + \psi_{\downarrow\uparrow}^2 + \psi_{\downarrow\downarrow}^2 = 1$$

Introduce Lagrangian multiplier

$$E(\psi_{\alpha\beta}, \lambda) = \psi_{\uparrow\uparrow}^2 - \psi_{\uparrow\downarrow}^2 - \psi_{\downarrow\uparrow}^2 + \psi_{\downarrow\downarrow}^2 + \lambda(\psi_{\uparrow\uparrow}^2 + \psi_{\uparrow\downarrow}^2 + \psi_{\downarrow\uparrow}^2 + \psi_{\downarrow\downarrow}^2 - 1)$$

To find the extrema, the following condition should be satisfied:

$$\frac{\partial E(\psi_{\alpha\beta},\lambda)}{\partial \psi_{\uparrow\uparrow}} = 2(1+\lambda)\psi_{\uparrow\uparrow} = 0$$

$$\frac{\partial E(\psi_{\alpha\beta},\lambda)}{\partial \psi_{\uparrow\downarrow}} = 2(-1+\lambda)\psi_{\uparrow\downarrow} = 0$$

$$\frac{\partial E(\psi_{\alpha\beta},\lambda)}{\partial \psi_{\downarrow\uparrow}} = 2(-1+\lambda)\psi_{\downarrow\uparrow} = 0$$

$$\frac{\partial E(\psi_{\alpha\beta},\lambda)}{\partial \psi_{\downarrow\downarrow}} = 2(1+\lambda)\psi_{\downarrow\downarrow} = 0$$

$$\frac{\partial E(\psi_{\alpha\beta},\lambda)}{\partial \lambda} = \psi_{\uparrow\uparrow}^2 + \psi_{\uparrow\downarrow}^2 + \psi_{\downarrow\uparrow}^2 + \psi_{\downarrow\downarrow}^2 - 1 = 0$$

There are several sets of solutions:

$$\lambda = -1 \qquad \psi_{\uparrow\uparrow} = 1 \qquad \psi_{\uparrow\downarrow} = \psi_{\downarrow\uparrow} = \psi_{\downarrow\downarrow} = 0$$

$$\lambda = -1 \qquad \psi_{\downarrow\downarrow} = 1 \qquad \psi_{\uparrow\uparrow} = \psi_{\uparrow\downarrow} = \psi_{\downarrow\uparrow} = 0$$

$$\lambda = 1 \qquad \psi_{\uparrow\downarrow} = \psi_{\downarrow\uparrow} = \frac{1}{\sqrt{2}} \qquad \psi_{\uparrow\uparrow} = \psi_{\downarrow\downarrow} = 0$$

Substitute the above result into the above equation, we know only the third one gives the desired result.

As we show above, the calculation of ground state is to differentiate the coefficients.

We can generalize the above process. The energy are function of high tensor

$$E(\psi^*_{\beta_1,\ldots,\beta_N},\psi_{\alpha_1,\ldots,\alpha_N}) = \mathrm{E}(\psi^*_i,\psi_j)$$

Thus to find the extrema of the function E with respect to the entries of the tensor, the differential of the function shall set to zero:

$$\frac{\partial \mathrm{E}(\psi^*_i,\psi_j)}{\partial \psi_j} = 0$$

Since we have

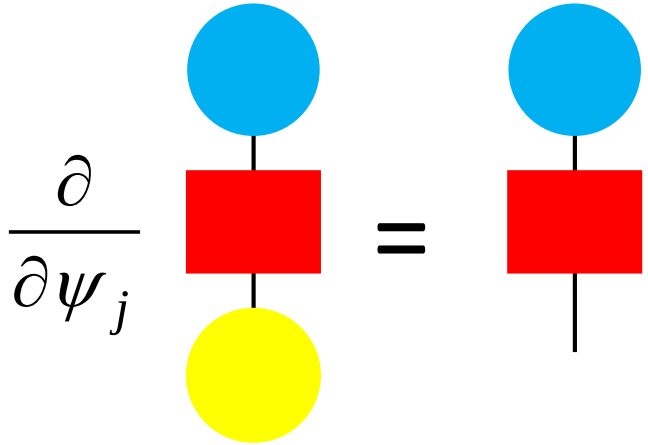
$$\mathrm{E}(\psi^*_i,\psi_j) = E(\psi^*_{\beta_1,\ldots,\beta_N},\psi_{\alpha_1,\ldots,\alpha_N}) = \langle\psi|H|\psi\rangle = \psi^{*i}H_i^j\psi_j$$

Therefore, the extrema condition is equivalent to the following eigenvalue problem;

$$\frac{\partial \mathrm{E}(\psi^*_i,\psi_j)}{\partial \psi_j} = 0 \Rightarrow \psi^{*i}H_i^j = 0$$

The derivative of any linear function of tensors (any function defined via a tensor network where each tensor appears only once) with respect to a particular tensor A, is given by the tensor network where the tensor A has been removed.

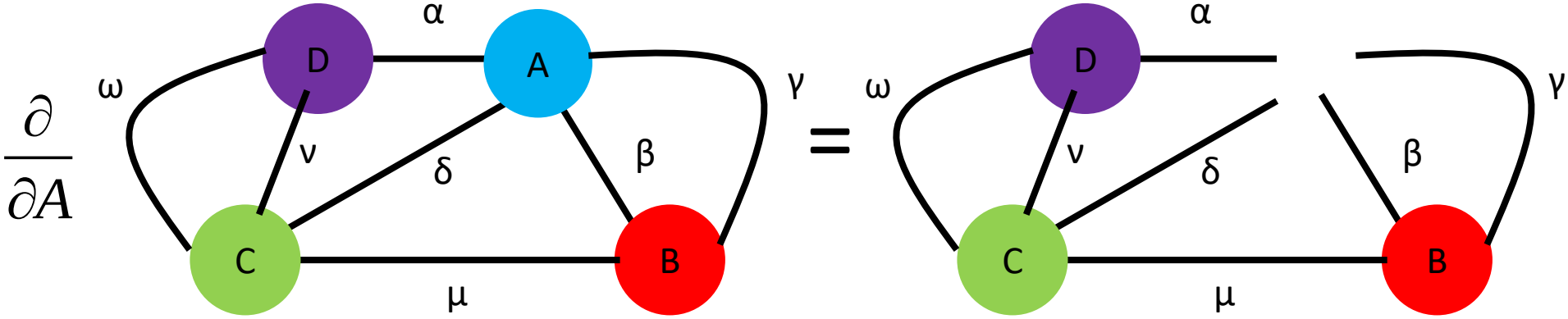
Graphic representation



Which correspond to

$$\frac{\partial \mathrm{E}(\psi_i^*, \psi_j)}{\partial \psi_j} = \psi^{*i} H_i^j$$

A more complex example:



Gauging and basis modification

An operation that plays a prominent role in tensor network algorithms is the gauging, that is, to change the tensors entries by means of local transformations, without changing the global state wo exploit some desired properties.

Indeed, whenever two tensors are contracted, it is possible to insert an identity operator between them without changing the overall tensor network properties. Using the equivalence:

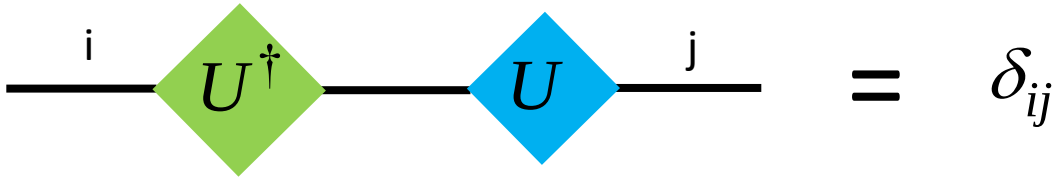
$$1 = U^{\dagger} U$$

that holds by definition for any unitary operator U, the tensor entries can be changed as

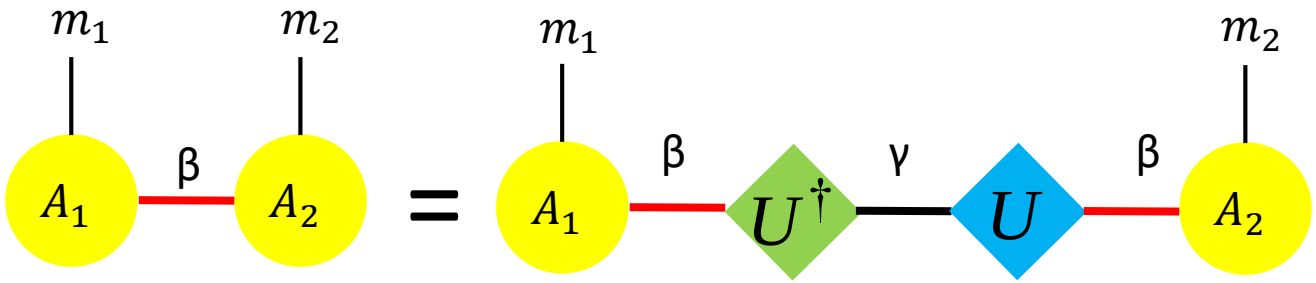
$$A_i B_i = A_i 1_{i,j} B_j = A_i (U^{\dagger} U)_{i,j} B_j = A_i U^{\dagger}_{ik} U_{kj} B_j = A_k B_k$$

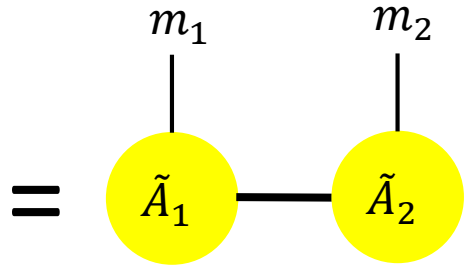
with the net result that the tensor structure remains unchanged, but the tensor entries have changed and thus can satisfy some desired properties.

Clearly, this freedom can be exploited at the level of the fusion indices I, j as well at each single indexes αj independently.



$$\delta_{ij} = (U^\dagger U)_{ij} = U^\dagger_{ik} U_{kj}$$





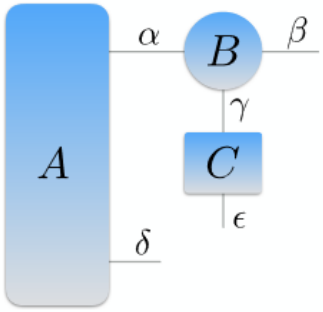
If we think a litter further, the gauging each time is equivalent to basis modification.

Numerical scaling

The question is as following: now we have a tensor network, we want to calculate the result (either a number of a low tensor), what's the most efficient way?

Here we briefly discuss how to read off the cost scaling from a tensor network diagram which is actually a quite simple procedure.
For a given contraction of two arbitrary tensors, we count the dimensions of all open indices of the two tensors as well as those of the indices that are contracted over.
The product of these numbers leads to the number of operators necessary to perform this specific contraction.

$$\sum_{\alpha,\gamma} A_{\alpha\delta} B_{\alpha\beta\gamma} C_{\gamma\epsilon} =$$



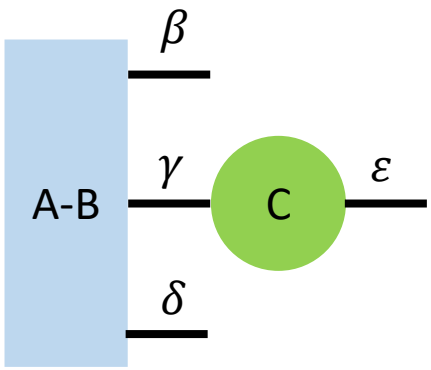
Contraction of A, B

$$\{\delta, \alpha, \beta, \gamma\} : \mathrm{O}(\mathrm{D}^3\mathrm{d})$$

$$\{\alpha, \beta, \delta\} : 1, \ldots, \mathrm{D}$$

$$\{\gamma, \varepsilon\} : 1, \ldots, \mathrm{d}$$

Contraction of A-B and C



$$\{\delta, \varepsilon, \beta, \gamma\} : \mathrm{O}(\mathrm{D}^2\mathrm{d}^2)$$

Contraction order and efficiency

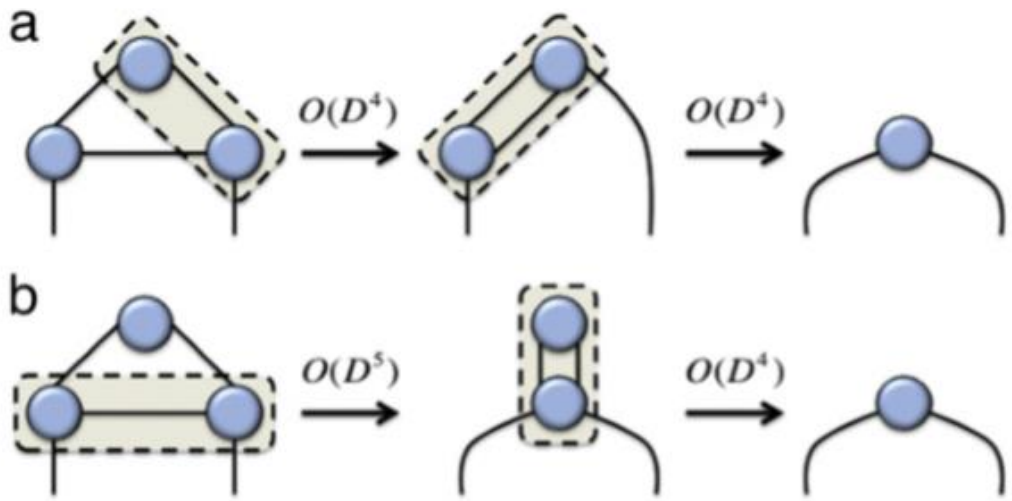


Fig. 8. (Color online) (a) Contraction of 3 tensors in $O(D^4)$ time; (b) contraction of the same 3 tensors in $O(D^5)$ time.

Since in tensor network, one has to deal with many contractions, and the aim is to make these as efficiently as possible. For this, finding the optimal order of indices to be contracted in a tensor network turn out to be a crucial step, specially when it comes to programming computer codes to implement the methods.

Eigenvalue decomposition

From linear algebra, we know an eigenvalue and eigenvector of a square matrix A are a scalar λ and a nonzero column vector x so that

$$Ax = x\lambda$$

Into matrix form, we have

$$AU = U\Lambda$$

where

$$U = \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix}$$

$$\Lambda = diag\{\lambda_1, \dots, \lambda_n\}$$

Single value decomposition

For a generic, m*n non-quadratic matrix A, a SVD decomposition yields

$$A = U \Sigma V^{\dagger}$$

$$m \times n \quad m \times m \quad m \times n \quad n \times n$$

where:

$$U^{\dagger} U = 1$$

$$\Sigma = diag\{\sigma_1, .., \sigma_k, 0, ..., 0\}$$

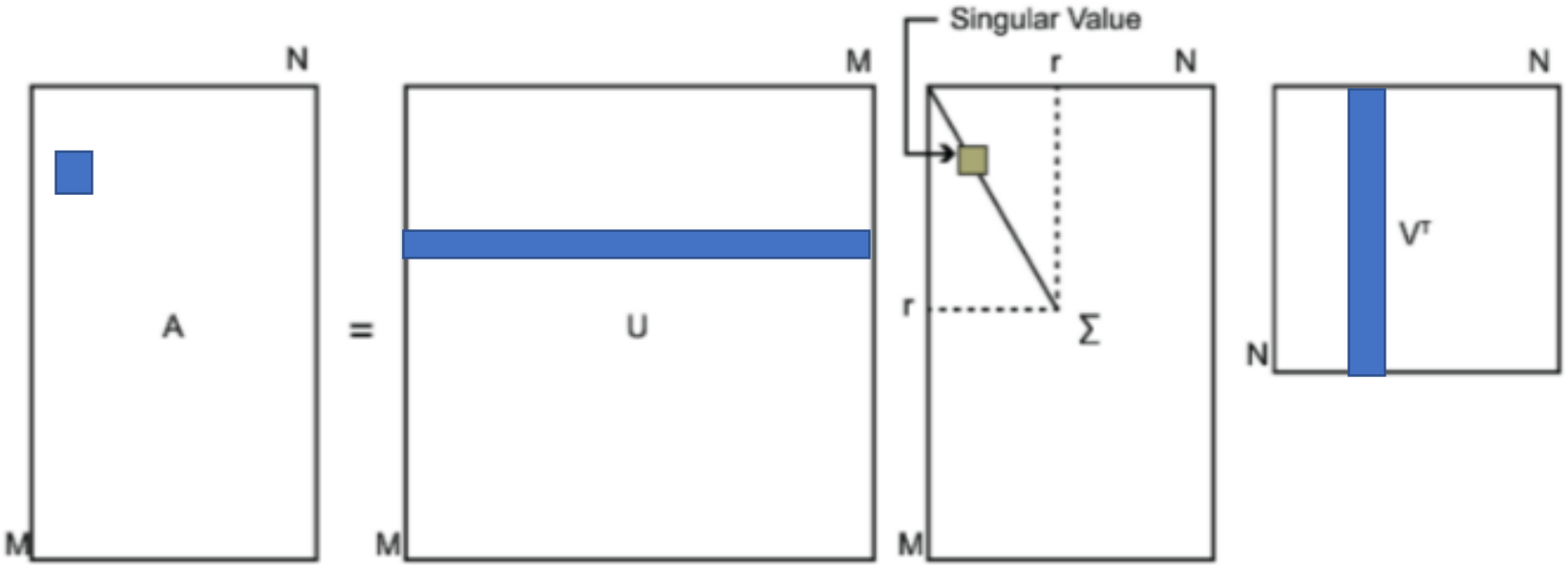
$$V^{\dagger} V = 1$$

Expand in matrix element form:

$$A_{m_1 m_2} = (U\Sigma V^\dagger)_{m_1 m_2} = \sum_i U_{m_1 i}\sigma_i (V^\dagger)_{i m_2}$$

row column

We have the following picture



SVD and eigenvalue decomposition

$$AA^{\dagger} = U\Sigma V^{\dagger}V\Sigma^{\dagger}U^{\dagger} = U\Sigma^2 U^{?}$$

$$\Rightarrow (AA^{\dagger})U = U\Sigma^2$$

$$A^{\dagger}A = V\Sigma^{\dagger}U^{\dagger}U\Sigma V^{\dagger} = V\Sigma^2 V^{?}$$

$$\Rightarrow (A^{\dagger}A)V = V\Sigma^2$$

Compared to eigenvalue decomposition, this means that the singular values squared are the eigenvalues of both $AA^{\dagger}$ and $A^{\dagger}$A, and the respective eigenvectors are the columns of U and V respectively.

This tells us how to find U, V and Σ.

Example:

$$A=\begin{pmatrix}4 & 0 & 5\\ 0 & 0 & 5\end{pmatrix}$$

$$AA^{\dagger}=\begin{pmatrix}41 & 25\\ 25 & 25\end{pmatrix}$$

Eigenvalue decomposition

$$\begin{pmatrix}41 & 25\\ 25 & 25\end{pmatrix}\begin{pmatrix}0.5896 & -0.8077\\ -0.8077 & -0.5896\end{pmatrix}=\begin{pmatrix}0.5896 & -0.8077\\ -0.8077 & -0.5896\end{pmatrix}\begin{pmatrix}6.7512 & \\ & 59.2488\end{pmatrix}$$

$$U=\begin{pmatrix}0.5896 & -0.8077\\ -0.8077 & -0.5896\end{pmatrix}$$

$$A^{\dagger}A = \begin{pmatrix} 16 & 0 & 20 \\ 0 & 0 & 0 \\ 20 & 0 & 50 \end{pmatrix}$$

Eigenvalue decomposition

$$\begin{pmatrix} 16 & 0 & 20 \\ 0 & 0 & 0 \\ 20 & 0 & 50 \end{pmatrix} \begin{pmatrix} 0 & 0.9076 & 0.4197 \\ -1 & 0 & 0 \\ 0 & -0.4197 & 0.9076 \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 0.9076 & 0.4197 \\ -1 & 0 & 0 \\ 0 & -0.4197 & 0.9076 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 6.7512 & 0 \\ 0 & 0 & 59.2483 \end{pmatrix}$$

$$V^{\dagger} = \begin{pmatrix} 0 & 0.9076 & 0.4197 \\ -1 & 0 & 0 \\ 0 & -0.4197 & 0.9076 \end{pmatrix}$$

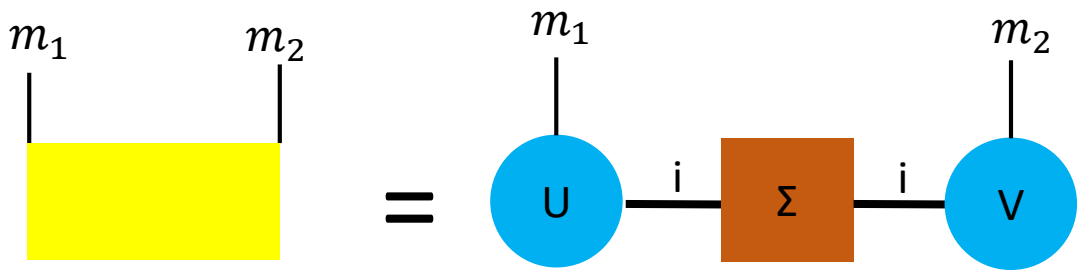
Therefore, we have

$$U = \begin{pmatrix} 0.5896 & -0.8077 \\ -0.8077 & -0.5896 \end{pmatrix}$$

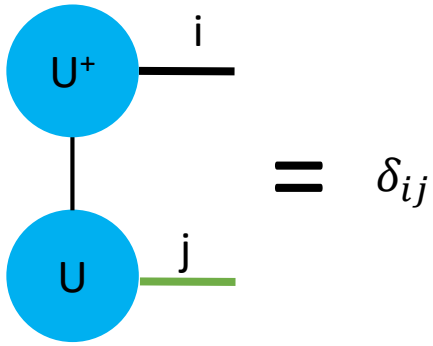
$$\Sigma = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 2.5983 & 0 \\ 0 & 0 & 7.6973 \end{pmatrix}$$

$$V^{\dagger} = \begin{pmatrix} 0 & 0.9076 & 0.4197 \\ -1 & 0 & 0 \\ 0 & -0.4197 & 0.9076 \end{pmatrix}$$

$$\psi^{m_1 m_2} = \sum_i U_i^{m_1} \sigma_i V_i^{m_2}$$

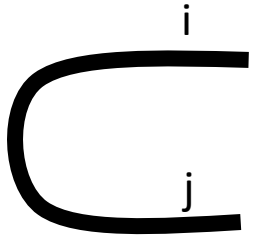


And the orthonormal condition written as:



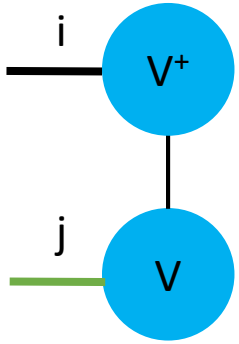
$= \delta_{ij}$

Or we can simplified as



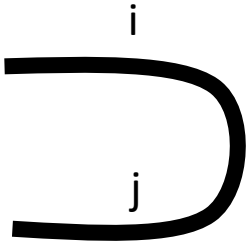
$= \delta_{ij}$

$\delta_{ij} =$



Or we can simplified as

$\delta_{ij} =$

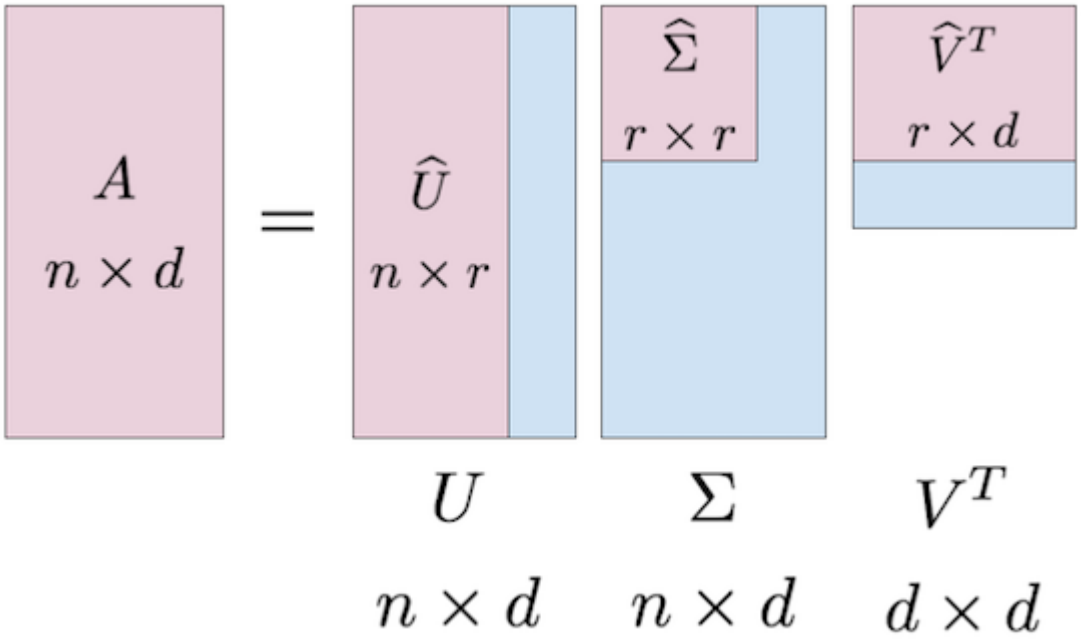


SVD and Compression

对于奇异值，在奇异值矩阵中也是按照从大到小排列，而且奇异值减小得特别快，在很多情况下，前10%甚至1%得奇异值的和就占了全部的奇异值之和的99%以上的比例。也就是说，我们也可以用最大的k个奇异值和对应的左右奇异向量来近似描述矩阵。即：

$$A_{m\times n} = U_{m\times m}\Sigma_{m\times n}(V^{\dagger})_{n\times n} \approx U_{m\times k}\Sigma_{k\times k}(V^{\dagger})_{k\times n}$$

其中k要比n小很多，也就是一个大的矩阵A可以用三个小的矩阵U, Σ, V来表示，如下图所示，现在我们的矩阵A只需要灰色的部分的三个小矩阵就可以近似描述了：



https://www.cnblogs.com/pinard/p/6251584.html

SVD and rank-lowering

Matrix product ansatz

Any tensor can be expressed as product of matrixes.

$$\psi^{m_1 m_2 \ldots m_n} \rightarrow C_{m_1, m_2 \ldots m_n} = \sum_{a_1} U_{m_1 a_1} \Sigma_{a_1 a_1} (\mathrm{V}^{\dagger})_{a_1, m_2 \ldots \mathrm{m}_n}$$

split into two

Here {m} is the physical d.o.f. and {a} is the auxiliary d.o.f..

Map:

$$U_{m_1 a_1} = A^{(1), m_1}_{1 \alpha_1}$$

$$C_{a_1 m_2 \ldots m_n} = \Sigma_{a_1 a_1} (\mathrm{V}^{\dagger})_{a_1, m_2 \ldots \mathrm{m}_n}$$

$$\psi^{m_1 m_2 \ldots m_n} \rightarrow C_{m_1 m_2 \ldots m_n} = \sum_{a_1} A^{(1), m_1}_{1 a_1} C_{a_1 m_2 \ldots m_n}$$

Do SVD for m_2

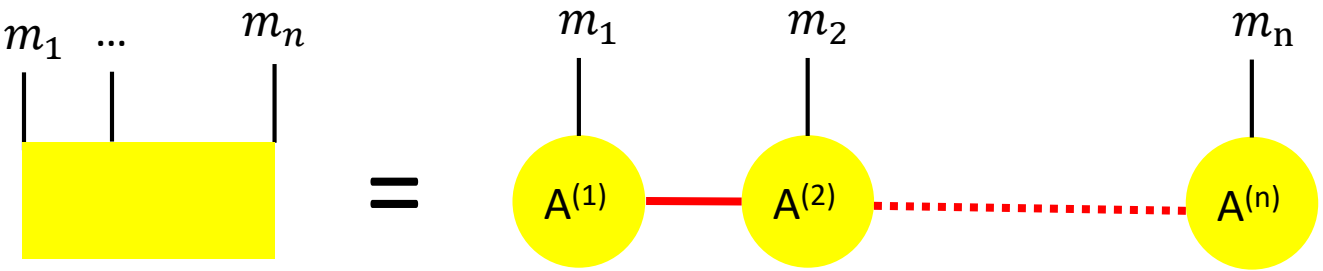
$$C_{a_1 m_2, \ldots m_n} = \sum_{a_2} A^{(2),m_2}_{a_1 a_2} C_{a_1 a_2 \, \mathrm{m}_3 \ldots m_n}$$

$$\psi^{m_1 m_2 \ldots m_n} \rightarrow C_{m_1 m_2 \ldots m_n} = \sum_{a_1, a_2} A^{(1),m_1}_{1 a_1} A^{(2),m_2}_{a_1 a_2} C_{a_1 a_2 m_3 \ldots m_n}$$

Until to the last physical d.o.f., we have

$$\psi^{m_1 m_2 \ldots m_n} = \sum_{\{a\}} A^{(1),m_1}_{1 a_1} A^{(2),m_2}_{a_1 a_2} \ldots A^{(\mathrm{L}),m_n}_{a_n 1}$$

Thus we can write degrade high rank tensor to low rank.



Generalize to higher dimension

For lattice higher than 1D, we have PEPS for 2D. And 3D and more higher dimension can be generalized in the same way.