Tensor Train Representation

$\sigma = (\sigma_1, \ldots, \sigma_L)$ :: Physical Multi-index

$|\psi\rangle = \sum_{\sigma \in \sigma} C_\sigma |\sigma\rangle$ :: Wavefunction coefficients

We illustrate the tensor C_σ as:

where lines represent physical indices (e.g. σ_i) and rectangles represent "sites" of coefficients.

Singular value decompositions allow us to distinguish each physical index by its own array as long as we introduce new "bond" indices to capture any non-classical correlations

To distingush a physical index, we reshape the tensor so that it is a matrix with the distinguished index on an axis is separate from the others: $(\sigma_1 \cdots \sigma_L) \longrightarrow (\sigma_1, \sigma_2 \cdots \sigma_L)$

In terms of lowered (row) indices and raised (column) indices, this operation means

$$C_{\sigma_1 \cdots \sigma_L} \rightarrow C_{\sigma_1}^{\sigma_2 \cdots \sigma_L}$$

Alternatively

$$C_{\sigma_1 \cdots \sigma_L} \rightarrow C^{\sigma_1}_{\sigma_2 \cdots \sigma_L}$$

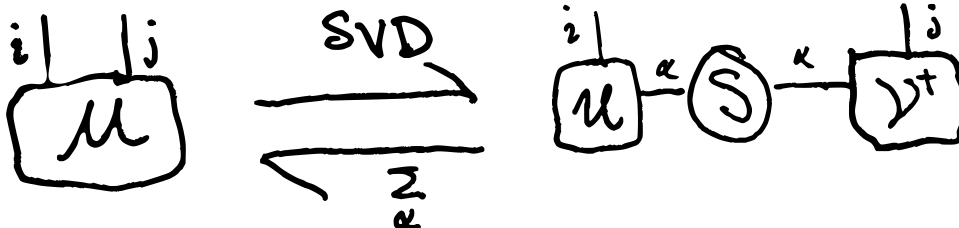
In general this is a matrix with upper and lower indices:

$$M_i^j$$

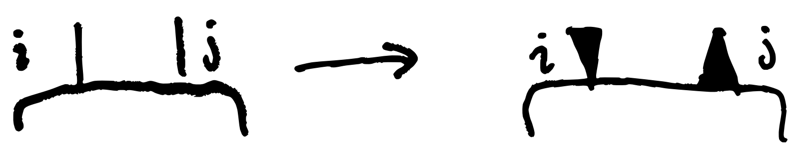
Then SVD separates these like so:

$$M_i^j = \sum_\alpha U_i^\alpha S_\alpha^\alpha V^{\dagger j}_\alpha$$

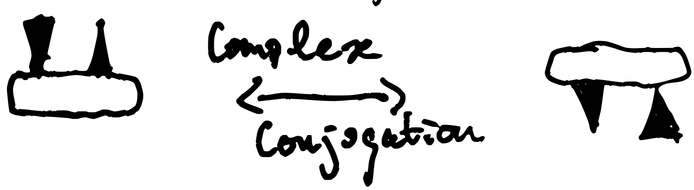
Indices i and j are symbolic or can take on particular values. For fixed values of i, j, we obtain a coefficient of $\mathcal{M}$ from the previous formula. If we think of it symbolically, it is a whole matrix multiplication. In any case,



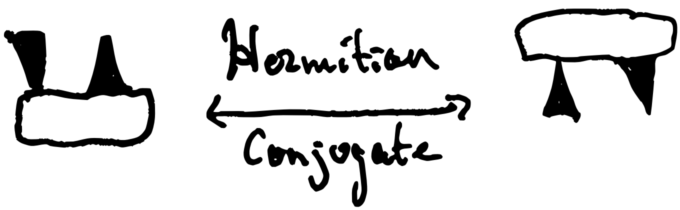
SVD guarantees that $\mathcal{U}, \mathcal{V}^{\dagger}$ are unitary and S diagonal. I think that the notation misses an important piece of information: whether an index is raised or lowered (column/row). I propose using triangles instead of sticks



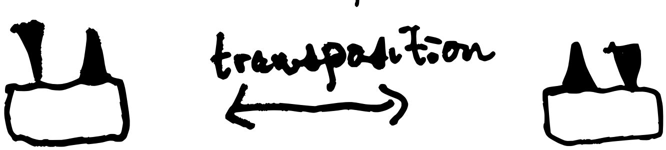
where ▼ means a lowered index, which is like $|\psi\rangle$ in Dirac's notation, and ▲ means a raised index, which is like $\langle\psi|$ in Dirac's notation EXCEPT for complex conjugation. Instead we represent complex conjugation by whether the bond is on the top or bottom of the rectangle:



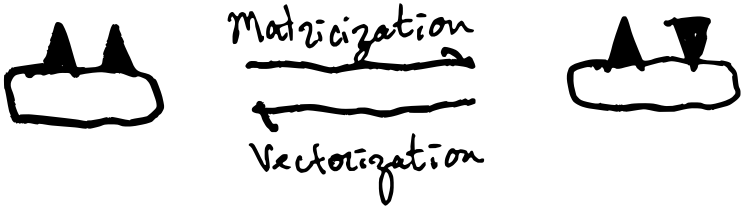
Thus the Hermitian conjugate also reflects the triangles:



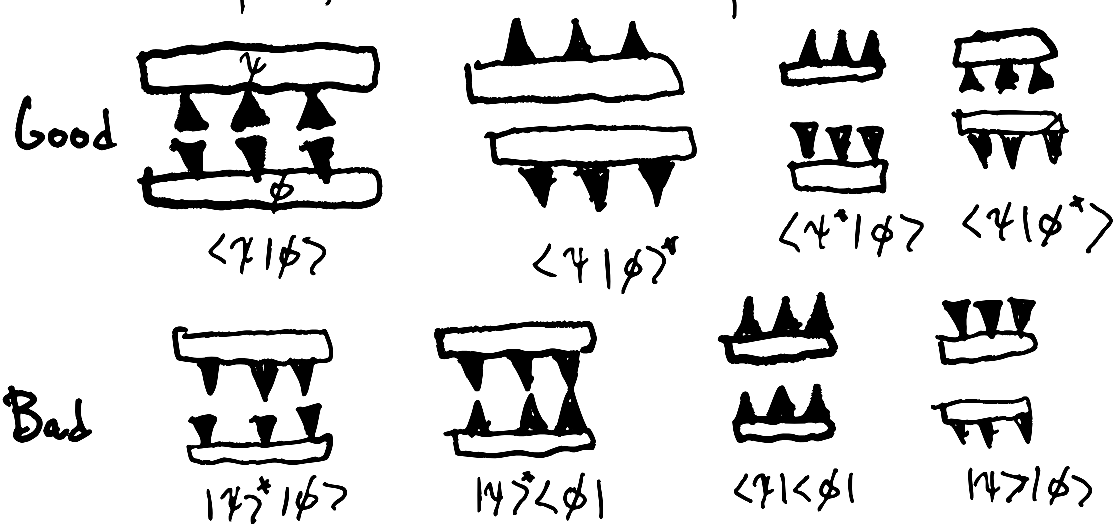
That means transposition is



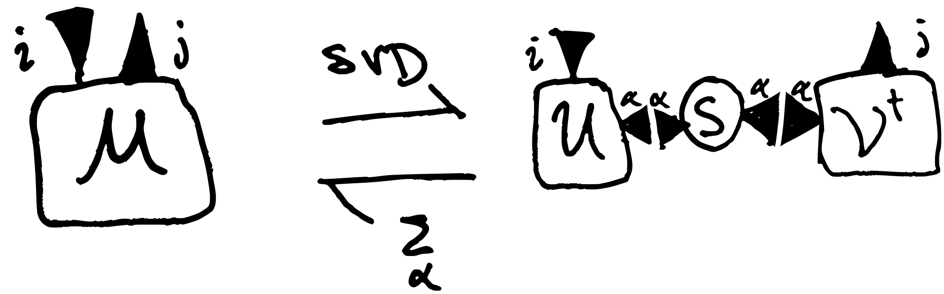
We can selectively flip indices by particular reshapes



Now the orientation of the triangles tells us whether indices are in compatible positions to be summed over directly: when bases meet, they can be contracted, with a matrix multiplication. For example, an inner product is:

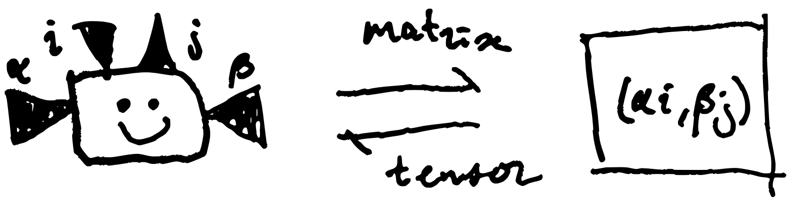


Our SVD diagram becomes:

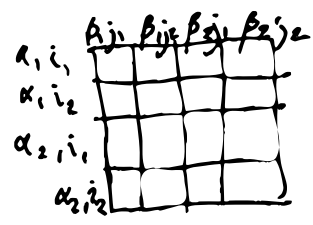


That's my proposal to make the notation even more difficult and exciting! It now gives a way to keep track of individual reshapes of sites / rectangles. I think it has a flavor of how chemists draw bonds in molecular diagrams! I should also explain about the bonds on the horizontal sides. Bonds are never allowed to move!

(i.e. raised or lowered). They can be conjugated with the rest of the matrix, but should not be transposed. They are reference points that preserve the topology of the tensor network by serving as pointers between sites. If a bond is on the right of a site, it is a raised/column index. If it is left, it is lowered/row. We write this as

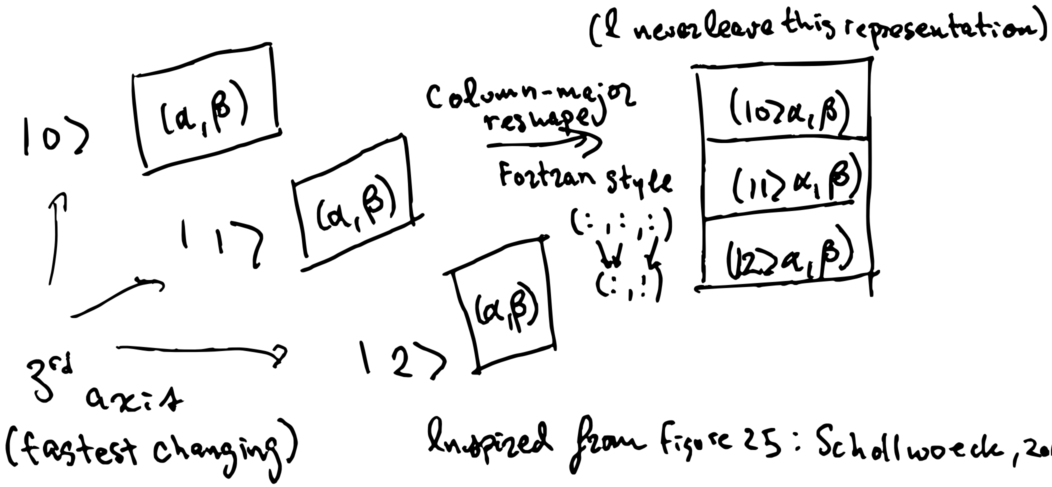


These grouped indices on an axis, multi-indices, partition a matrix into chunks, like so:



One should choose an endianness convention

The other approach to sites than matrices with multi-indices is to use higher dimensional arrays where each axis is its own index. Then liberal use of numpy's "einsum" can enable tensor contractions and products. However one still has to reshape everything into a matrix each time you do SVD, i.e.:



Inspired from figure 25: Schollwoeck, 2011

While the tensor network diagrams are a nice simplification, they are a severe simplification with respect to the implementation details. Hopefully my remarks cleared that up a bit. With this diagrammatic language, we can express tensor algorithms. That takes too long to explain well so I refer you to TensorNetwork.org for a good introduction to the manipulations which one uses to do these.