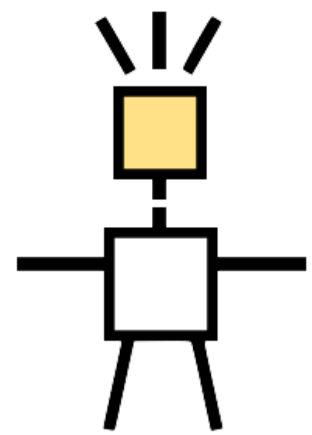
ITensor



E.M. Stoudenmire

Apr 2018 - Sao Paulo

FLATIRON INSTITUTE

SIMONS FOUNDATION

Tensor Renormalization Group

The tensor renormalization group or TRG
is an elegant algorithm to contract a lattice of tensors

This algorithm was proposed by Levin and Nave
(cond-mat/0611687)

It is called a "renormalization group" because it decimates the tensor network in a heirarchical fashion

Classical Partition Function

A key application of TRG is computing properties of classical lattice models

We will use it to evaluate the partition function, but can also evaluate local observables, and even conformal field theory scaling dimensions!

Classical Ising Model

The partition function of the classical Ising model is

$$Z = \sum_{\sigma_1 \sigma_2 \sigma_3 \ldots} e^{-E(\sigma_1,\sigma_2,\sigma_3,\ldots)/T}$$

Where $E(\sigma_1, \sigma_2, \sigma_3, \ldots)$ is the energy function

and T is the temperature

The spin variables take the values $\sigma = +1, -1$

One Dimensional Ising Model

In one dimension, the energy of the Ising model is

$$E(\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_N) = \sigma_1\sigma_2 + \sigma_2\sigma_3 + \sigma_3\sigma_4 + \ldots + \sigma_N\sigma_1$$

So anti-aligned (antiferromagnetic) configurations have lowest energy

Taking periodic boundary conditions, so interaction between first and last Ising spin

Transfer Matrix Trick

To compute the partition function, use following trick

$$Z = \sum_{\{\sigma\}} \exp\left(\frac{-1}{T}\sum_n \sigma_n \sigma_{n+1}\right)$$

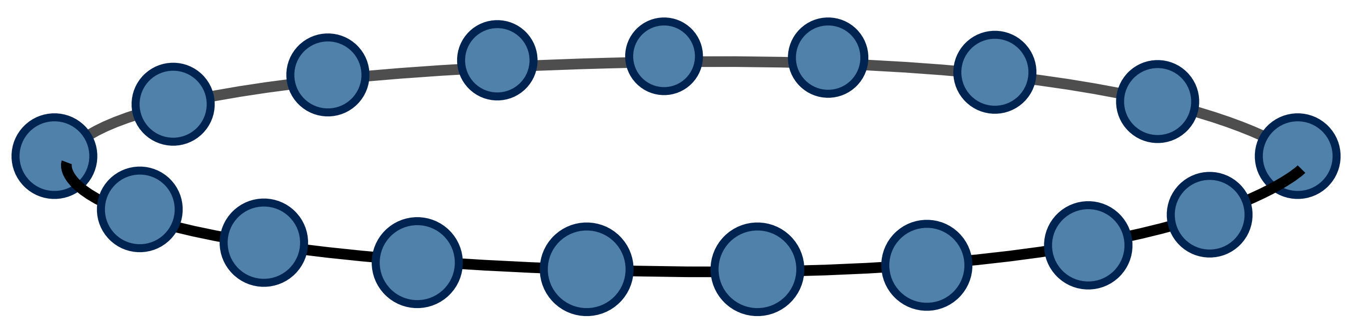
$$= \sum_{\{\sigma\}} \prod_n e^{-(\sigma_n \sigma_{n+1})/T} = \mathrm{Tr}\left(M^N\right)$$

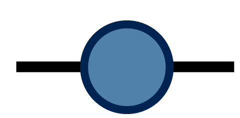
Where the matrix M is defined as $M_{\sigma\sigma'} = e^{-(\sigma\sigma')/T}$

Transfer Matrix Trick

Can visualize the transfer-matrix form of Z as a tensor network

$$Z = \mathrm{Tr}\left(M^N\right)$$



 $= M$

Transfer Matrix Trick

One-dimension case exactly solvable

1) Diagonalize M to get its eigenvalues: $\{\lambda_1, \lambda_2\}$

2) Observe that:

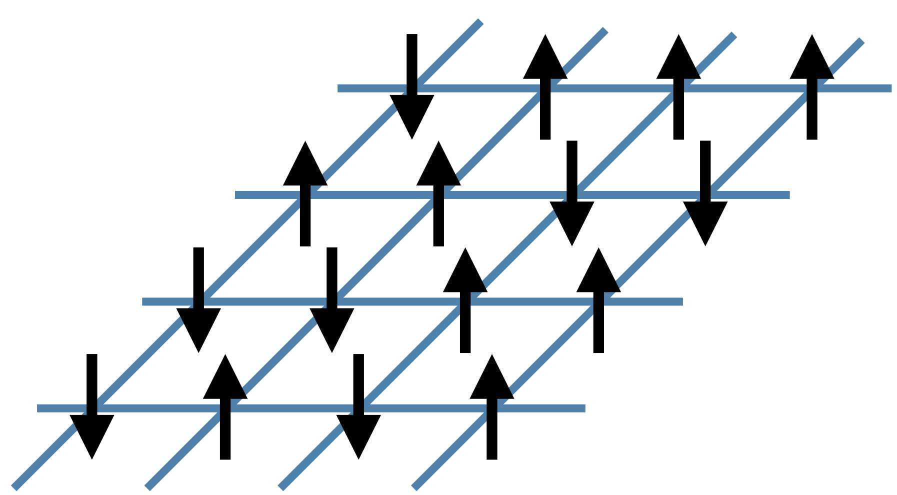
$$Z = \mathrm{Tr}\left(M^N\right) = \lambda_1^N + \lambda_2^N$$

Using basis-invariance property of the trace

Two-dimensional Ising Model

Here we're primarily interested in the two-dimensional Ising model

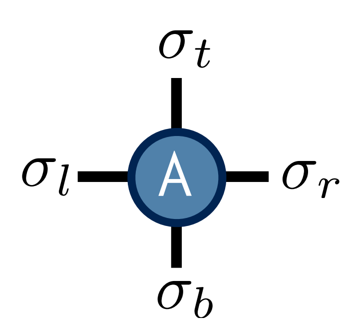
$$E(\sigma_1, \sigma_2, \ldots) = \sum_{\langle ij \rangle} \sigma_i \sigma_j$$



Two-dimensional Transfer Tensor

Again possible to define Z by replacing transfer matrix, with a "transfer tensor"

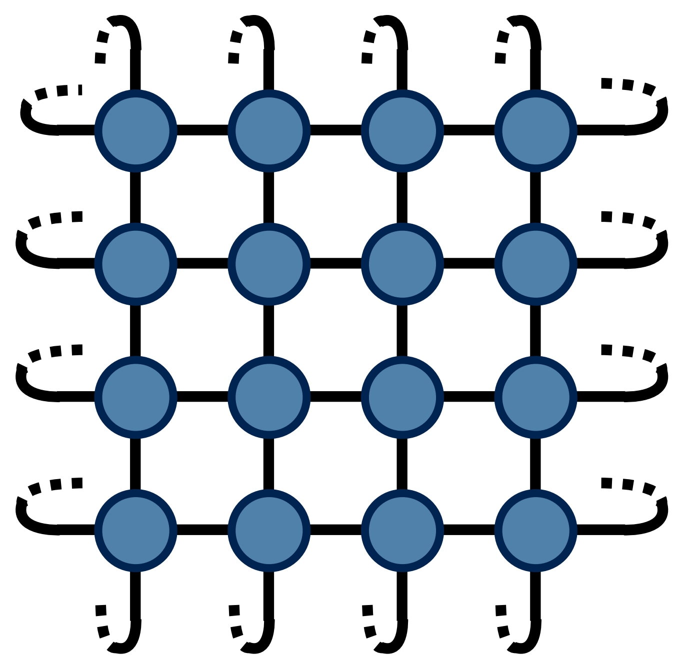
$$A^{\sigma_t \sigma_r \sigma_b \sigma_l} = e^{-(\sigma_t \sigma_r + \sigma_r \sigma_b + \sigma_b \sigma_l + \sigma_l \sigma_t)/T}$$



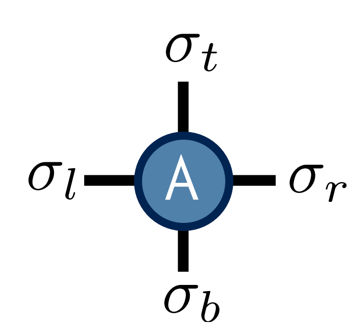
Two-dimensional Transfer Tensor

In terms of tensor "A", partition function Z is

$Z =$



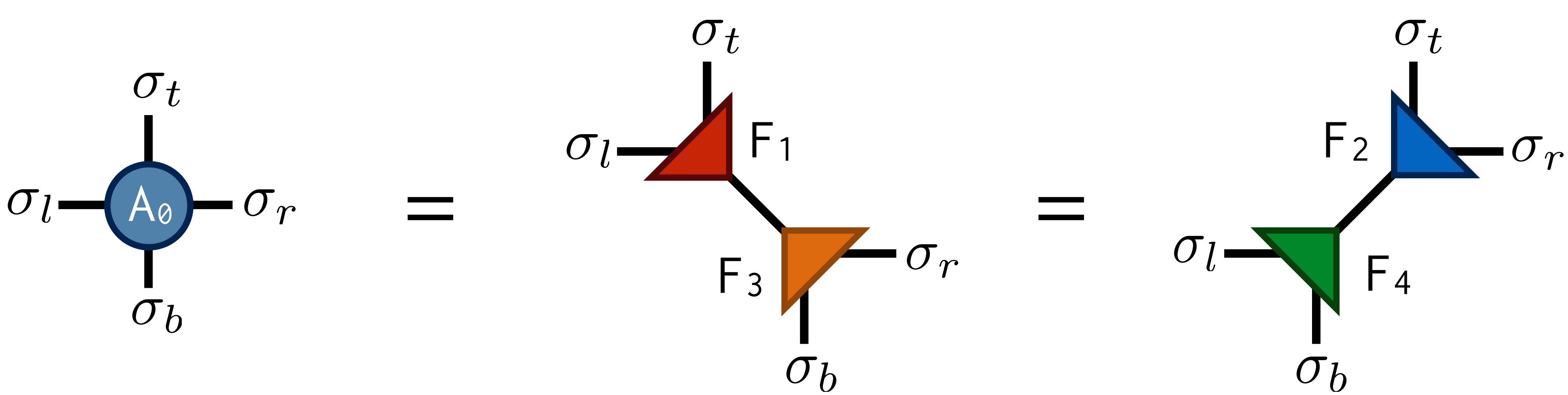
$$A^{\sigma_t \sigma_r \sigma_b \sigma_l} = e^{-(\sigma_t\sigma_r + \sigma_r\sigma_b + \sigma_b\sigma_l + \sigma_l\sigma_t)/T}$$



Tensor Renormalization Group

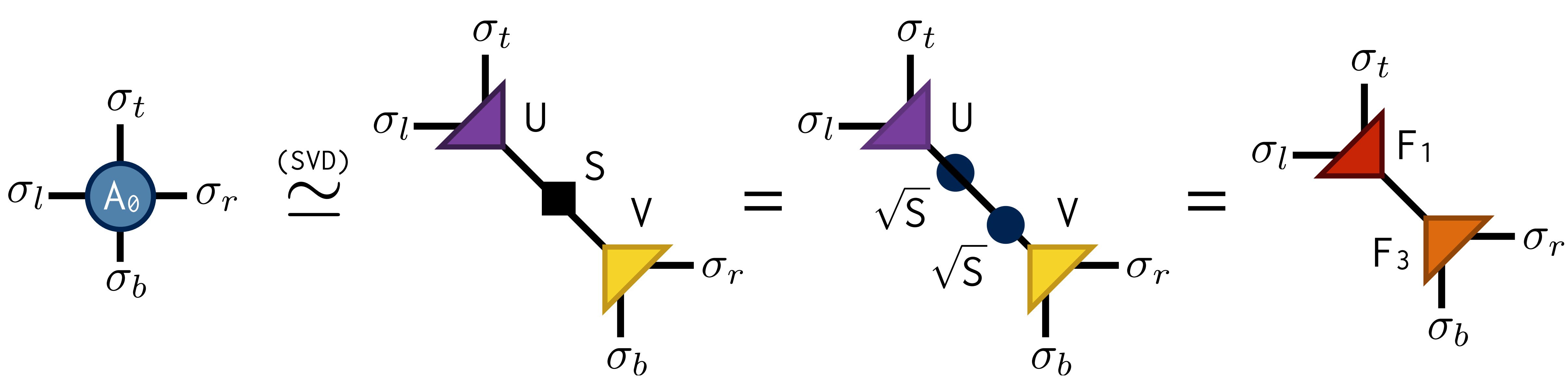
The TRG strategy to compute Z is in two steps

First, use the SVD to factorize the A tensor two ways:



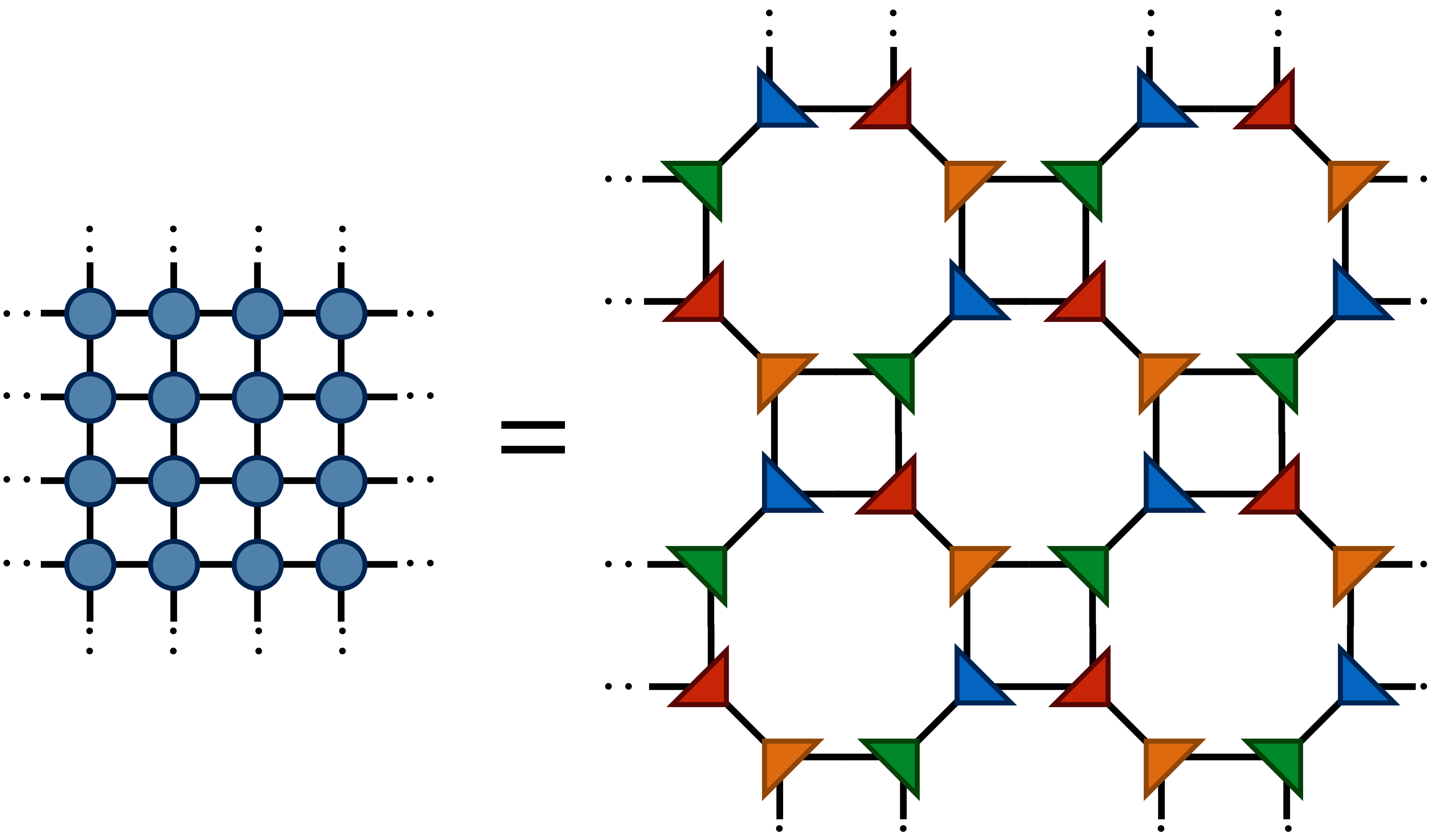
Tensor Renormalization Group

Each factorization can be derived from an SVD by grouping the square roots of singular values with unitary tensors:



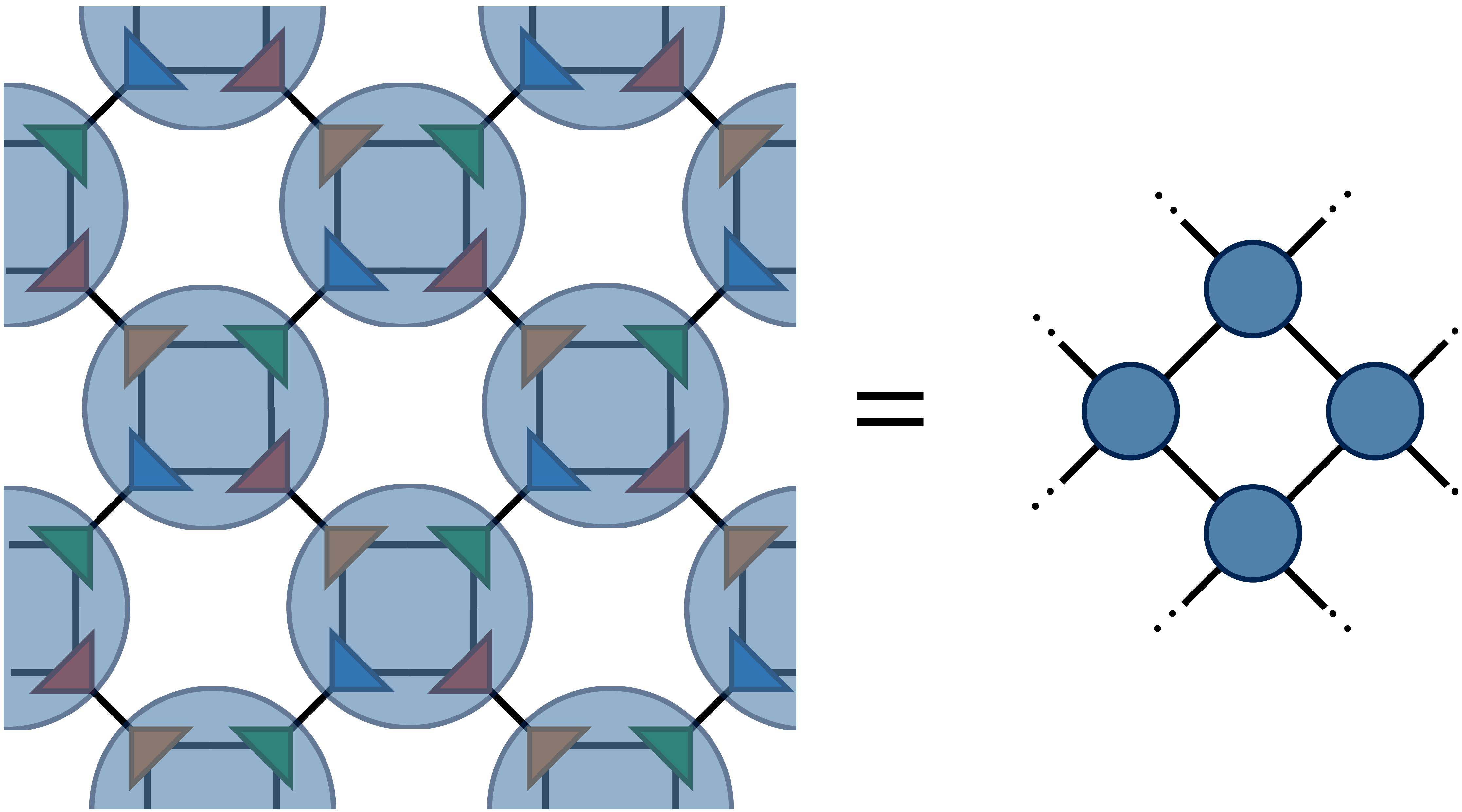
Tensor Renormalization Group

Insert these factorizations to rewrite the network as:



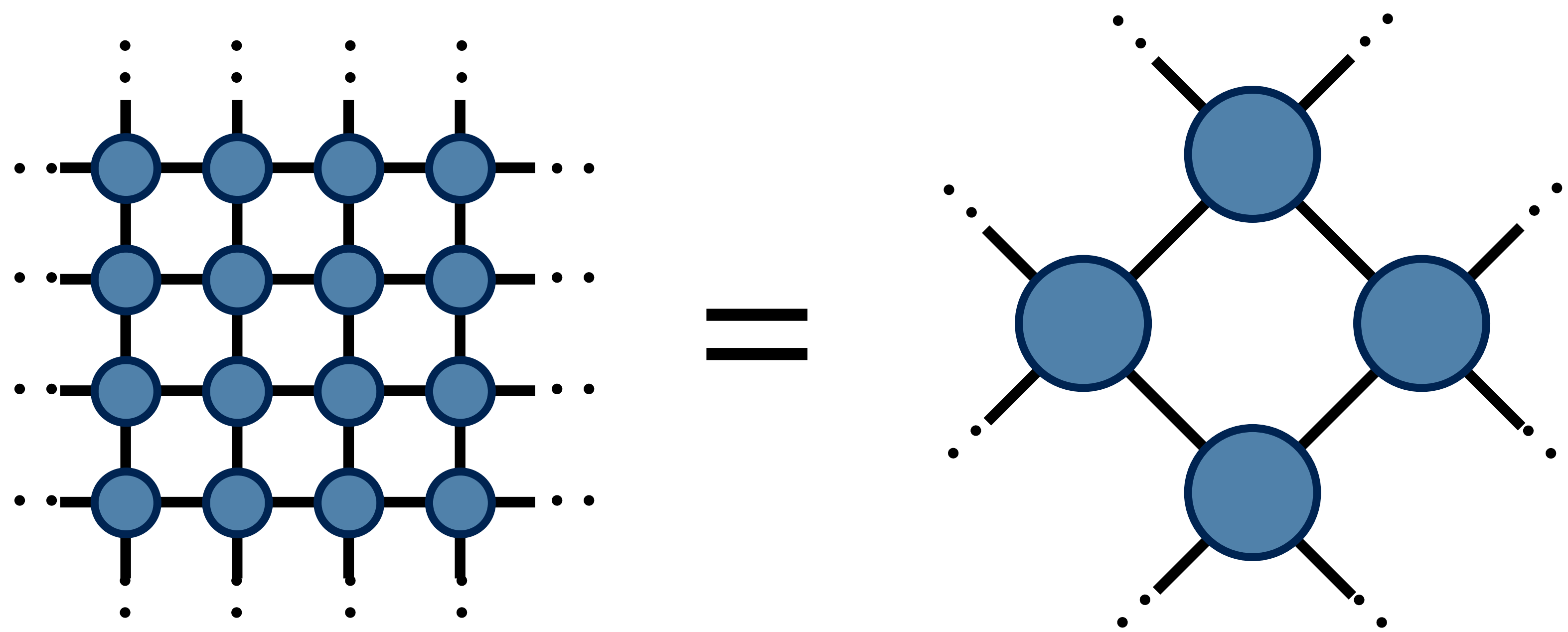
Tensor Renormalization Group

Then group blocks of four tensors as follows:



Tensor Renormalization Group

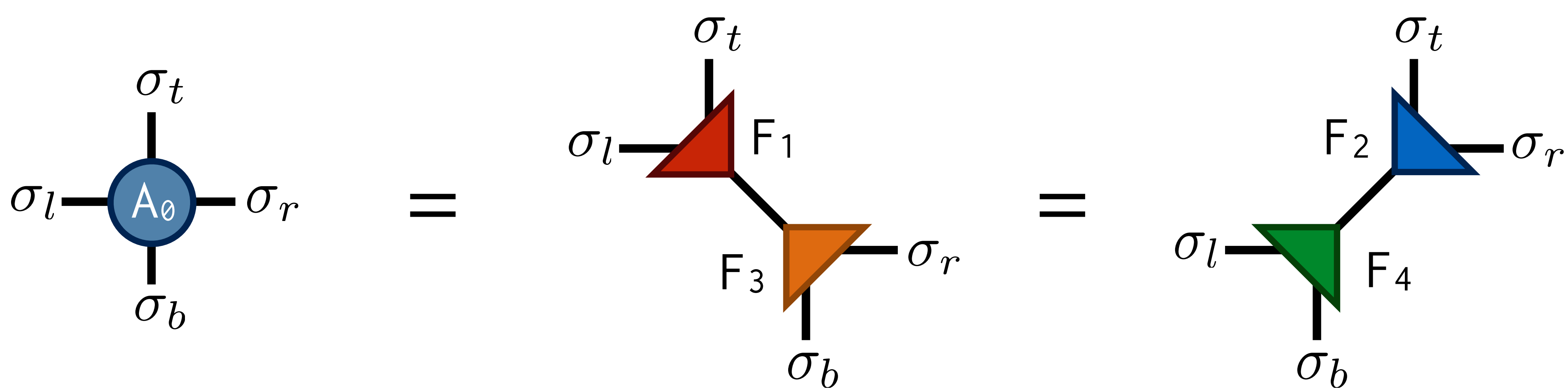
Effect of Step 1 followed by Step 2



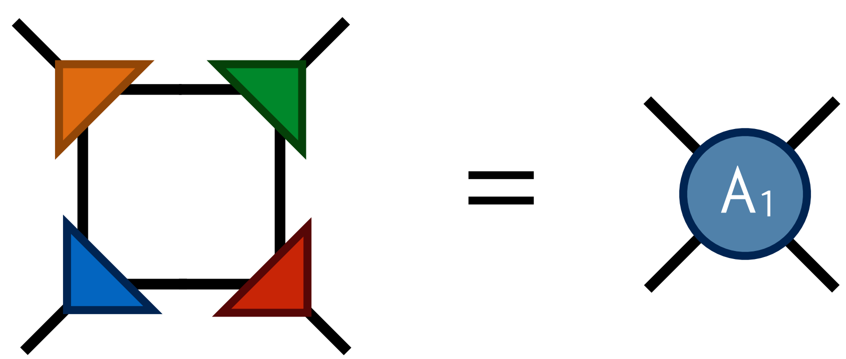
Tensor Renormalization Group

Two steps of TRG algorithm:

Step #1:



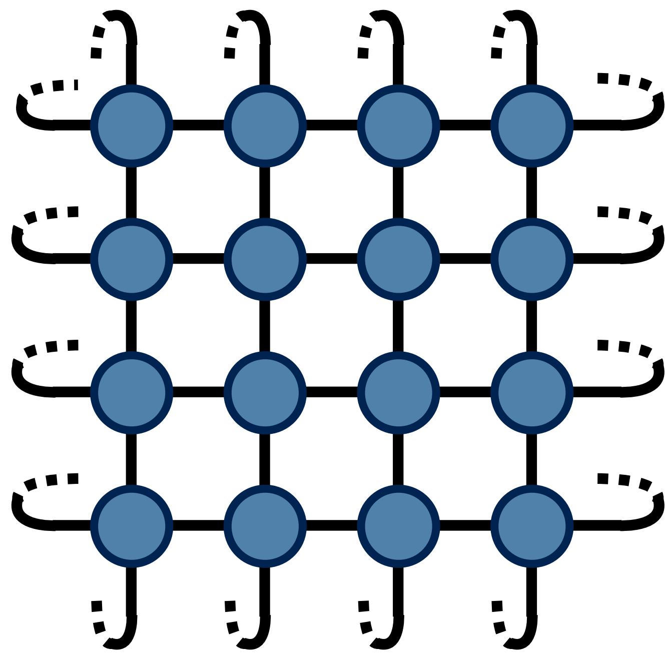
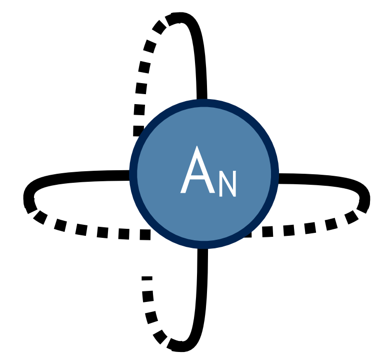
Step #2:



Can be repeated to turn network of A_1's into A_2's etc.

Tensor Renormalization Group

After iterating enough steps (how many?) entire network become a single A tensor A_N

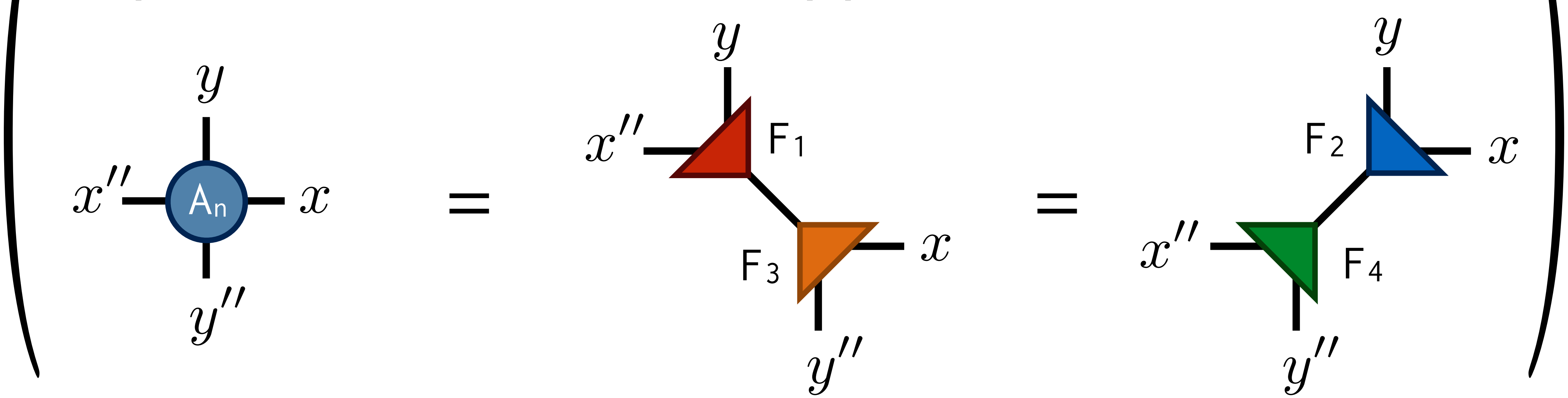
$Z =$  $=$ 

Tracing the A_N tensor gives the partition function

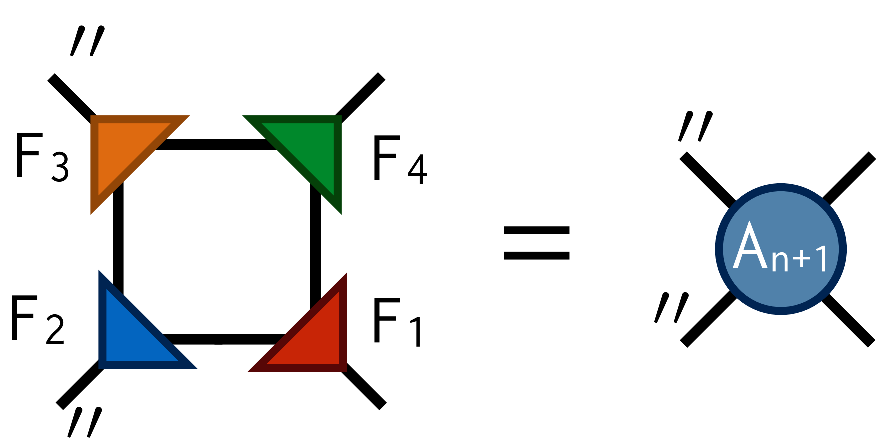
ITensor Code Activity

Read through the code `tutorial/trg/trg.cc`

Step 1 – factor two ways (approximately)



Your job: Step 2 – recombine factors, defining new A



Try following parameters:
T = 3.0
maxm = 20
topscale = 8

./trg [log(Z)/Ns = 0.149216]