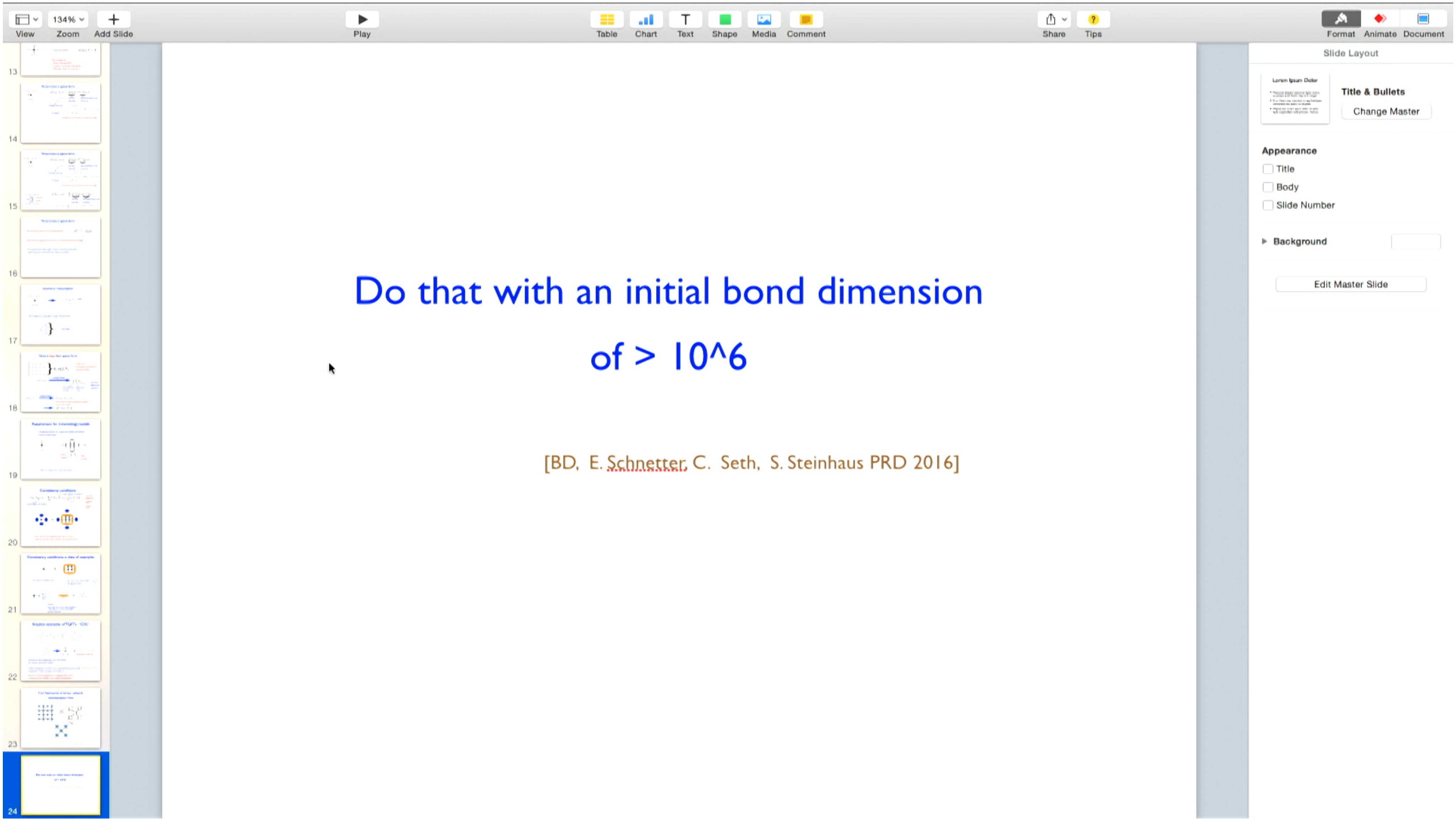
Title: Coarse-graining of Spin Foams - 4

Date: Jun 22, 2017 12:00 PM

URL: http://pirsa.org/17060085

Abstract:


Do that with an initial bond dimension
of > 10^6
[BD, E. Schnetter, C. Seth, S. Steinhaus PRD 2016]

Facilitate the computation

To allow for sufficiently complicated models:

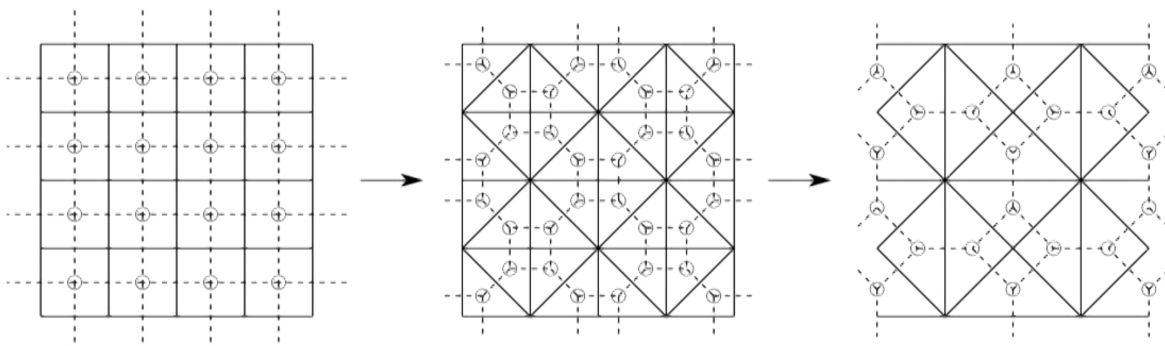
- Protect symmetry: built in into algorithm instead of trusting that it will be preserved (as it usually is not due to numerical errors)
- Use the split of the tensor into 'free data' and 'symmetry determined structure' to save computational time and memory
- Make use coupling rules to avoid wasting memory and time (big factor!)
- Use the (coarse) observable induced by the symmetry to characterize your fixed point tensors and flow

Facilitate the computation

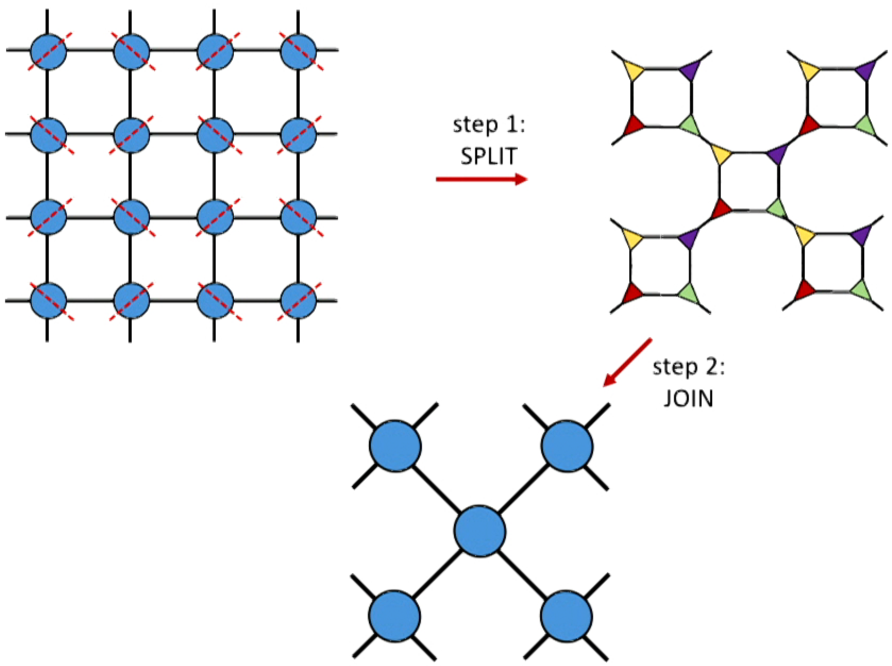
To allow for sufficiently complicated models:

- Protect symmetry: built in into algorithm instead of trusting that it will be preserved (as it usually is not due to numerical errors)
- Use the split of the tensor into 'free data' and 'symmetry determined structure' to save computational time and memory
- Make use coupling rules to avoid wasting memory and time (big factor!)
- Use the (coarse) observable induced by the symmetry to characterize your fixed point tensors and flow

- Use triangular version of the algorithm:
  This algorithm with t~chi^6 can be reorganized into one with t~chi^5.
  We will also need to save smaller objects (e.g. by a factor of 100)



Find fixed points of tensor network renormalization flow



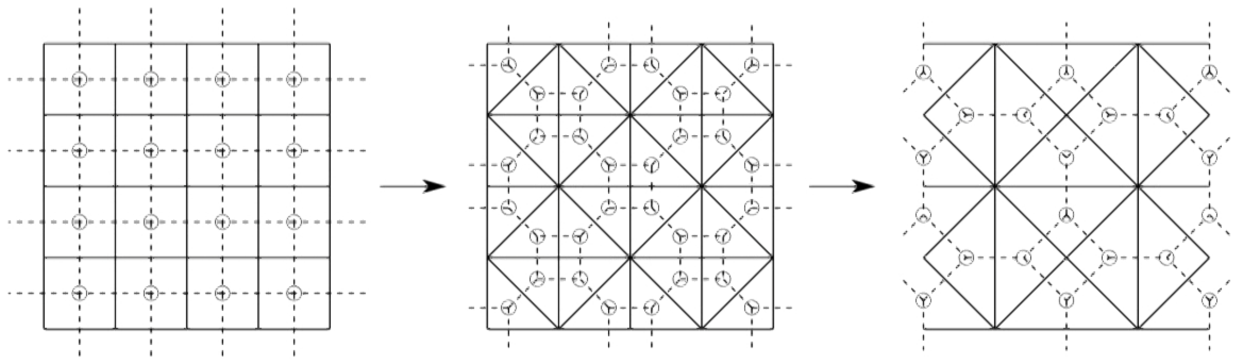
[from Guifre]

Facilitate the computation

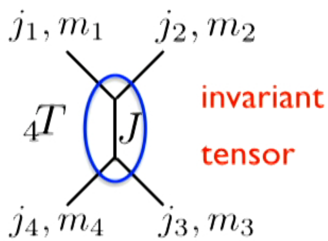
To allow for sufficiently complicated models:

- Protect symmetry: built in into algorithm instead of trusting that it will be preserved (as it usually is not due to numerical errors)
- Use the split of the tensor into 'free data' and 'symmetry determined structure' to save computational time and memory
- Make use coupling rules to avoid wasting memory and time (big factor!)
- Use the (coarse) observable induced by the symmetry to characterize your fixed point tensors and flow

- Use triangular version of the algorithm:
  This algorithm with t~chi^6 can be reorganized into one with t~chi^5.
  We will also need to save smaller objects (e.g. by a factor of 100)



Symmetry protecting algorithm



$$ {}_4T(\{(j_i, m_i)\}) = \sum_J \underbrace{{}_4\hat{T}^J(\{j_i\})}_{\text{(almost) free data}} \; \underbrace{{}_4C^{j_i;J}(\{m_i\})}_{\text{symmetry-determined structure}} $$

Symmetry protection:
- Only the 'free data' (hat T) undergoes the renormalization flow.

Splitting:
- The matrix to be split appears in block form, with blocks labelled by 'coupling channel' J.
- The singular values per channel are interesting observables.

Superindices:
- To make use of coupling rules introduce index mappings. E.g.:
  K(J)=(j1,j2) labels all

Gluing:
- Sum only over spin values. 'Precontract' symmetry-determined structures to recouping symbols (6j symbols). These have to be convoluted into the renormalization flow:

Coarse graining flow

Gluing step gives output tensor:

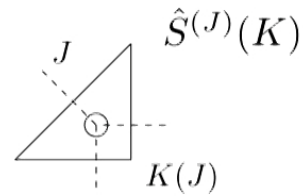
$$\hat{\mathcal{T}}^{(\{\bar{J}\})}(\{j_1\},\{j_2\};\{j_c\},\{j_a\}) = \sum_{\{j_b\}} \frac{\sqrt{(-1)^{j_c^+ + j_a^+ + \bar{J}^+}}}{\sqrt{d_{\bar{J}^+}}\sqrt{d_{j_b^+}}} \frac{\sqrt{(-1)^{j_c^- + j_a^- + \bar{J}^-}}}{\sqrt{d_{\bar{J}^-}}\sqrt{d_{j_b^-}}} \frac{\sqrt{(-1)^{(j_c^+)' + (j_a^+)' + (\bar{J}^+)'}}}{\sqrt{d_{(\bar{J}^+)'}}\sqrt{d_{(j_b^+)'}}} \frac{\sqrt{(-1)^{(j_c^-)' + (j_a^-)' + (\bar{J}^-)'}}}{\sqrt{d_{(\bar{J}^-)'}}\sqrt{d_{(j_b^-)'}}} \times$$
$$\times \sqrt{d_{j_1^+} d_{j_2^+}} \sqrt{d_{j_1^-} d_{j_2^-}} \sqrt{d_{(j_1^+)'} d_{(j_2^+)'}} \sqrt{d_{(j_1^-)'} d_{(j_2^-)'}} \times$$
$$\times \begin{bmatrix} j_c^+ & j_a^+ & \bar{J}^+ \\ j_1^+ & j_2^+ & j_b^+ \end{bmatrix} \begin{bmatrix} j_c^- & j_a^- & \bar{J}^- \\ j_1^- & j_2^- & j_b^- \end{bmatrix} \begin{bmatrix} (j_c^+)' & (j_a^+)' & (\bar{J}^+)' \\ (j_1^+)' & (j_2^+)' & (j_b^+)' \end{bmatrix} \begin{bmatrix} (j_c^-)' & (j_a^-)' & (\bar{J}^-)' \\ (j_1^-)' & (j_2^-)' & (j_b^-)' \end{bmatrix}$$
$$\times (\hat{S})^{(\{j_1\})}(\{j_b\},\{j_a\})(\hat{S})^{(\{j_2\})}(\{j_c\},\{j_b\}) \quad . \qquad \text{(C1)}$$

symmetry determined structure

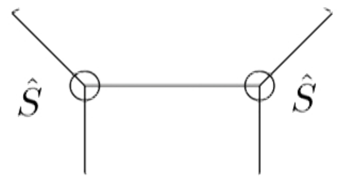
from splitting input tensor

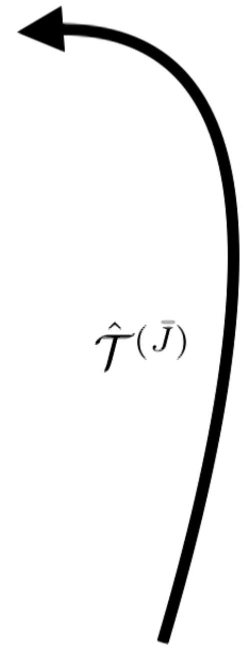
Triangular algorithm

input three-valent tensor:

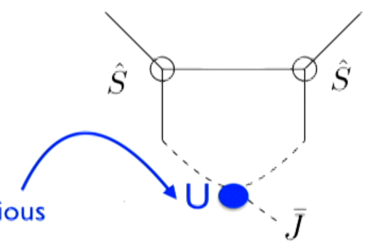


gluing two gives a four-talent tensor:





summarize (and truncate) two indices into one
to get back the output three-valent tensor:



from SVD of previous
four-talent tensor

The second and third step can be performed at fixed coupling channel.
We do not need to save the full four-talent tensor.

Savings

- savings by using super-indices for k=4: factor 1000
- savings by triangular algorithm:

Level k	$j_{\max}$	Maximal $K(j)$	Size of $\hat{S}$	Size of $\hat{S}$ in GB	Size of $\hat{T}$	Size of $\hat{T}$ in GB	Size of block in GB
4	2	$K(1) = 5$	11^4	~ 0.00022	43^4	~ 0.051	~ 0.0058
5	2	$K(1) = 6$	14^4	~ 0.0006	70^4	~ 0.36	~ 0.025
6	3	$K(1) = K(2) = 8$	24^4	~ 0.005	160^4	~ 9.77	~ 0.25
7	3	$K(2) = 10$	30^4	~ 0.013	246^4	~ 54.6	~ 1.5
8	4	$K(2) = 13$	45^4	~ 0.062	461^4	~ 673.1	~ 12.2
9	4	$K(2) = 15$	55^4	~ 0.14	671^4	~ 3021	~ 38.2
10	5	$K(2) = K(3) = 18$	76^4	~ 0.5	1112^4	~ 23000	~ 165

Higher dimensions

and

models with gauge symmetries

[BD, S. Mizera, S. Steinhaus NJP 2016]

[C. Delcamp, BD, 2016]

Savings

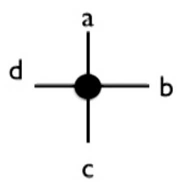
- savings by using super-indices for k=4:   factor 1000
- savings by triangular algorithm:

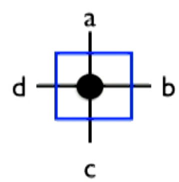
Level k	$j_{\max}$	Maximal $K(j)$	Size of $\hat{S}$	Size of $\hat{S}$ in GB	Size of $\hat{T}$	Size of $\hat{T}$ in GB	Size of block in GB
4	2	$K(1)=5$	11^4	~ 0.00022	43^4	~ 0.051	~ 0.0058
5	2	$K(1)=6$	14^4	~ 0.0006	70^4	~ 0.36	~ 0.025
6	3	$K(1)=K(2)=8$	24^4	~ 0.005	160^4	~ 9.77	~ 0.25
7	3	$K(2)=10$	30^4	~ 0.013	246^4	~ 54.6	~ 1.5
8	4	$K(2)=13$	45^4	~ 0.062	461^4	~ 673.1	~ 12.2
9	4	$K(2)=15$	55^4	~ 0.14	671^4	~ 3021	~ 38.2
10	5	$K(2)=K(3)=18$	76^4	~ 0.5	1112^4	~ 23000	~ 165

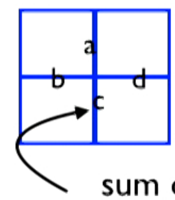
Making TNWs more flexible:

Decorated TNWs

TNW's in 2D:

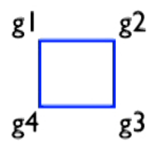


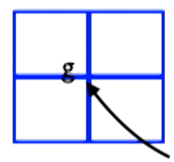




sum over variables
associated edges

Other possibilities:

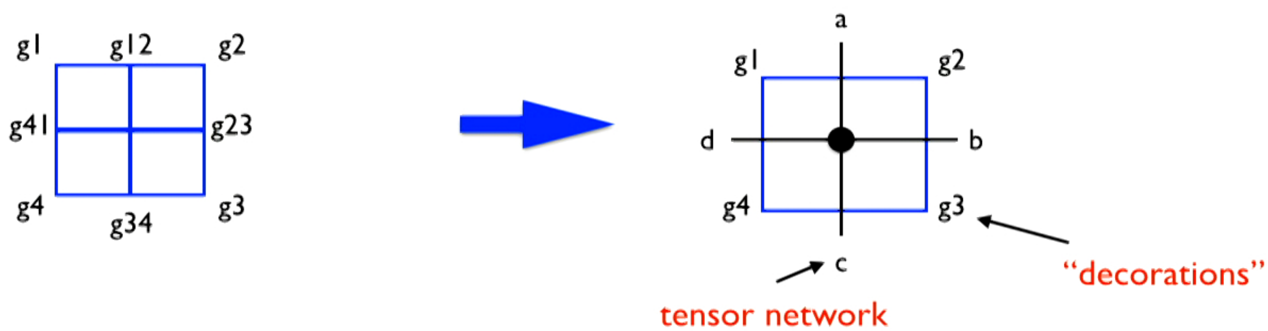




sum over variables
associated to vertices

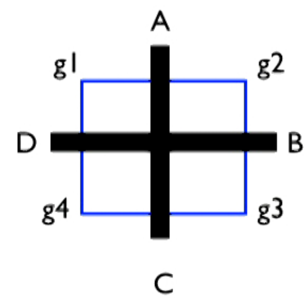
An algorithm which keeps access to some of the original variables?
These can be used as observables.

Decorated TNW



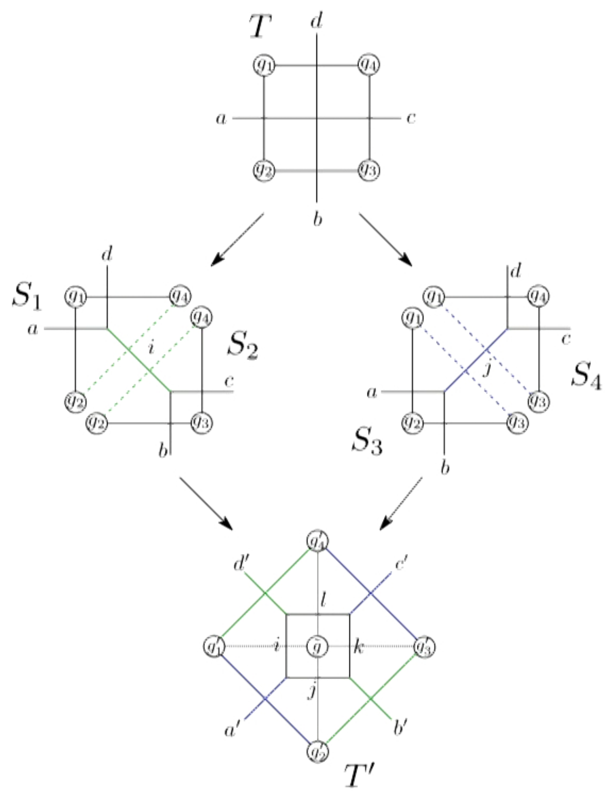
Gluing more squares

More decoration variables are turned into tensor network labels.



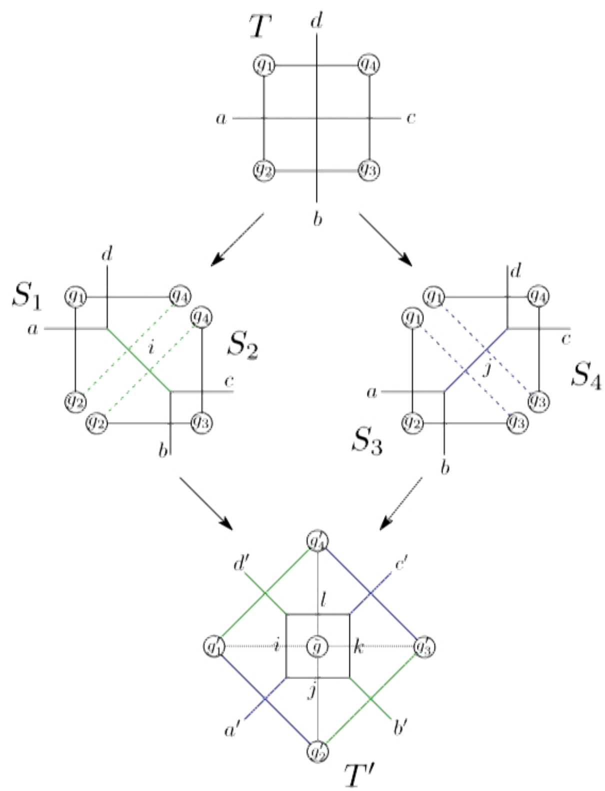
Decorated TNW coarse graining

Algorithm very similar to what we had before.
But the SVD decomposed into "SVD per block" very similar to what we discussed for the symmetry protected algorithm.
Indeed in both cases we want to preserve certain observables.



Decorated TNW coarse graining

Algorithm very similar to what we had before.
But the SVD decomposed into "SVD per block" very similar to what we discussed for the symmetry protected algorithm.
Indeed in both cases we want to preserve certain observables.



Application to gauge theories (in 3D … or 4D)

Gauge theories:

- There are unphysical gauge degrees of freedom. Do not want to deal with these!

(Even much more than in the symmetry protected algorithm this leads to huge savings.)

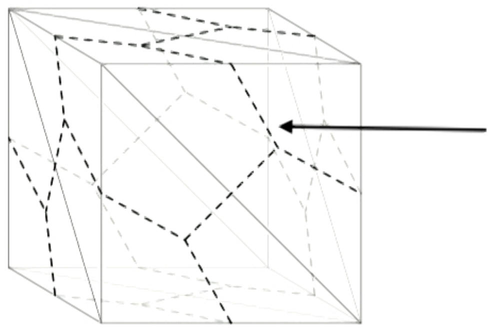
- Gauge invariant information encoded in "spin networks":
  - want to preserve observables described by spin networks as it has interesting geometric interpretation
  - expect that it also coarse grains in a geometric way

  (similar to what we discussed in the symmetry protected algorithm)

Decorated tensor networks are sufficiently flexible:
Lead to a doable and testable algorithm.

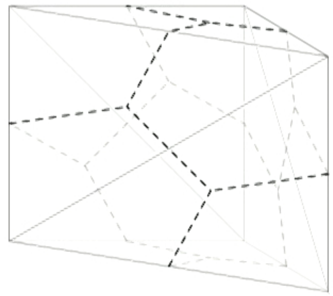
(Abelian) gauge theories in 3D

Basic building block:

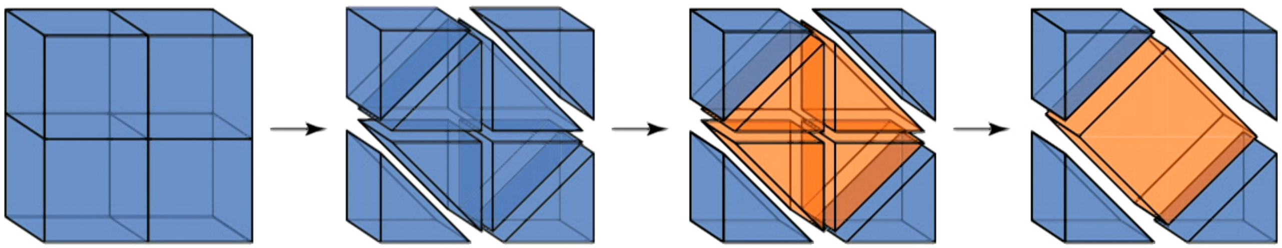


Decoration:
"Spin network":
Graph on boundary of building block
labelled with group representations.

Satisfies coupling rules (at nodes):
for Abelian models labels on 7 links
determine all the remaining ones (11 links).

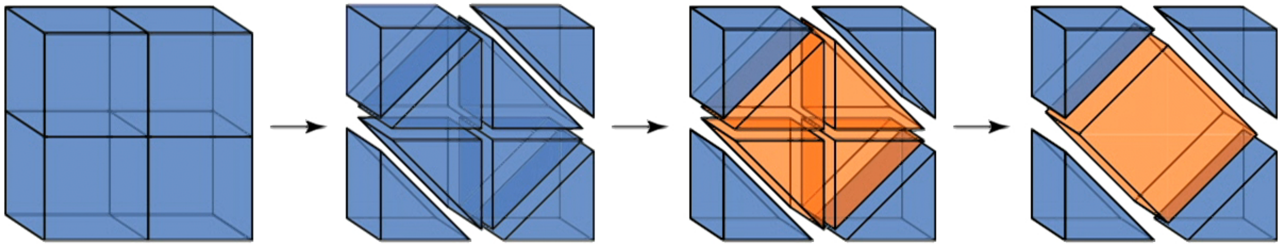


Algorithm in 3D

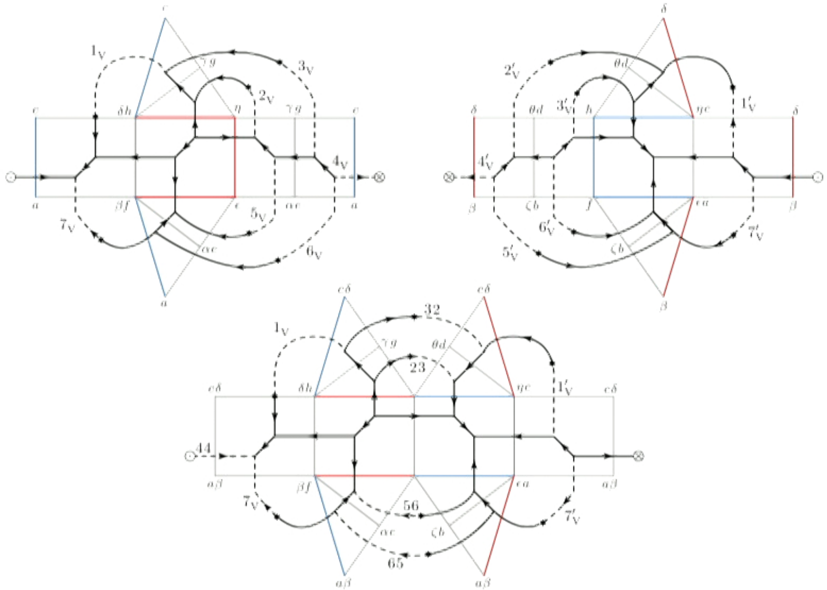


… repeat after rotation …

Algorithm in 3D



in more detail:
(Non-Abelian model)



Features

- decorations given by (boundary) spin network:
  interesting observable and coarse grained in a geometric way

- splitting procedure via SVD: again with a block structure due to decorations
  encodes interesting information on how intertwiner degrees of freedom flow
  (relevant information for spin foams)

- refined boundary states: can be either encoded in tensor network structure
  or refined boundary spin networks

Non-Abelian models

… allow for key ingredient of spin foam models: non-trivial intertwiner degrees of freedom

Main question: Do spin foam models flow to new TQFTs (new fixed points)?
Do intertwiner degrees of freedom lead to new phases?

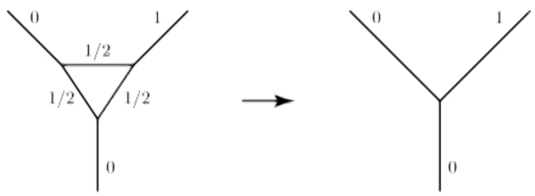
Non-Abelian models

… allow for key ingredient of spin foam models: non-trivial intertwiner degrees of freedom

Main question: Do spin foam models flow to new TQFTs (new fixed points)?
Do intertwiner degrees of freedom lead to new phases?

Non-Abelian models

- decorations given by (boundary) spin network:
  interesting observable and coarse grained in a geometric way
- but …
- for non-Abelian models coupling rules are not preserved under coarse graining:



Geometric interpretation: Curvature requires deformation of Gauss constraints.
(Curvature leads to torsion.)