Title: TBA

Date: Sep 30, 2015 12:00 PM

URL: http://pirsa.org/15090085

Abstract:


K
singular value
1
0,8
0,6
0,51
$S^3 \times \mathbb{R}$

Coarse graining spin net models

Sebastian Steinhaus

sebastian.steinhaus@desy.de

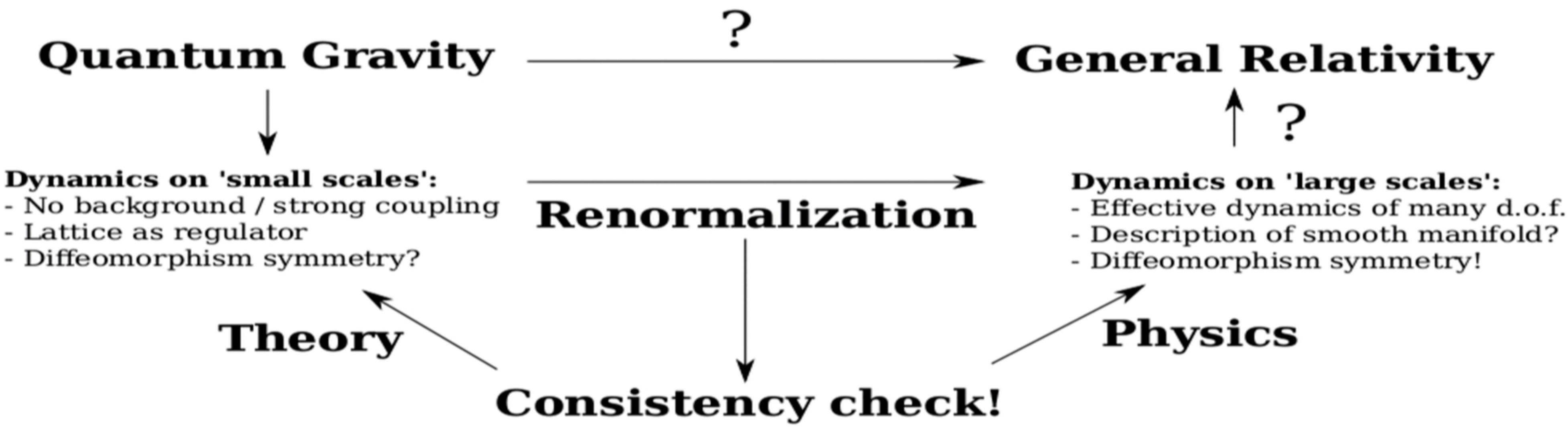
II. Institute for Theoretical Physics
University of Hamburg

Renormalization in Background Independent Theories
@ Perimeter Institute, Waterloo
30th September 2015

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Emmy Noether-Programm
DFG Deutsche Forschungsgemeinschaft

Renormalizing quantum gravity



In order to address these crucial questions, renormalization techniques must use both **analytical and numerical techniques!**

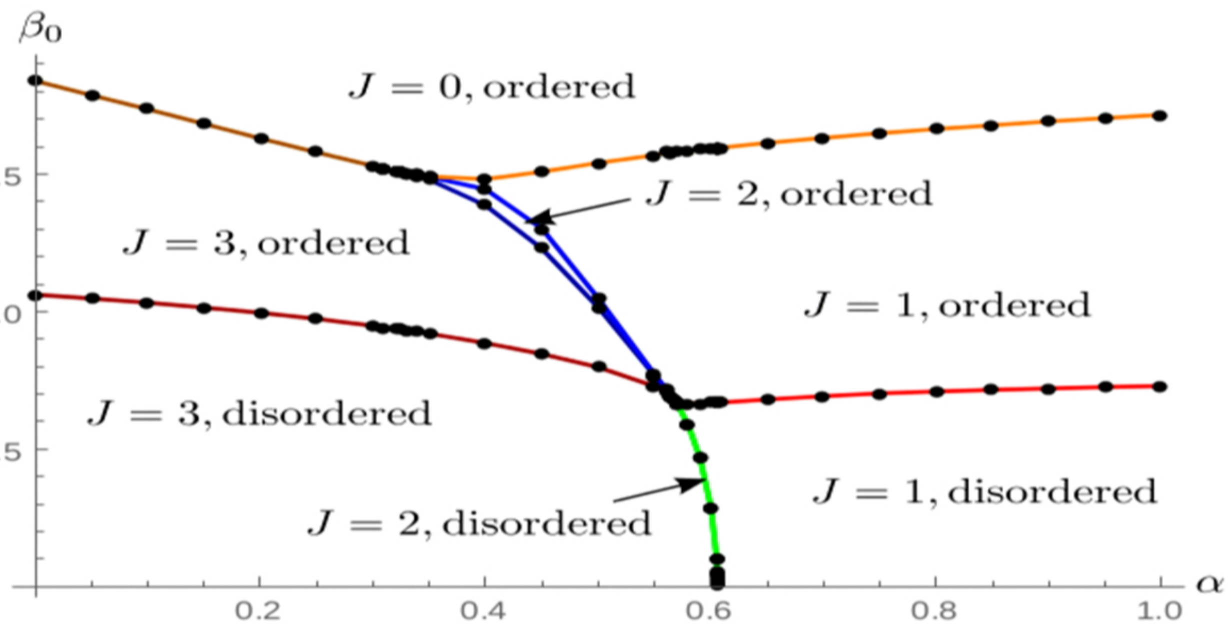
Tensor network renormalization as an example

- **Tensor network renormalization** [Levin, Nave '07, Gu, Wen '09, Vidal, Evenbly '14] is a **numerical** tool to efficiently study systems with many d.o.f.
  - **Coarse grain** tensor network encoding dynamics.
  - Evaluate (and **approximate**) partition function in parts.
  - Study **effective dynamics** at coarser scales.
  - A priori **no reference to background structure**.
- Successfully applied to **(analogue) spin foam models**. [Dittrich, Eckert, Martin-Benito '11; Dittrich, Martin-Benito, Schnetter '13; Dittrich, Martin-Benito, S.St. '13; S.St. '15; Dittrich, Girelli, Schnetter, Seth, S.St. w.i.p.]
- **Lattice gauge theories** [Dittrich, Mizera, S.St. '14] → Clement's talk!

Purpose of this talk:

- Explain algorithm for Ising model (take-home example)
- **Analytical improvements** make numerical investigation **feasible**.

What we are looking for! [S.St. '15]

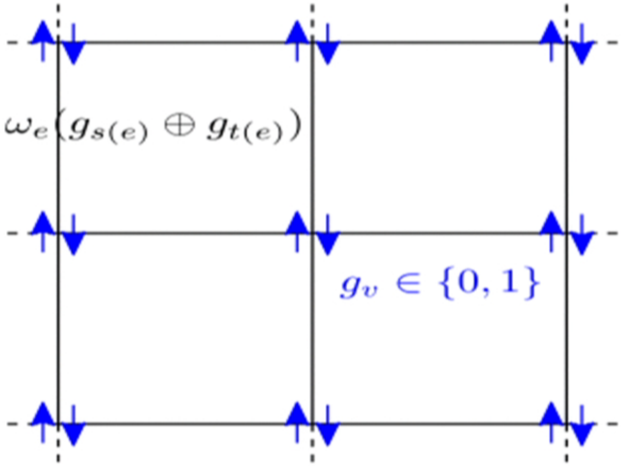


Uncover **different phases** of spin foam models (**geometric meaning?**) and phase transitions (**continuum limit?**).

Outline

1. Motivation
2. Tensor network renormalization - Ising model
   - Definition as a tensor network
   - The algorithm – general scheme
3. Improving the algorithm I - Symmetries
4. Improving the algorithm II - Triangular
5. Going beyond the Ising model - spin nets
6. Summary

The Ising model



- Study the Ising model on a 2D square lattice.
- Vertex v carries an **Ising spin** $g_v \in \mathbb{Z}_2 = \{0,1\}$ with $\oplus$: sum mod 2.
- Edge e carries an **edge weight**:

$$\omega_e(g_{s(e)} \oplus g_{t(e)}) = \exp\left(-2\beta(g_{s(e)} \oplus g_{t(e)}) + \beta\right)$$

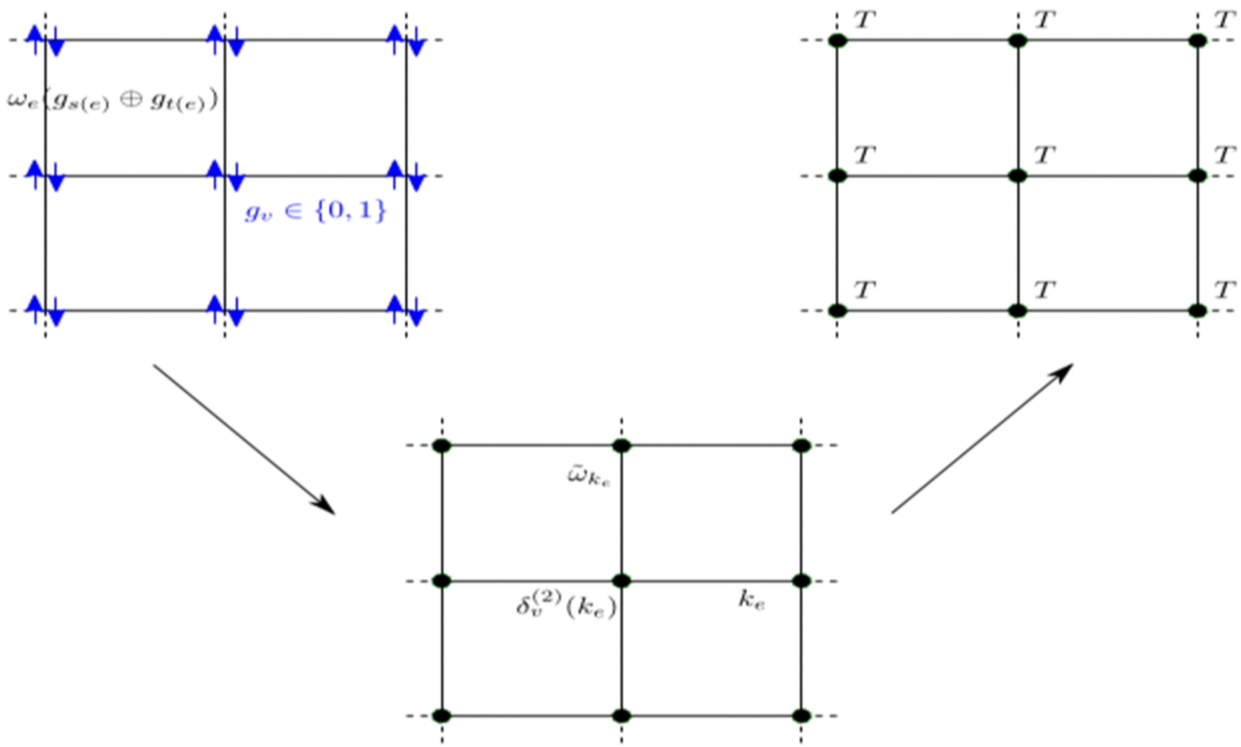
- β: **coupling constant**.
- The **partition function** is defined as:

$$Z = \sum_{\{g_v\}} \prod_e \omega_e(g_{s(e)} \oplus g_{t(e)})$$

Writing the Ising model as a tensor network

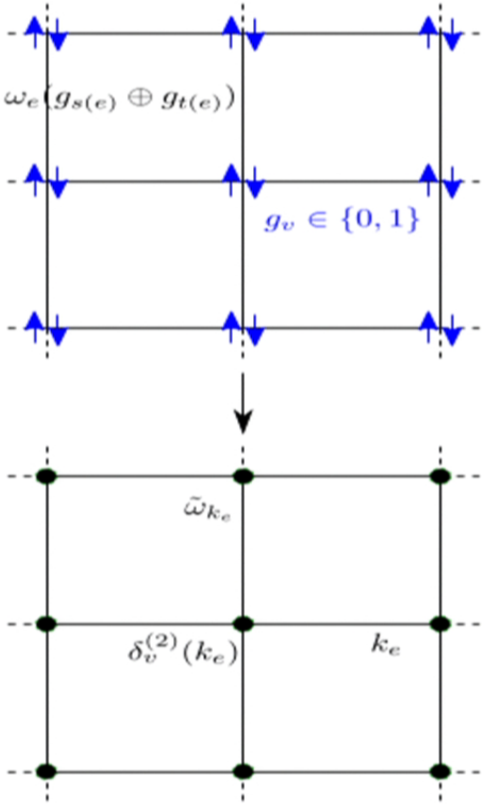
$Z = \sum_{\{g_v\}} \prod_e \omega_e(\{g_v\}_{v \subset e})$

$Z = \mathrm{Ttr}(T \ldots T)$



$Z = \sum_{\{k_e\}} \prod_e \tilde{\omega}_{k_e} \prod_v \delta^{(2)}(\sum_{e \supset v} k_e)$

The Fourier transformed Ising model [Savit '80]



- ω_e can be expanded in **characters** χ:

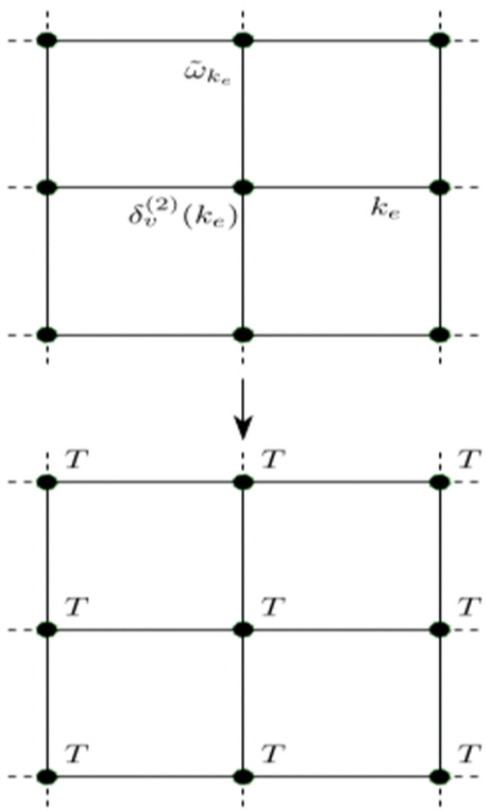
$$\omega(g) = \frac{1}{2}\sum_{k=0}^{1} \tilde{\omega}_k \chi_k(g), \quad \tilde{\omega}_k = \sum_{g=0}^{1} \omega(g)\overline{\chi_k(g)}\,,$$

  with $\chi_k(g) = \exp(i\pi(k \cdot g))$ and $\chi_k(g_1 \oplus g_2) = \chi_k(g_1)\chi_k(g_2)$.

- Rewrite the **partition function** as follows:

$$Z = \frac{1}{2^E}\sum_{\{k_e\}}\left(\prod_e \tilde{\omega}_k\right)\sum_{\{g_v\}}\prod_v\left(\prod_{e\supset v}\chi_{k_e}(g_v)\right)$$
$$= \sum_{\{k_e\}}\left(\prod_e \tilde{\omega}_k\right)\prod_v \delta^{(2)}\left(\sum_{e\supset v} k_e\right) \quad .$$

The Ising model as a tensor network

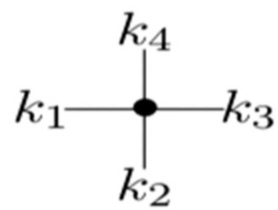


- The idea is to write the **partition function** as a **contraction of a tensor network**:

$$Z = \mathrm{Ttr}(T \dots T),$$

with

$$T_{k_1 k_2 k_3 k_4} =$$

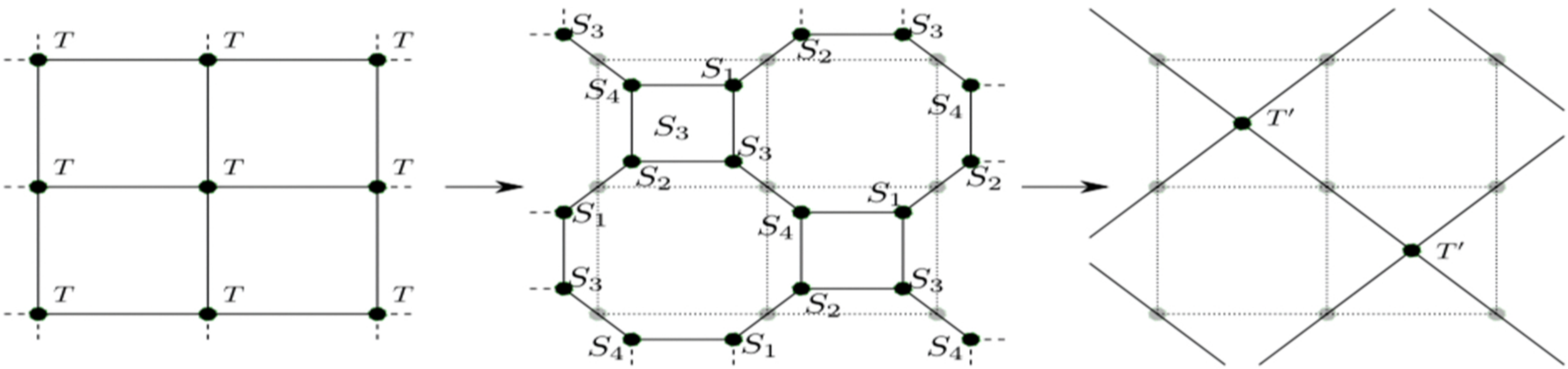


$$= \left(\prod_{i=1}^{4} \sqrt{\tilde{\omega}_{k_i}} \right) \delta \left(\sum_{i=1}^{4} k_i \right) .$$

- Code: 4-dim array $T(k_1, k_2, k_3, k_4)$, $2 \times 2 \times 2 \times 2$.

The algorithm – Overview [Levin, Nave '07, Gu, Wen '09]

- Three steps:
  - **'Reshape'** tensor into a **matrix**.
  - Perform a **singular value decomposition** (SVD) → truncate.
  - Sum over fine degrees of freedom.



Shape the tensor into a matrix

- Reshape the tensor into a matrix:

$$T_{(k_1k_2)(k_3k_4)} =: M^{(1)}_{\underbrace{(k_1k_2)}_{A}\underbrace{(k_3k_4)}_{B}} \quad , \quad T_{(k_1k_4)(k_2k_3)} =: M^{(2)}_{\underbrace{(k_1k_4)}_{A}\underbrace{(k_2k_3)}_{B}}$$

- Concretely, we get the following 4×4 matrix:

$$M^{(1)}_{AB} = \begin{array}{c} \\ (0,0) \\ (0,1) \\ (1,0) \\ (1,1) \end{array} \begin{array}{c} \begin{array}{cccc} (0,0) & (0,1) & (1,0) & (1,1) \end{array} \\ \begin{pmatrix} * & 0 & 0 & * \\ 0 & * & * & 0 \\ 0 & * & * & 0 \\ * & 0 & 0 & * \end{pmatrix} \end{array}$$

- Realize in code either by hand (for loops), but many languages offer easy options. (Mathematica "ArrayReshape" etc.)

Shape the tensor into a matrix

- Reshape the tensor into a matrix:

$$T_{(k_1k_2)(k_3k_4)} =: M^{(1)}_{\underbrace{(k_1k_2)}_{A}\underbrace{(k_3k_4)}_{B}} \quad , \quad T_{(k_1k_4)(k_2k_3)} =: M^{(2)}_{\underbrace{(k_1k_4)}_{A}\underbrace{(k_2k_3)}_{B}}$$

- Concretely, we get the following 4×4 matrix:

$$M^{(1)}_{AB} = \begin{array}{c} \\ (0,0) \\ (0,1) \\ (1,0) \\ (1,1) \end{array} \begin{array}{c} \begin{array}{cccc} (0,0) & (0,1) & (1,0) & (1,1) \end{array} \\ \begin{pmatrix} * & 0 & 0 & * \\ 0 & * & * & 0 \\ 0 & * & * & 0 \\ * & 0 & 0 & * \end{pmatrix} \end{array}$$

- Realize in code either by hand (for loops), but many languages offer easy options. (Mathematica "ArrayReshape" etc.)

Singular Value Decomposition

- Diagonalize $MM^\dagger = UDU^\dagger$ and $M^\dagger M = VDV^\dagger$.
  - Eigenvalues are all positive and real, U, V unitary.
- **Any matrix** can be decomposed by a **singular value decomposition**:

$$M_{AB}^{(1)} = \sum_{i=1}^{4} U_{A,i}^{(1)}\, \lambda_i\, (V_{B,i}^{(1)})^\dagger \approx \sum_{i=1}^{2} U_{A,i}^{(1)}\, \lambda_i\, (V_{B,i}^{(1)})^\dagger$$

  - λ is diagonal matrix of singular values, with $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \lambda_4$.

Thm.: **Reconstruction** from **truncated SVD** is **best approximation** of M by a matrix of rank 2.

- Most programming languages offer a package with a SVD algorithm. (If necessary add an additional library, e.g. in C++.)

Singular Value Decomposition

- Diagonalize $MM^\dagger = UDU^\dagger$ and $M^\dagger M = VDV^\dagger$.
  - Eigenvalues are all positive and real, U, V unitary.
- **Any matrix** can be decomposed by a **singular value decomposition**:

$$M_{AB}^{(1)} = \sum_{i=1}^{4} U_{A,i}^{(1)}\, \lambda_i\, (V_{B,i}^{(1)})^\dagger \approx \sum_{i=1}^{2} U_{A,i}^{(1)}\, \lambda_i\, (V_{B,i}^{(1)})^\dagger$$

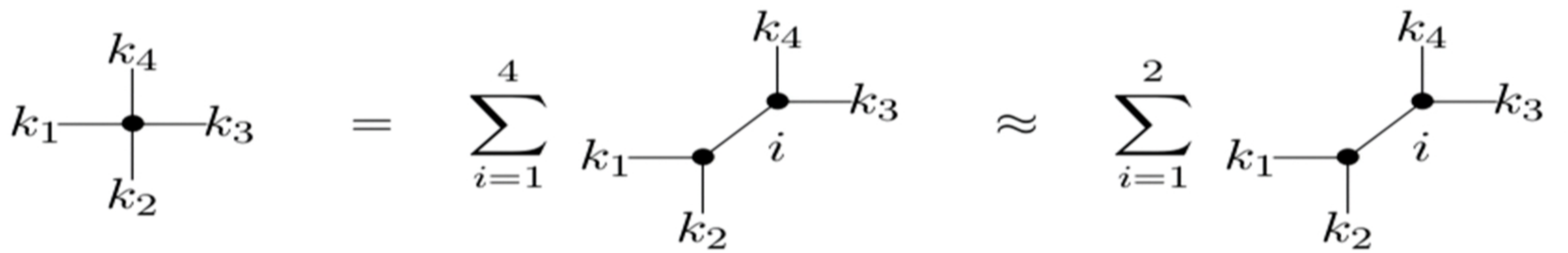
  - λ is diagonal matrix of singular values, with $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \lambda_4$.

Thm.: **Reconstruction** from **truncated SVD** is **best approximation** of M by a matrix of rank 2.

- Most programming languages offer a package with a SVD algorithm. (If necessary add an additional library, e.g. in C++.)

Splitting the tensor

- Given U, V and λ, we **split and truncate** the tensor T:



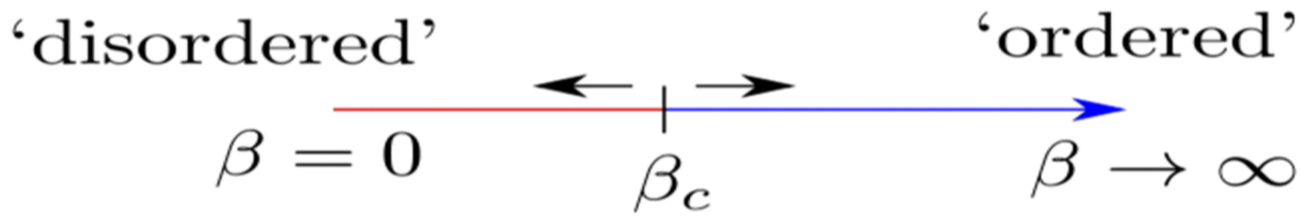
- After the SVD of both matrices we define:

$$S_{A,i}^{1,3} = U_A^{(1,2)} \sqrt{\lambda_i^{(1,2)}}\,, \quad S_{B,i}^{2,4} = V_B^{(1,2)} \sqrt{\lambda_i^{(1,2)}}.$$

We interpret labels i as **new coarse d.o.f. / variables**. As a last step, we sum over the fine variables $\{k_i\}$.

Results - Phase transition

- Despite the crude approximation (just 2 singular values), we observe the **phase transition** (at wrong β_c) of the Ising model.

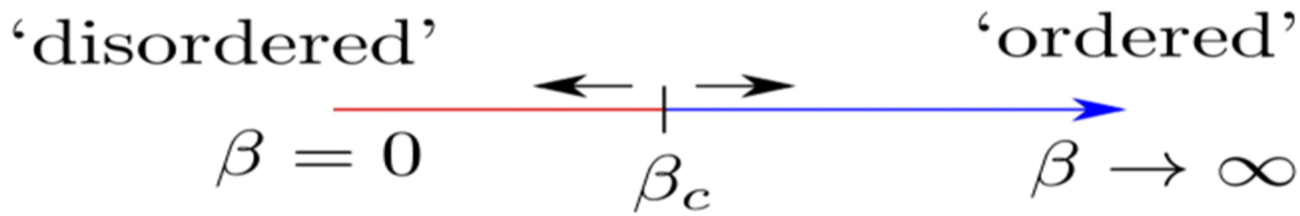


- **'disordered'**: 1 non-vanishing singular value.
  - No correlation between Ising spins.
- **'ordered'**: 2 non-vanishing singular values (equal size).
  - All Ising spins parallel, 2 possible states.
- Close to β_c: (almost) **scale invariant** (Indicates 2nd order phase transition).

What is the **meaning** of these singular values?
To which d.o.f. do they correspond?

Results - Phase transition

- Despite the crude approximation (just 2 singular values), we observe the **phase transition** (at wrong β_c) of the Ising model.



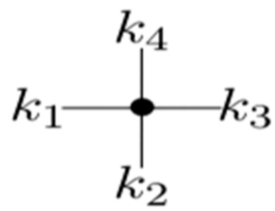
- **'disordered'**: 1 non-vanishing singular value.
  - No correlation between Ising spins.
- **'ordered'**: 2 non-vanishing singular values (equal size).
  - All Ising spins parallel, 2 possible states.
- Close to β_c: (almost) **scale invariant** (Indicates 2nd order phase transition).

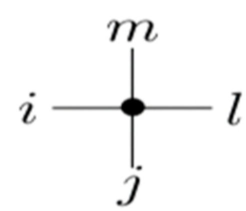
What is the **meaning** of these singular values?
To which d.o.f. do they correspond?

Outlook

1. Motivation
2. Tensor network renormalization - Ising model
   - Definition as a tensor network
   - The algorithm – general scheme
3. Improving the algorithm I - Symmetries
4. Improving the algorithm II - Triangular
5. Going beyond the Ising model - spin nets
6. Summary

T looked different before....

$T_{k_1 k_2 k_3 k_4} =$ 

$\rightarrow$ 

$= T'_{ijlm}$

- Compare $T_{k_1 k_2 k_3 k_4}$ and T'_{ijlm}:
  - Indices k_i are $\mathbb{Z}_2$ representations, i label SVs.
  - Indeed, U, V are **variable redefinitions**: $(k_1, k_2) \rightarrow i$.
- What is the **interpretation** of i after 17 redefinitions?
  - Would have to keep track of all 17 U, V!
- What happened to the δ function on the vertices?
  - Hidden in the meaning of the d.o.f. labelled by i!

Can we **preserve the symmetries** under coarse graining and use this to also use **less resources** (computational time and memory)?

Block diagonal form

- Reconsider the $M^{(1)}_{AB}$:

$$M^{(1)}_{AB} = \begin{array}{c} \\ (0,0) \\ (0,1) \\ (1,0) \\ (1,1) \end{array} \begin{array}{c} \begin{array}{cccc} (0,0) & (0,1) & (1,0) & (1,1) \end{array} \\ \begin{pmatrix} * & 0 & 0 & * \\ 0 & * & * & 0 \\ 0 & * & * & 0 \\ * & 0 & 0 & * \end{pmatrix} \end{array}$$

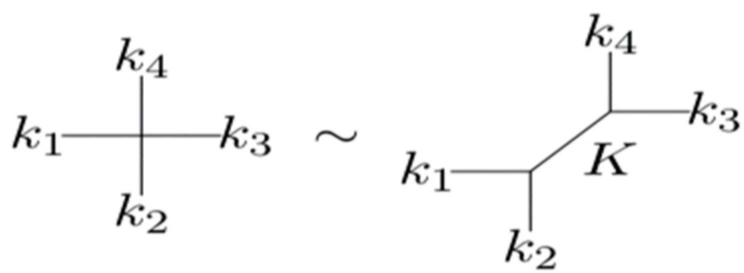
- If the entries are reorganized:

$$M^{(1)}_{AB} = \begin{array}{c} \\ (0,0) \\ (1,1) \\ (0,1) \\ (1,0) \end{array} \begin{array}{c} \begin{array}{cccc} (0,0) & (1,1) & (0,1) & (1,0) \end{array} \\ \begin{pmatrix} * & * & 0 & 0 \\ * & * & 0 & 0 \\ 0 & 0 & * & * \\ 0 & 0 & * & * \end{pmatrix} \end{array}$$

Intertwiner channels

- Exploiting the δ function:

$$\delta^{(2)}(k_1+k_2+k_3+k_4)=\sum_{K=0}^{1}\delta^{(2)}(k_1+k_2+K)\delta^{(2)}(k_3+k_4+K)$$

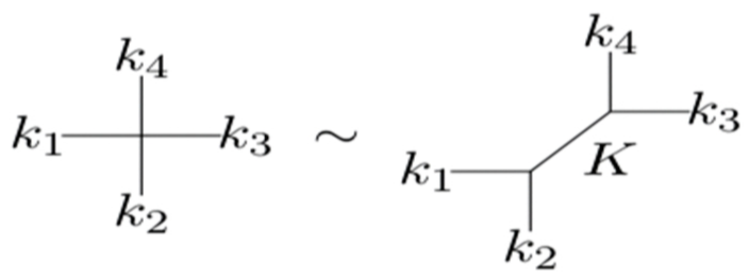


- Turn $T_{k_1k_2k_3k_4}$ into $T^K_{k_1k_2k_3k_4}$.
  - Only compute components $\{k_i\}$ **compatible** with K!
  - 2 2×2 matrices $M^{K,(1,2)}_{AB}$.
  - **One SVD per matrix.**
  - **One singular value per block.**

Intertwiner channels

- Exploiting the δ function:

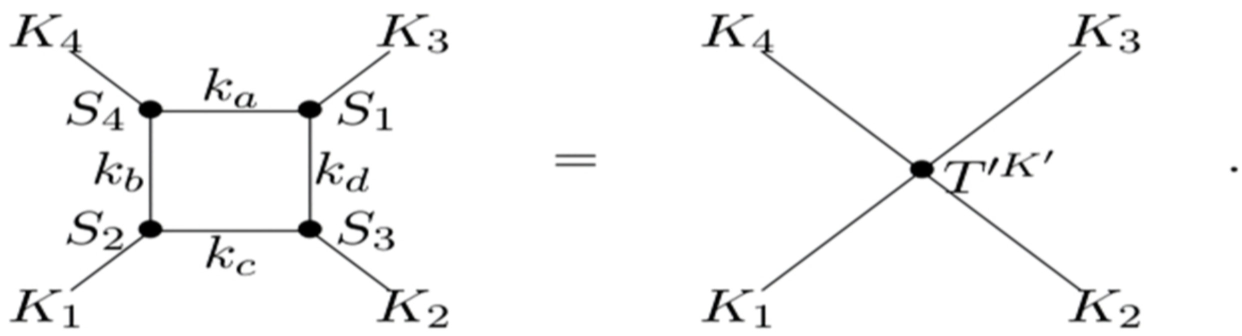
$$\delta^{(2)}(k_1 + k_2 + k_3 + k_4) = \sum_{K=0}^{1} \delta^{(2)}(k_1 + k_2 + K)\delta^{(2)}(k_3 + k_4 + K)$$



- Turn $T_{k_1 k_2 k_3 k_4}$ into $T^{K}_{k_1 k_2 k_3 k_4}$.
  - Only compute components $\{k_i\}$ **compatible** with K!
  - 2 2×2 matrices $M^{K,(1,2)}_{AB}$.
  - **One SVD per matrix.**
  - **One singular value per block.**

Symmetries are preserved

- The tensors $S^{m,K}_{k_a k_b,i} \sim \delta^{(2)}(k_a + k_b + K)$.
  - Sum only over k_a, k_b compatible with K.
- The new tensor has an **interpretation of the old variables**:

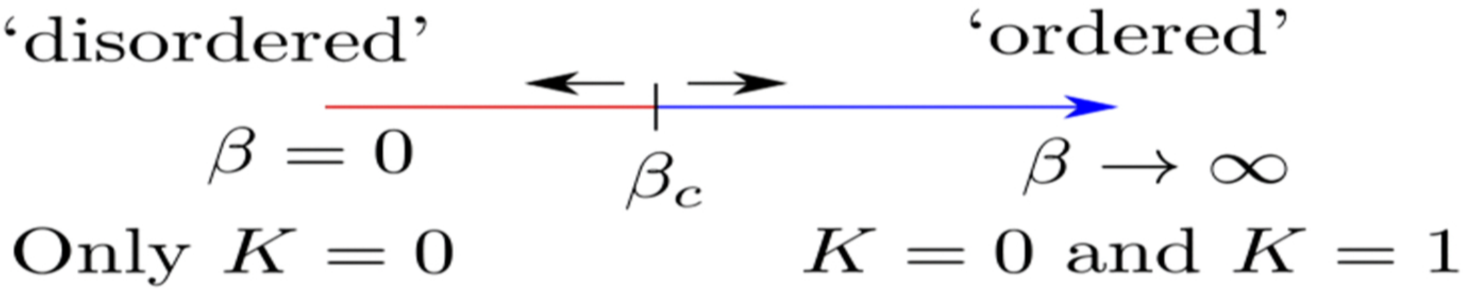


- $T'_{K_1K_2K_3K_4} \sim \delta^{(2)}(K_1 + K_2 + K_3 + K_4)$:
  - Save only **blockdiagonal** form $T'^{K'}$.

T'^{K} is of same size (and similar form) as the initial T^K.

Results - Phase transition II

- Despite the crude approximation (**one singular value per block** K), we observe the phase transition (at wrong β_c) of the Ising model.

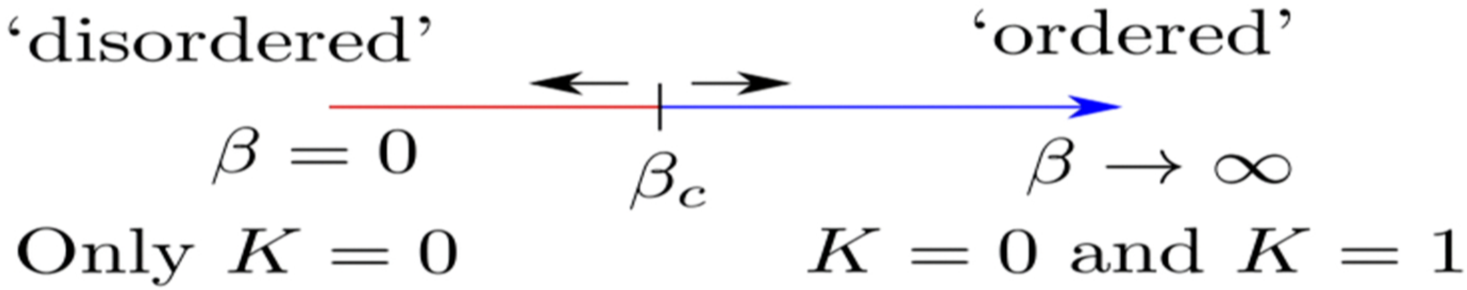


- **'disordered'**: Only representations $K = 0$ allowed
  - Matches **initial model** for $\beta = 0$.
- **'ordered'**: $K = 0$ and $K = 1$ allowed with equal weights.
  - Matches **initial model** for $\beta \to \infty$.

Due to **explicit symmetry preservation**, we obtain an **interpretation** of the phases from within the model.

Results - Phase transition II

- Despite the crude approximation (**one singular value per block** K), we observe the phase transition (at wrong β_c) of the Ising model.

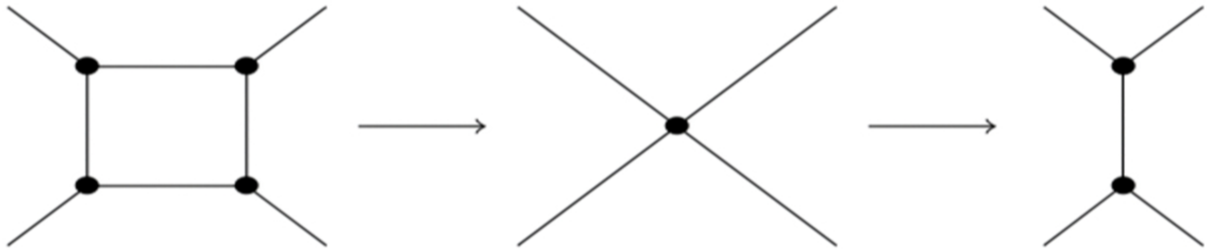


- **'disordered'**: Only representations $K = 0$ allowed
  - Matches **initial model** for $\beta = 0$.
- **'ordered'**: $K = 0$ and $K = 1$ allowed with equal weights.
  - Matches **initial model** for $\beta \to \infty$.

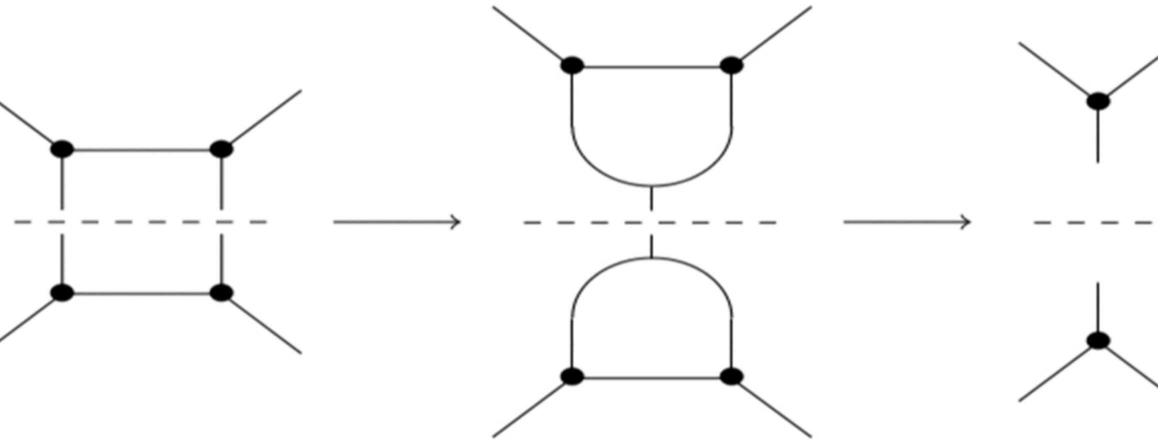
Due to **explicit symmetry preservation**, we obtain an **interpretation** of the phases from within the model.

Can we optimize this?

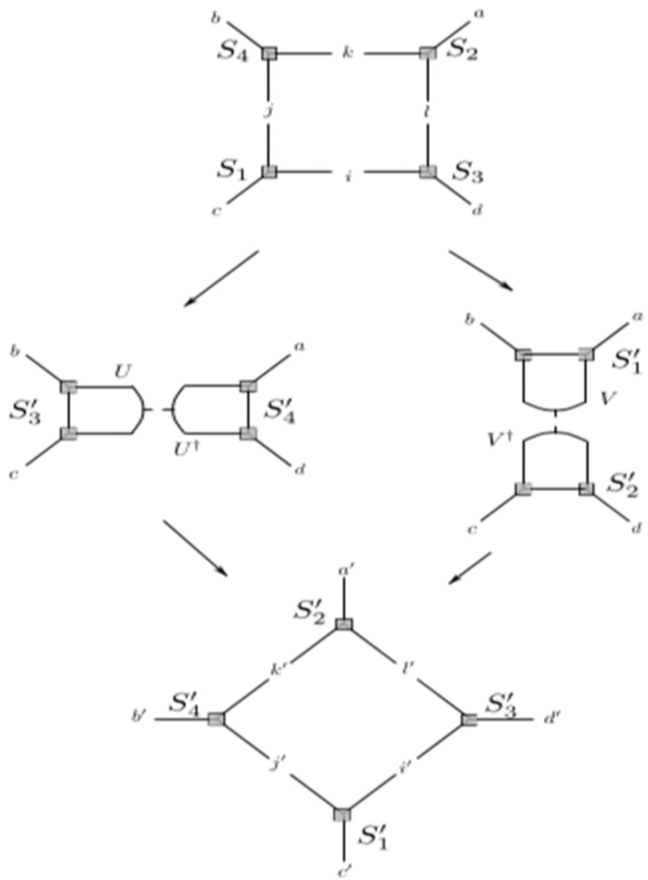
- We glue 3-valent tensors only to split them again:



- Why not construct new 3-valent tensors?



The triangular algorithm



- From 4-valent T to 3-valent S.
  - **Less memory** required to store 3-valent tensors.
- Compute one 4-valent tensor from two 3-valent ones.
- Perform SVD between 'fine' and 'coarser' d.o.f.
- Symmetry preserving!
  - To compute **one** block of new S, just compute **one** block of intermediate T.

Key ideas for memory reduction

- Save smaller building blocks.
- Exploit symmetries to only compute what is necessary in this particular step!

Summary - Improving performance and interpretation

Tensor network renormalization

- Coarse graining algorithm with a **controlled** truncation scheme.
- SVD determines new **effective** d.o.f. and their **relevance**.

↓

Explicit symmetry preservation

- Use symmetry to only store, compute and sum over non-zero parts.
- Keep **interpretation** in terms of **original variables**.

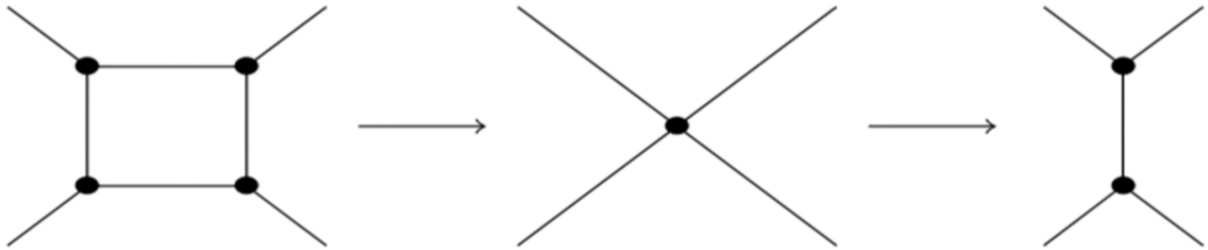
↓

Triangular algorithm

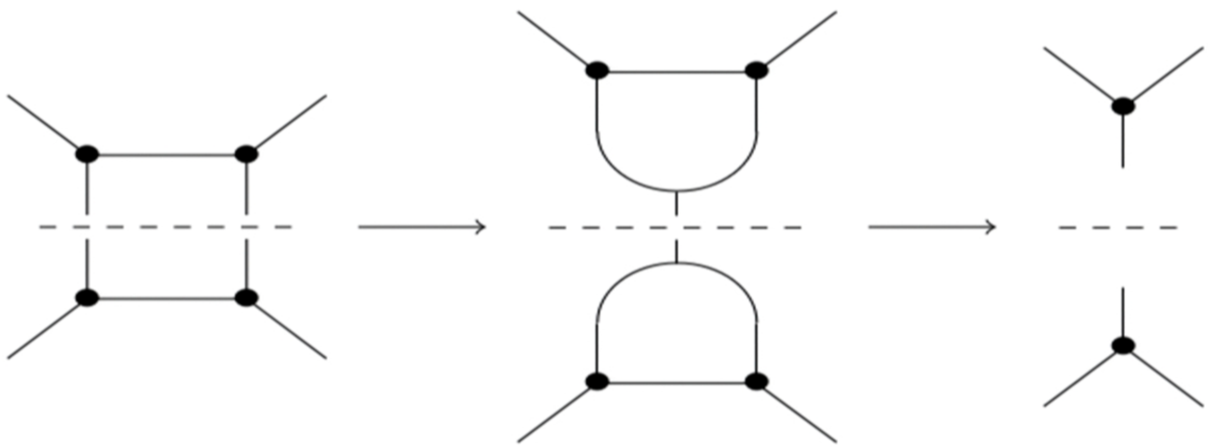
- Work with **smaller** building blocks.
- Compute only necessary tensors for current calculation.

Can we optimize this?

- We glue 3-valent tensors only to split them again:



- Why not construct new 3-valent tensors?



Summary - Improving performance and interpretation

Tensor network renormalization

- Coarse graining algorithm with a **controlled** truncation scheme.
- SVD determines new **effective** d.o.f. and their **relevance**.

↓

Explicit symmetry preservation

- Use symmetry to only store, compute and sum over non-zero parts.
- Keep **interpretation** in terms of **original variables**.

↓

Triangular algorithm

- Work with **smaller** building blocks.
- Compute only necessary tensors for current calculation.

Outlook

1. Motivation
2. Tensor network renormalization - Ising model
   - Definition as a tensor network
   - The algorithm – general scheme
3. Improving the algorithm I - Symmetries
4. Improving the algorithm II - Triangular
5. Going beyond the Ising model - spin nets
6. Summary

A short history of spin net models

- Spin net models are statistical models **related to spin foam models**:
  - Ising model in 2D is related to $\mathbb{Z}_2$ gauge theory in 4D.
- **Symmetry preserving 4-valent algorithm**:
  - Abelian finite groups $\mathbb{Z}_q$ [Dittrich, Eckert, Martin-Benito '11; Dittrich, Eckert, '11]
  - Non-Abelian finite group S_3 [Dittrich, Martin-Benito, Schnetter '13]
  - Quantum group $SU(2)_k$ [Dittrich, Martin-Benito, S.St. '13]
- **Triangular algorithm (symmetry preserving)**:
  - Analogue Barrett-Crane model for $SU(2)_k \times SU(2)_k$ [Dittrich, Girelli, Schnetter, Seth, S.St. w.i.p.]
  - Ising model coupled to dynamical $SU(2)_k$ background [S.St. '15]

Actually all optimizations described in this talk are **necessary** to allow us to coarse grain $SU(2)_k \times SU(2)_k$ spin nets!

Dictionary - Ising model and $SU(2)_k \times SU(2)_k$ spin nets

	Ising	$SU(2)_k \times SU(2)_k$
initial tensor T	$\sim \delta^{(2)}(\sum_{e \supset v} k_e)$	$\sim \mathcal{P}^{\{m_e^\pm\}_{e \supset v}}_{\{n_e^\pm\}_{e \supset v}}(\{j_e^\pm\}_{e \supset v})$
'Size' of the tensor	2^4	$\prod_{i=1}^8 \sum_{j_i=0}^{j_{\max}} (2j_i^\pm + 1)$
Symmetry preserving	T^K	$T^{(j_5^+, j'^+_5, j_5^-, j'^-_5)}(\{j_e^\pm\})$
No. of blocks	2	$(j_{\max} + 1)^4$ 256 for $k = 6, j_{\max} = 3$ 625 for $k = 8, j_{\max} = 4$
Size of largest matrix	2×2	$8^4 \times 8^4 \sim 0.25$ GB for $k = 6$ $13^4 \times 13^4 \sim 12$ GB for $k = 8$

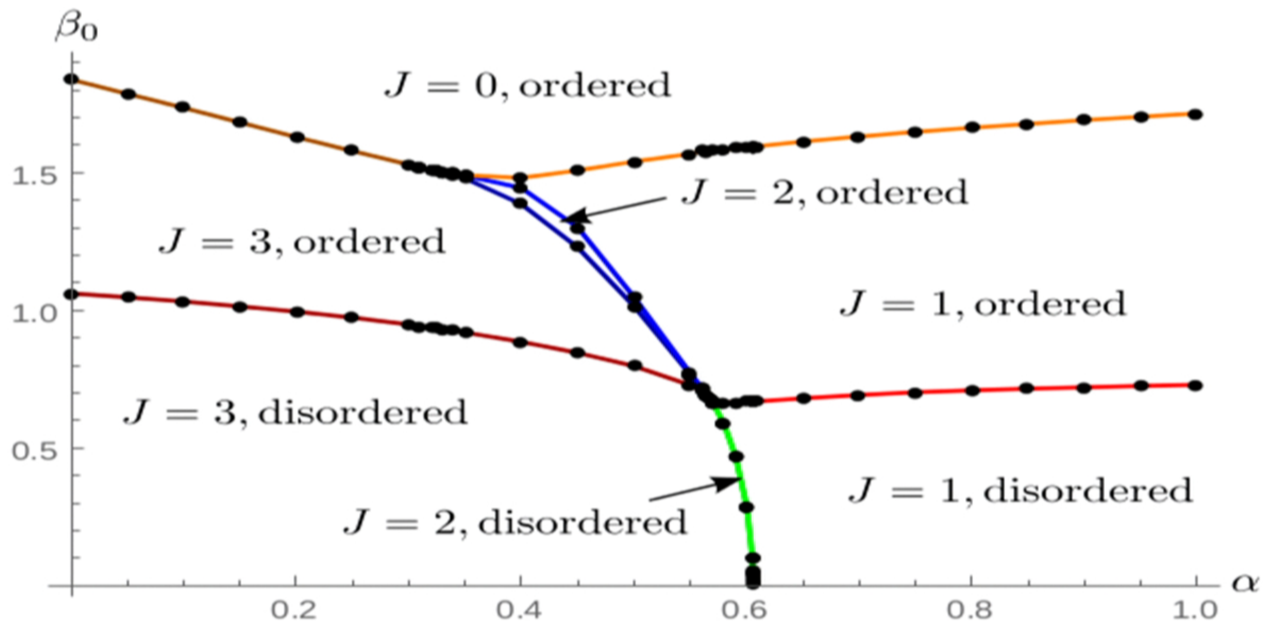
- 'Super-index': $(j_1, j_2) \to j$, only save allowed couplings.
- 'Super-index' for $\{6j\}$: Only compute, save and sum over non-vanishing $6j$-symbols.

Summary

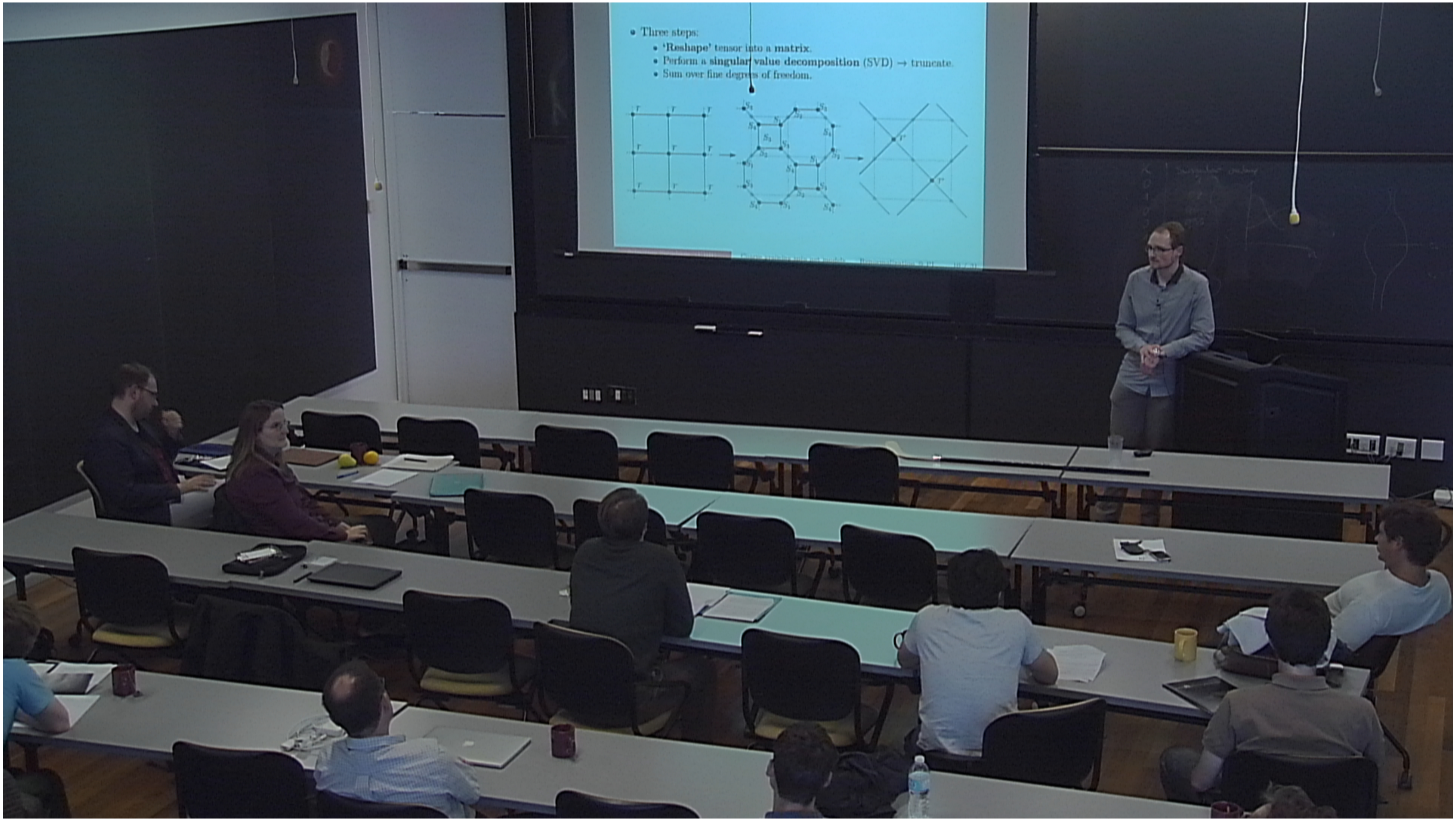
- In depth presentation of **tensor network renormalization** for the Ising model.
- Many **analytical improvements** to make this algorithm feasible, both in terms of interpretation and computational resources.
  - Explicitly keeping track of **symmetries** allows to express tensor in originial variables.
  - Also **smaller** tensors, matrices and thus **less computational cost**.
  - **Triangular algorithm** to reduce memory usage **significantly**.

Take home messages

- Numerical methods are a promising tool to advance quantum gravity!
- Analytical and numerical techniques **supplement** each other!
- We have to employ both to develop **new tools** for studying quantum gravity!



Thank you for your attention!


Three steps:
'Reshape' tensor into a matrix.
Perform a singular value decomposition (SVD) → truncate.
Sum over fine degrees of freedom.