Advancing the optimization of iPEPS: Direct energy minimization with automatic differentiation

Juraj Hasik

(13.2.2020)

Outline

1) Introduction

- Problems of interest – (spin) lattice models
- Tensor networks – motivation & main concepts of iPEPS

2) Solving 2D spin systems with iPEPS

- Observables – Corner transfer matrix
- Optimization I – imaginary time evolution
- Case study: Coupled ladders
- Optimization II – gradients & algorithmic differentiation
- Application to J1-J2 model

3) Conclusions & Future directions

Intro: Spin Models

Many-body electron problem **is reduced** to interacting magnetic moments (**spins**) …

$$H = \sum_{\langle i,j \rangle} J_{1,ij} S_i . S_j + \sum_{\langle\langle i,j \rangle\rangle} J_{2,ij} S_i . S_j + \dots$$
$$+ \sum_{\langle i,j,k,l \rangle} Q_{ijkl} (S_i . S_j)(S_k . S_l) + \dots$$

… usually arranged on a regular **lattice**

Rich physics – Landau symm. breaking theory, deconfined critical point, topological order, …

Intro: Tensor Networks

Variational states targeting GS lattice models

F. Verstraete and J. I. Cirac, arXiv:cond-mat/0407066, (2004)

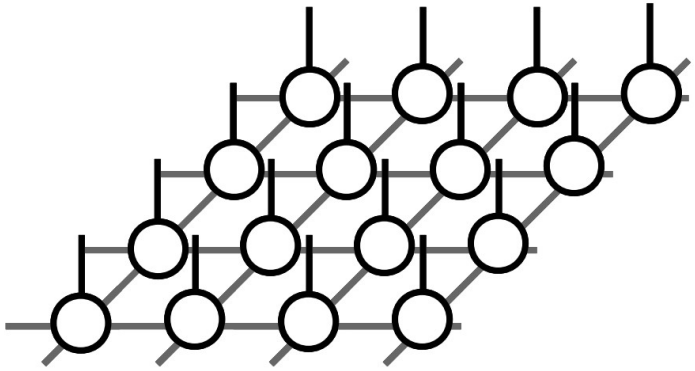
- **area law** by construction
- no sign problem
- **no FS effects**

- can break or impose **lattice symmetries** and/or **internal symmetries**

$$|\psi\rangle = \sum_{s_1 s_2 \ldots} c_{s_1 s_2 \ldots} |s_1 s_2 \ldots\rangle$$

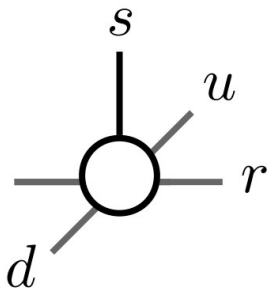
#parameters: 2^#spins

iPEPS



$$\mathrm{Tr}_{aux}(a^{s_1} a^{s_2} \ldots)$$

$$a^s_{ulrd} :=$$

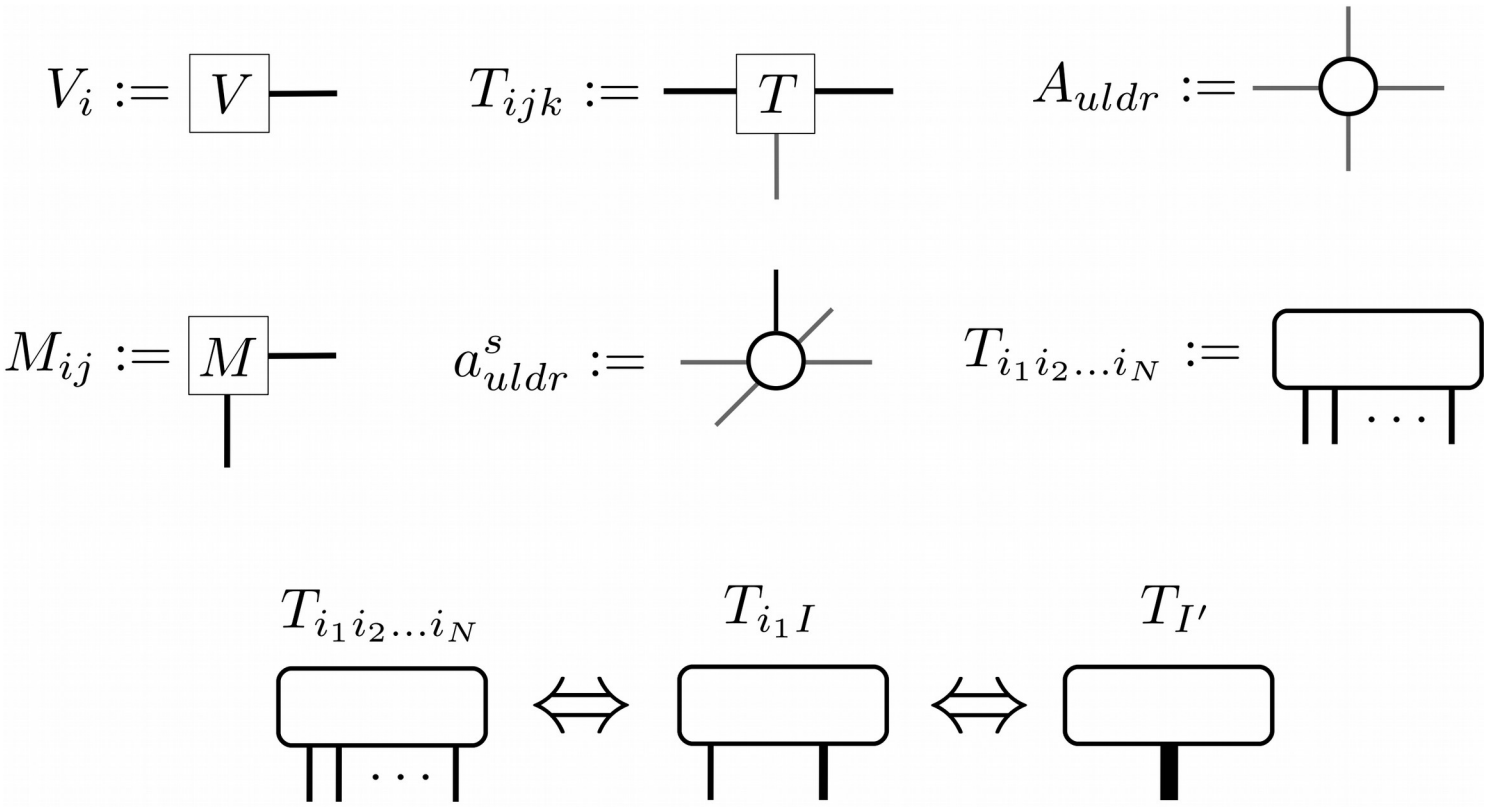


dimension(*u,l,d,r*) = D

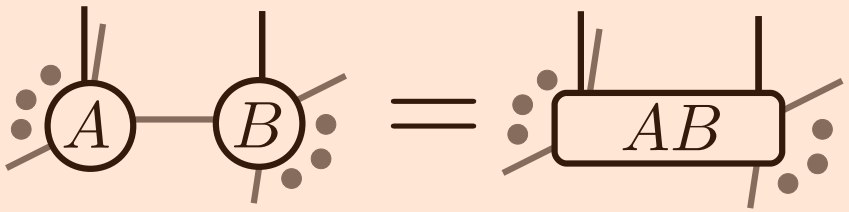
$$|\psi\rangle = \sum_{s_1 s_2 \ldots} Tr_{aux}(a^{s_1} a^{s_2} \ldots)|s_1 s_2 \ldots\rangle$$

#parameters:
2D^4 per tensor

Intro: Diagrammatics



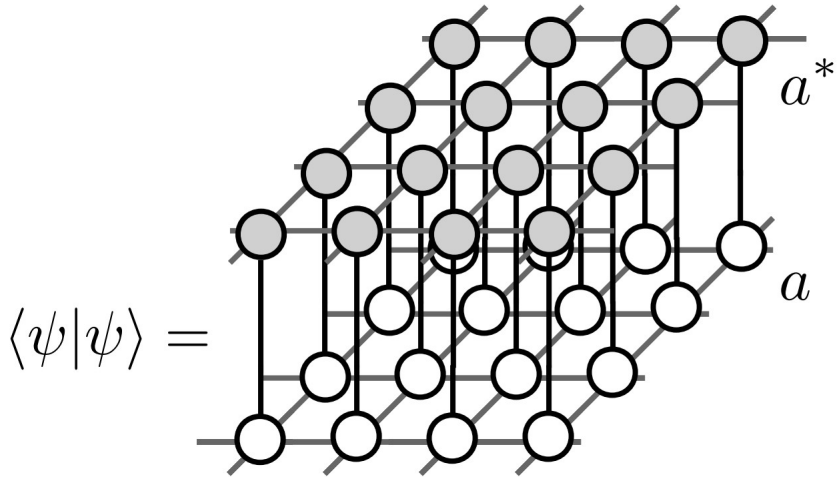
Basic operation: **Tensor contraction**

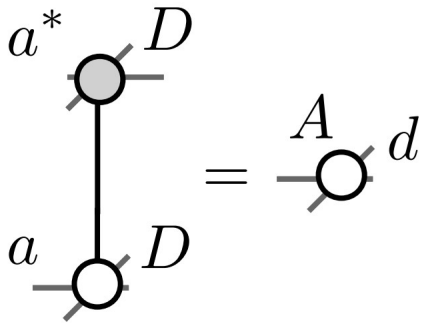


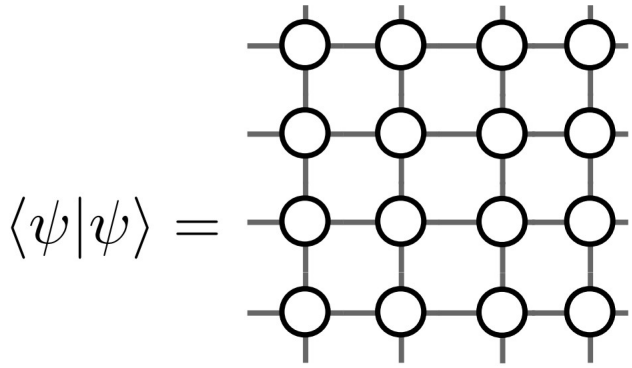
$$\Leftrightarrow \sum_{i_1 i_1'} A^{s}_{i_1 i_2 \ldots} \delta_{i_1 i_1'} B^{s'}{}_{i_1' i_2' \ldots} = (AB)^{s s'}_{i_2 \ldots i_2' \ldots}.$$

Observables of iPEPS

Expectation values must be **approximated**







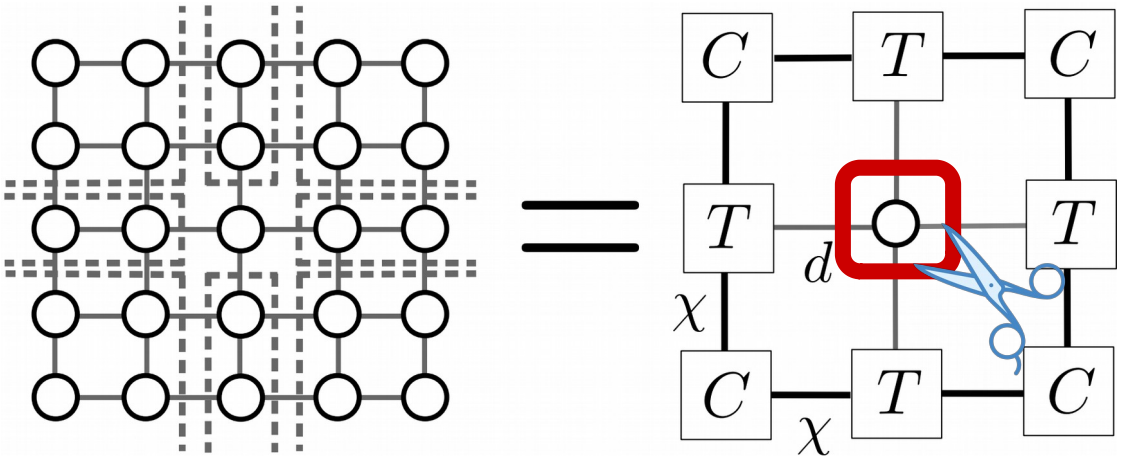
Construct **environment**

- corners **C**
- half-row/-columns **T**

Baxter, J. Stat. Phys. 17, 1 (1977)

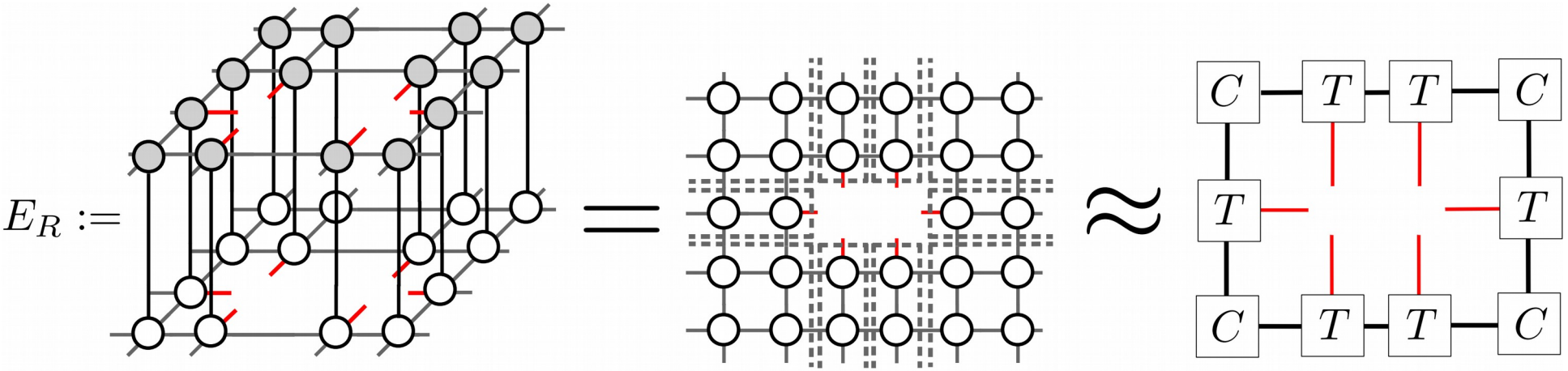
New control parameter:

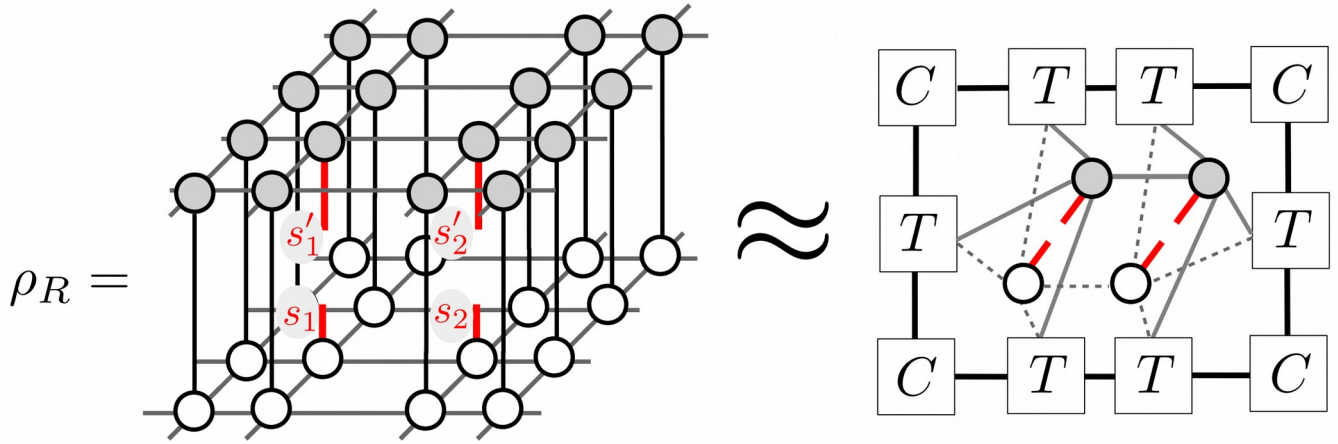
env. dimension χ



Observables of iPEPS

From **reduced environments** (E_R) of region R build **reduced density matrices** (ρ_R) of region R



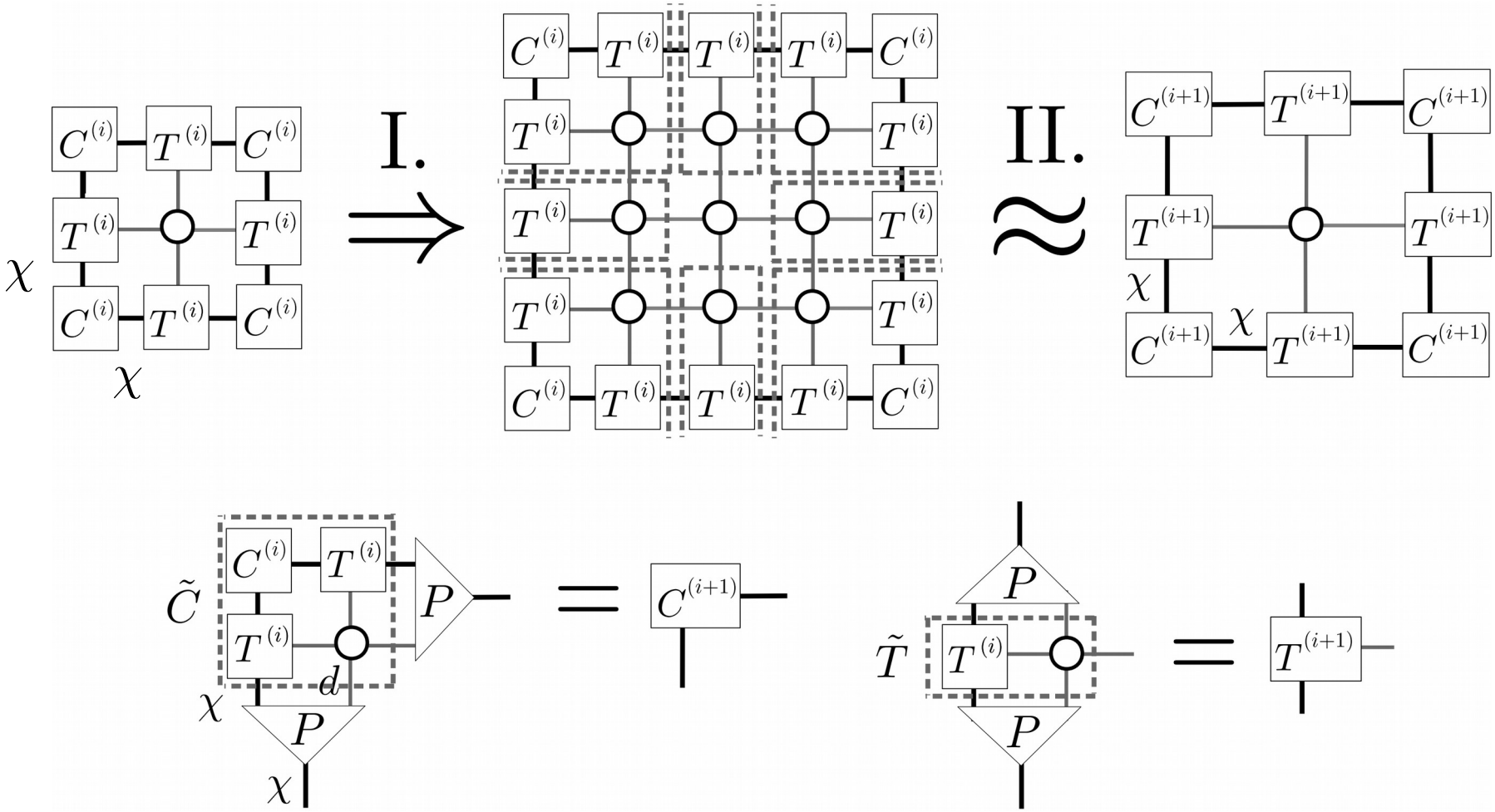


Any observable inside the region R is: $\langle \mathcal{O} \rangle_\chi \approx \mathrm{Tr}(\rho_R(\chi)\mathcal{O})$

Corner Transfer Matrix

Corner transfer matrix renormalization group (**CTMRG**)

Complexity **$O(\chi^3 D^6)$** T. Nishino and K. Okunishi, JPSJ 65, 891 (1996), R. Orús and G. Vidal, Phys. Rev. B 78, 155117 (2008)
Corboz et al., Phys. Rev. Lett. 113, 046402, (2014)



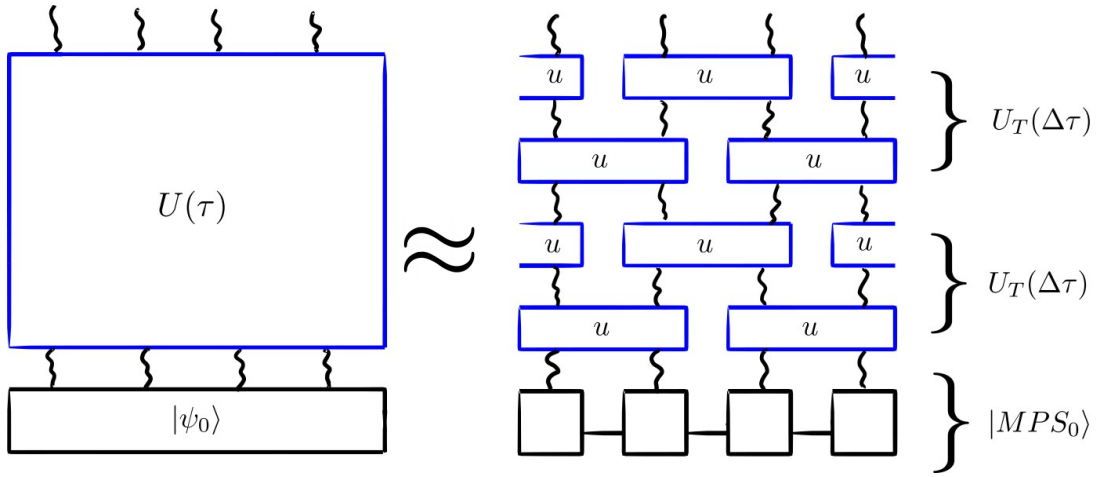
... iterate until **fixed point** C,T

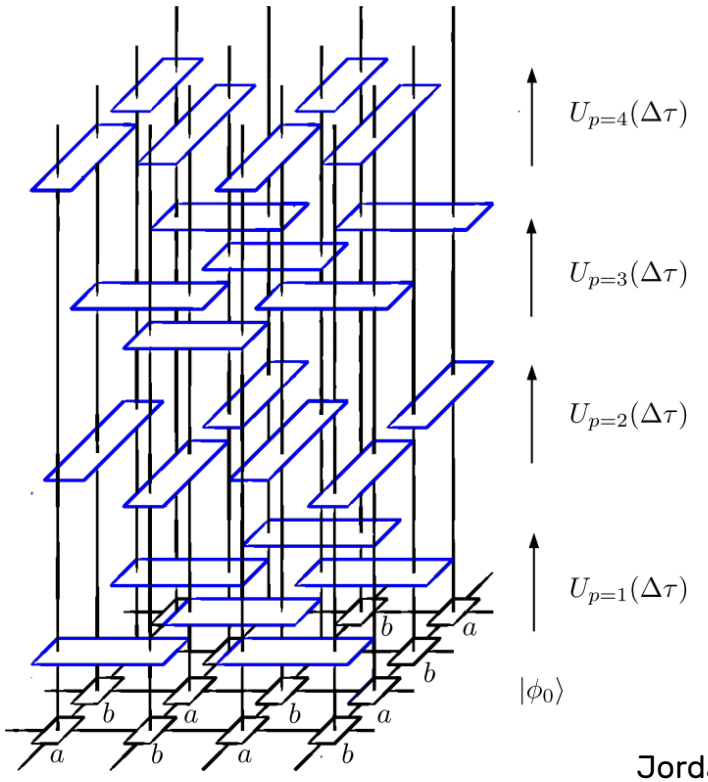
Optimization I

Warning: **Optimization is hard !**

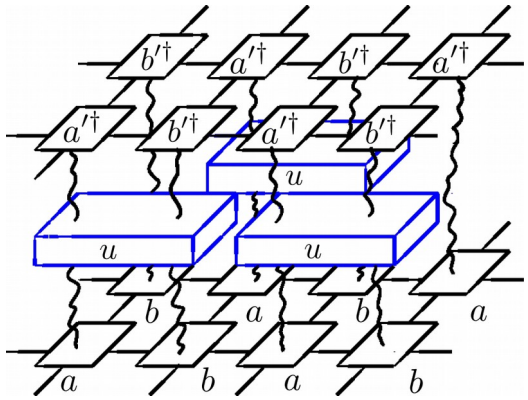
(I) Find the **fixed point** of **imag. time evolution** ...

$$U(\tau) = exp(-\tau H)$$





... use **Trotter decomposition**



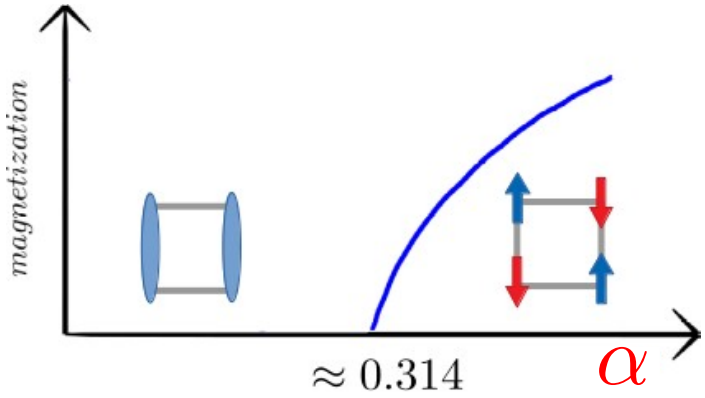
Simple and **Full Update** contract layer by layer

Jordan et al., Phys. Rev. Lett. 101, 250602, (2008); Phien et al., Phys. Rev. B 92, 035142 (2015)

Coupled Spin-1/2 Ladders

Non-frustrated model featuring transition from **Néel phase** to **paramagnet**

$$H = J \sum_{r} \vec{S}_r . \vec{S}_{r+\hat{x}} + J \sum_{r|r_y \in even} \vec{S}_r . \vec{S}_{r+\hat{y}} + \alpha \sum_{r|r_y \in odd} \vec{S}_r . \vec{S}_{r+\hat{y}}$$

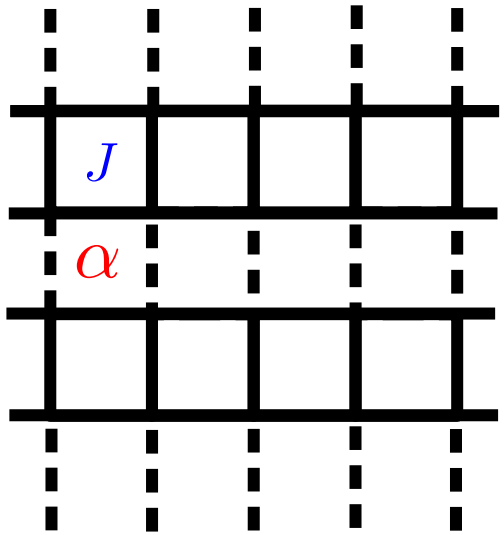


For small inter-ladder coupling α

- GS is a **gapped** and **paramagnetic**
- **"VBS"** - Singlets form along rungs of the ladder

Continuous phase transition at $\alpha_c \approx 0.314$

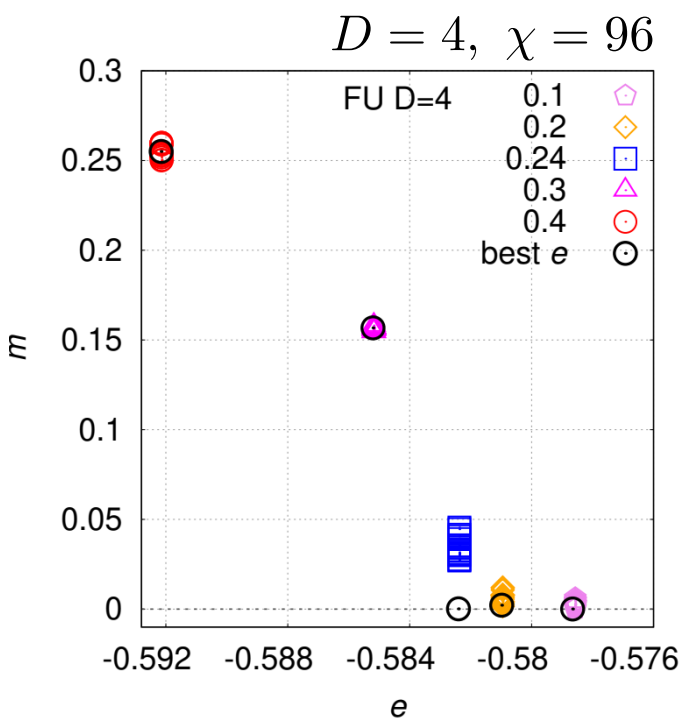
QMC: M. Matsumoto et. al, PRB 65 (2001); L. Capriotti, F. Becca, PRB 65 (2002)

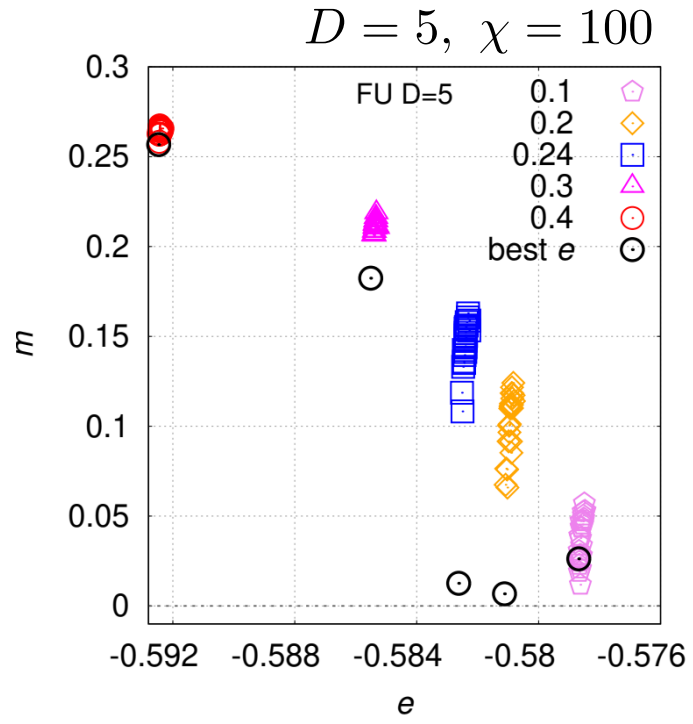


Realized in $C_9H_{18}N_2CuBr_4$ Hong et al., Nat. Phys. 13, 2017;

Coupled Ladders: Full Update

- Initialize by a set of 24 VBS states with noise
- Fast-full update (FFU), use adaptive timestep





Large α (Néel):

JH and F. Becca, Phys. Rev. B 100, 054429 (2019)

- FU shows **narrow distributions of minima** (in *e* and *m*)
- slight quantitative difference between **D=4** and **D=5**

Small α (paramagnet):

- **striking difference** between FU for **D=4** and **D=5**
- at **D=5** minima are **broadly distributed in magnetization** !

Optimizing II

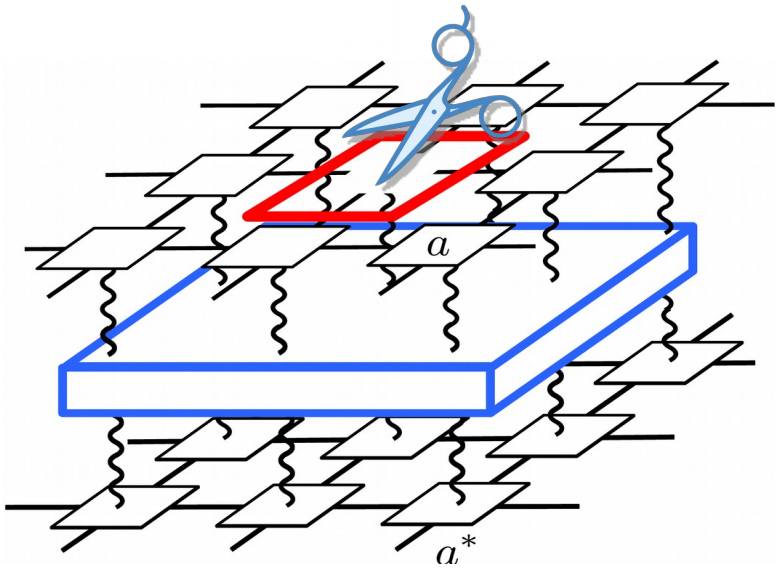
(II) Direct energy minimization

$$min\langle\psi|H|\psi\rangle$$

1. get **gradient**

$$\partial_a\langle\psi|H|\psi\rangle$$

2. steepest descent, CG, L-BFGS, ...



Caveat: How to evaluate the gradient for iPEPS ?

- Finite-Difference: simple, but only for few parameters
  D. Poilblanc and M. Mambrini, Phys. Rev. B 96, 014414 (2017)
- Summation schemes: harder with increasing range of *H*-terms
  P. Corboz, Phys. Rev. B 94, 035133 (2016); Vanderstraeten et al., Phys. Rev. B 94, 155123 (2016)
- **Algorithmic differentiation** (AD): ???
  Liao et al., Phys. Rev. X 9, 031041 (2019)

Primer: Algorithmic differentiation

Central question:

How to evaluate the gradient of a complicated scalar function of many variables ?

Simple model of a variational energy:

$$E : \mathbb{R}^N \xrightarrow{F^1} \mathbb{R}^{M_2} \xrightarrow{F^2} \mathbb{R}^{M_3} \xrightarrow{F^3} \mathbb{R}^{M_4} \xrightarrow{F^4} \mathbb{R}$$

$$F^4(F^3(F^2(F^1(\mathbf{x})))) = F^4(F^3(F^2(\mathbf{v}^2))) = F^4(F^3(\mathbf{v}^3)) = F^4(\mathbf{v}^4) =: E$$

Option 1: **Finite difference**

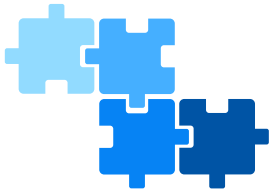
pick a direction $\mathbf{e}_i$ in the space of parameters and a small h

$$(\mathbf{g}_0)_i \approx \frac{E(\mathbf{x}_0 + h\mathbf{e}_i) - E(\mathbf{x}_0)}{h},$$

- finite precision error, **complexity O(N) x O(E)**

Primer: Algorithmic differentiation

Core premise of Algorithmic differentiation:



Functions are ultimately composed of (many) simple operations as +, -, /, *, exp, log, sin, ...

Assume that **Jacobians** are known: $J^n(\mathbf{v}_0^n) = \left(\frac{\partial F^n}{\partial \mathbf{v}^n}\right)\Big|_{\mathbf{v}^n=\mathbf{v}_0^n}$.

The **forward mode** AD

$$
\begin{aligned}
&\mathbf{x}_0 \equiv \mathbf{v}_0^1 \quad \rightarrow \mathbf{v}_0^2 = F^1(\mathbf{v}_0^1) \quad \rightarrow \mathbf{v}_0^3 = F^2(\mathbf{v}_0^2) \quad \rightarrow \mathbf{v}_0^4 = F^3(\mathbf{v}_0^3) \\
&\qquad\qquad \rightarrow E = F^4(\mathbf{v}_0^4), \\
&\mathbf{e}_i \equiv \mathbf{g}_{0,i}^1 \quad \rightarrow \mathbf{g}_{0,i}^2 = J^1(\mathbf{v}_0^1)\cdot \mathbf{g}_{0,i}^1 \quad \rightarrow \mathbf{g}_{0,i}^3 = J^2(\mathbf{v}_0^2)\cdot \mathbf{g}_{0,i}^2 \quad \rightarrow \mathbf{g}_{0,i}^4 = J^3(\mathbf{v}_0^3)\cdot \mathbf{g}_{0,i}^3 \\
&\qquad\qquad \rightarrow (\mathbf{g}_0)_i = J^4(\mathbf{v}_0^4)\cdot \mathbf{g}_{0,i}^4.
\end{aligned}
$$

In short: $(\mathbf{g}_0)_i = J^4(\mathbf{v}_0^4)\cdot(J^3(\mathbf{v}_0^3)\cdot(J^2(\mathbf{v}_0^2)\cdot(J^1(\mathbf{x}_0)\cdot\mathbf{e}_i)))$ Cost: **O(N) x O(E)**

Primer: Algorithmic differentiation

The **reverse mode** AD

I. Evaluate E($\mathbf{x}_0$) and **store all the intermediate variables**

$$\mathbf{x}_0 \equiv \mathbf{v}_0^1 \rightarrow \ \mathbf{v}_0^2 = F^1(\mathbf{v}_0^1) \rightarrow \ \mathbf{v}_0^3 = F^2(\mathbf{v}_0^2) \rightarrow \ \mathbf{v}_0^4 = F^3(\mathbf{v}_0^3) \rightarrow \ E = F^4(\mathbf{v}_0^4)$$

II. Accumulate the gradient in the reverse order

$$1 \cdot J^4(\mathbf{v}_0^4) = \bar{\mathbf{v}}_0^4 \rightarrow \bar{\mathbf{v}}_0^4 \cdot J^3(\mathbf{v}_0^3) = \bar{\mathbf{v}}_0^3 \rightarrow \bar{\mathbf{v}}_0^3 \cdot J^2(\mathbf{v}_0^2) = \bar{\mathbf{v}}_0^2 \rightarrow \bar{\mathbf{v}}_0^2 \cdot J^1(\mathbf{x}_0) = \bar{\mathbf{x}}_0$$

Observe: $\bar{\mathbf{x}}_0$ holds all components of the gradient

$$\bar{\mathbf{x}}_0 \cdot \mathbf{e}_i = (((J^4(\mathbf{v}_0^4) \cdot J^3(\mathbf{v}_0^3)) \cdot J^2(\mathbf{v}_0^2)) \cdot J^1(\mathbf{x}_0)) \cdot \mathbf{e}_i = (\mathbf{g}_0)_i$$

Define vector-matrix products - **Adjoint functions**

$$\bar{F}^n : \mathbb{R}^{M_n} \times \mathbb{R}^{M_{n+1}} \xrightarrow{\bar{F}^n} \mathbb{R}^{M_n}$$
$$\bar{F}^n(\mathbf{v}^n, \bar{\mathbf{v}}^{n+1}) := \bar{\mathbf{v}}^{n+1} \cdot J^n(\mathbf{v}^n) = \bar{\mathbf{v}}^n$$

$$\Rightarrow \quad F^4(F^3(F^2(F^1(\mathbf{x}_0)))) \qquad \bar{F}^1(\mathbf{x}_0, \bar{F}^2(\mathbf{v}_0^2, \bar{F}^3(\mathbf{v}_0^3, \bar{F}^4(\mathbf{v}_0^4, 1)))) = \bar{\mathbf{x}}_0$$

Primer: Algorithmic differentiation

A (central) example of the **adjoint function**

$$C = f(A,B) \longrightarrow \mathrm{d}C = \frac{\partial f}{\partial A}\mathrm{d}A + \frac{\partial f}{\partial B}\mathrm{d}B \qquad E = E(C) \longrightarrow \mathrm{d}E =: Tr\left(\overline{C}^T \mathrm{d}C\right)$$

$$dE = Tr\left(\overline{C}^T \frac{\partial f}{\partial A}\mathrm{d}A\right) + \left(\overline{C}^T \frac{\partial f}{\partial B}\mathrm{d}B\right) \longrightarrow \begin{cases} \overline{A} = \left(\frac{\partial f}{\partial A}\right)^T \overline{C} \\ \overline{B} = \left(\frac{\partial f}{\partial B}\right)^T \overline{C} \end{cases}$$

Take **matrix multiplication** (= tensor contraction)

$$C = AB \longrightarrow \mathrm{d}C = \mathrm{d}AB + A\mathrm{d}B \longrightarrow \begin{cases} \overline{A} = \overline{C}B^T \\ \overline{B} = A^T\overline{C} \end{cases}$$

Many other matrix functions (ED, SVD, Inverse, …)

M. Giles, https://people.maths.ox.ac.uk/gilesm/files/NA-08-01.pdf

Fresh developments: Complex SVD, Lanczos, …

Z.Q. Wan, S.X. Zhang arXiv:1909.02659, H. Xie, J.G. Liu, L. Wang arXiv:2001.04121

Algorithmic Differentiation

- Both **Forward mode** and **Reverse mode** evaluate derivatives with **machine precision**

- **Forward mode** has complexity O(N) * O(E)

- **Reverse mode has complexity O(1) * O(E)**

- Caveat: Memory requirements **are not bounded** !
- Implemented in major machine-learning frameworks: **TensorFlow**, **PyTorch**, …

  … or in one of the libraries for your favorite language Fortran, C++, Julia, etc. (see http://www.autodiff.org)

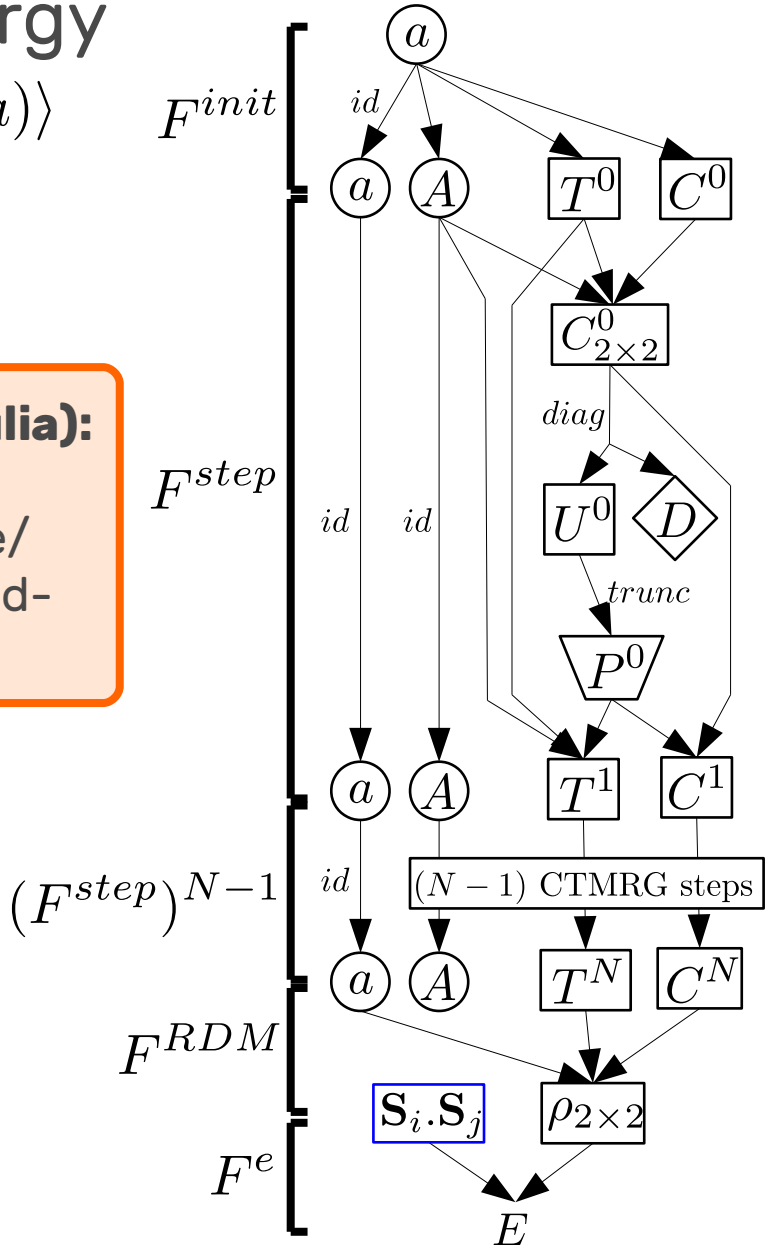
J1-J2 Model: Energy as DAG

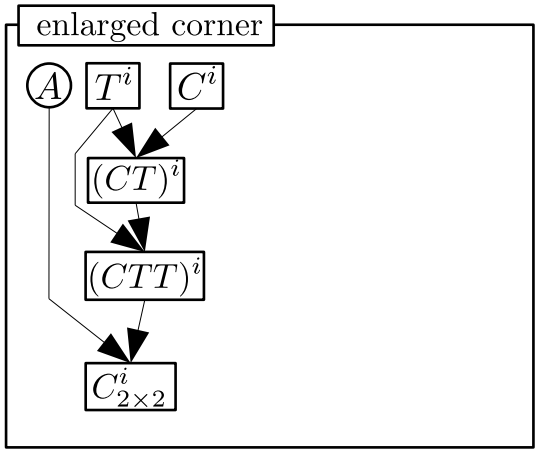
Variational energy

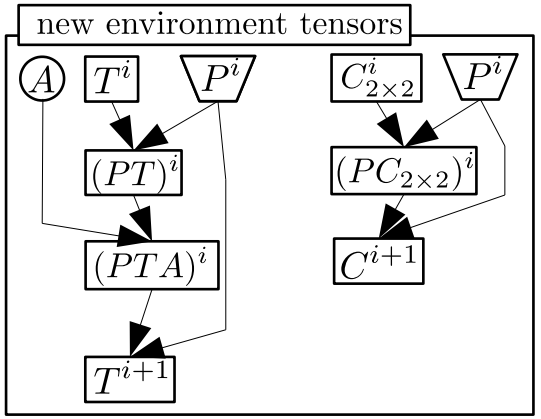
$$E(a) = \langle \psi(a) | H | \psi(a) \rangle$$

Great hands-on (in Julia):

http://blog.rogerluo.me/2018/10/23/write-an-ad-in-one-day/



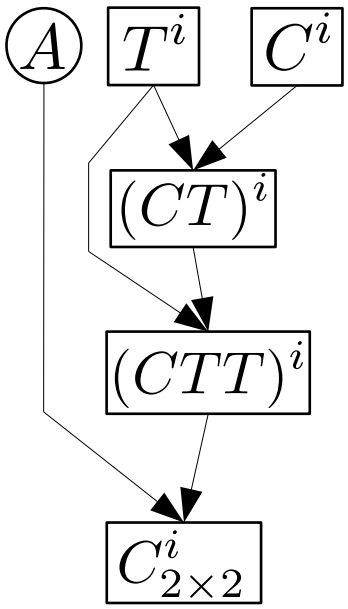




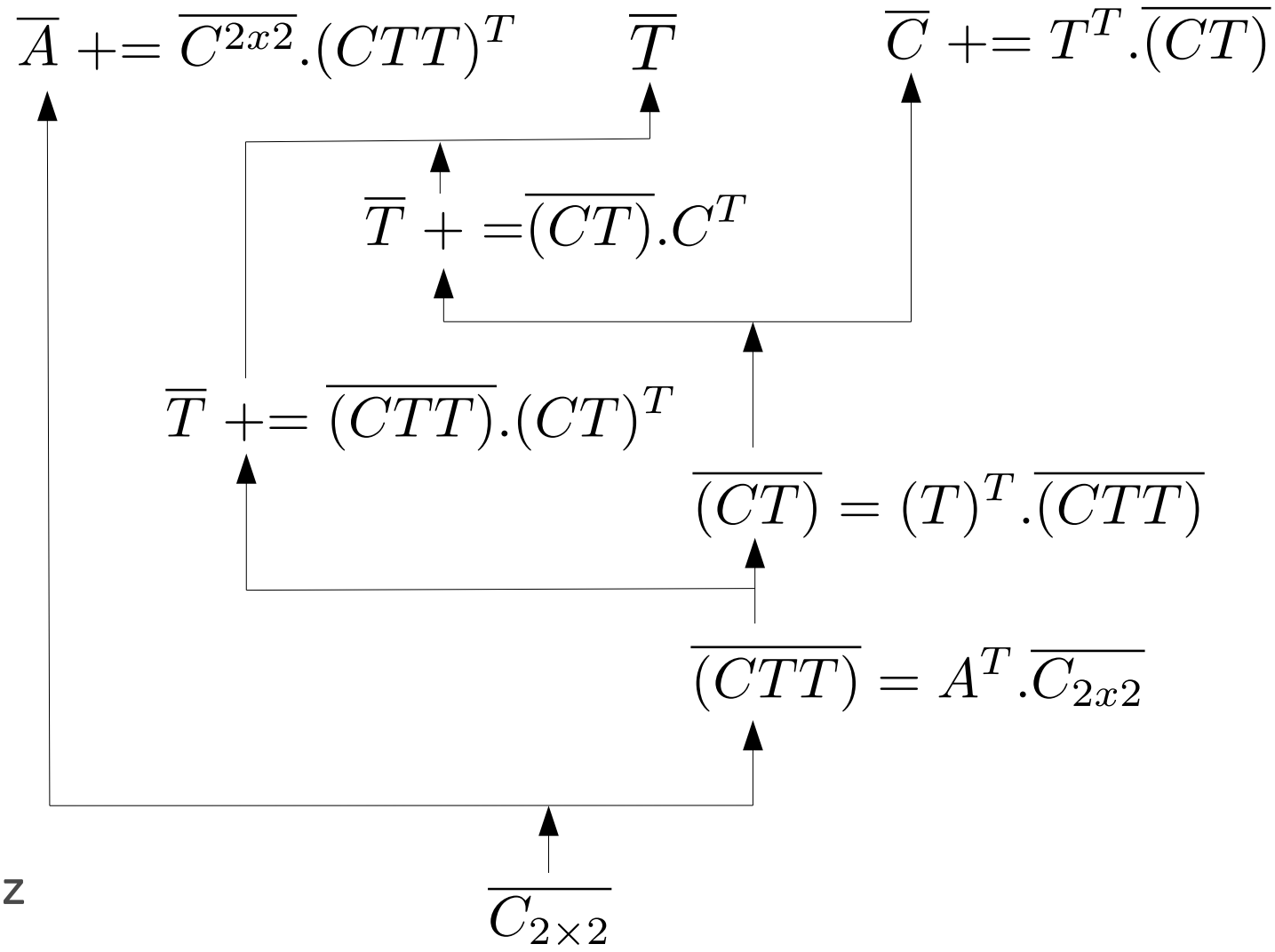
J1-J2 Model: Energy as DAG

Enlarged corner

Forward



Backward

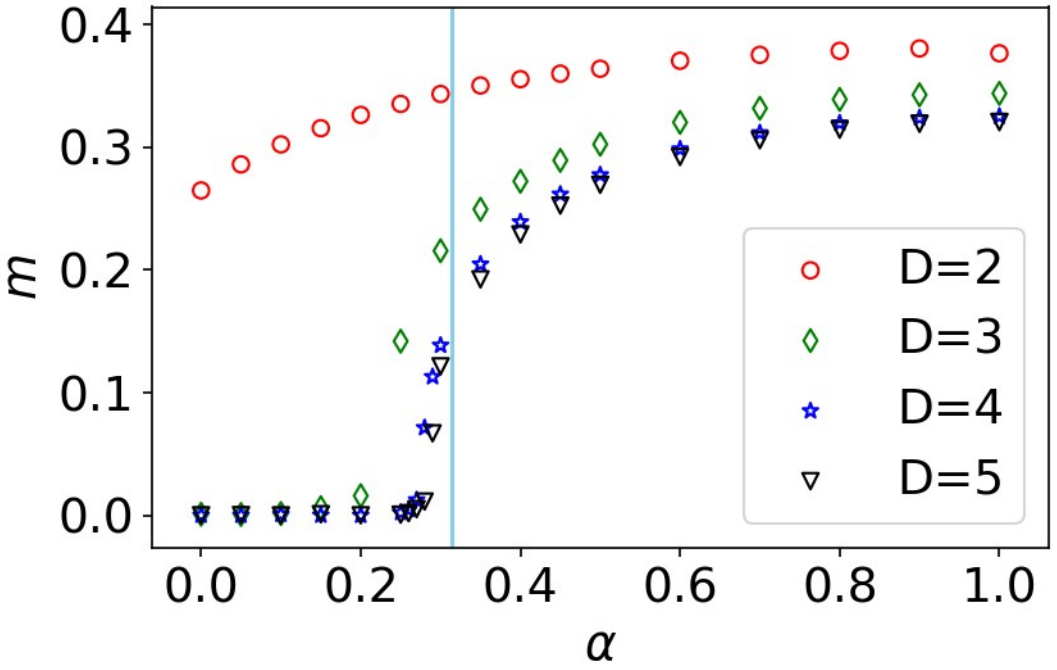


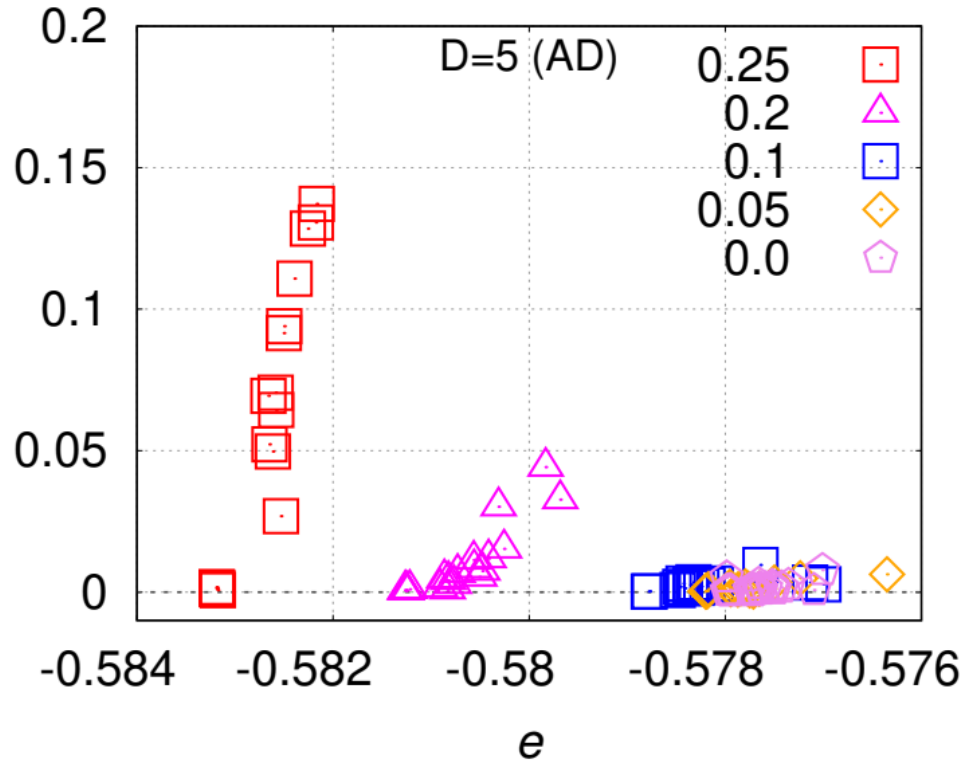
Analogy ?
S.P.G. Crone, P. Corboz
arXiv:1912.00908

Ladders: Revisited with AD

Identical protocol: Initialize by a set of 24 VBS states with noise

- energy minimization gives sound result even at **D=3** !
- at **D=5**, despite **rough landscape** the expected magnetization curve is recovered

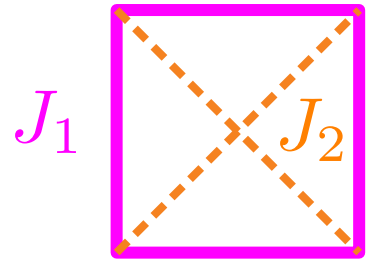




Intro: Square Lattice J1-J2 Model

Prototypical example of a frustrated magnet

$$H = J_1 \sum_{\langle i,j \rangle} S_i . S_j + J_2 \sum_{\langle\langle i,j \rangle\rangle} S_i . S_j$$

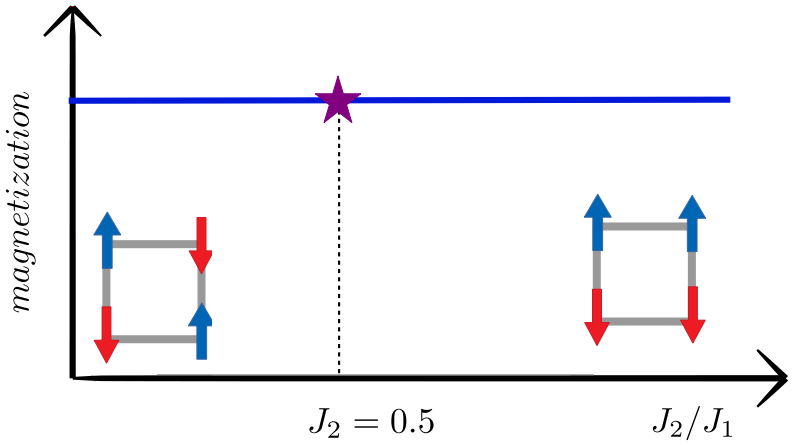


- Classically: transition at **$J_2/J_1 = 0.5$**

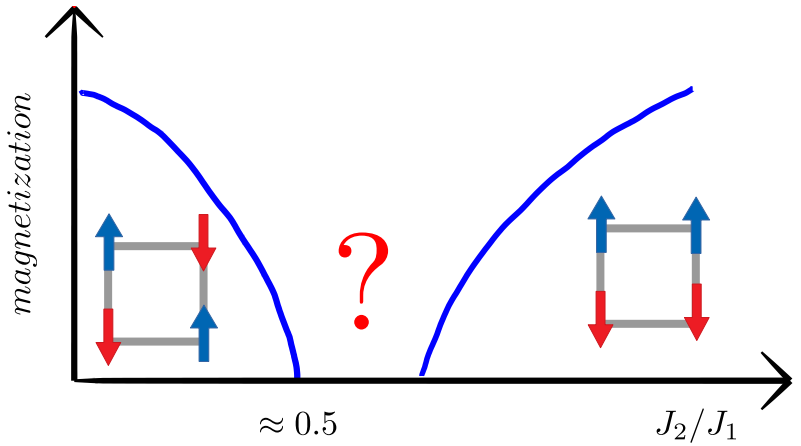
Spin waves:

P. Chandra and B. Doucot, Phys. Rev. B 38, 9335 (1988)

- **Transition** from **Néel** to **paramagnetic** phase near maximally frustrated point **$J_2/J_1 \approx 0.5$**
- For **$J_2/J_1 \gtrsim 0.6$** system orders again in stripes



$\Longrightarrow$ QM



Intro: Square Lattice J1-J2 Model

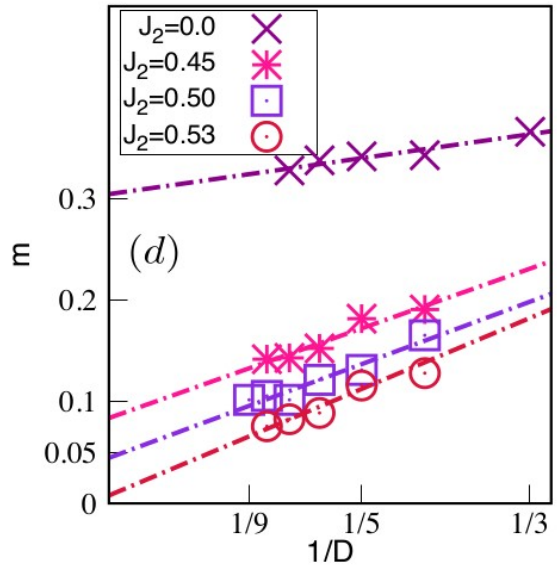
Open question: the intermediate phase ?

Spin liquid: gapless U(1), gapped $\mathbb{Z}_2$, gapless $\mathbb{Z}_2$

DMRG: Jiang et al., Phys. Rev. B 86, 024424 (2012)
VMC: Hu et al., Phys. Rev. B 88, 060402 (2013)
PFFRG: Herring et al., Phys. Rev. B 99, 100405(R) (2019)
PEPS: Liu et al., Phys. Rev. B 98, 241109(R) (2018)
iPEPS: D. Poilblanc and M. Mambrini, Phys. Rev. B 96, 014414 (2017)

Valence bond solids: columnar, plaquette

PEPS: Wang et al., Phys. Rev. B 94, 075143 (2016)
iPEPS: R. Haghshenas and D. N. Sheng, Phys. Rev. B 97, 174408 (2018)

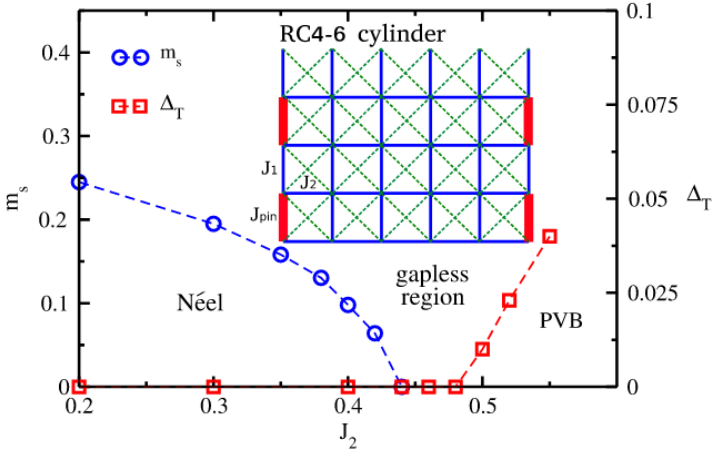


Gapless spin liquid into VBS

DMRG: L. Wang and A. Sandvik, PRL 121, 107202 (2018)

Gapless(?) spin liquid into Plaquettes

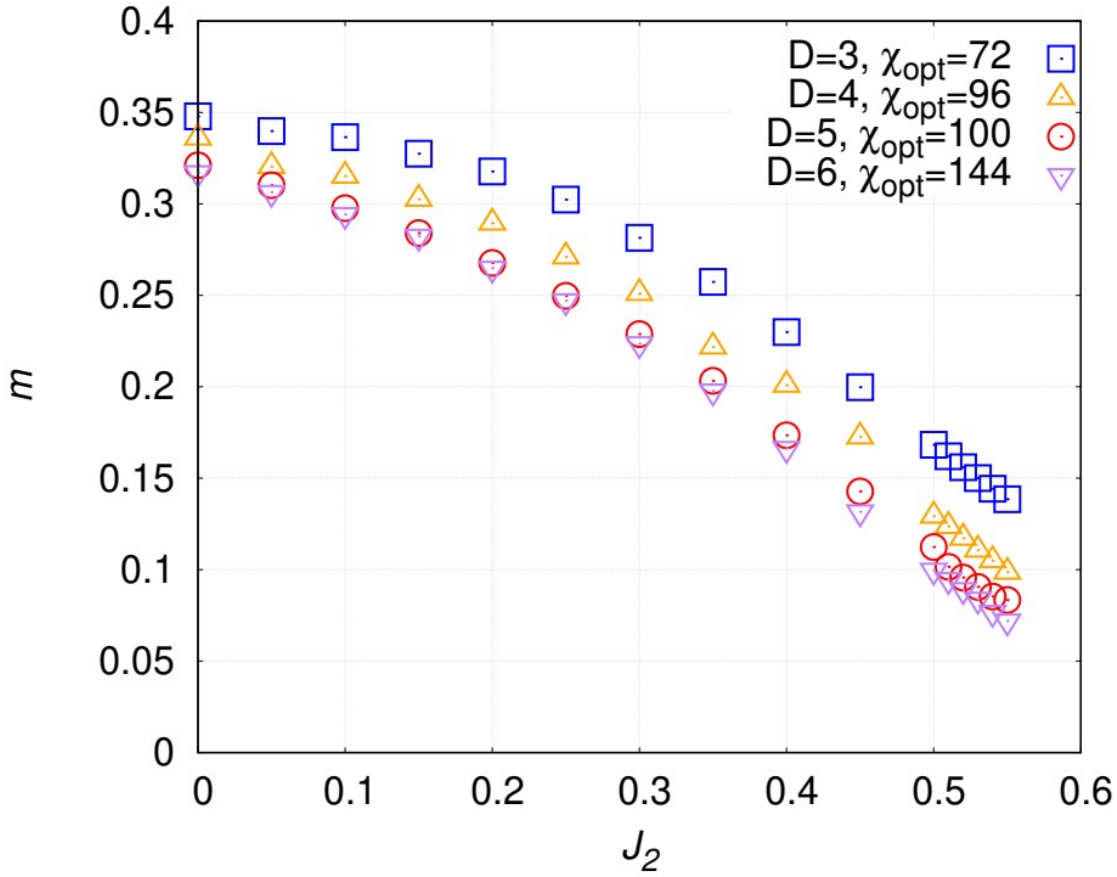
DMRG: Gong et al., Phys. Rev. Lett. 113, 027201 (2014)

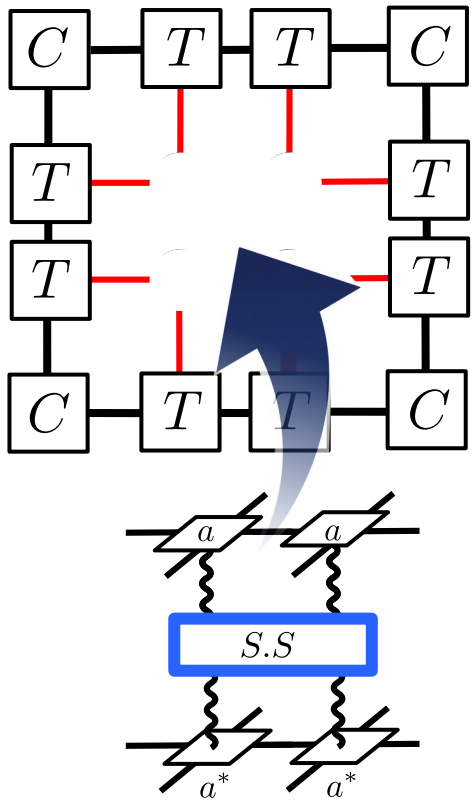


J1-J2 Model: Phase diagram with AD

Gradient of the energy is obtained by AD

- iPEPS: **single C_{4v} invariant tensor**
- several random tensors initializations
- **χ** used in optimization: 72, 96, 100 and 144

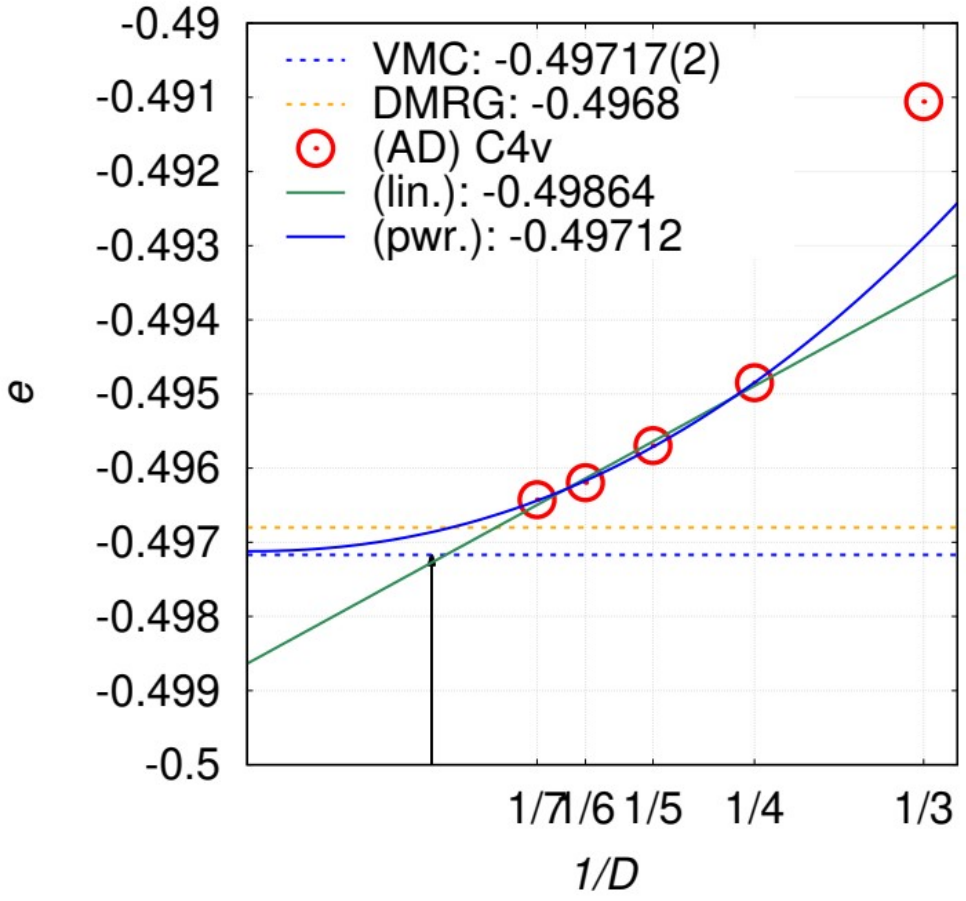


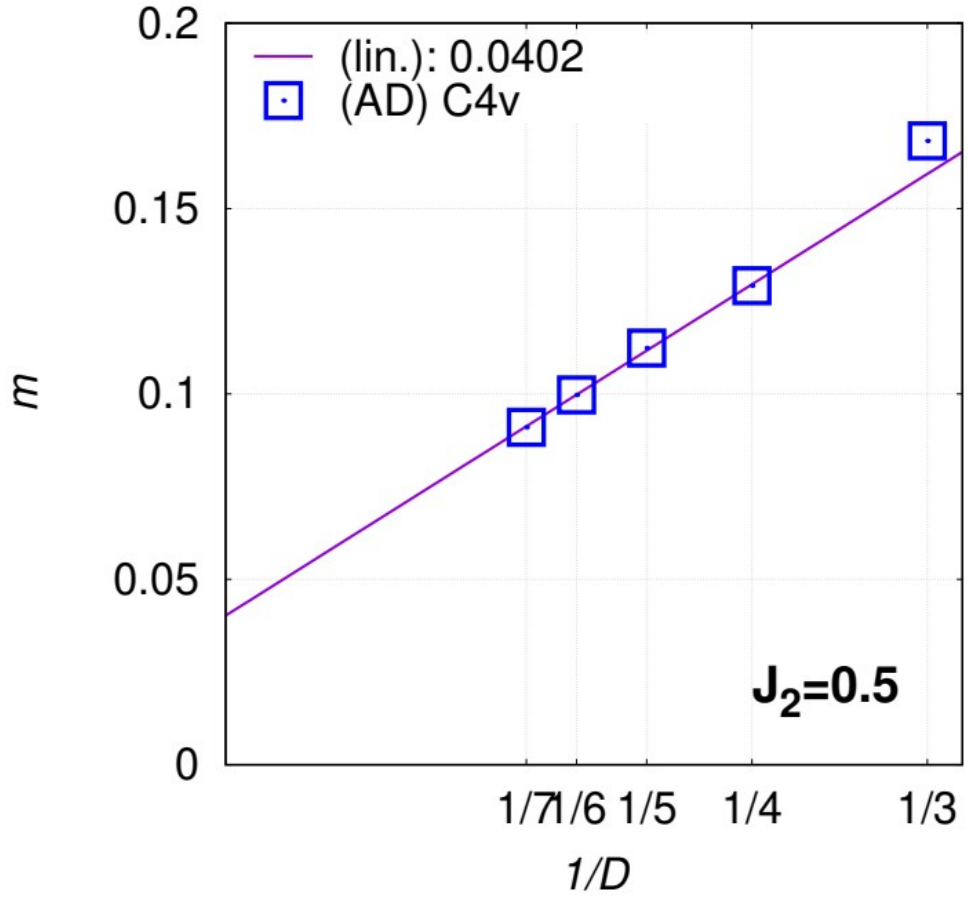


J1-J2 Model: Highly-frustrated

Analysis at highly-frustrated point J_2=0.5

- power law extrapolation of energy compatible with VMC
- **small but finite** magnetization

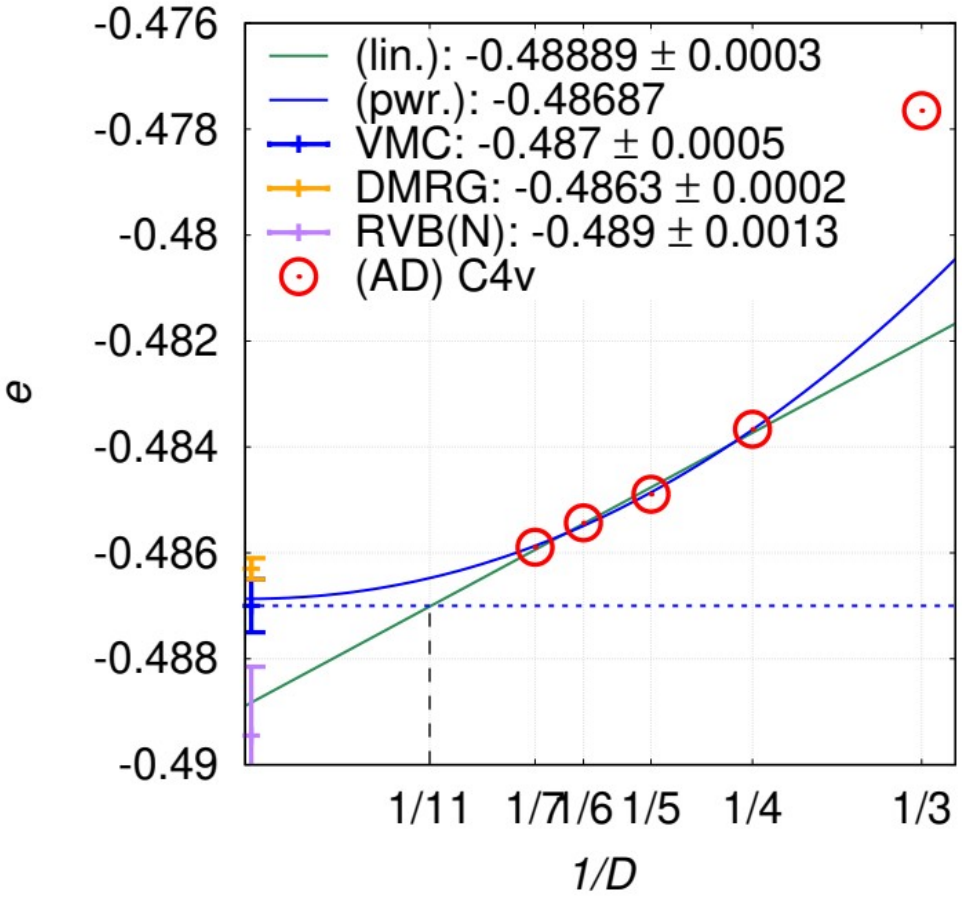


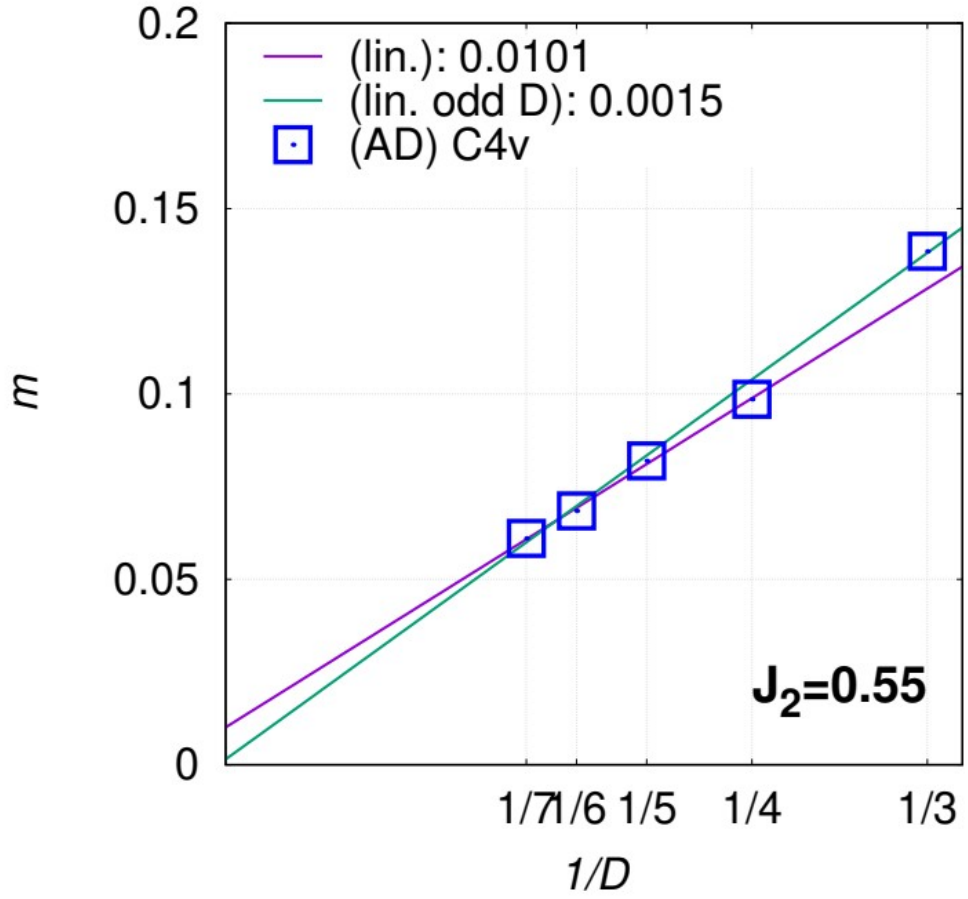


J1-J2 Model: Deep in the phase

Analysis at point J_2=0.55, inside non-magnetic region

- **very good** iPEPS variational energy
- odd-D data point to non-magnetic GS

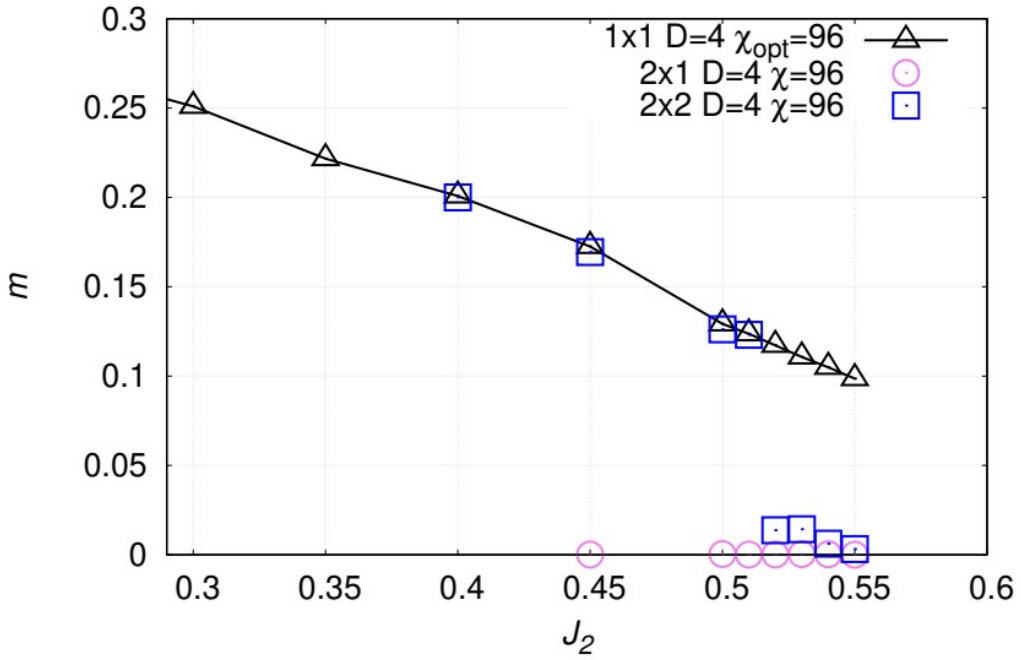


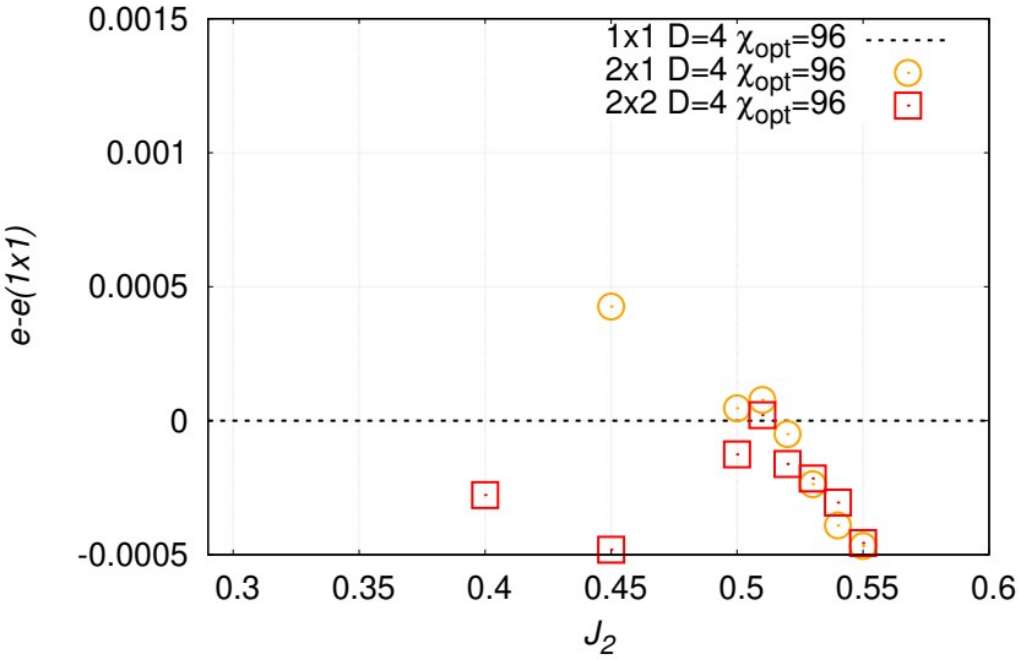


J1-J2 Model: Spin-liquid vs VBS

Generalize to larger unit cells – **2x1, 2x2**

- **transition to VBS** at $J_2 \approx 0.52$ (1st order ?)
- Finite-D effect or genuine order in thermodynamic limit ?





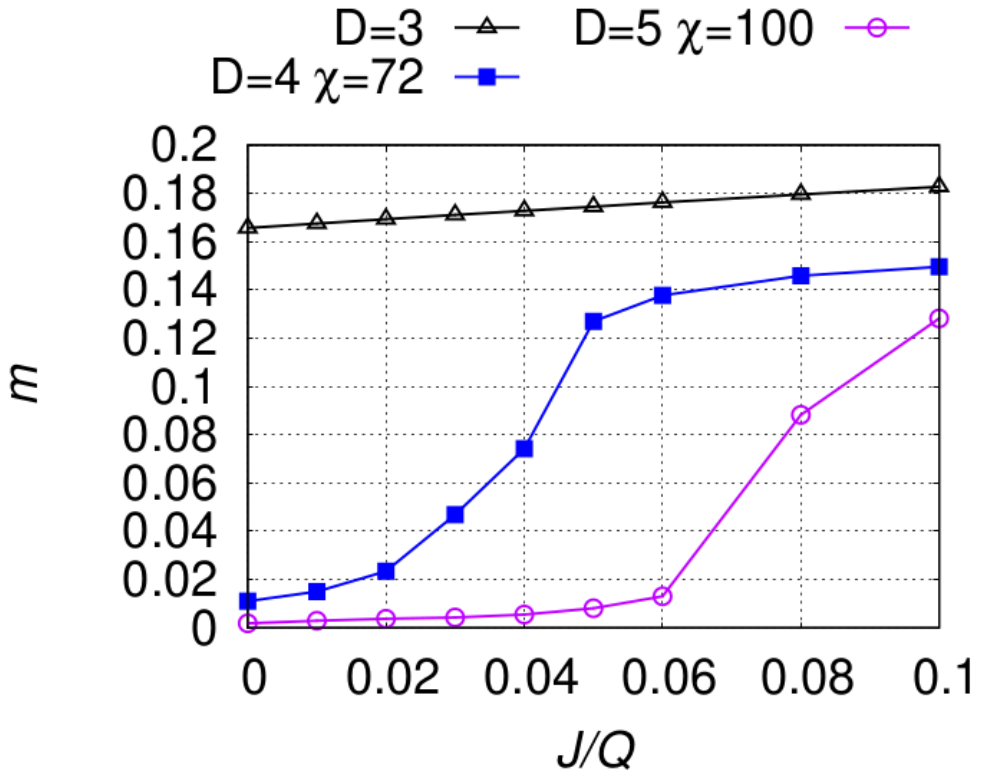
J1-J2 Model vs J-Q Model

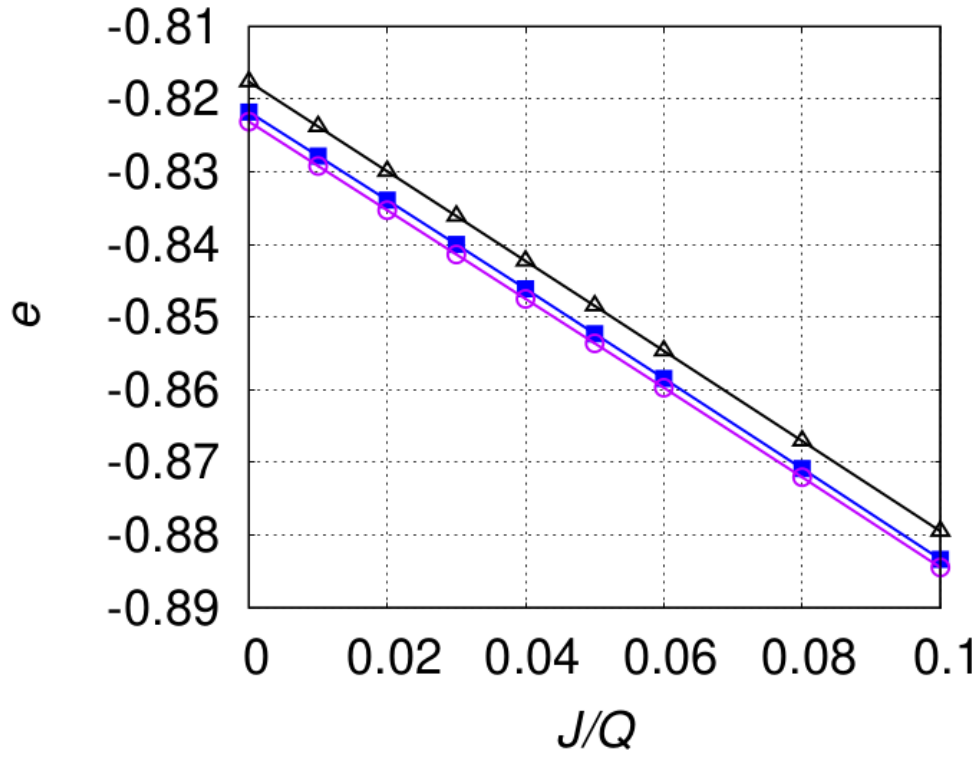
J-Q model

$$J\sum_{\langle ij\rangle}\mathbf{S}_i\cdot\mathbf{S}_j - Q\sum_{\langle ijkl\rangle}(\mathbf{S}_i\cdot\mathbf{S}_j-1/4)(\mathbf{S}_k\cdot\mathbf{S}_l-1/4)$$

DQCP around J/Q = 0.04 (A. Sandvik PRL 98, 227202 (2007))

- [Prelim.] Finite-D seems to converge toward a sharp feature





Concluding remarks I

Open-source suite of codes: **tn-torch**

github.com/jurajHasik/tn-torch
J. Hasik and G. Mbeng

CTM + AD optimization based on PyTorch with CPU and/or GPU (https://pytorch.org/)

- CTM for **arbitrary unit cells**
- Examples for NN, NNN, plaquette, ...
- Some extra stuff: correlation functions, transfer matrix spectrum, ...

Concluding remarks II

Optimization with FU can be problematic

Gradient optimization (with AD) is a way forward

- easy to extend beyond the nearest-neighbour Hamiltonians
- highly-optimized implementation out-of-the box (TensorFlow, PyTorch, ...)

iPEPS + AD: a step closer to DMRG-like variational method in two dimensions

Devil is in the details: adjoint for iterative SVD / ED, not enough memory, derivative at the fixed point ?, ...