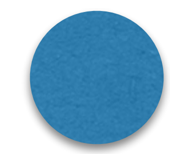
From DMRG to TNS

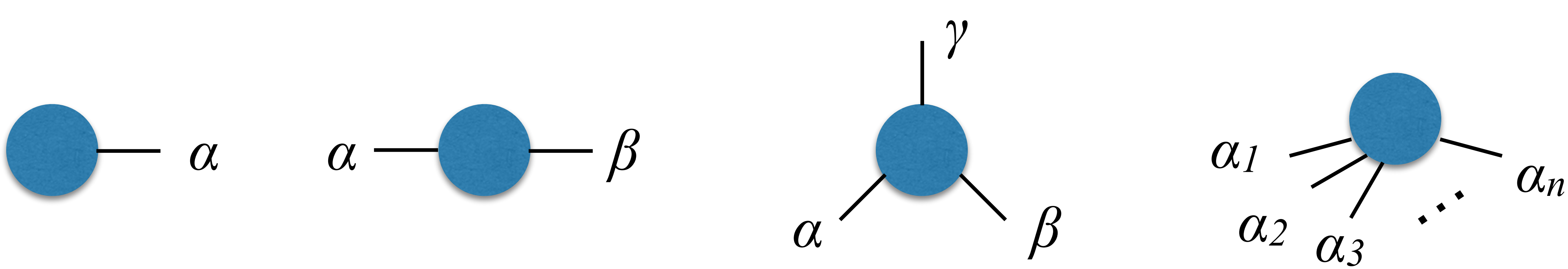
Ying-Jer Kao
Department of Physics
National Taiwan University

Graphical Representation

$$A = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ \vdots \\ A_N \end{bmatrix} \quad B = \begin{bmatrix} B_{11} & \cdots & B_{1n} \\ \vdots & \ddots & \vdots \\ B_{m1} & \cdots & B_{mn} \end{bmatrix}$$

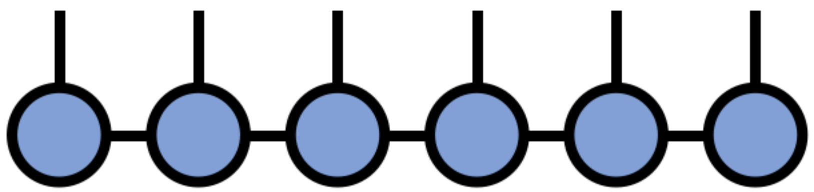
scalar  S

vector	matrix	rank-3 tensor	rank-*n* tensor
A_α	$B_{\alpha\beta}$	$C_{\alpha\beta\gamma}$	$T_{\alpha_1\alpha_2\alpha_3\ldots\alpha_n}$

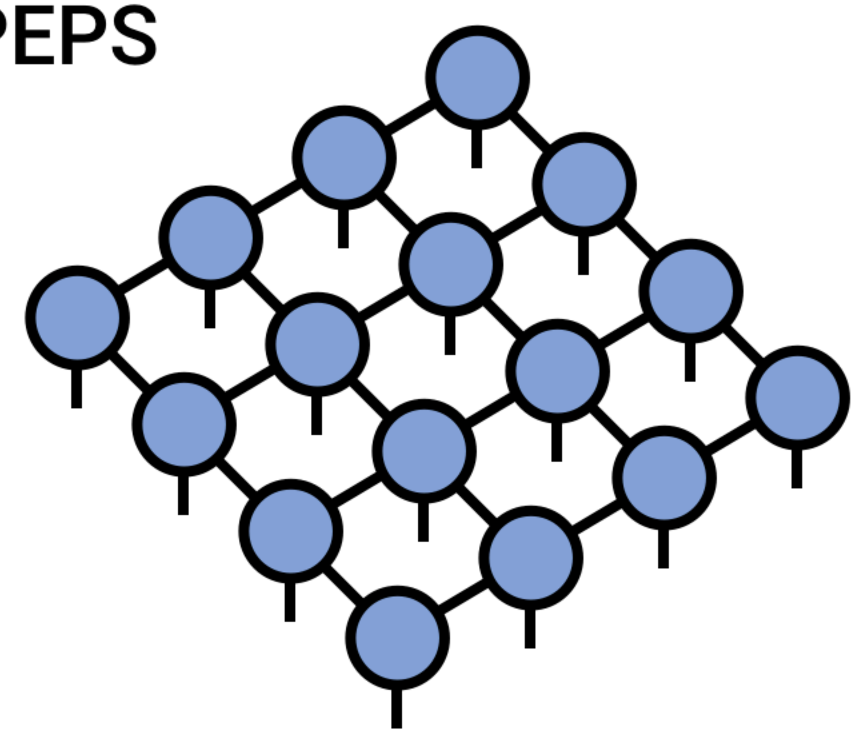


Tensor Network States

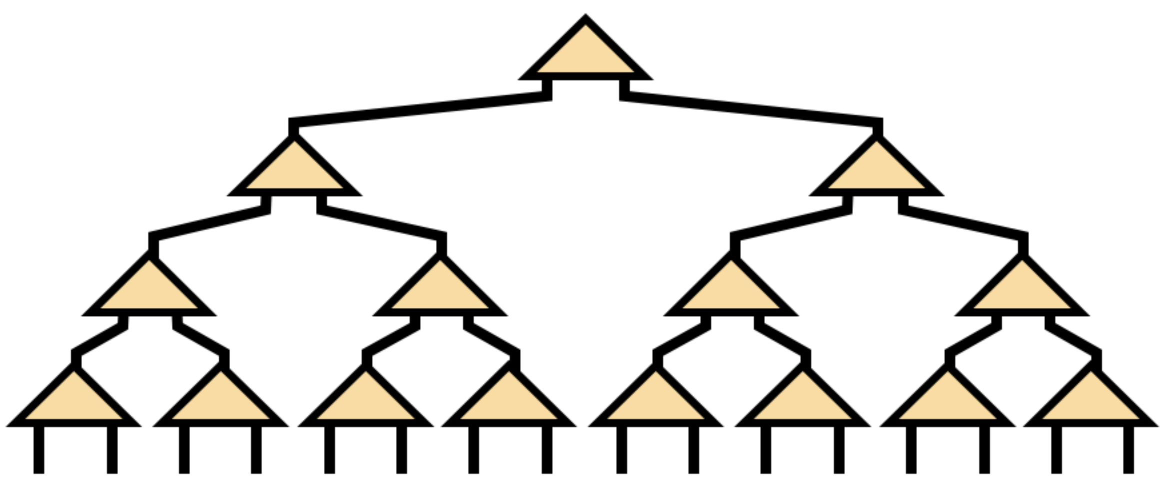
Matrix Product State / Tensor Train



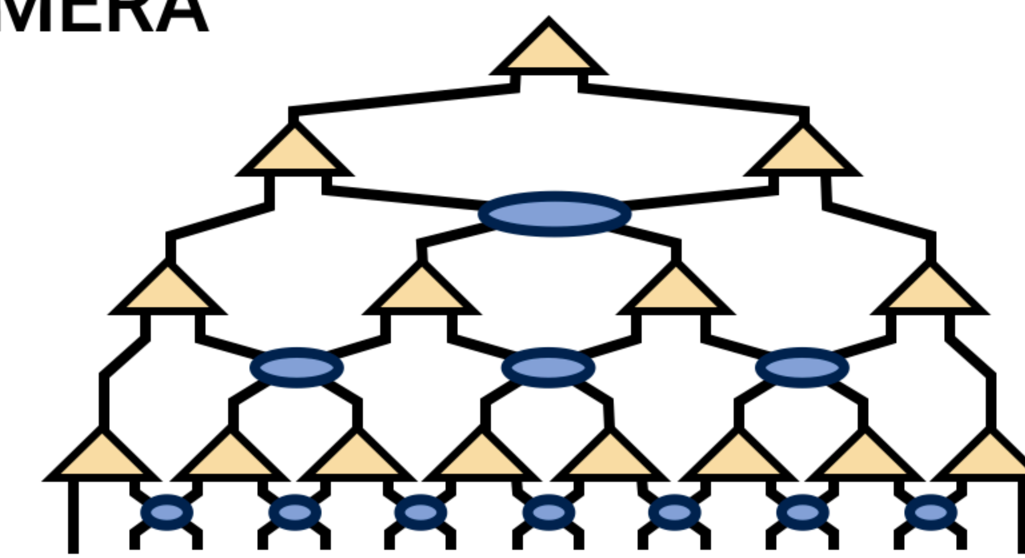
PEPS



Tree Tensor Network / Hierarchical Tucker



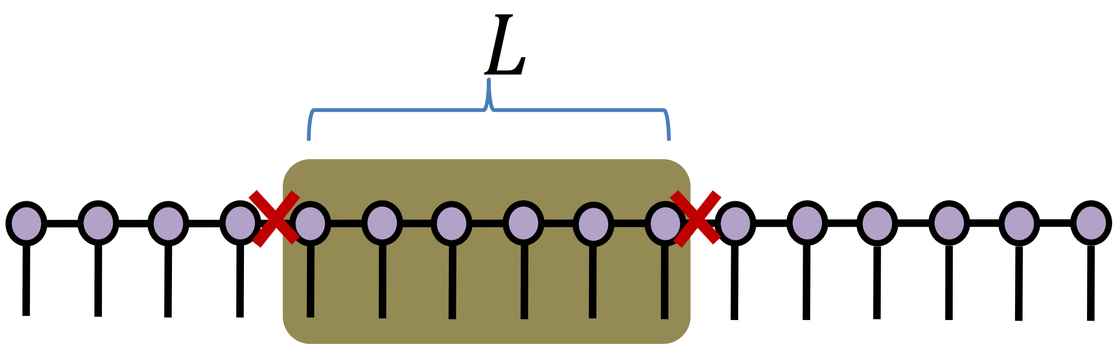
MERA



http://tensornetwork.org/

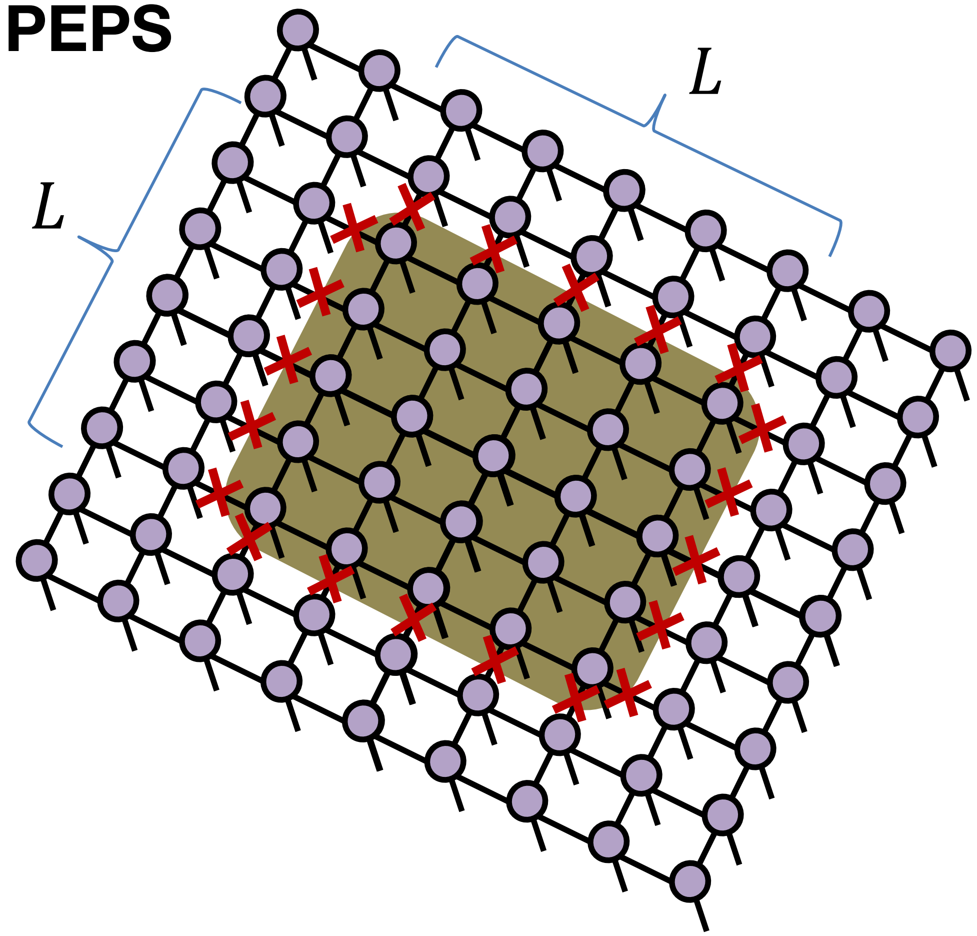
Entanglement

MPS



$S_L \sim L^0 \sim$ **const.**

PEPS

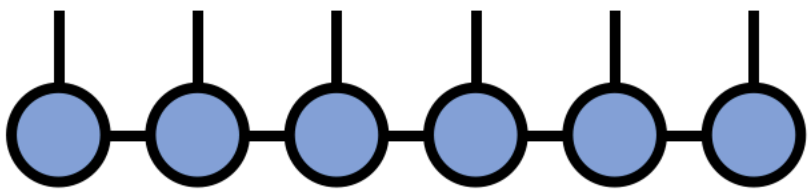


$S_{L^2} \sim L^1$

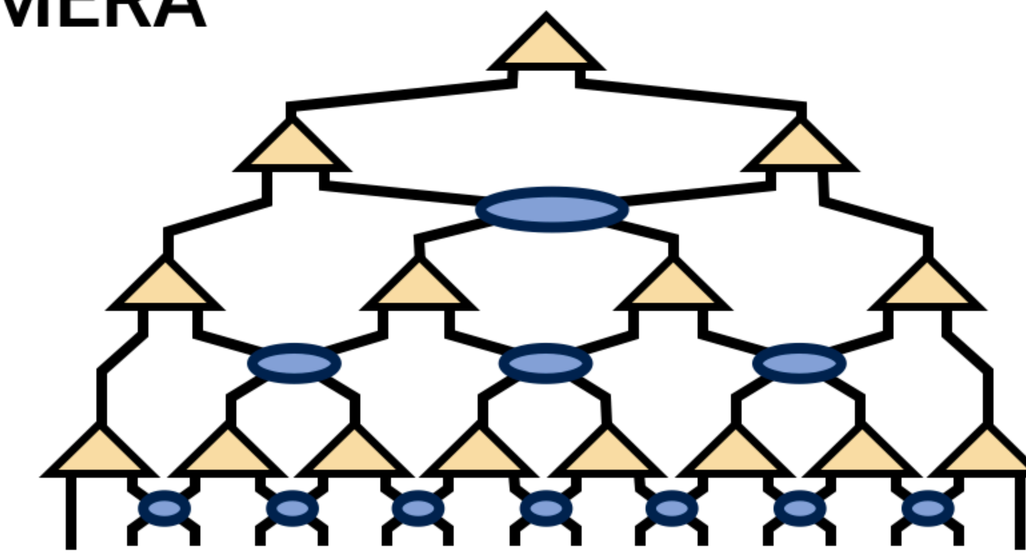
Entanglement area law: $S_{L^D} \sim L^{D-1}$

Tensor Network States

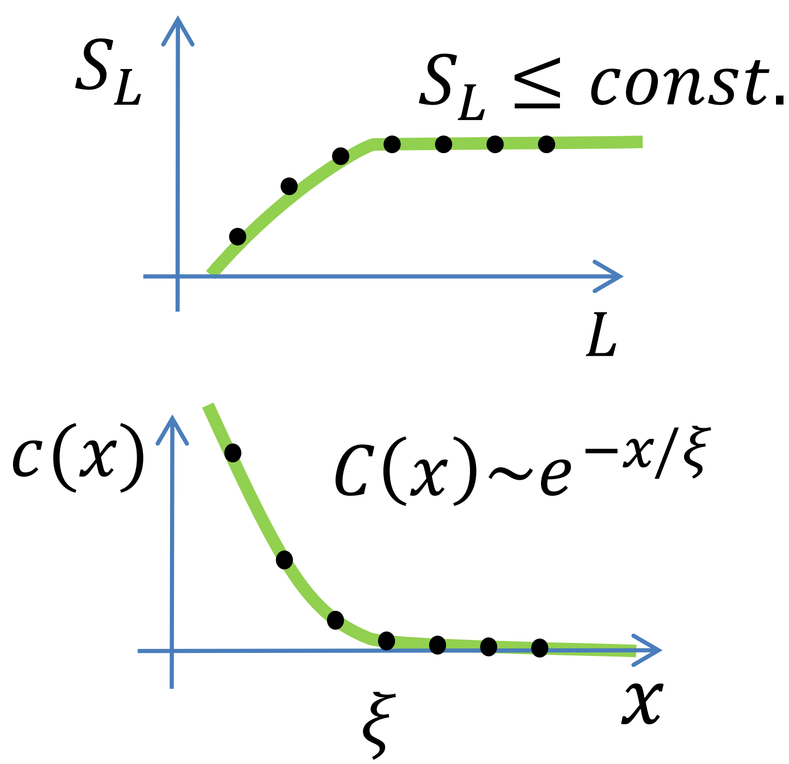
Matrix Product State / Tensor Train



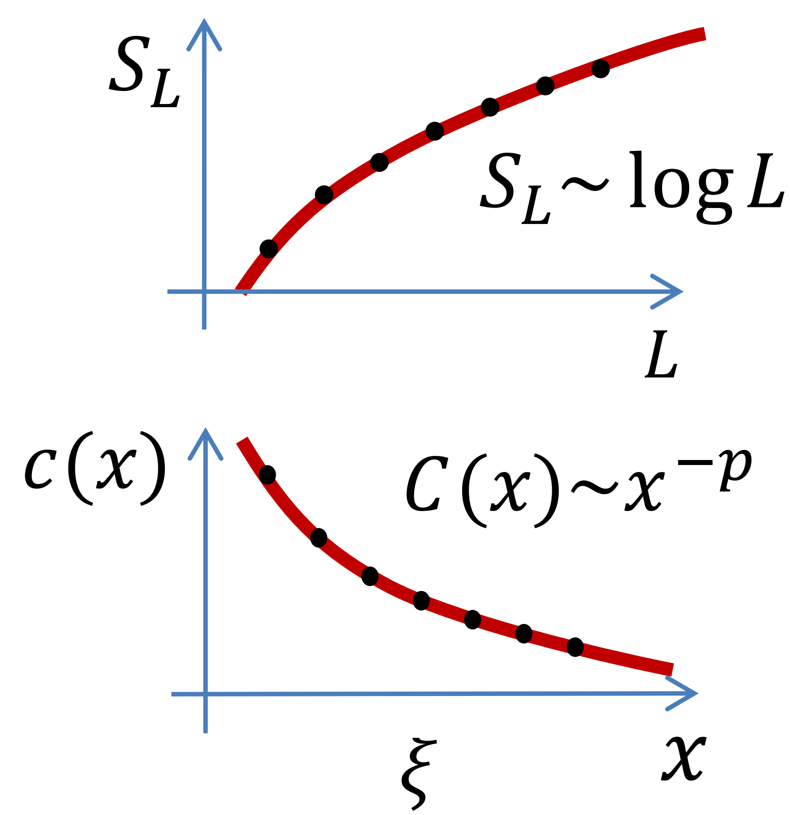
MERA



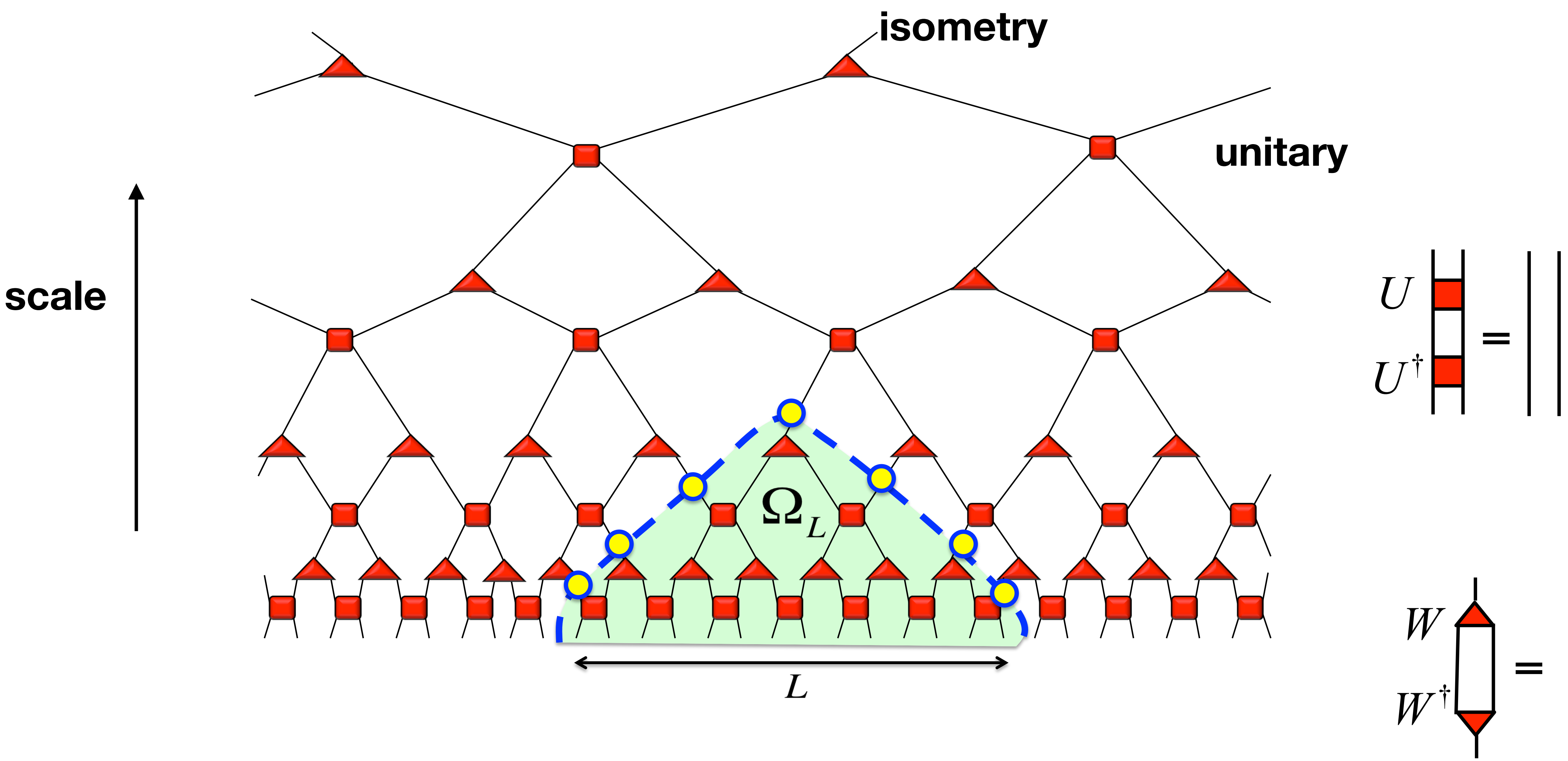
gapped Hamiltonian



gapless Hamiltonian



MERA

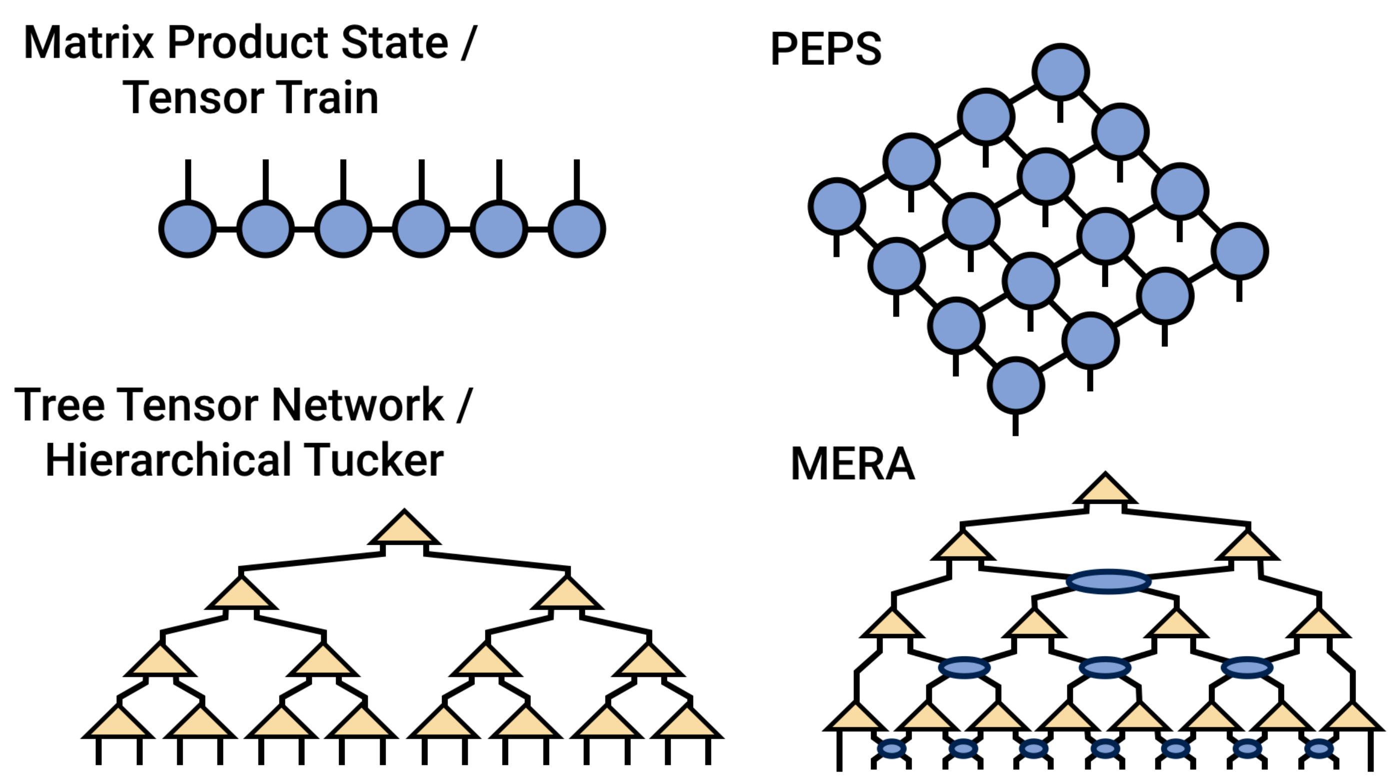


Entanglement Entropy ~ number of bonds cut

For 1D scale invariant MERA, $S \sim \log L$

Entanglement Scaling

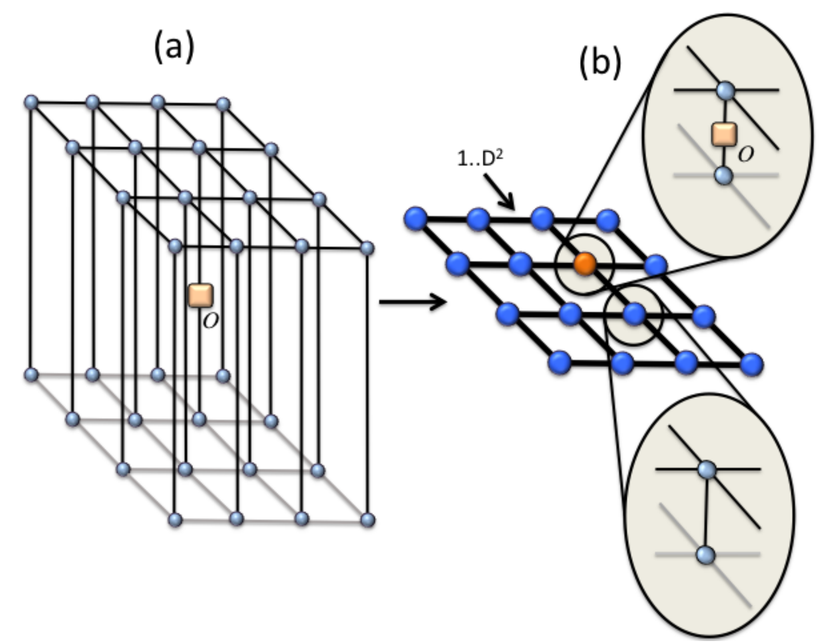
	MPS	2d PEPS	TTN	1d MERA	1d bMERA
$S(L)$	$O(1)$	$O(L)$	$O(1)$	$O(\log L)$	$O(L)$
$\langle O \rangle$	exact	approx.	exact	exact	exact
ξ	$< \infty$	$\leq \infty$	$< \infty$	$\leq \infty$	$\leq \infty$
Tensors	any	any	any	unit./isom.	unit./isom.
Can. form	obc, ∞	no	yes	–	–



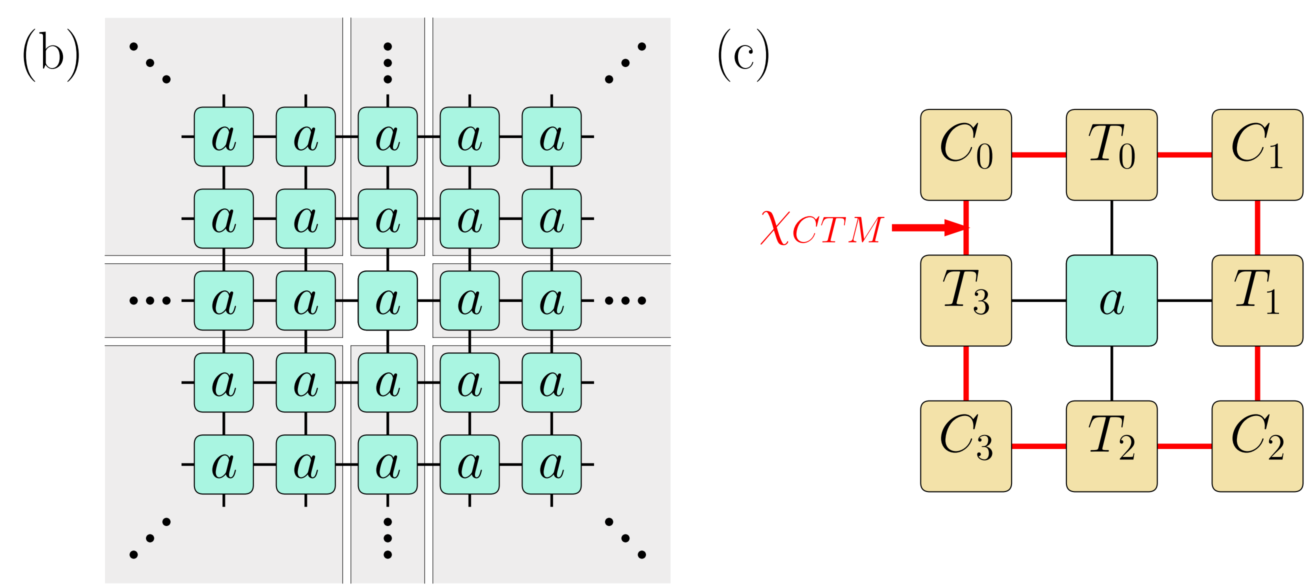
Algorithms

- Finding ground state wave function $|\psi_g\rangle$
  - Imaginary time evolution/ Simple update:
    consider only local environment (Fast, less accurate)
  - Variational update/ Full update:
    consider the global environment (Slow, more accurate)

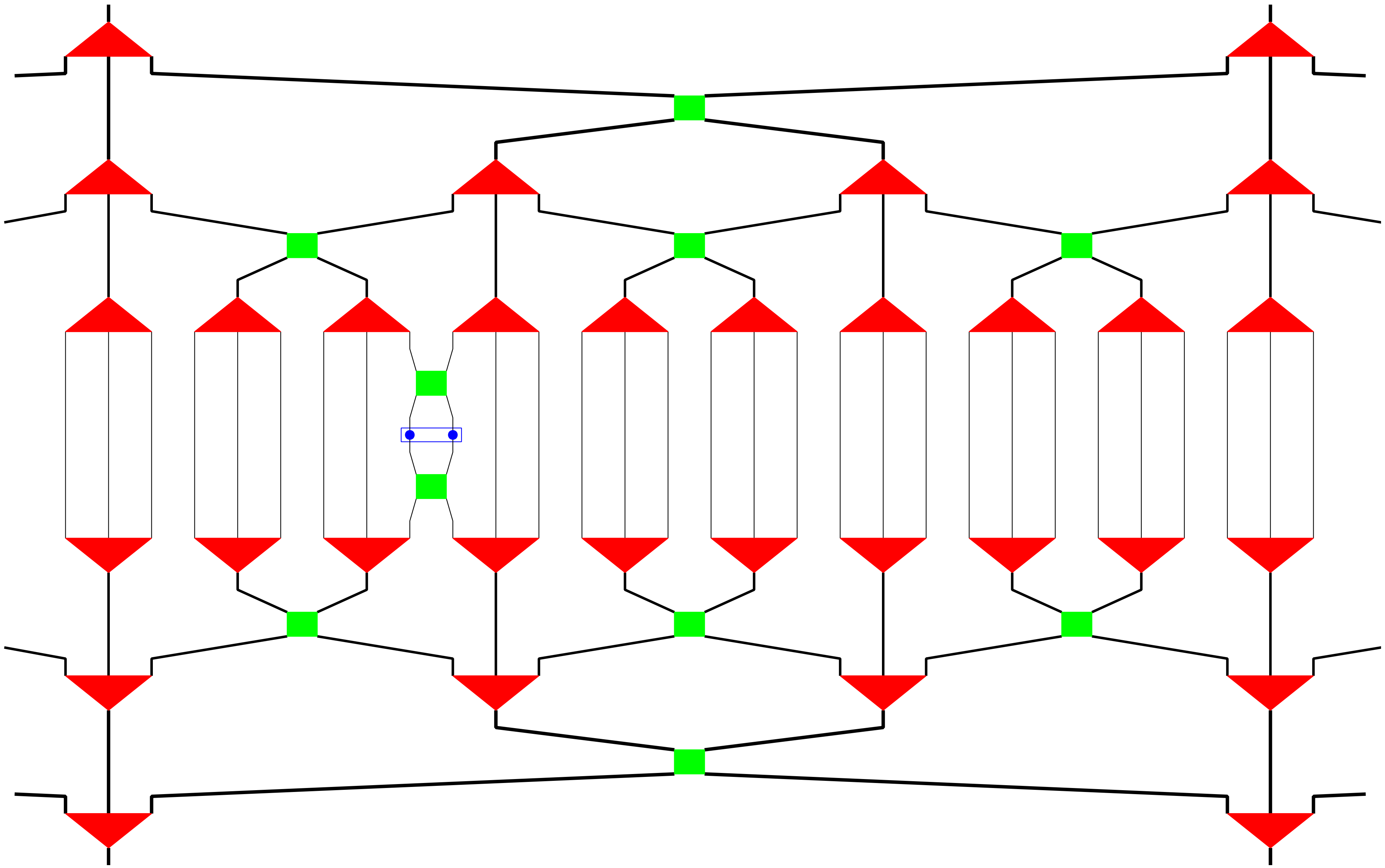
Algorithms



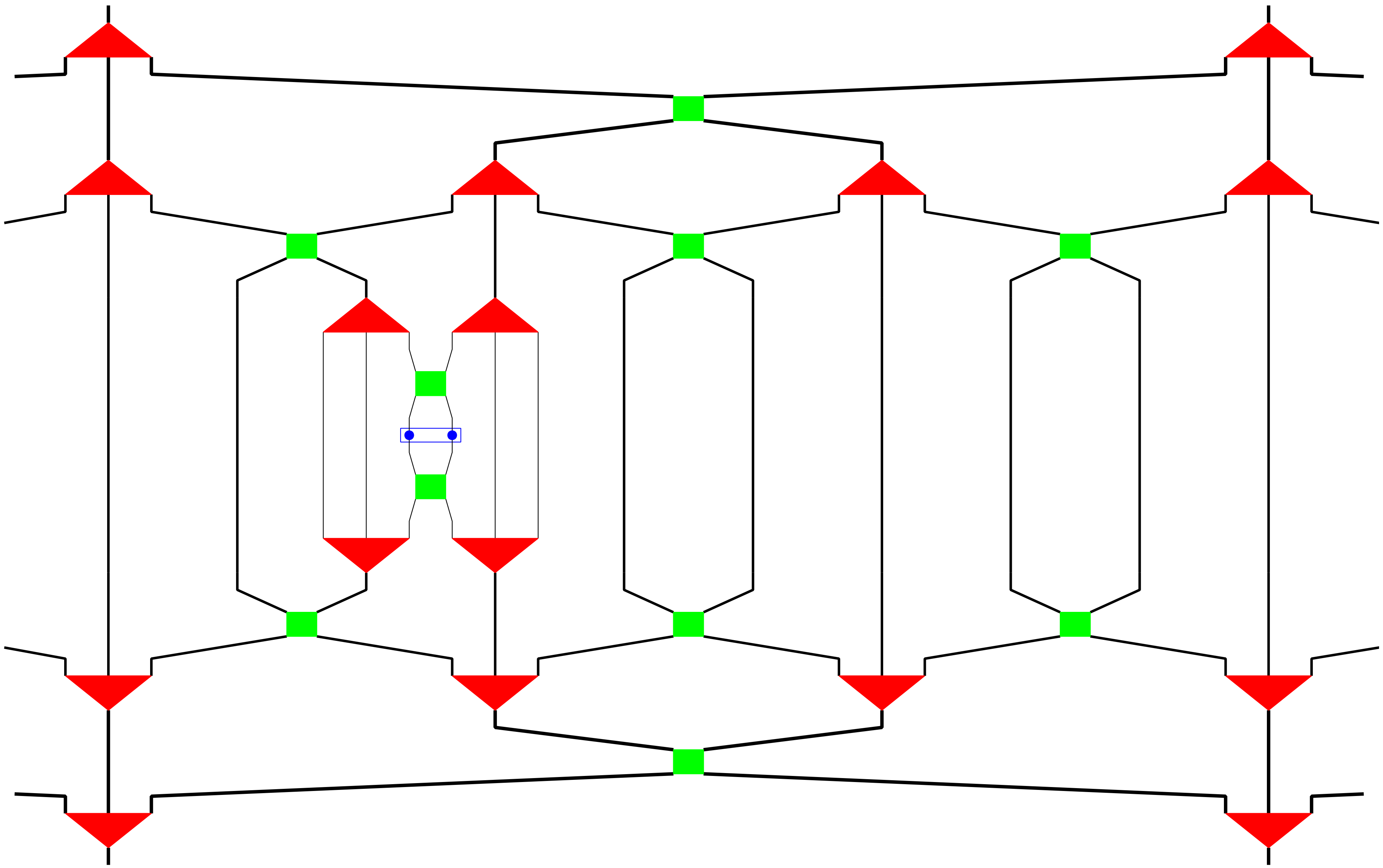
- Expectation value $\langle \psi_g | O | \psi_g \rangle$
  - Finite PEPS: boundary MPS
  - Infinite PEPS: Corner Transfer Matrix, boundary MPS, channel method
  - MERA: exact contraction



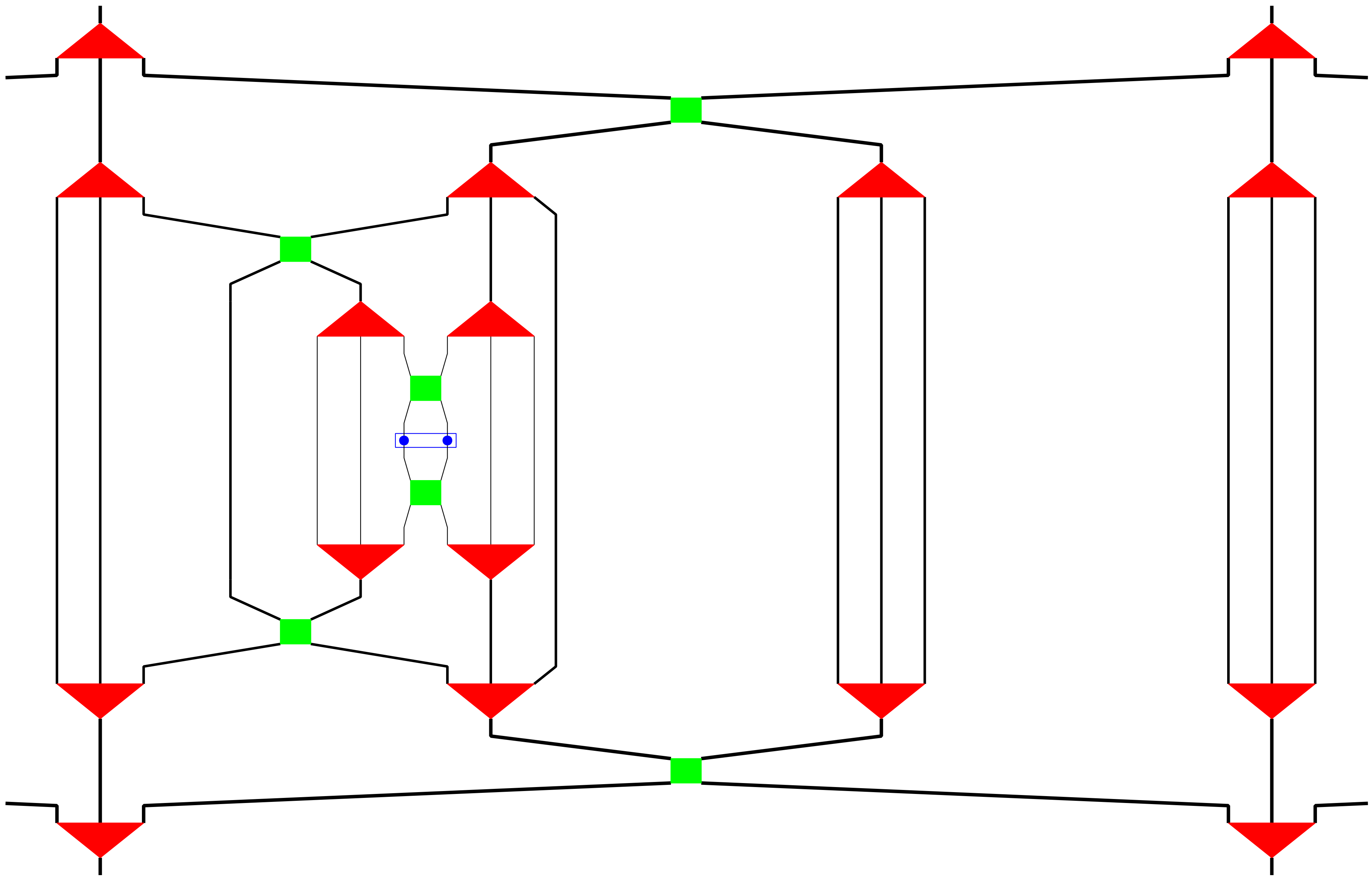
MERA: expectation value



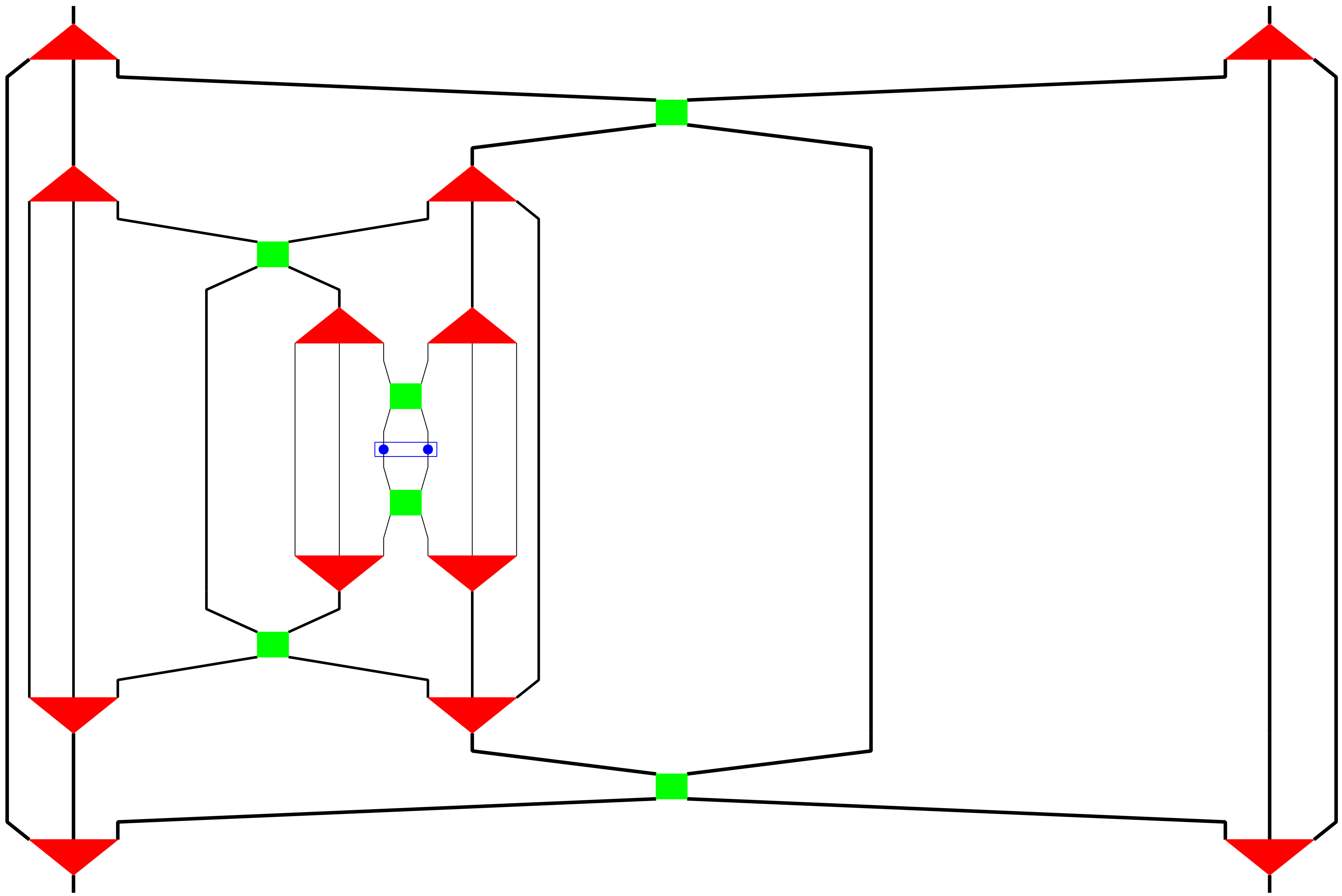
MERA: expectation value



MERA: expectation value



MERA: expectation value



Applications

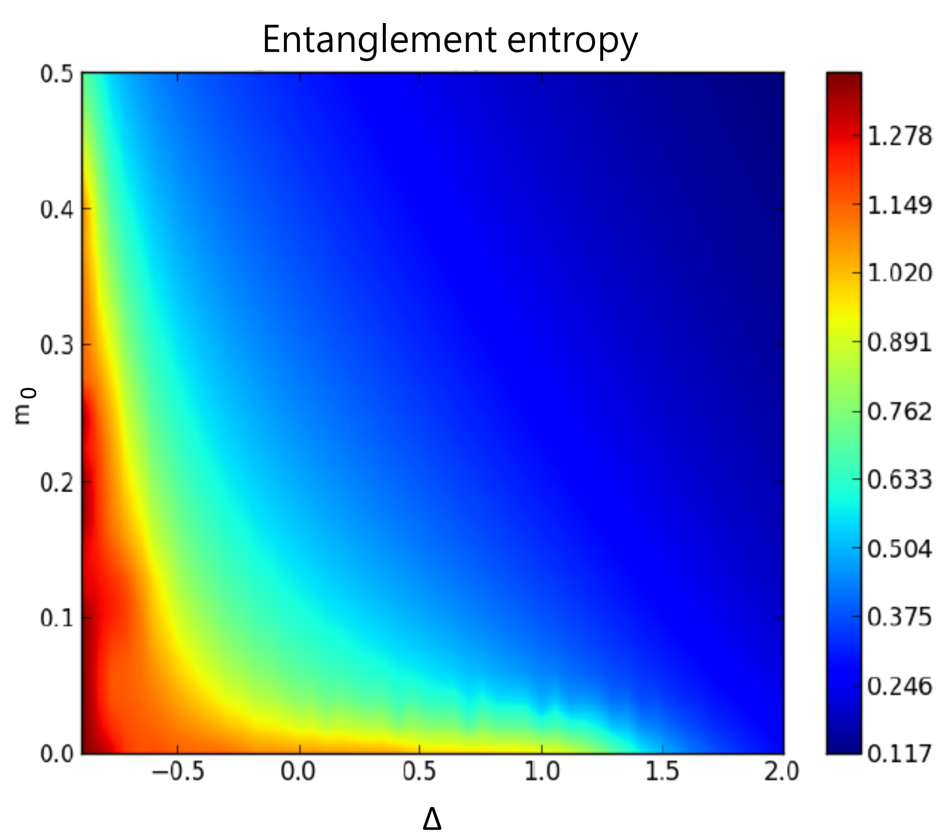
- Quantum Frustrated Magnets (DMRG, iPEPS/iPESS)
- Topological order (DMRG, PESS)
- Disordered system (Tree TN, PEPS)
- Dynamics (Mostly tDMRG/ TDVP)
- Open systems (MPS, PEPS)
- Conformal Field Theory (sMERA, iDMRG)
- Classical Statistical Mechanics (PEPS)
- Boundary CFT (bMERA, DMRG+IBC)
- Holography (MERA, other)
- Quantum Field Theory (MPS, PEPS)
- Quantum-classical programming (MPS)
- Machine Learning (MPS, MERA-like)

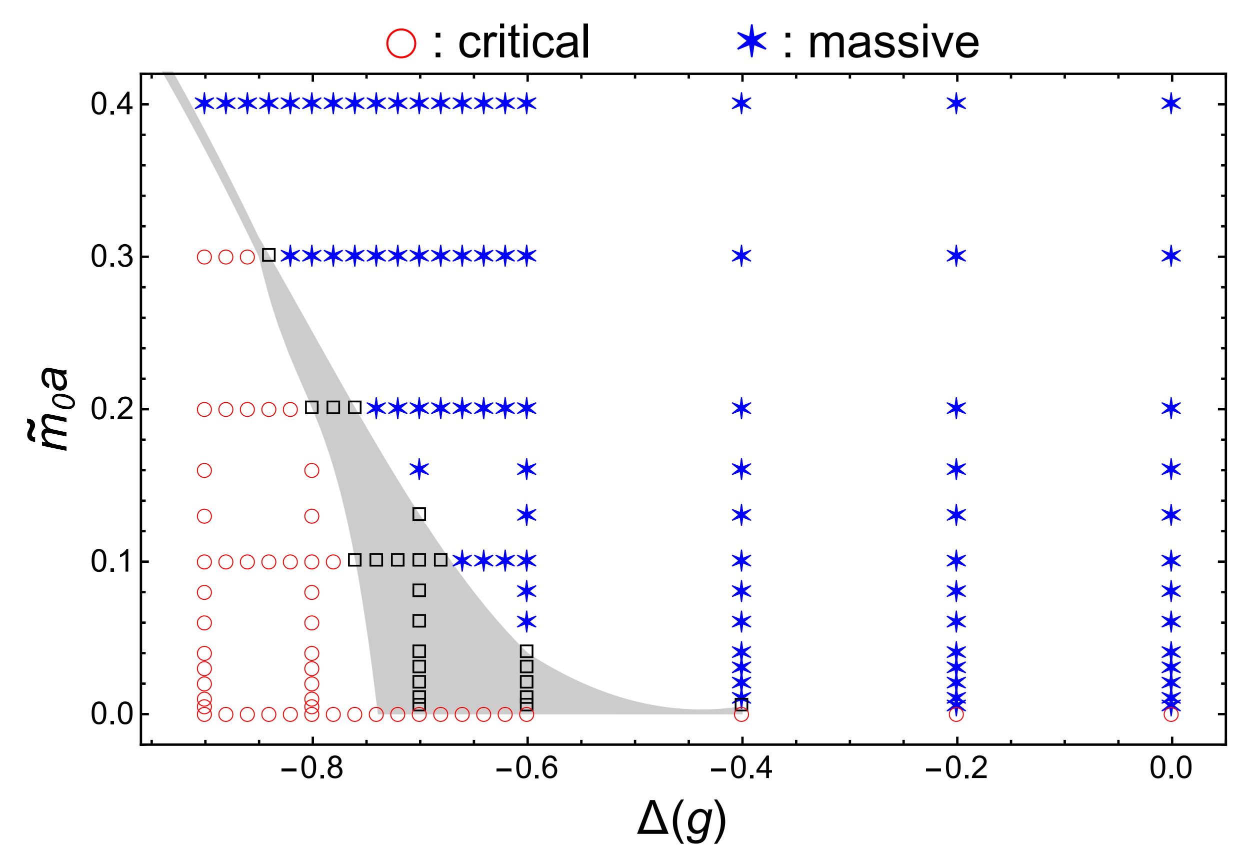
Example: (1+1)D Thirring Model

$$S_{\mathrm{Th}}[\psi,\bar{\psi}] = \int d^2x \left[\bar{\psi} i\gamma^\mu \partial_\mu \psi - m_0 \bar{\psi}\psi - \frac{g}{2}\left(\bar{\psi}\gamma_\mu\psi\right)^2\right]$$

$$\bar{H}_{XXZ} = \nu(g)\left[-\frac{1}{2}\sum_n^{N-2}\left(S_n^+ S_{n+1}^- + S_{n+1}^+ S_n^-\right) + a\tilde{m}_0 \sum_n^{N-1}(-1)^n\left(S_n^z + \frac{1}{2}\right) + \Delta(g)\sum_n^{N-1}\left(S_n^z + \frac{1}{2}\right)\left(S_{n+1}^z + \frac{1}{2}\right)\right]$$

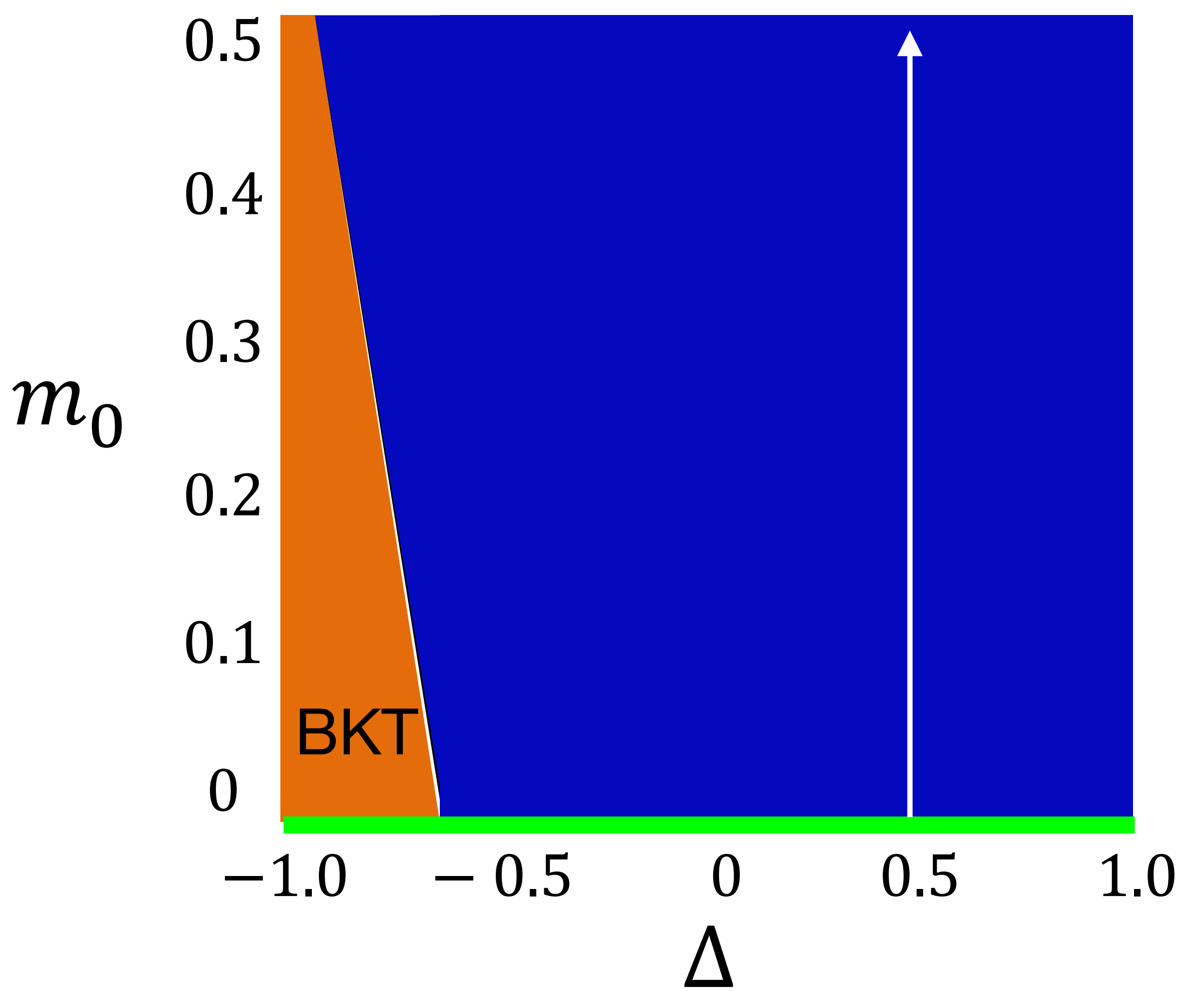
$$\nu(g) = \frac{2\gamma}{\pi\sin(\gamma)}, \tilde{m}_0 = \frac{m_0}{\nu(g)}, \Delta(g) = \cos(\gamma), \text{ with } \gamma = \frac{\pi - g}{2}$$



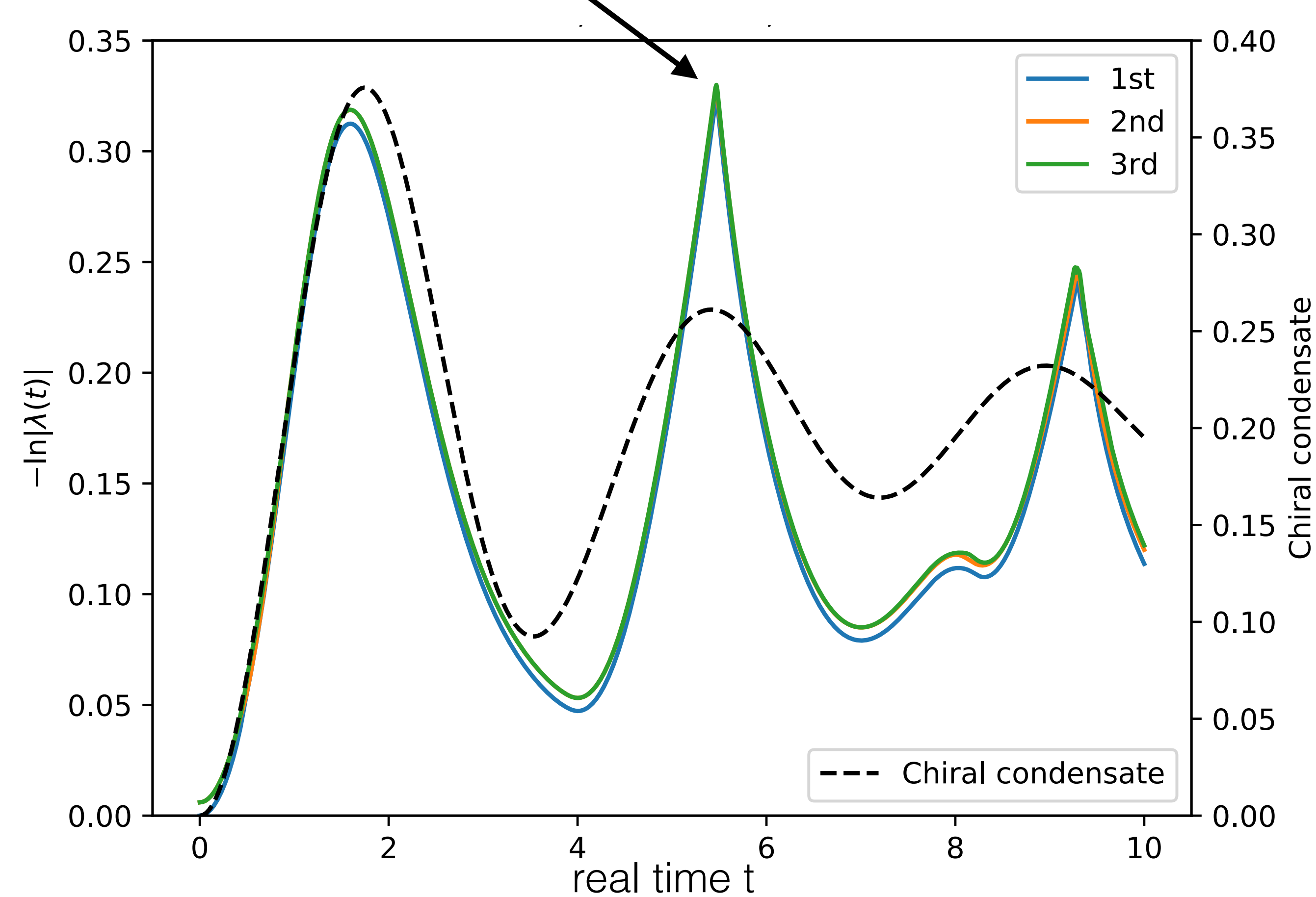


M.-C. Bañuls, K. Cichy, Y.-J. Kao, C.-J. D. Lin, Y.-P. Lin, D. T.-L. Tan arXiv preprint arXiv:1908.04536

Example: (1+1)D Thirring Model

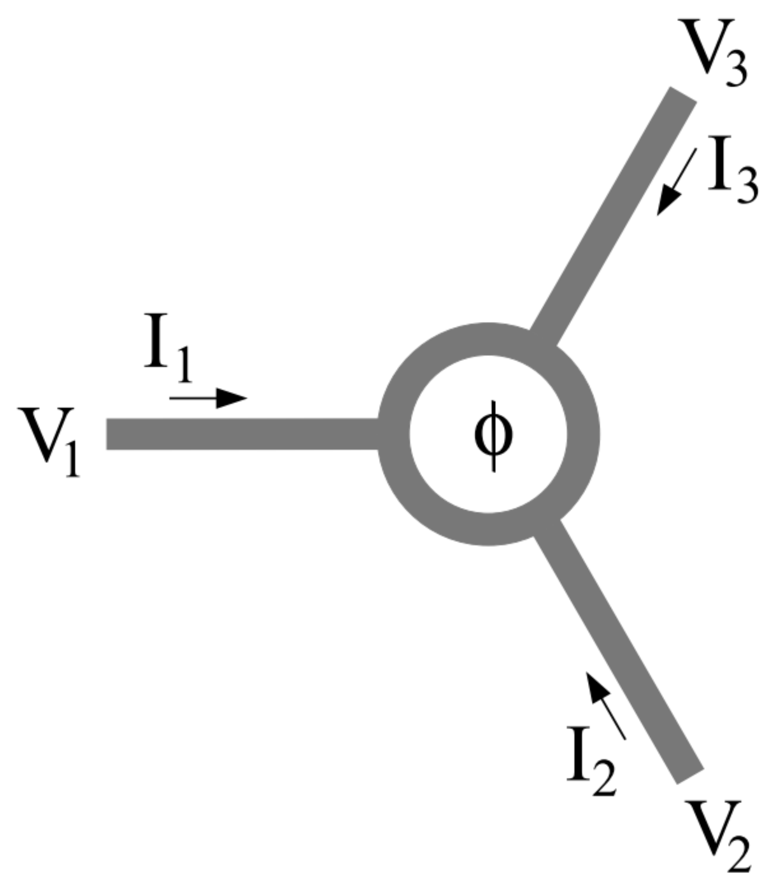


Dynamical phase transition



M.-C. Bañuls, K. Cichy, H.-T. Hung, Y.-J. Kao, C.-J. D. Lin, unpublished.

Example: Y-junction of TLL wires

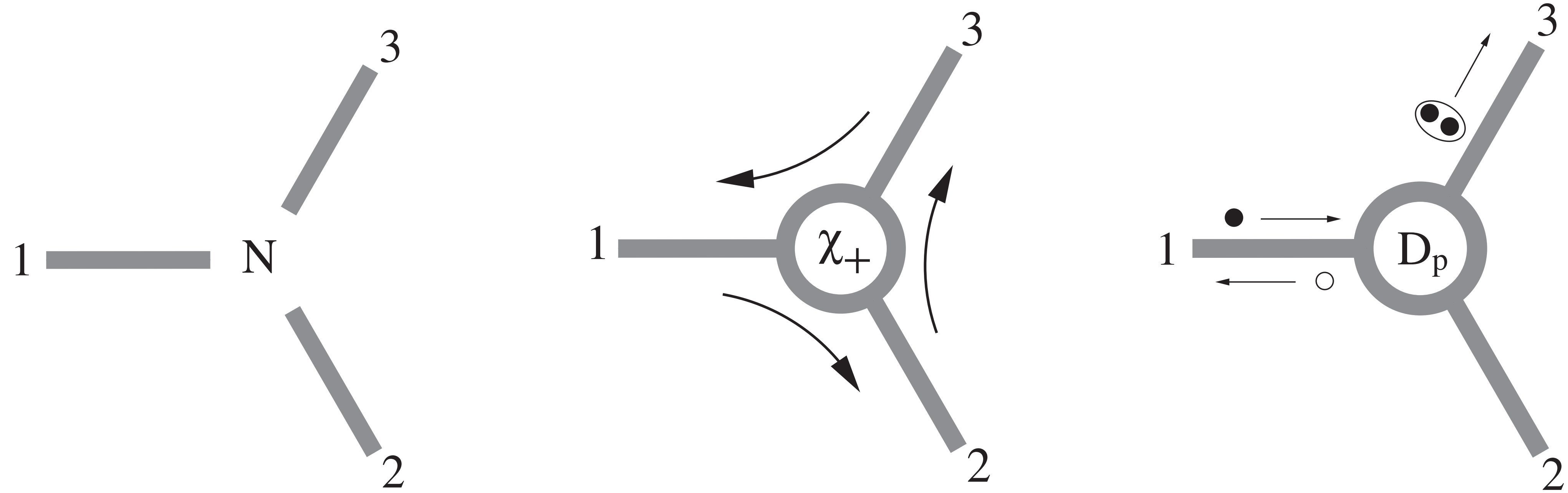


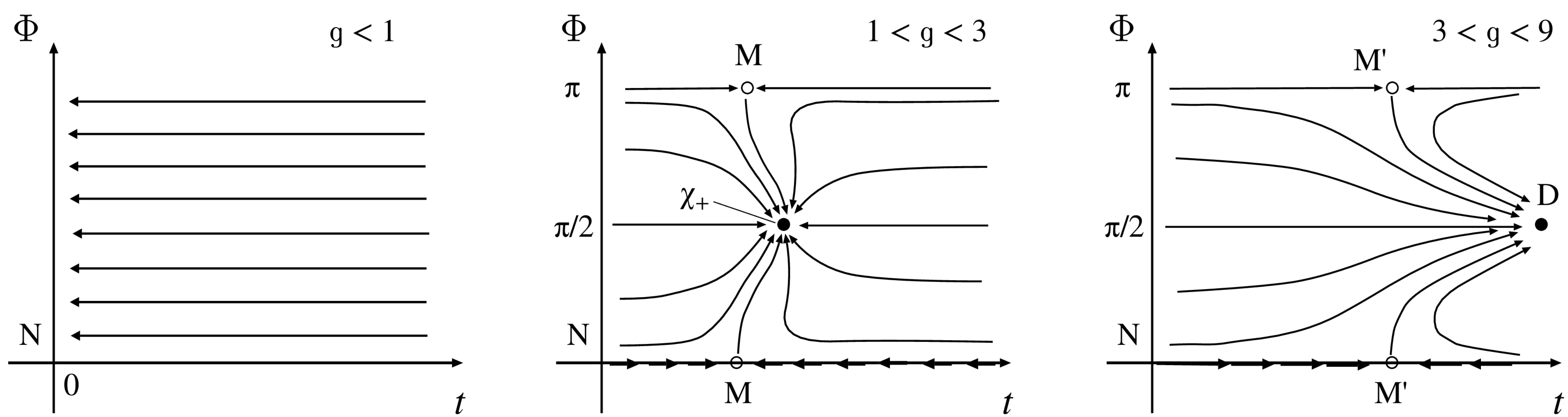
Oshikawa et al. J. Stat. Mech. (2006) P02008

- Y-junction of interacting quantum wires: Tomonaga-Luttinger Liquid wires
- RG fixed point determined by the interaction in the wires and flux in the junction
- DMRG+Infinite BC

Chung-Yo Luo, Yoshiki Fukusumi, Masaki Oshikawa,YJK and Pochung Chen, PRB **99**, 121103(R) (2019)

RG Fixed Points



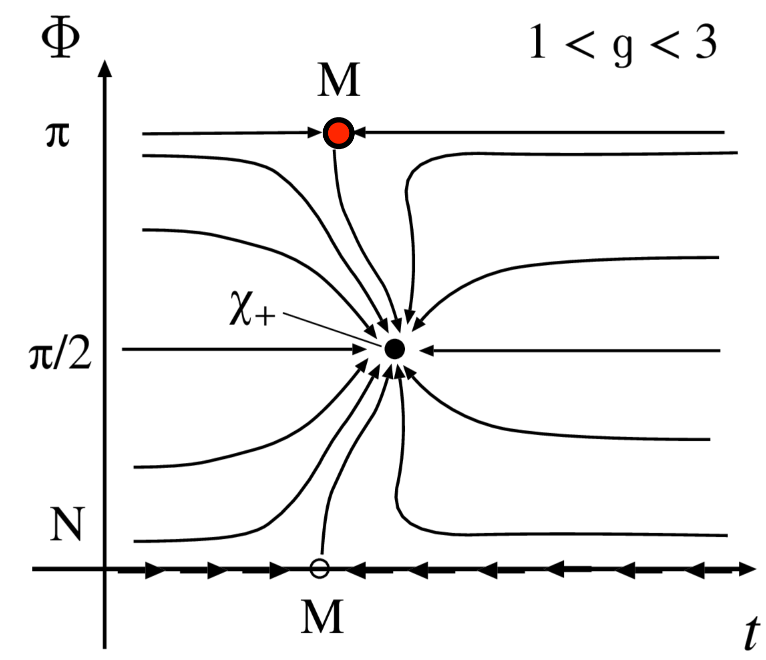


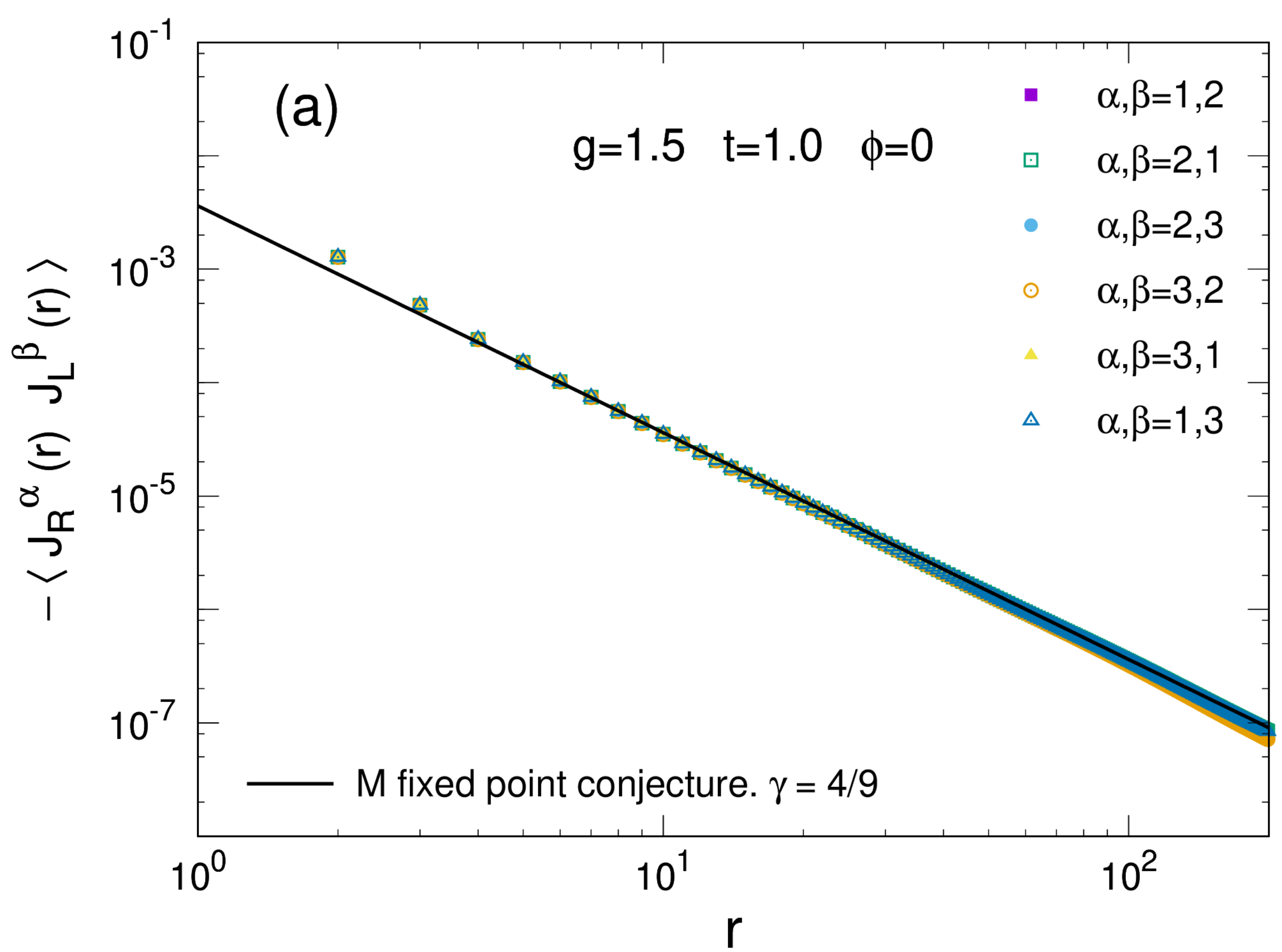
Oshikawa et al. J. Stat. Mech. (2006) P02008

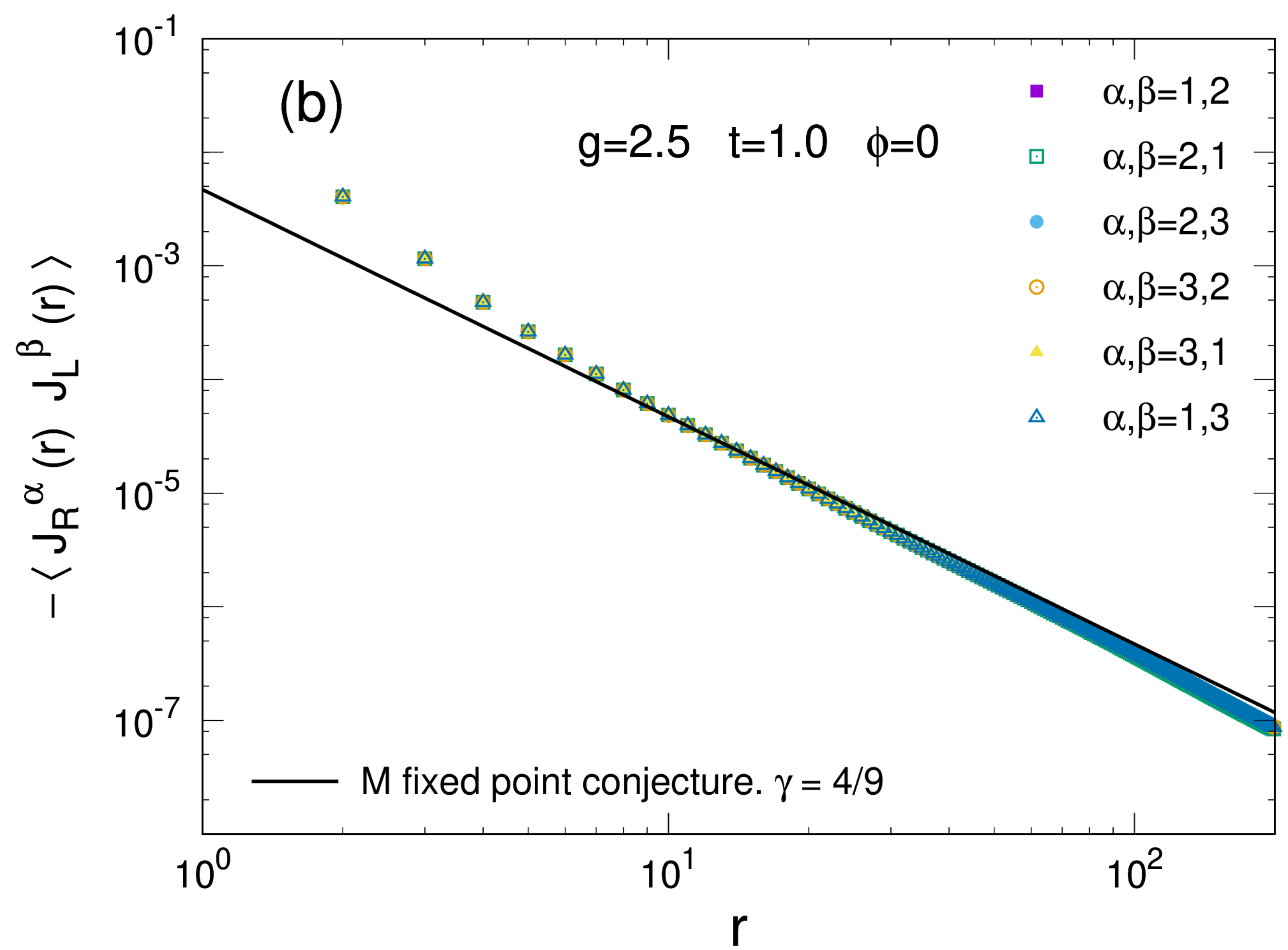
$1 < g < 3$: M **Fixed Point**

- Time-reversal symmetric unstable fixed point

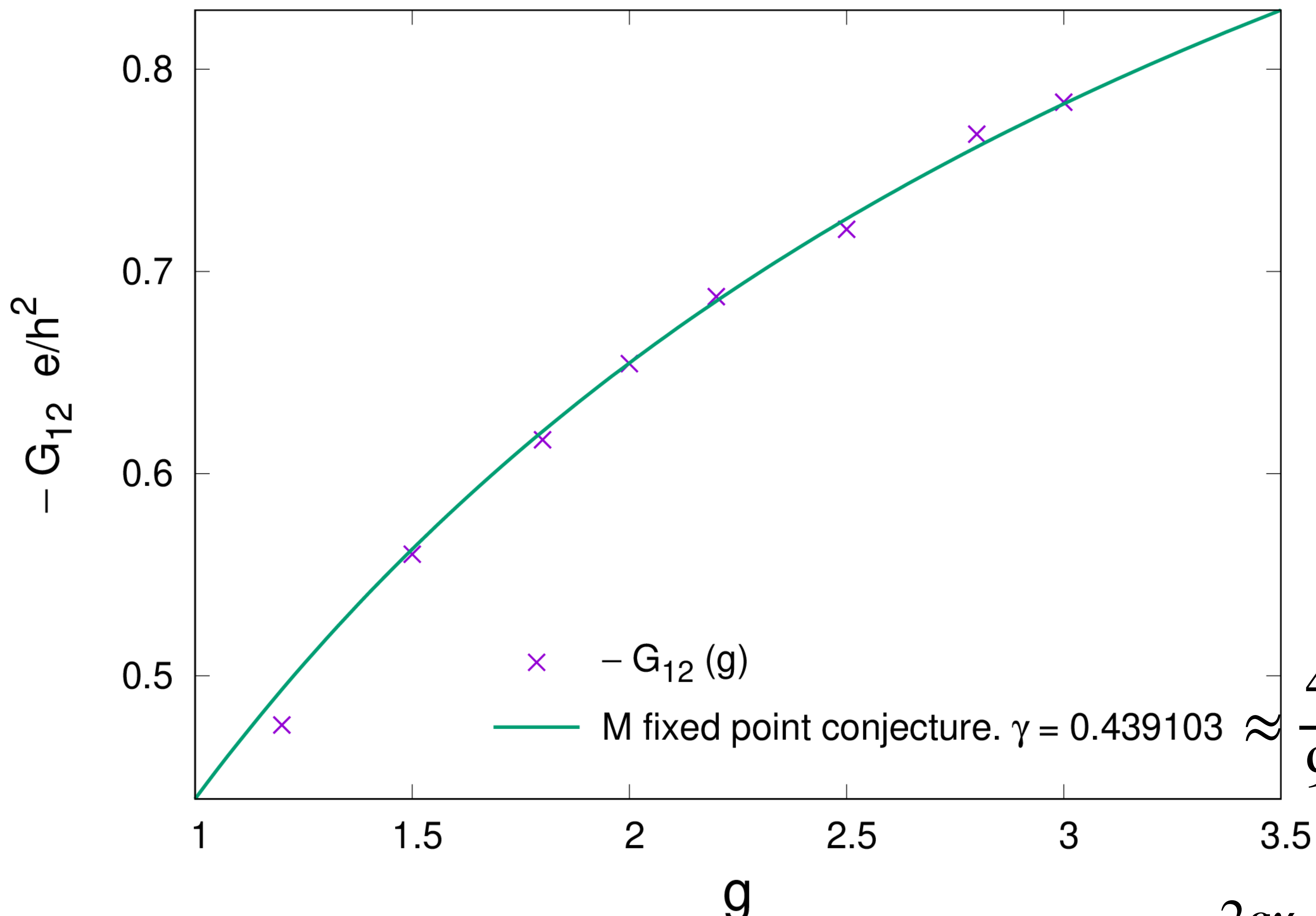
$$G^M_{\alpha\beta} = \frac{2g\gamma}{2g + 3\gamma - 3g\gamma}\frac{e^2}{h}, \gamma = \frac{4}{9}$$







$1 < g < 3$: M Fixed Point



$\approx \frac{4}{9}$

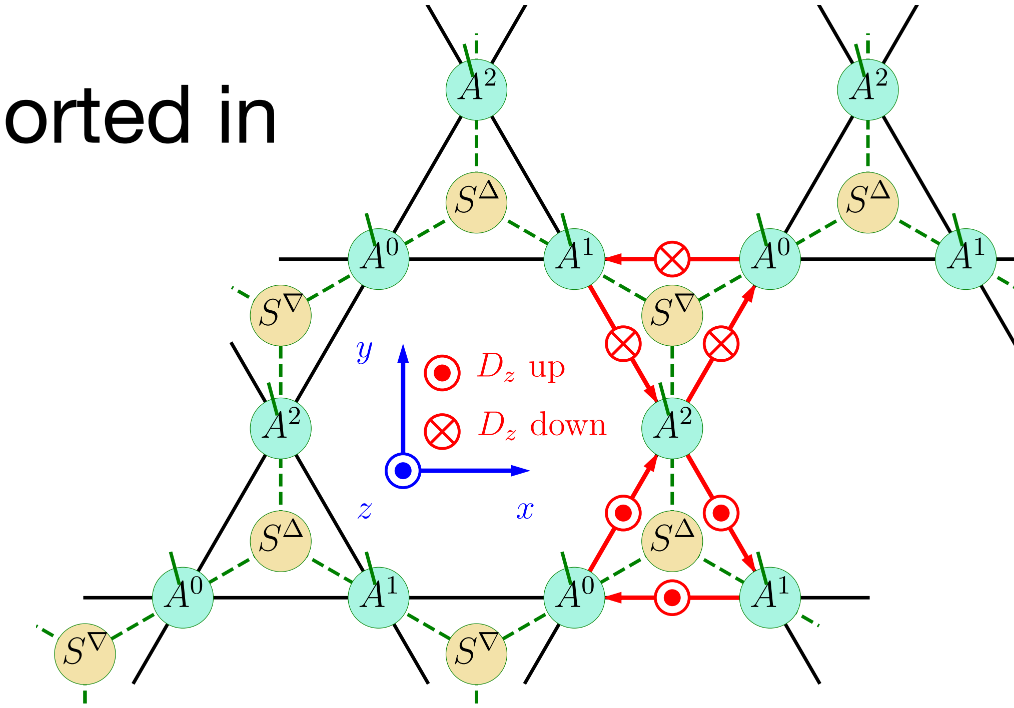
$$G^M_{\alpha\beta} = \frac{2g\gamma}{2g + 3\gamma - 3g\gamma}\frac{e^2}{h}, \gamma = \frac{4}{9}$$

Chung-Yo Luo, Masaki Oshikawa, YJK and Pochung Chen, unpublished.

Example: Kagome AFM+ DM interaction

$$H = \sum_{\langle i,j \rangle} \mathbf{S}_i \cdot \mathbf{S}_j + D\hat{z} \cdot (\mathbf{S}_i \times \mathbf{S}_j)$$

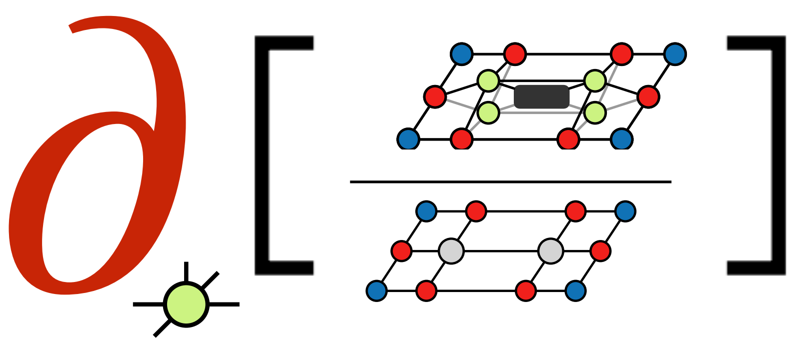
- Kagome AF Heisenberg model: Gapless spin liquid
- $D_z \approx 0.08J, D_\perp \approx 0.01J$ in Herbertsmithite
- Infinite Projected-Entangled Symplex State (iPESS)
- $D_c \approx 0.012(2)J$, spin liquid physics reported in Herbertsmithite needs to be reaccessed

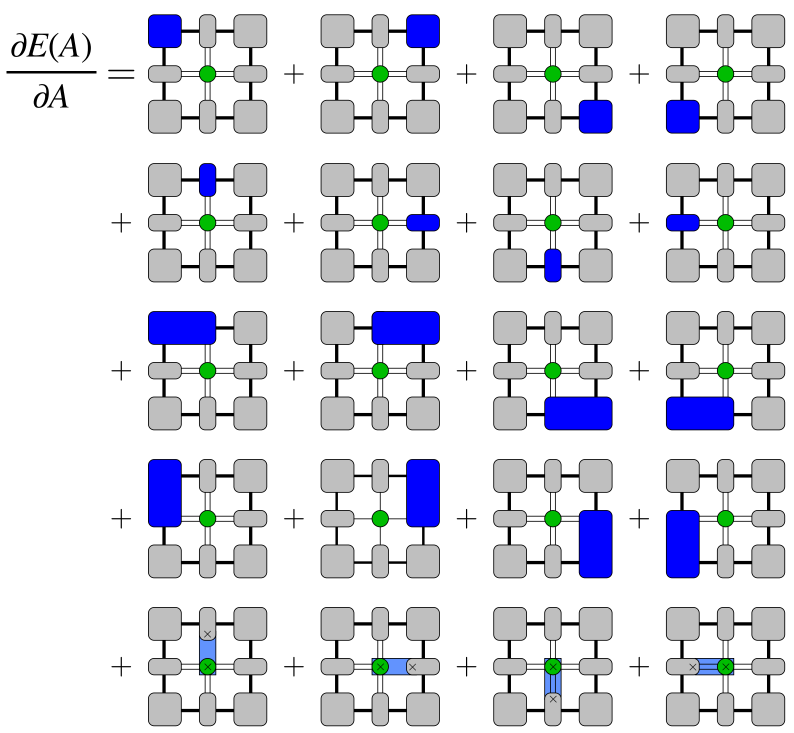


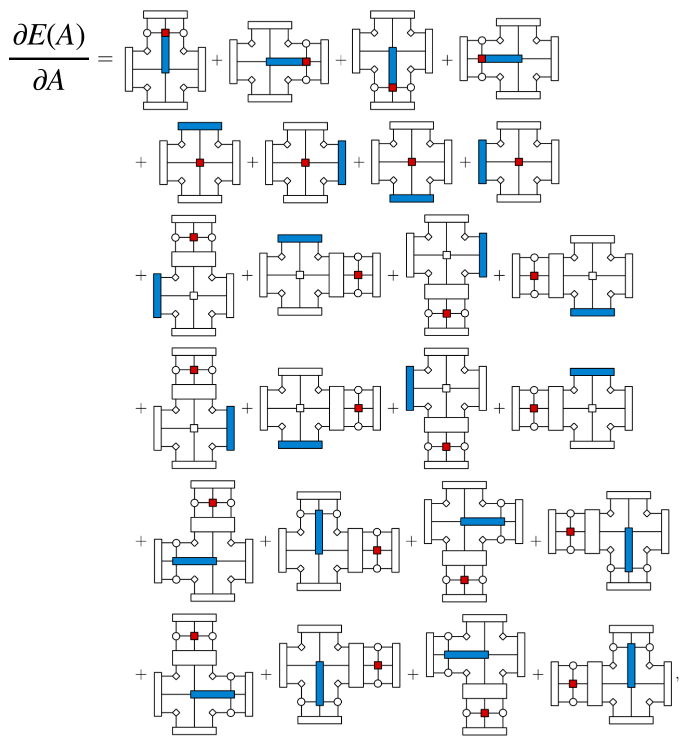
H. J. Liao, et al., Phys. Rev. Lett. 118, 137202 (2017).
C.-Y. Lee, B. Normand, YJK Phys. Rev. B 98, 224414 (2018)

Outlook: Learn from DL community

- Differentiable Programming



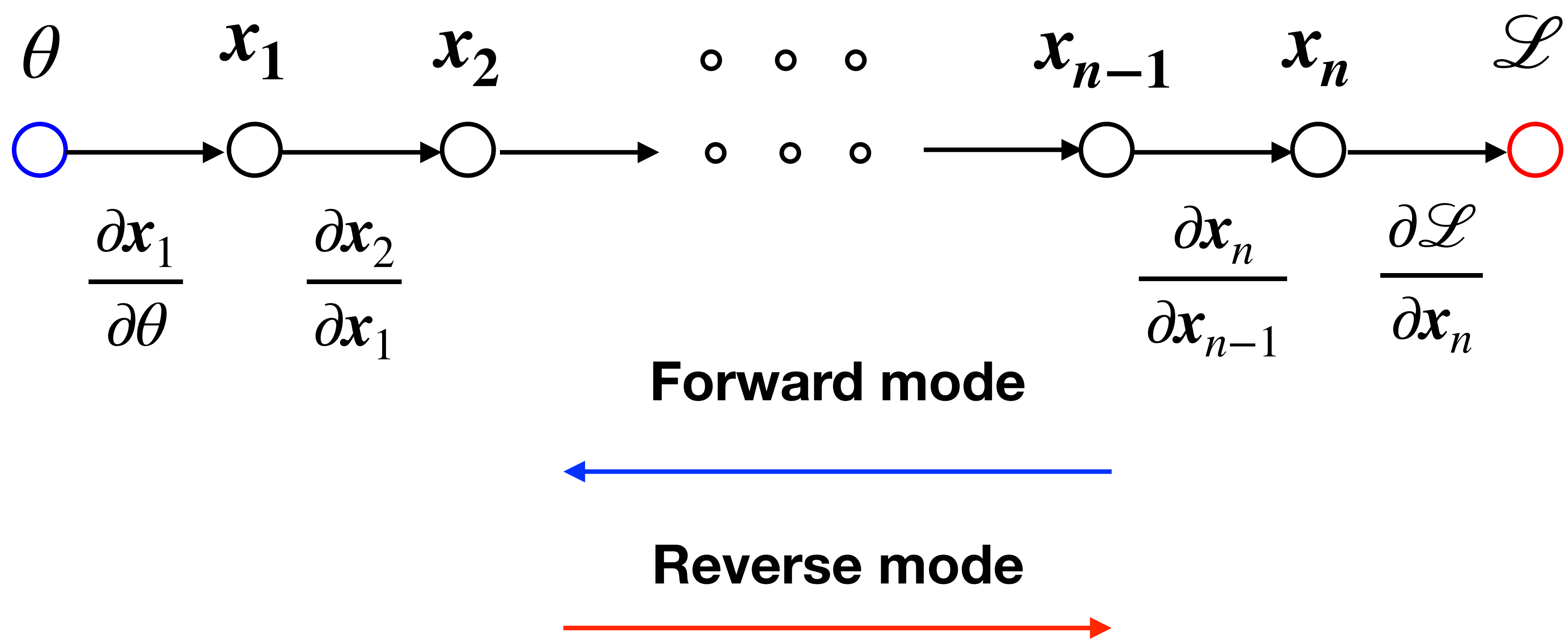




- Automatic differentiation! AutoGrad

Hai-Jun Liao, Jin-Guo Liu, Lei Wang, and Tao Xiang Phys. Rev. X 9, 031041

Outlook: Learn from DL community



$$\frac{\partial \mathscr{L}}{\partial \boldsymbol{\theta}} = \frac{\partial \boldsymbol{x}_1}{\partial \theta} \frac{\partial \boldsymbol{x}_2}{\partial \boldsymbol{x}_1} \cdots \frac{\partial \boldsymbol{x}_n}{\partial \boldsymbol{x}_{n-1}} \frac{\partial \mathscr{L}}{\partial \boldsymbol{x}_n}$$

- Automatic differentiation in DL (Tensorflow, PyTorch, Flux/ Zygote)

Hai-Jun Liao, Jin-Guo Liu, Lei Wang, and Tao Xiang Phys. Rev. X 9, 031041

Outlook: Bring TN computation to HPC

- Tensor network software
  - ITensor (C++, Julia) Abelian symmetry/GPU
  - mptoolkit (C++) non-Abelian symmetry/GPU
  - uni10 (C++/python) Abelian symmetry/GPU
  - TensorKit.jl (Julia) non-Abelian symmetry
  - mptensor (C++/python) non\-symmetric/HPC
  - TensorNetwork (python+ Tensorflow) non-symmetric/Cloud computing (CPU+GPU+TPU?)
  - Tor10 (python +PyTorch) symmetric/ML frame work (work in progress)
  - TensorNetworkAD.jl (Julia) Tensor Network with AD