Efficient simulations of low-dimensional systems

Frank Pollmann

Max Planck Institute for the Physics of Complex Systems

mpipks

Trieste, Sep. 14 2015

Efficient simulations of low-dimensional systems

Overview

(1) **Matrix-product states and probes for topological phases**

- Review: Entanglement and matrix-product states (MPS)
- MPS for infinite systems
- Extracting fingerprints of topological order

(2) **Efficient simulation of dynamical properties**

- Time-evolving block decimation (TEBD)
- Quench dynamics and entanglement growth
- MPO based time evolution

(3) **Tutorial: Hands on session**

(2) Efficient simulation of dynamical properties

$$i\hbar\frac{\partial}{\partial t}|\psi\rangle = H|\psi\rangle$$

Time evolution of MPS

- How to efficiently simulate the time evolution of MPS?

$$|\psi_t\rangle = \exp(-iHt)|\psi_{t=0}\rangle$$

— **Time evolving block decimation**
  [Vidal '03]

— Time dependent DMRG
  [White & Feiguin '04, Daley et al. '04,…]

— Krylov space based methods
  [Schmitteckert '04,…]

— Time dependent variational principle
  [Haegemann et al. '11 / '15]

— **Matrix-product operator based time evolutions**
  [Zaletel et al. '15]

Time evolving block decimation

Time evolving block decimation

- Assume we have a Hamiltonian of the form

$$H = \sum_j h^{[j,j+1]}$$

- Time evolution in real time

$$|\psi_t\rangle = \exp(-iHt)|\psi_{t=0}\rangle$$

- Time evolution in imaginary time

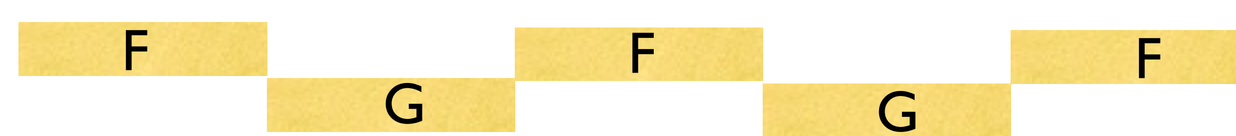
$$|\psi_0\rangle = \lim_{\tau\to\infty} \frac{\exp(-H\tau)|\psi_i\rangle}{||\exp(-H\tau)|\psi_i\rangle||}$$

Time evolving block decimation

- Consider the Hamiltonian $H = \sum_j h^{[j,j+1]}$
- Decompose the Hamiltonian as $\mathrm{H{=}F{+}G}$

$$F \equiv \sum_{\text{even } j} F^{[j]} \equiv \sum_{\text{even } j} h^{[j,j+1]}$$

$$G \equiv \sum_{\text{odd } j} G^{[j]} \equiv \sum_{\text{odd } j} h^{[j,j+1]}$$



- We observe $[F^{[r]}, F^{[r']}] = 0$ $([G^{[r]}, G^{[r']}] = 0)$
but $[G, F] \neq 0$

Time evolving block decimation

- Apply Suzuki-Trotter decomposition of order p

$$\exp\left(-i(F+G)\delta t\right) \approx f_p\left[\exp(-F\delta t), \exp(-G\delta t)\right]$$

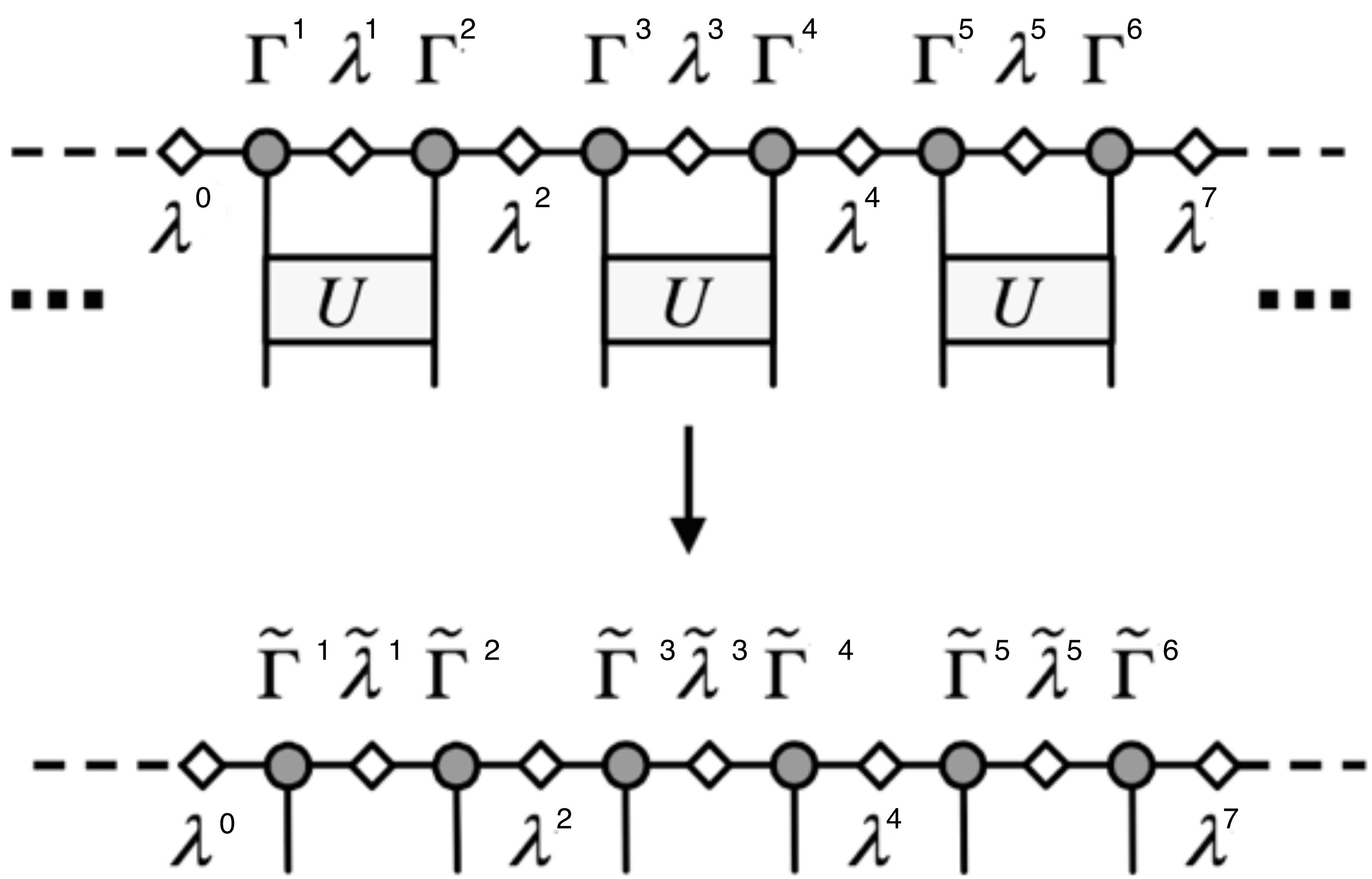
with $f_1(x,y) = xy$, $f_2(x,y) = x^{1/2} y x^{1/2}$, etc.

- Two chains of two-site gates

$$\begin{aligned} U_F &= \prod_{\text{even } r} \exp(-iF^{[r]}\delta t) \\ U_G &= \prod_{\text{odd } r} \exp(-iG^{[r]}\delta t) \end{aligned}$$

Time evolving block decimation

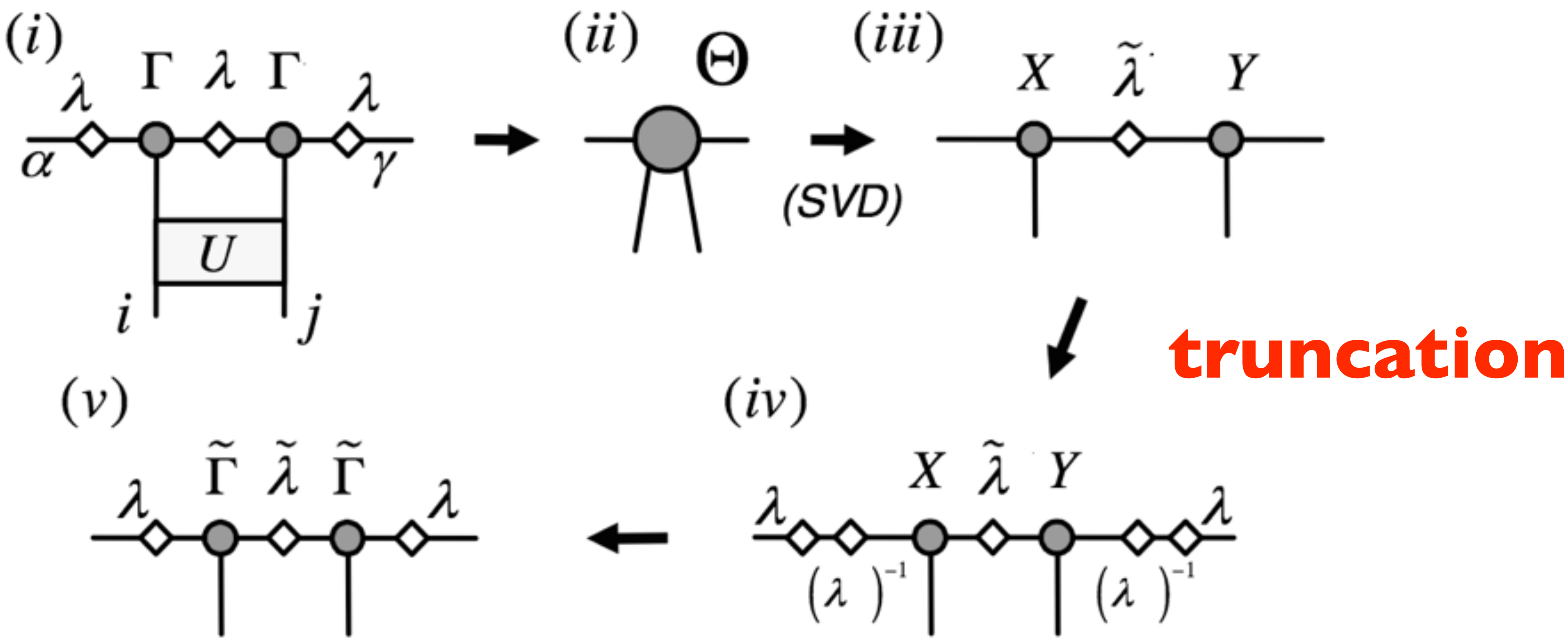
- Time Evolving Block Decimation algorithm (TEBD)



- How do we get the original form back?

Time evolving block decimation

- Time Evolving Block Decimation algorithm (TEBD)



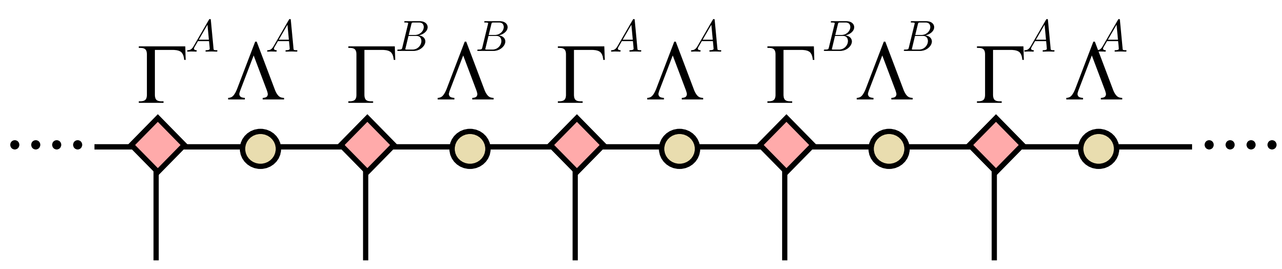
- Scales with the matrix dimension as χ^3

Time evolving block decimation

- Assume that $|\psi\rangle$ is translational invariant and $N = \infty$: infinite Time Evolving Block Decimation algorithm (iTEBD)

- Partially break translational symmetry to simulate the action of the gates

$$\Gamma^{[2r]} = \Gamma^A,\ \lambda^{[2r]} = \lambda^A,\ \Gamma^{[2r+1]} = \Gamma^B,\ \lambda^{[2r+1]} = \lambda^B$$



- Time evolution achieved by repeated local application of gates (parallel)

Time evolving block decimation

- Python + numpy provide useful tools to simply implement the algorithm as key functions are already implemented

```
X=tensordot(Y,Z,axes=(1,0))
```

$$X_{ijk} = \sum_m Y_{im} Z_{mjk}$$

```
X=reshape(X,(dim1*dim2,dim3))
```

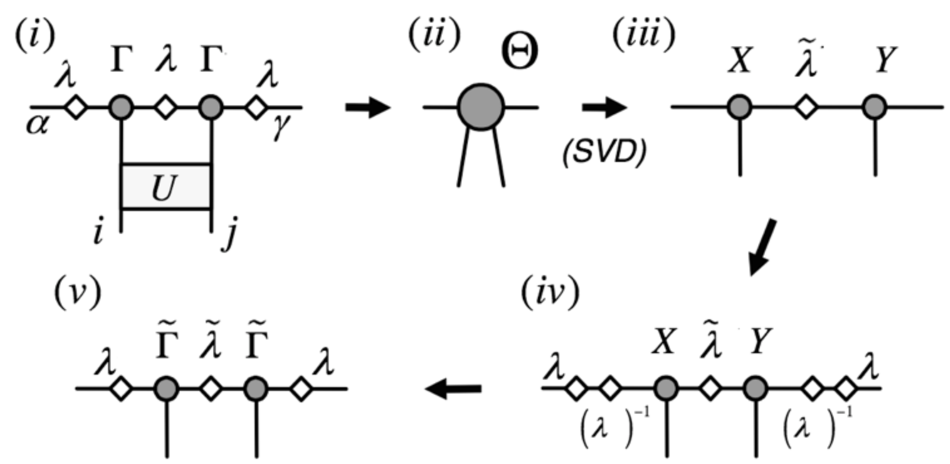
$$X_{ijk} \to X_{(ij)k}$$

```
X=transpose(X,(0,2,1))
```

$$X_{ijk} \to X_{ikj}$$

Time evolving block decimation

```
# First define the parameters of the model / simulation
J=1.0; g=0.5; chi=5; d=2; delta=0.01; N=1000;
G = np.random.rand(2,d,chi,chi);l = np.random.rand(2,chi)

# Generate the two-site time evolution operator
H = np.array( [[J,-g/2,-g/2,0], [-g/2,-J,0,-g/2], [-g/2,0,-J,-g/2], [0,-g/2,-g/2,J]] )
U = np.reshape(expm(-delta*H),(2,2,2,2))

# Perform the imaginary time evolution alternating on A and B bonds
for step in range(0, N):
    A = np.mod(step,2); B = np.mod(step+1,2)

    # Construct theta
    theta = np.tensordot(np.diag(l[B,:]),G[A,:,:,:],axes=(1,1))
    theta = np.tensordot(theta,np.diag(l[A,:],0),axes=(2,0))
    theta = np.tensordot(theta,G[B,:,:,:],axes=(2,1))
    theta = np.tensordot(theta,np.diag(l[B,:],0),axes=(3,0))

    # Apply U
    theta = np.tensordot(theta,U,axes=([1,2],[0,1]))

    # SVD
    theta = np.reshape(np.transpose(theta,(2,0,3,1)),(d*chi,d*chi))
    X, Y, Z = np.linalg.svd(theta); Z = Z.T

    # Truncate
    l[A,0:chi]=Y[0:chi]/np.sqrt(sum(Y[0:chi]**2))

    X=np.reshape(X[0:d*chi,0:chi],(d,chi,chi))
    G[A,:,:,:]=np.transpose(np.tensordot(np.diag(l[B,:]**(-1)),X,axes=(1,1)),(1,0,2))

    Z=np.transpose(np.reshape(Z[0:d*chi,0:chi],(d,chi,chi)),(0,2,1))
    G[B,:,:,:]=np.tensordot(Z,np.diag(l[B,:]**(-1)),axes=(2,0))

print "E_iTEBD =", -np.log(np.sum(theta**2))/delta/2
```

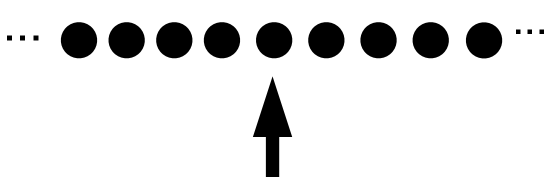


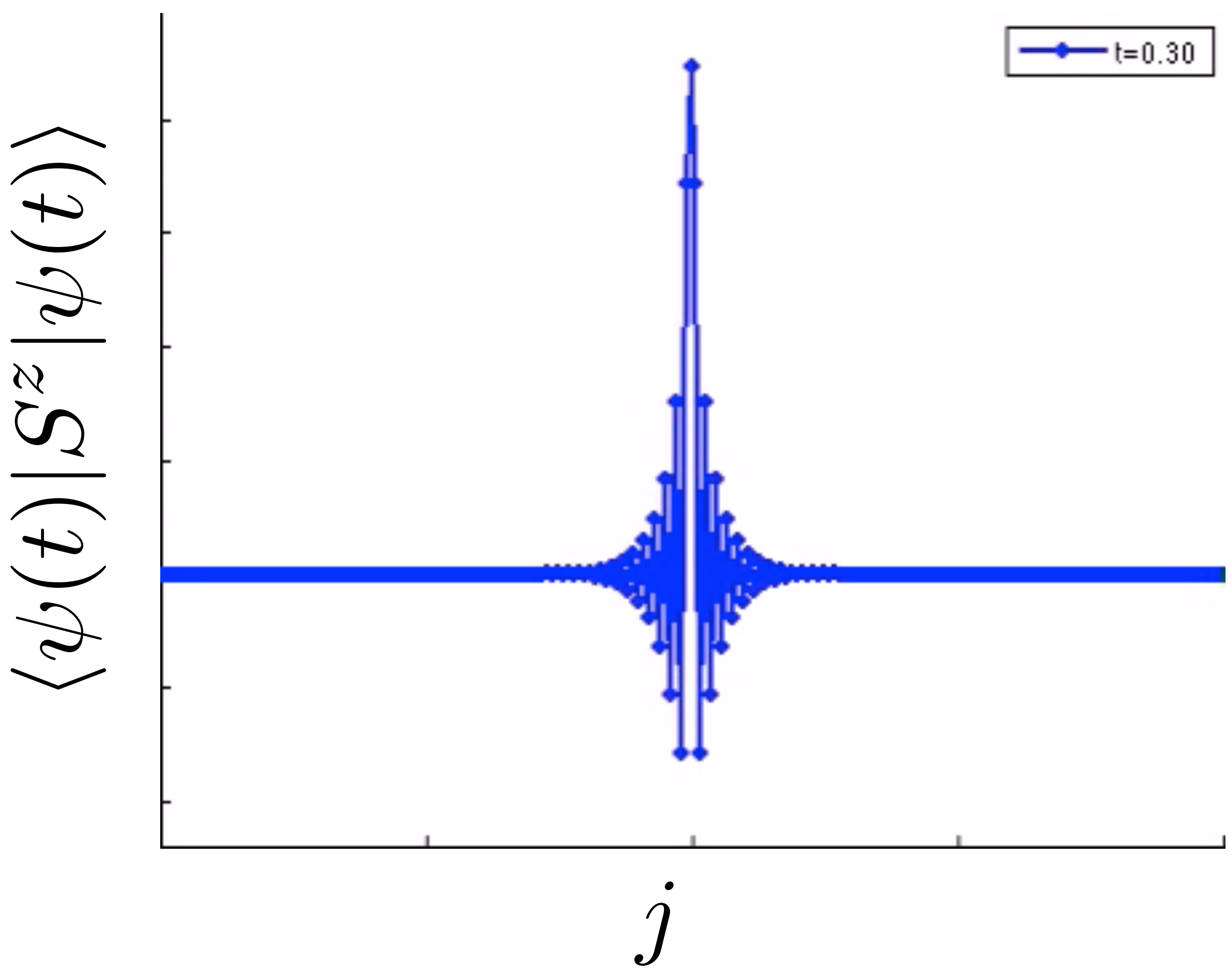
Quench dynamics and entanglement growth

Dynamical Response

- Spin-1 Heisenberg model: $H = \sum_j \vec{S}_j \cdot \vec{S}_{j+1}$

- Time evolution of $S^+_{j_0}|\psi_0\rangle$



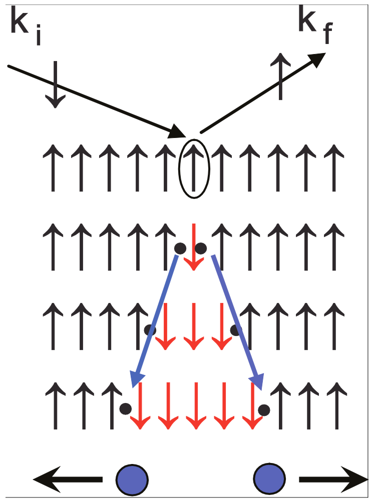


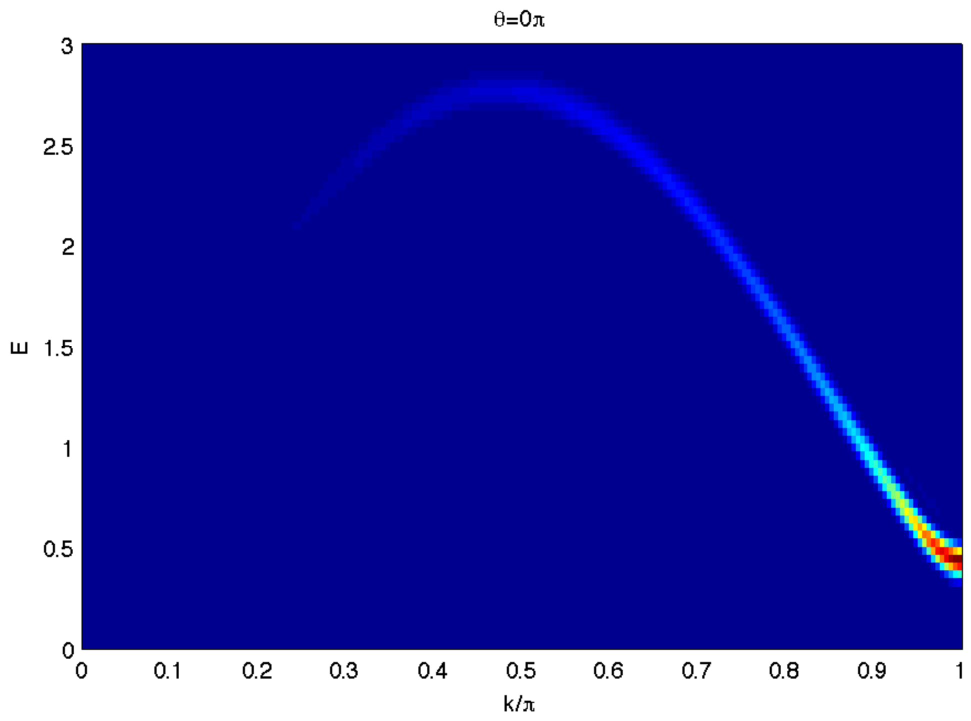
Dynamical Response

- Dynamical structure factor $S(k,\omega)$

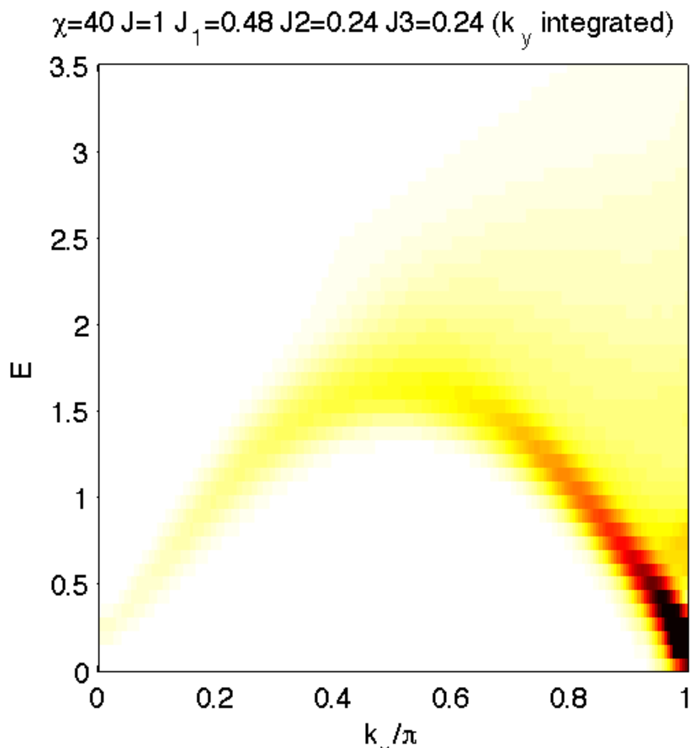
$$C(x,t) = \langle\psi_0|S_x^-(t)S_0^+(0)|\psi_0\rangle$$

$$S(k,\omega) = \sum_x \int_{-\infty}^{\infty} dt e^{-i(kx+\omega t)} C(x,t)$$





Spin-1 Heisenberg



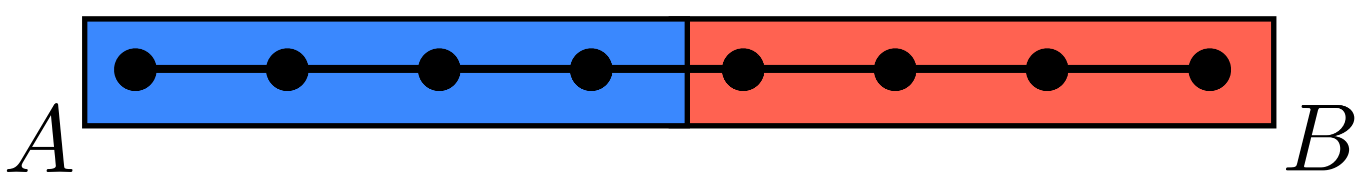
Spin-1/2 Ladder

Global Quenches

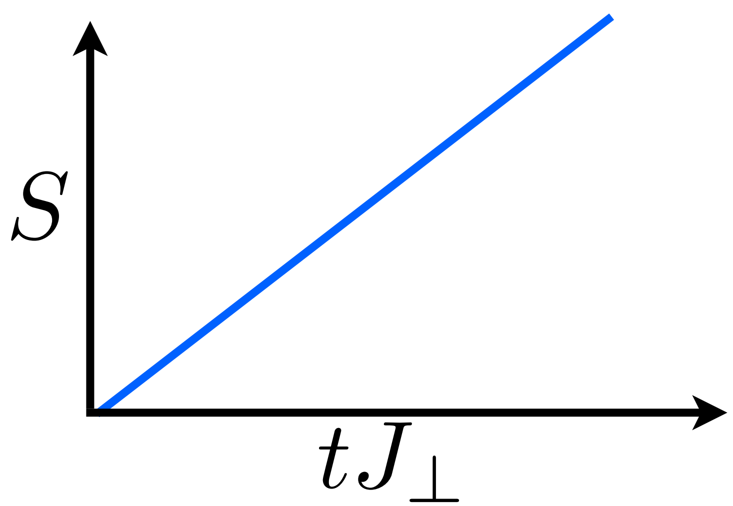
- Start from an unentangled product state ($S = 0$)

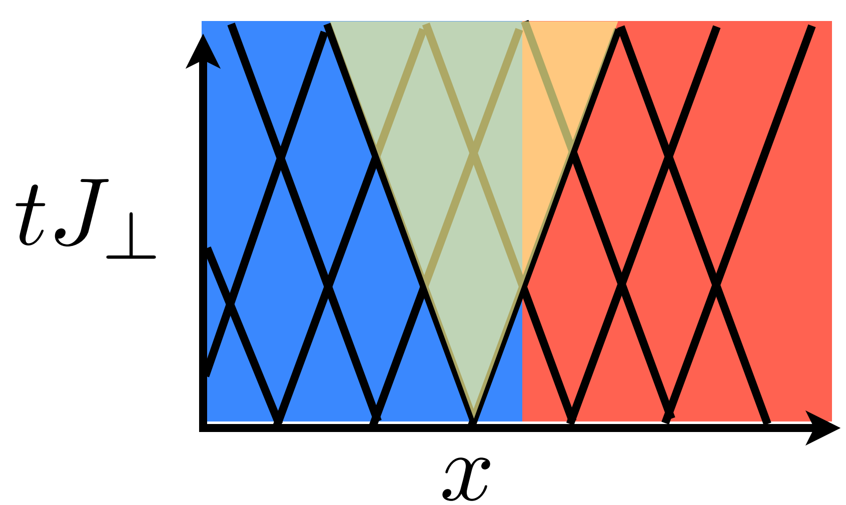
$$|\psi_0\rangle = |\uparrow\downarrow\uparrow\downarrow\uparrow\downarrow\uparrow\downarrow\uparrow\downarrow\uparrow\downarrow\uparrow\downarrow\uparrow\downarrow\uparrow\downarrow\uparrow\downarrow\rangle$$

- Measure the entanglement after quench and the time evolution with $U(t) = e^{-itH}$



- Time evolution with a Heisenberg Hamiltonian:

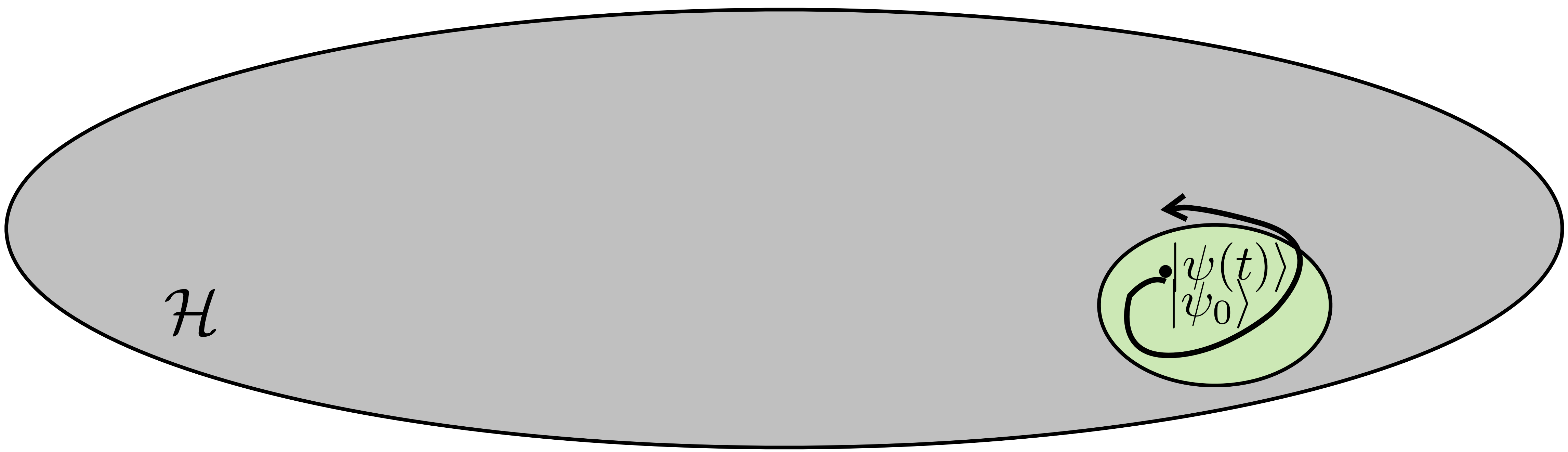




Lieb and Robinson (1972)
P. Calabrese and J. Cardy (2006)

Global Quenches

- Quickly leaving the comfort zone:
  Exponential growth of the bond dimension!



- Only short times can be simulated!

Hands on session!

MPO based time evolution

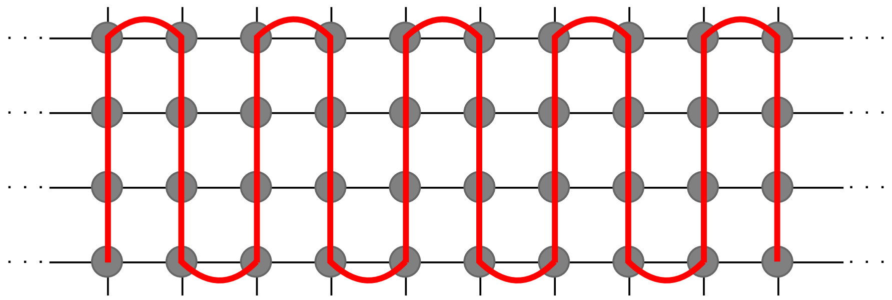
MPO based time evolution

- Desirable to have a method that can be…

  (i) … applied to any long-ranged Hamiltonian

  (ii) … applied to an infinitely long system

  (iii)… easily implemented



MPO based time evolution

- Hamiltonian expressed as a sum of terms $H = \sum_x H_x$
  Expand $U = \exp(-itH)$ for $t \ll 1$:

$$\underbrace{1 + t\sum_x H_x}_{\epsilon \sim \underline{L^2} t^2} \to \underbrace{\prod_x (1 + tH_x)}_{\epsilon \sim \underline{L} t^2}$$

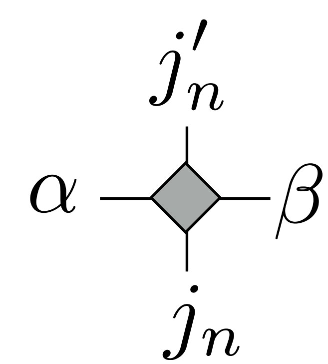
Neglect overlapping terms in expansion

$$\approx 1 + t\sum_x H_x + t^2 \sum_{x<y} H_x H_y$$

$$+ t^3 \sum_{x<y<z} H_x H_y H_z + \ldots$$

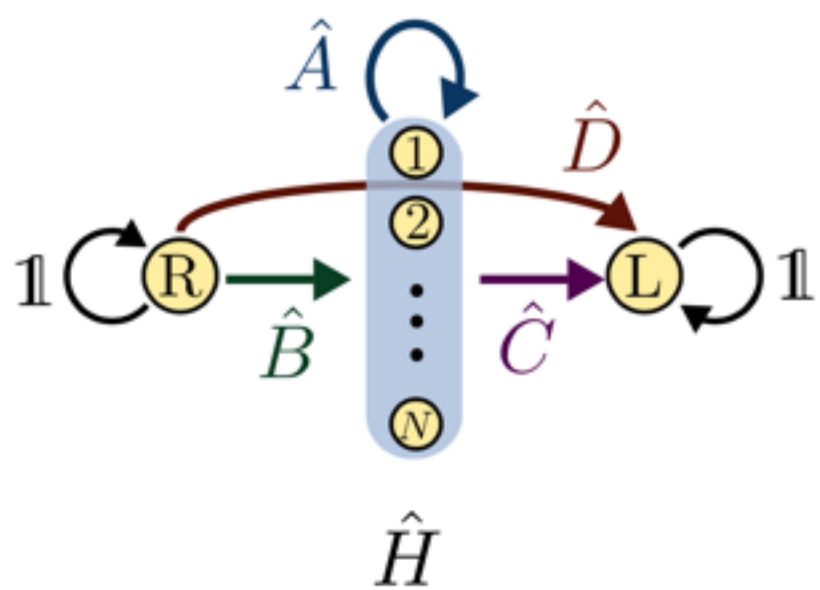
Compact matrix product operator representation

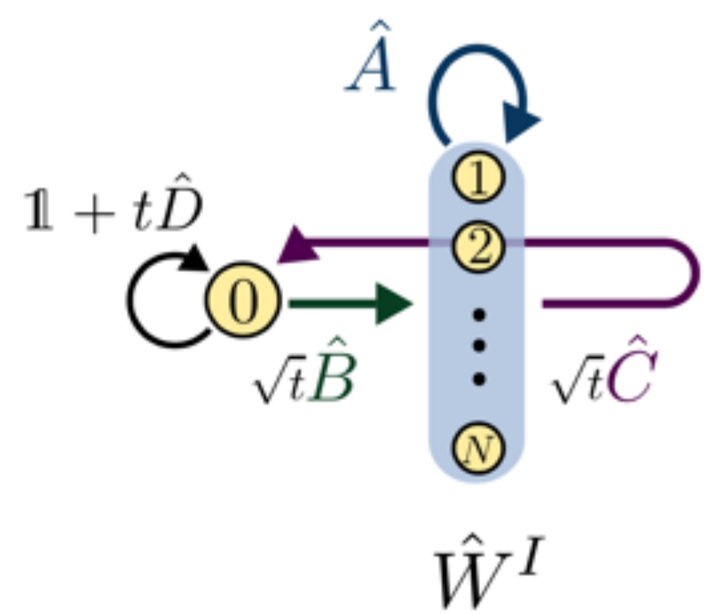
$$W^{[n] j_n j'_n}_{\alpha\beta} =$$



MPO based time evolution

- For experts on matrix product operators….



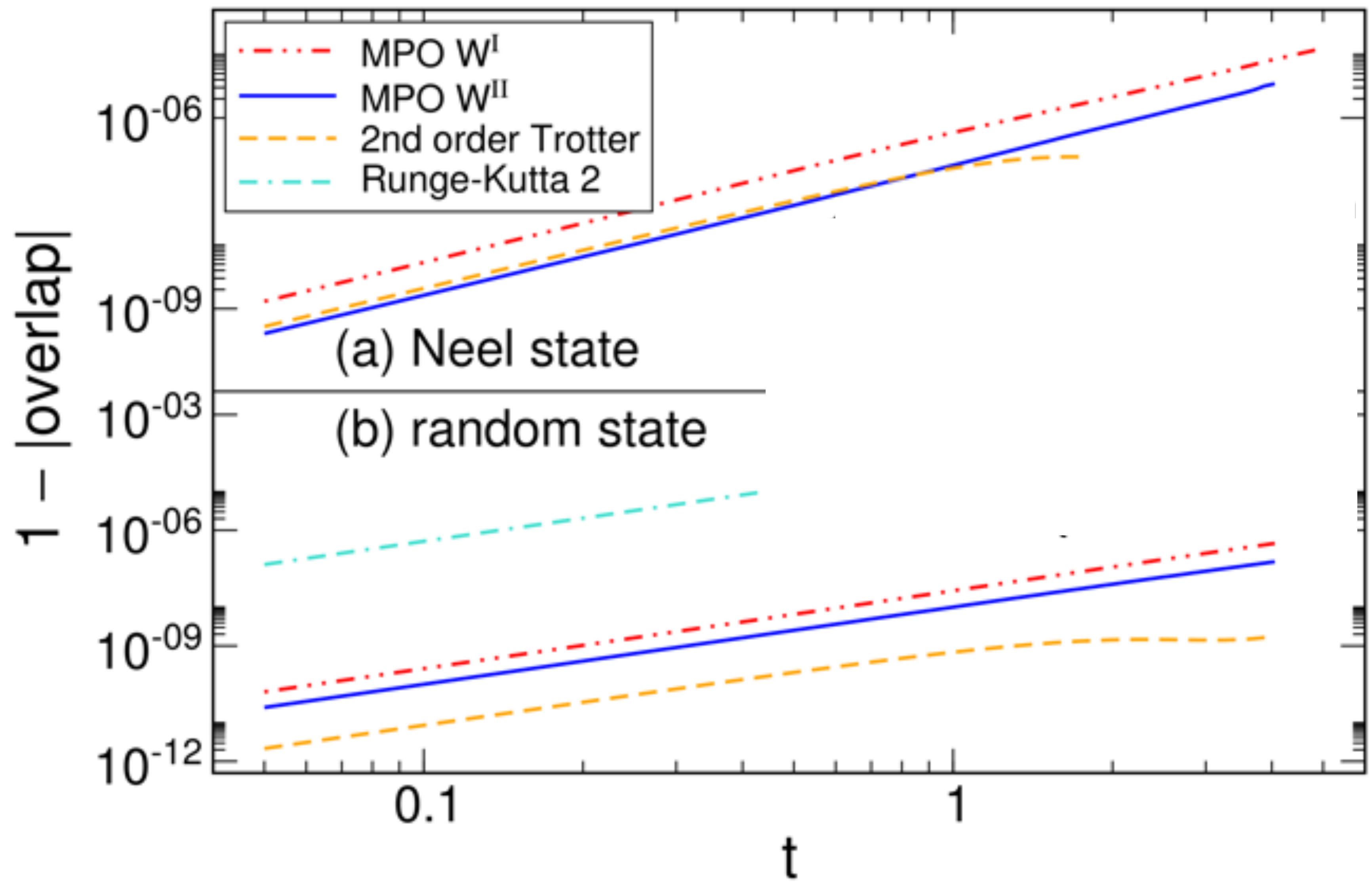


D dimensional
Hamiltonian MPO

$D-1$ dimensional
time evolution MPO

MPO based time evolution

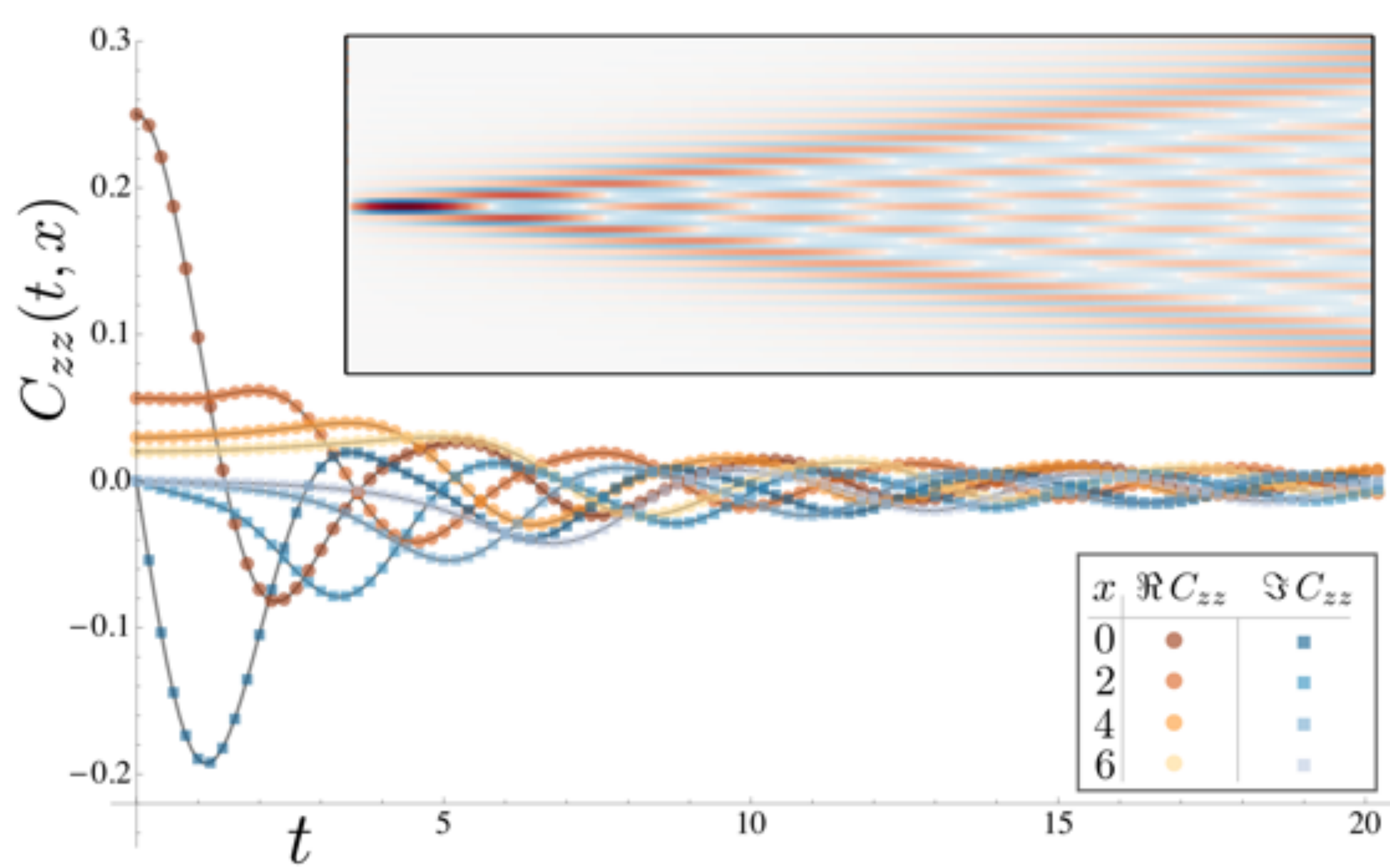
- Quench in the spin-1/2 Heisenberg chain



MPO based time evolution

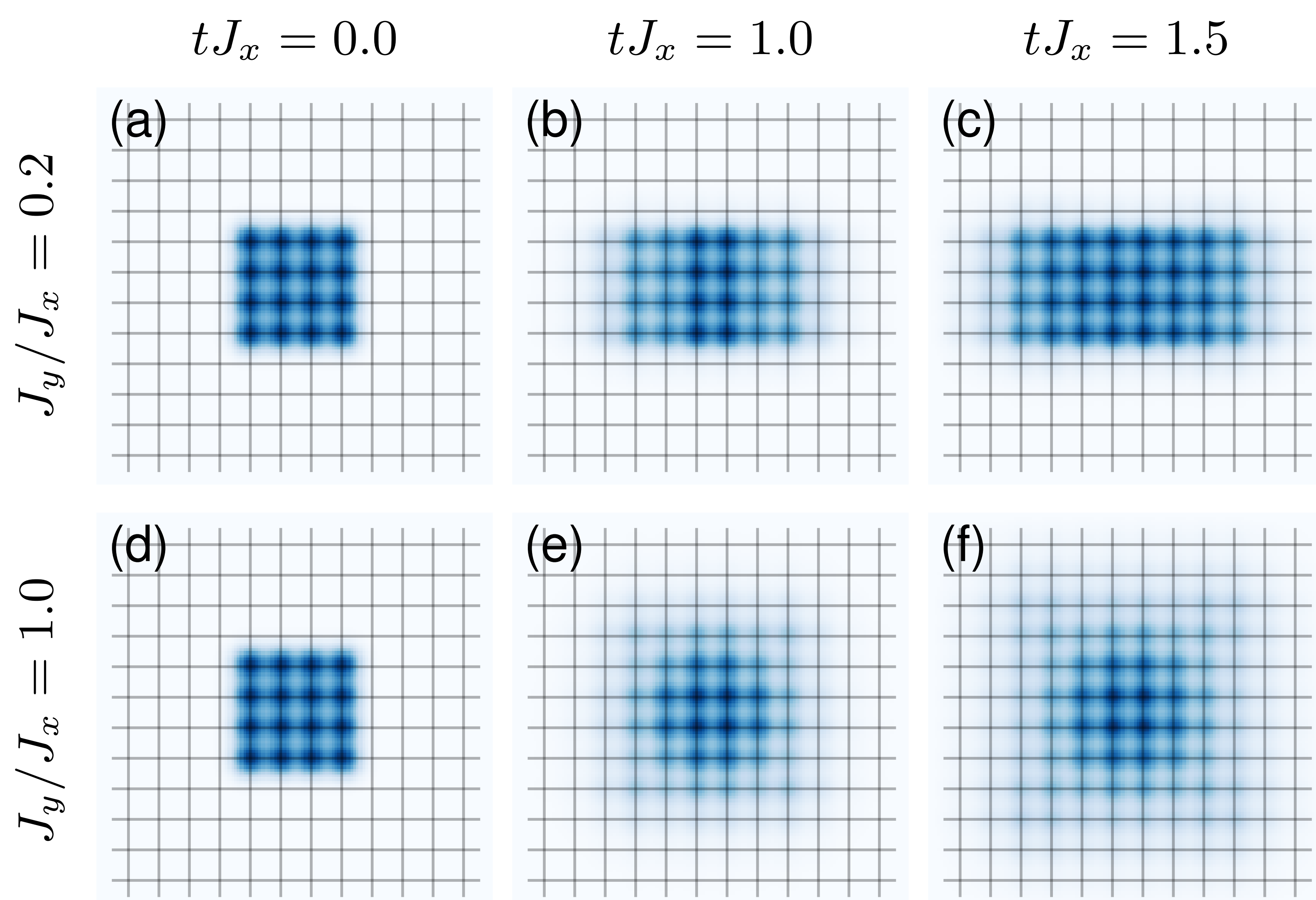
- Dynamical correlation functions in the **Haldane Shastry** model [Haldane & Zirnbauer '93]

$$H_{\mathrm{HS}} = \sum_{x,r>0} \frac{\mathbf{S}_x \cdot \mathbf{S}_{x+r}}{r^2}.$$



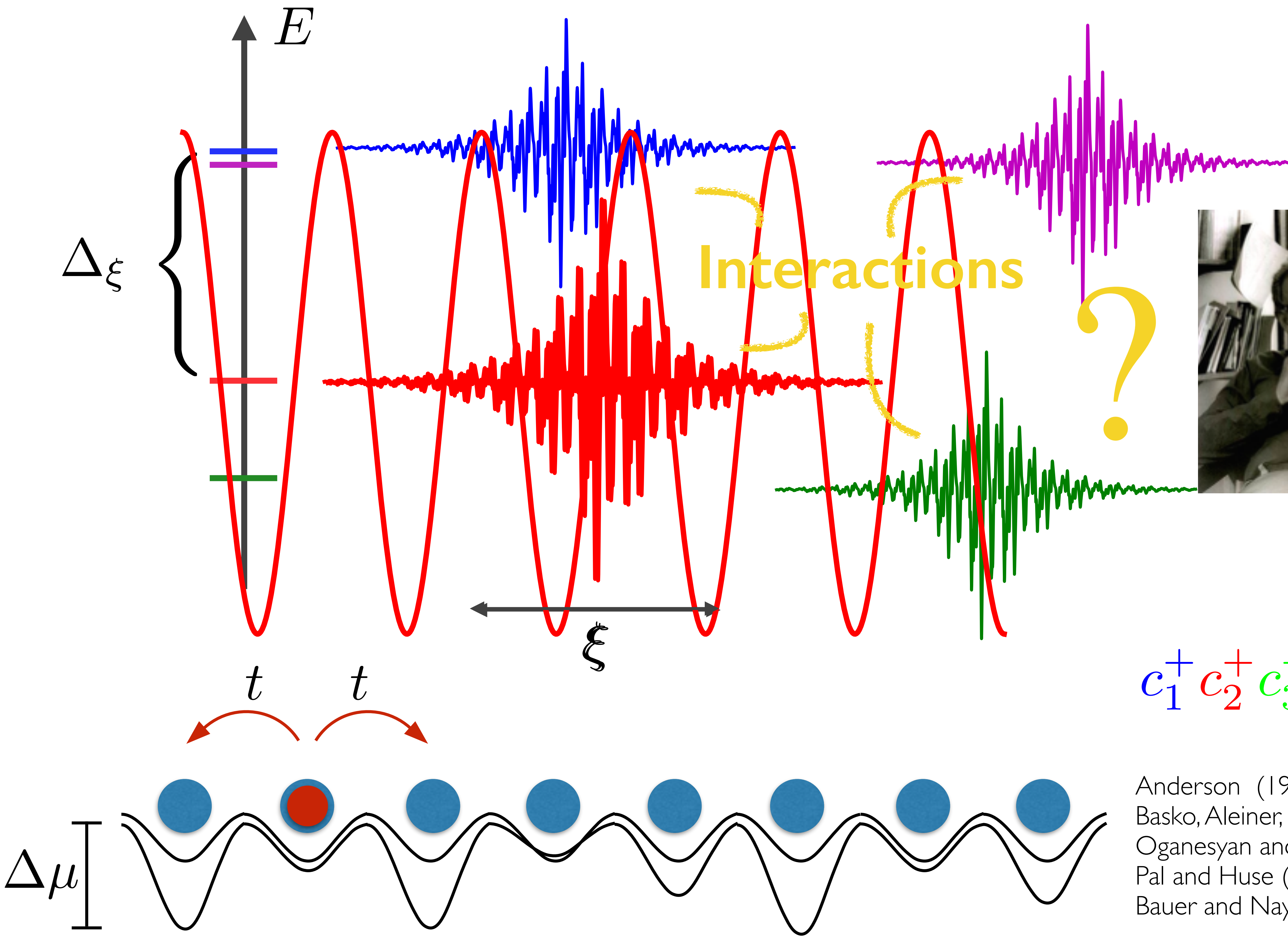
MPO based time evolution

- Expansion of bosonic clouds in 2D [Hauschild et al. '15]



Many-body localization

Many-body localization



$c_1^\dagger c_2^\dagger c_3^\dagger c_4^\dagger |0\rangle$

Anderson (1958)
Basko, Aleiner, Altshuler (2006)
Oganesyan and Huse (2007)
Pal and Huse (2010)
Bauer and Nayak (2013)
...

Many-body localization

Extended

$\sigma > 0$

Volume law

ETH

Localized

$\sigma = 0$

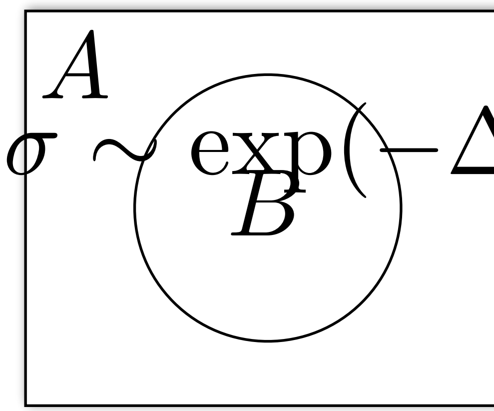
Area law

ETH breaks down

$\epsilon > \epsilon_0.$

Band insulator

$\sigma \sim \exp(-\Delta/kT)$



$\rho_B = \mathrm{Tr}_A |\psi\rangle\langle\psi|$

$S = -\mathrm{Tr}_B \rho_B \log \rho_B$

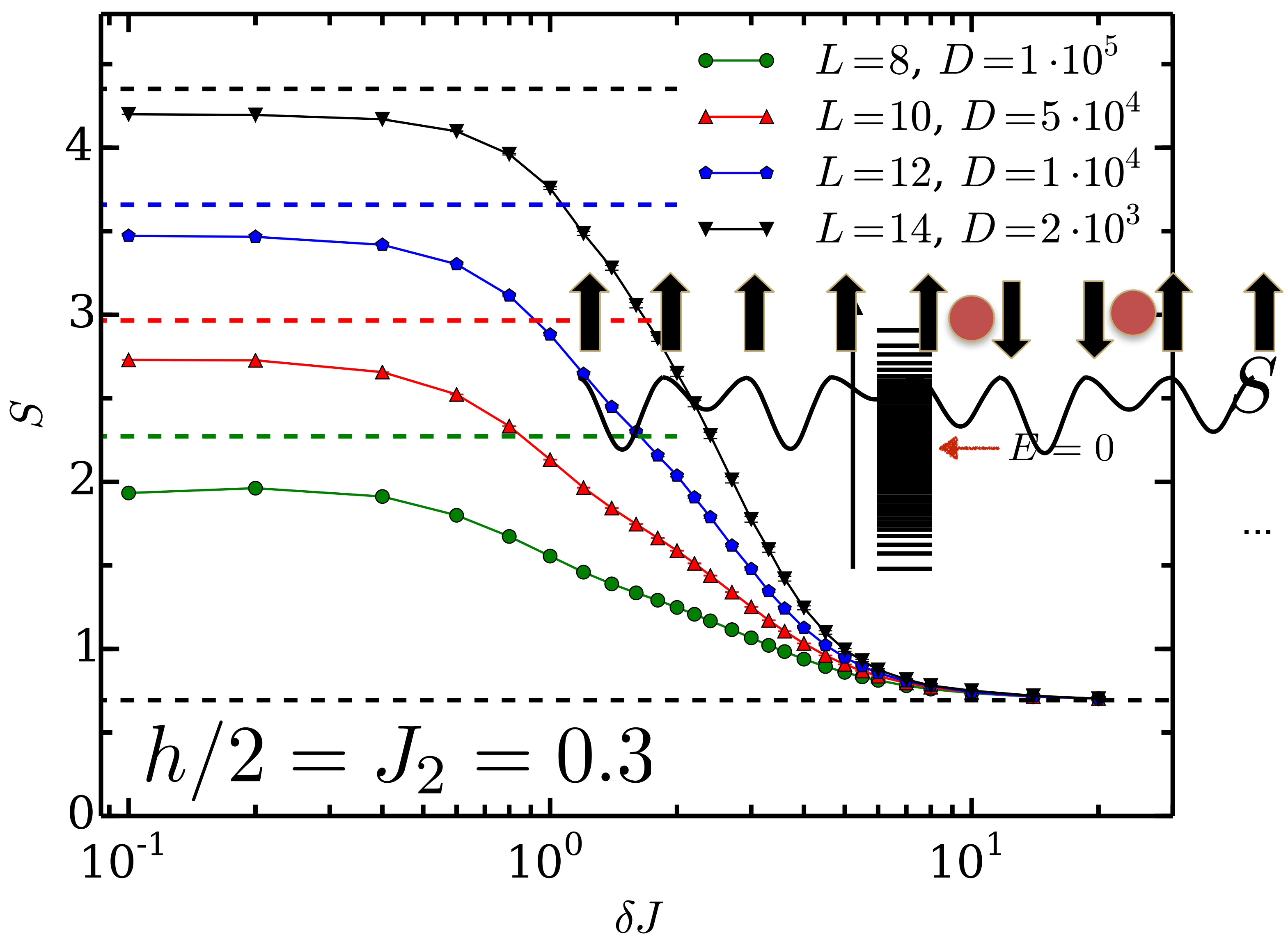
disorder strength

Anderson (1958)
Basko, Aleiner, Altshuler (2006)
Oganesyan and Huse (2007)
Pal and Huse (2010)
Bauer and Nayak (2013)
...

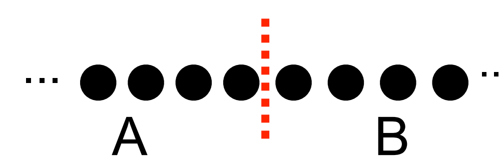
Many-body localization transition

- Localized and extended phase: **AREA vs. VOLUME law**

$$H = -\sum_i (1+\delta J_i)\sigma_i^z \sigma_{i+1}^z + h\sum_i \sigma_i^x + J_2 \sum_i \sigma_i^z \sigma_{i+2}^z$$



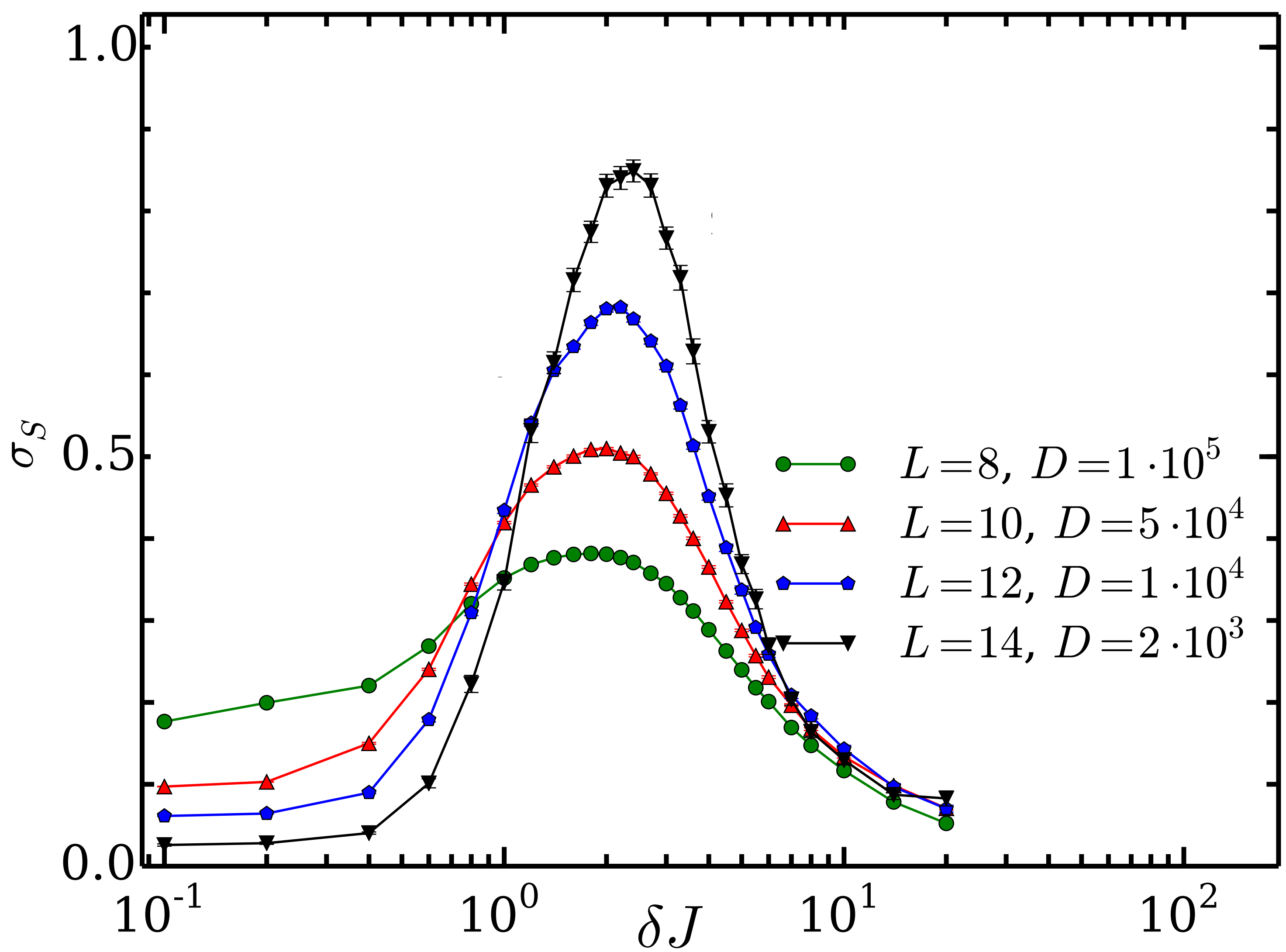
$$S = -\mathrm{Tr}_B \rho_B \log \rho_B$$



Kjäll, Bárðarson, FP, PRL **113**, 107204 (2014)

Many-body localization transition

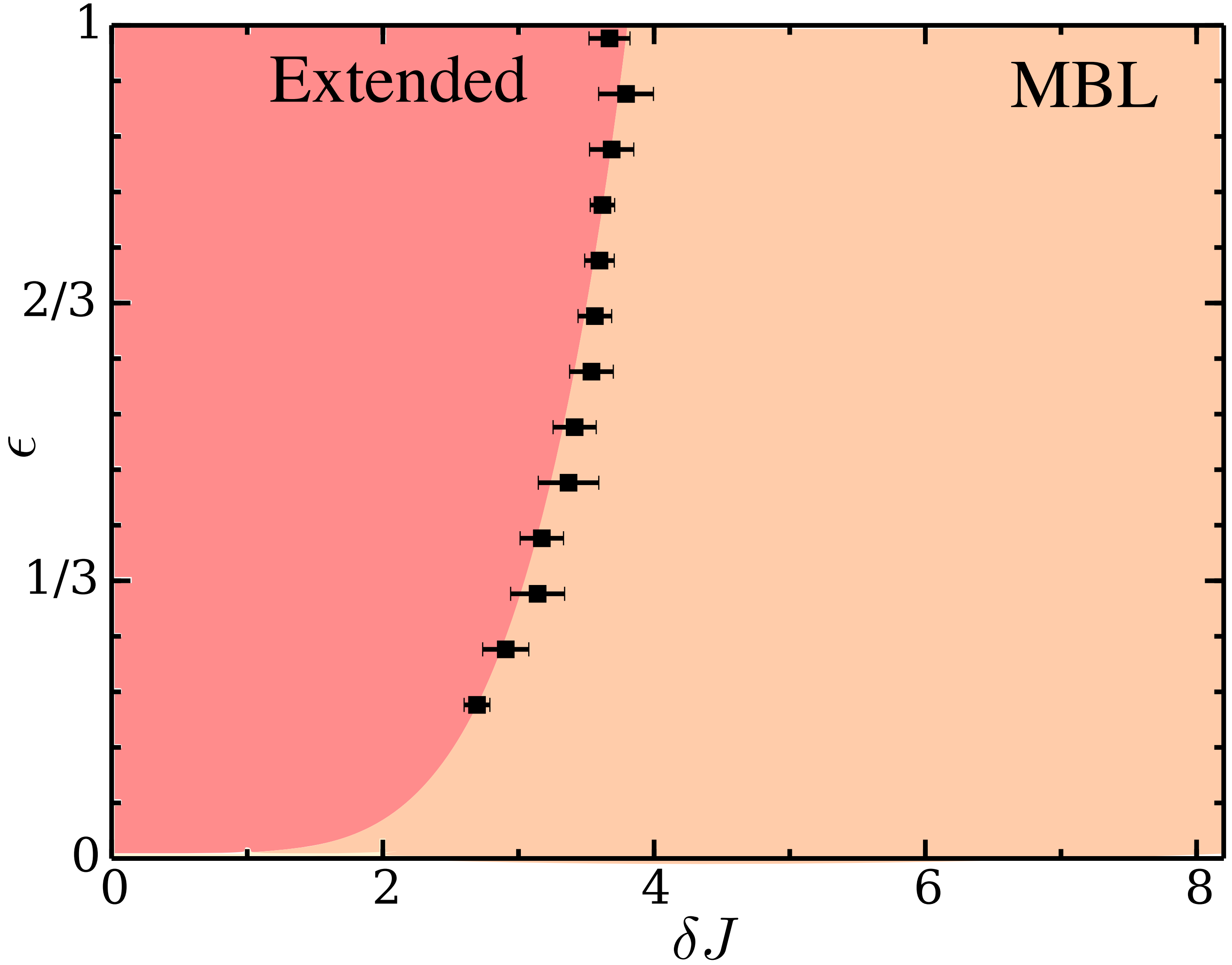
- Localized and extended phase: **AREA vs. VOLUME** law
  - ➡ Variance of S diverges at the transition point



Kjäll, Bárðarson, FP, PRL **113**, 107204 (2014)

Many-body localization transition

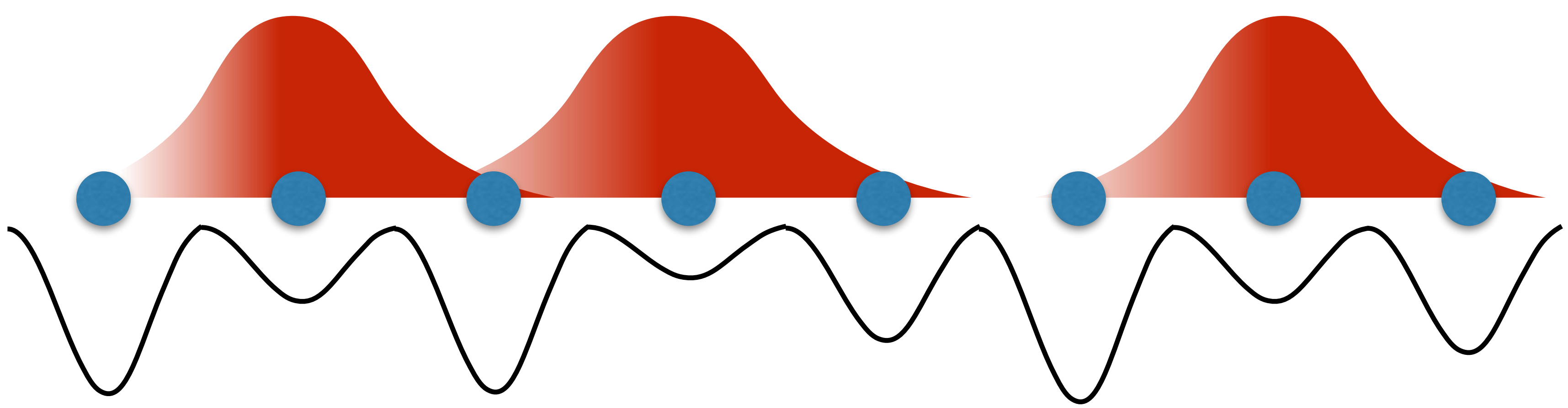
- Repeating the scaling for various energy densities yields the phase diagram



Kjäll, Bárðarson, FP, PRL **113**, 107204 (2014)

Quasi local integrals of motion

- Many-body eigenstates of Anderson insulator



- "Quasi local" product state representation of 2^L states

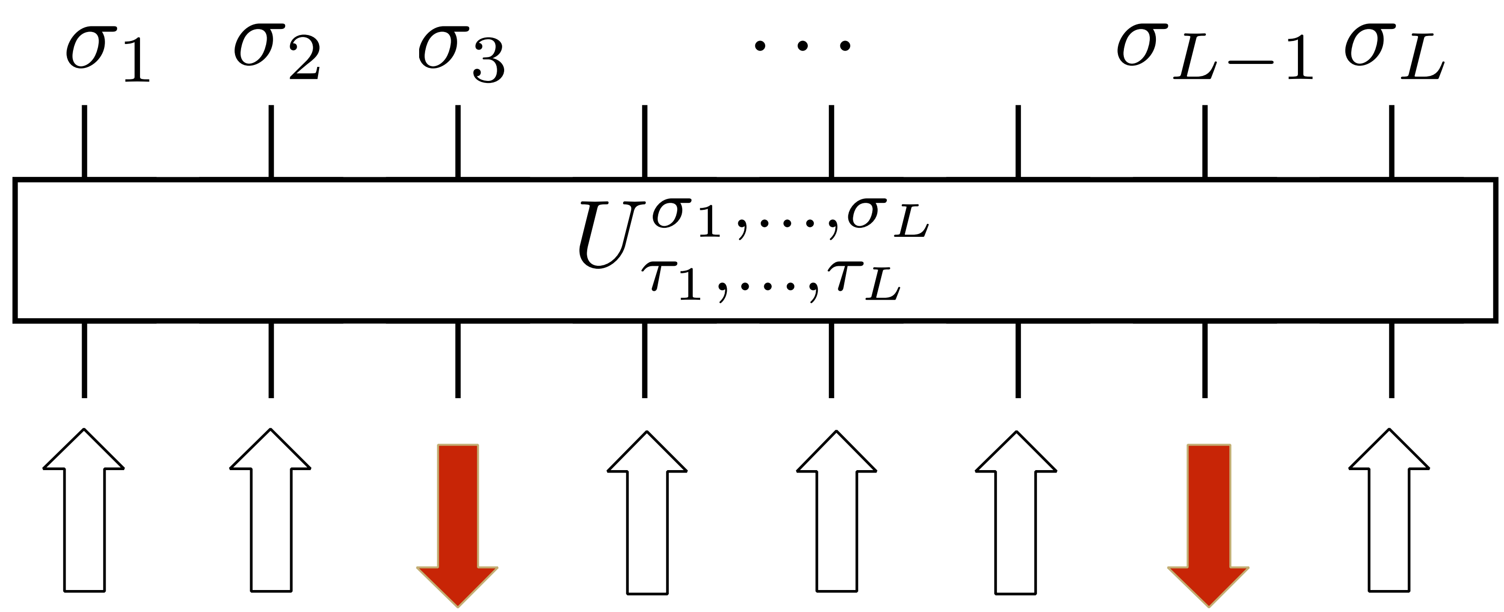
$$|\psi_{n_1,n_2,\ldots,n_L}\rangle = (c_1^\dagger)^{n_1}(c_2^\dagger)^{n_2}\ldots(c_L^\dagger)^{n_L}|0\rangle$$

Quasi local integrals of motion

- Many-body localization: "**p-bits**" (σ) and "**l-bits**" (τ):
  [Huse & Oganesyan '13, Serbyn, Papic, Abanin '13]

$$|\psi_{\tau_1,\tau_2,\ldots,\tau_L}\rangle =$$

$$|\tau_1, \tau_2, \ldots, \tau_L\rangle =$$

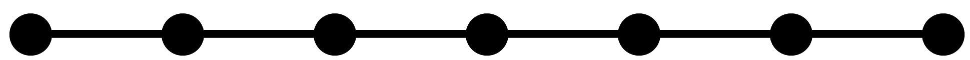


- All 2^L many-body eigenstates given by a **"quasi local"** unitary

- Efficient representation as **Matrix-Product Operator** ???

Disordered Anisotropic Heisenberg Chain

- Toy model to study the MBL phases [Anderson '58]



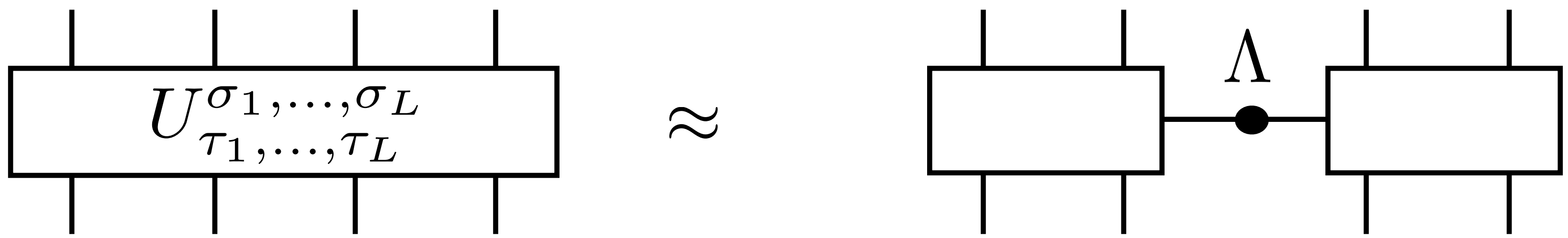
$$H = J_\perp \sum_i \underbrace{(S_i^x S_{i+1}^x + S_i^y S_{i+1}^y)}_{\text{hopping}} + \underbrace{\sum_i h_i S_i^z}_{\text{random potential}} + \underbrace{J_z \sum_i S_i^z S_{i+1}^z}_{\text{interaction}}$$

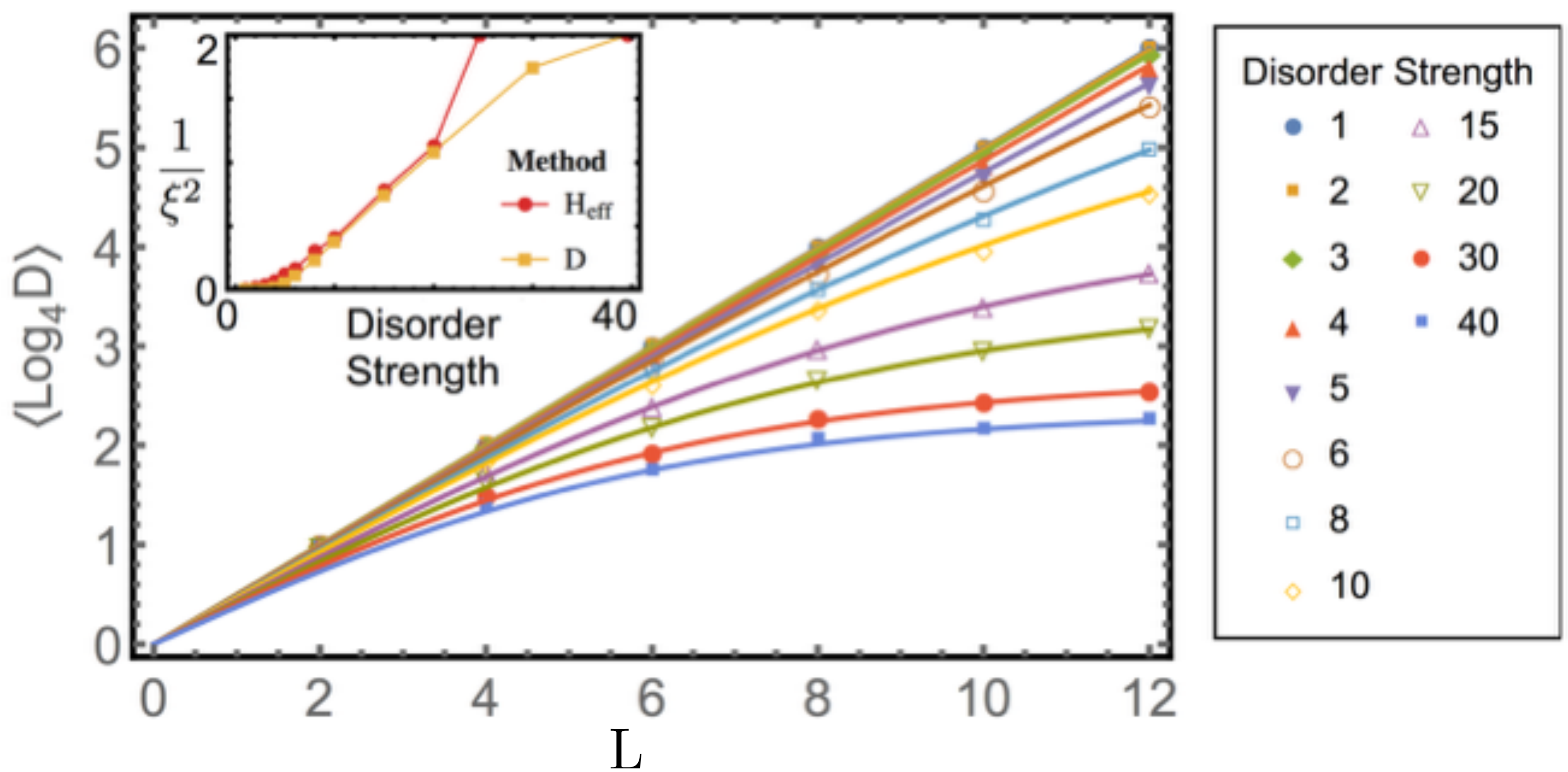
with $h_i \in [-W, W]$

- All single particle states localized for $W \neq 0$
- $J_\perp = J_z = 1$: fully MBL for $W \gtrsim 3.5$ [Pal & Huse '10]

Quasi local integrals of motion

- Compression using exact diagonalization (ED) [Pekker & Clark '14]



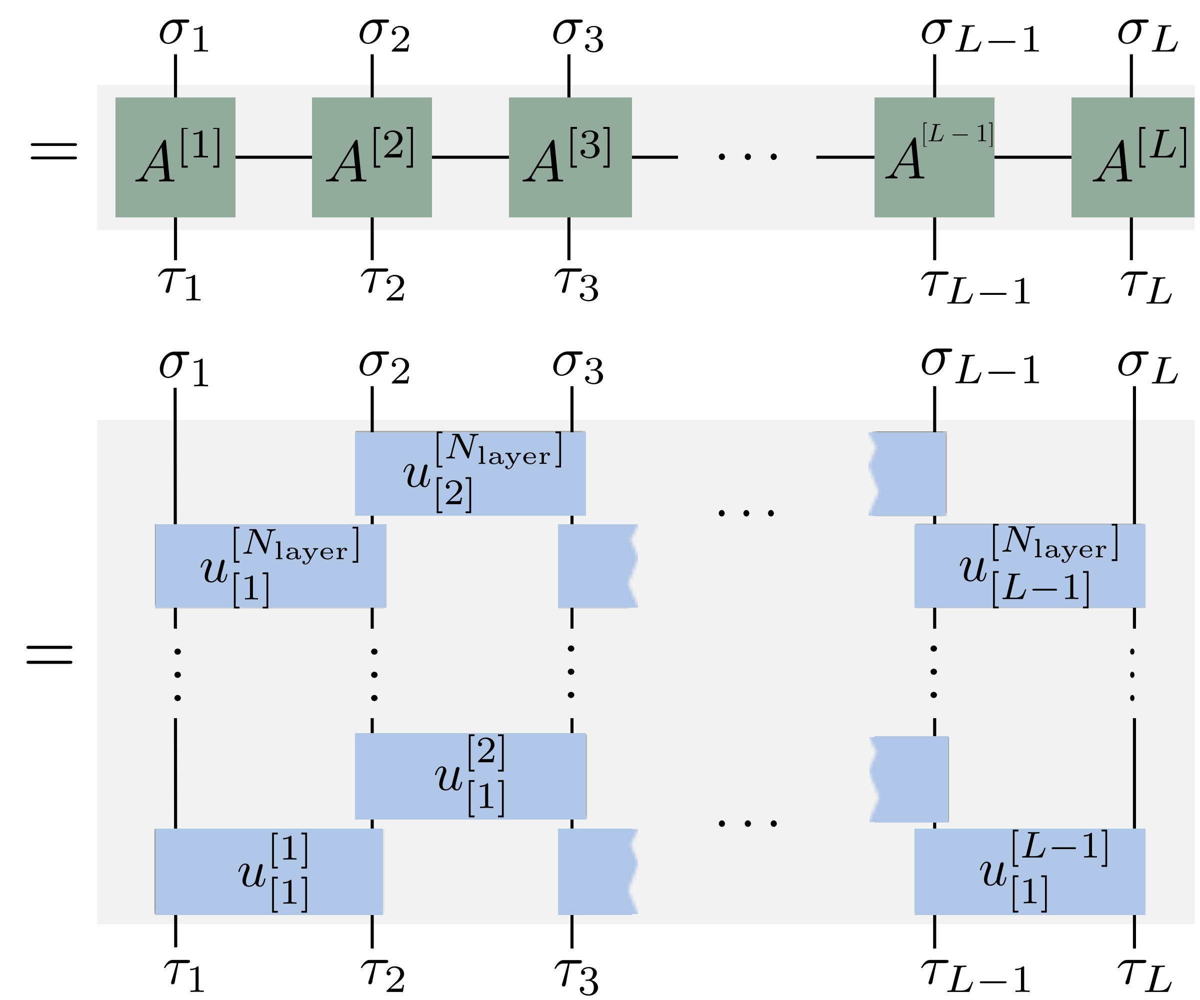


- ED exponential in size! Gauge of $U^{\sigma_1,\ldots,\sigma_L}_{\tau_1,\ldots,\tau_L}$? Unitarity?

Variational Ansatz:

- Finite depth local unitary network

$\tilde{U}^{\sigma_1,\dots\sigma_L}_{\tau_1,\dots\tau_L} =$



Different unitary networks possible…

- Locally minimize the cost function using CG

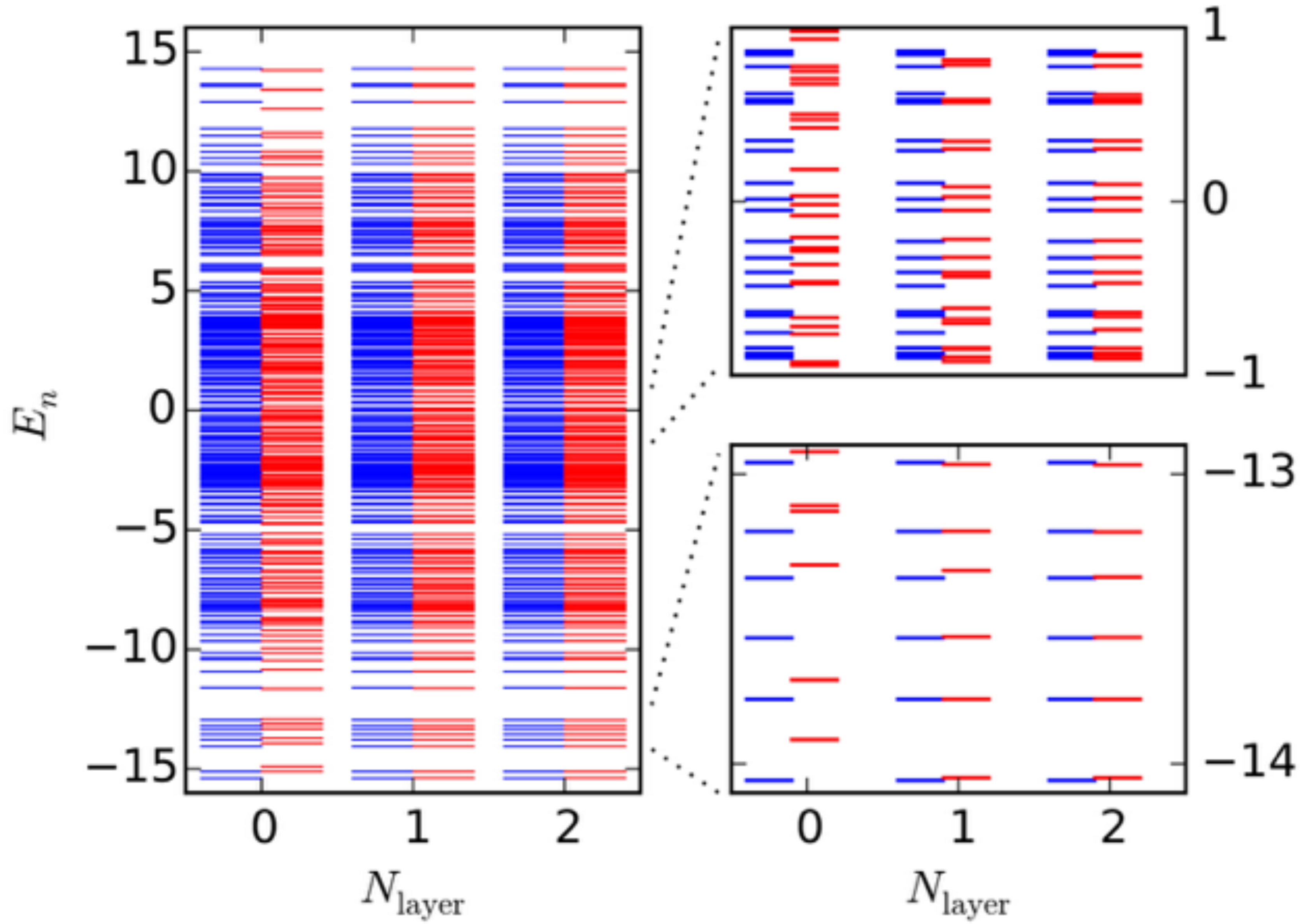
$$f(\{A^{[n]}\}) = \sum_{\{\boldsymbol{\tau}\}} \langle \psi_{\boldsymbol{\tau}} | H^2 | \psi_{\boldsymbol{\tau}} \rangle - \langle \psi_{\boldsymbol{\tau}} | H | \psi_{\boldsymbol{\tau}} \rangle^2 \geq 0$$

Scaling: **Linear in L and exponential in N_{Layer}**

FP, Khemani, Cirac, Sondhi, arXiv:1506.07179 (2015)

Comparison with exact results

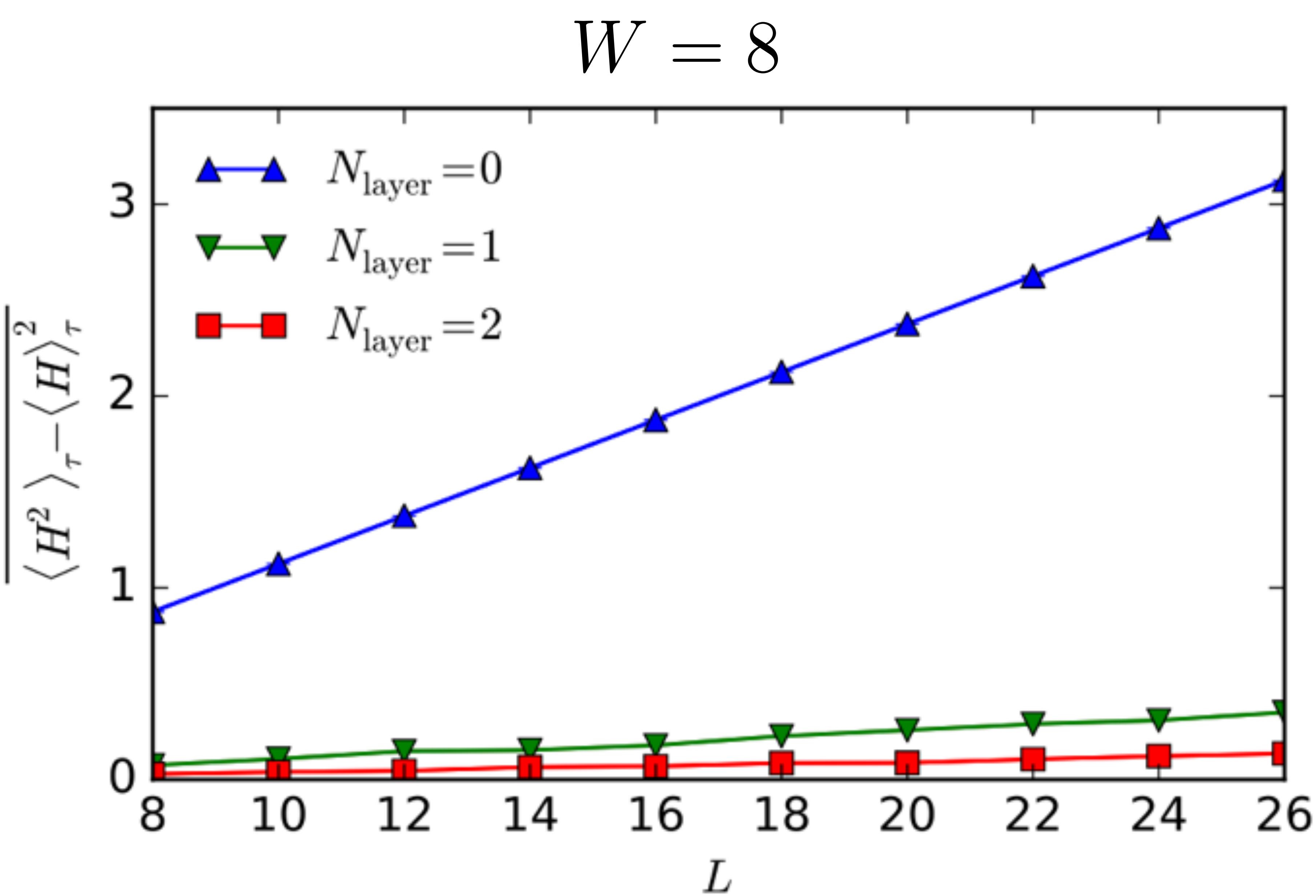
- Deep in localized phase with $W = 8$ and $L = 8$:



FP, Khemani, Cirac, Sondhi, arXiv:1506.07179 (2015)

Comparison with exact results

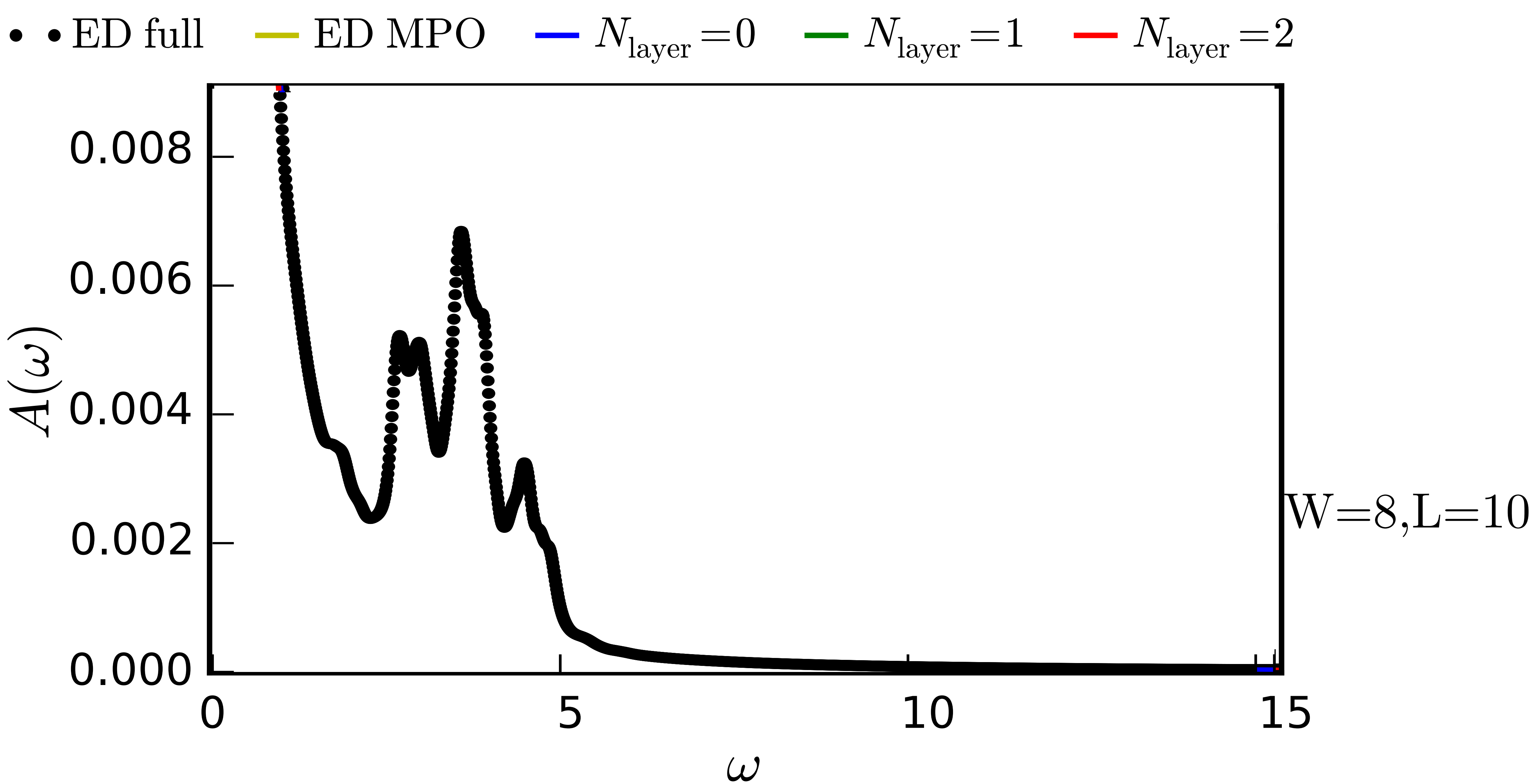
- Linear scaling of the mean variance: **Constant error density**



FP, Khemani, Cirac, Sondhi, arXiv:1506.07179 (2015)

Comparison with exact results

- Spectral function: $A(\omega) = \frac{1}{2^L} \sum_{\{\boldsymbol{\tau_1}\},\{\boldsymbol{\tau_2}\}} |\langle \boldsymbol{\tau_1} | S^z_{L/2} | \boldsymbol{\tau_2} \rangle|^2 \delta(\omega - E_{\boldsymbol{\tau_1}} + E_{\boldsymbol{\tau_2}})$



FP, Khemani, Cirac, Sondhi, arXiv:1506.07179 (2015)