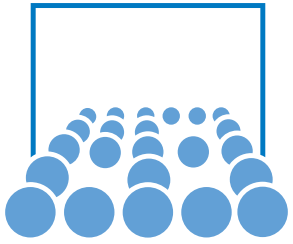
Numerical Linear and Multilinear Algebraic Approaches to Quantum Tensor Networks

Konrad Waldherr

November 8, 2013



Joint work with Thomas Huckle

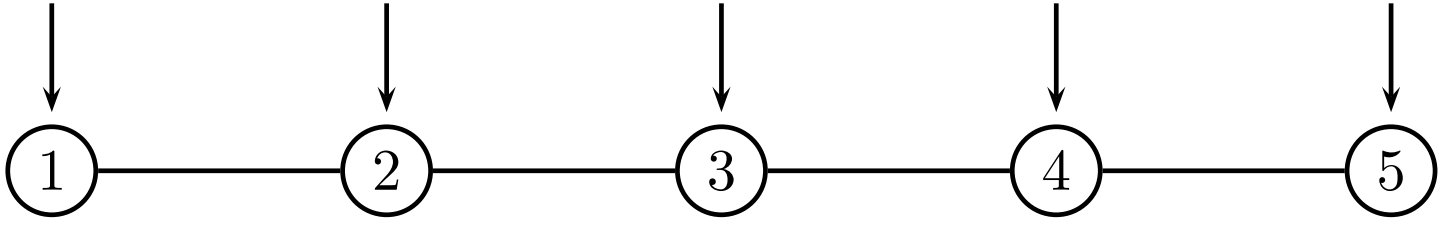
Problem: Computation of Ground States

Given:

- Physical system with N particles (e.g., 1D spin chain)

- Couplings within the system (e.g., nearest-neighbor interaction)

- External interaction (e.g., exterior magnetic field)



Goal:

- Find ground state, i.e., the state related to the smallest energy of the system.

Mathematical Model

- The state of an individual spin is described by a unit vector in $\mathbb{C}^2$.
- The state space of a composed system is the **tensor product** of the state spaces of the constituent subsystems.
- The state of the entire system is represented by a vector in $\mathbb{C}^{2^N}$.

Mathematical Model

- The state of an individual spin is described by a unit vector in $\mathbb{C}^2$.
- The state space of a composed system is the **tensor product** of the state spaces of the constituent subsystems.
- The state of the entire system is represented by a vector in $\mathbb{C}^{2^N}$.
- The overall energy of the physical system is described by the Hamiltonian $\boldsymbol{H} \in \mathbb{C}^{2^N \times 2^N}$.
- The Hamiltonian has a representation

$$\boldsymbol{H} = \sum_{k=1}^{M} \alpha_k \boldsymbol{Q}_{\boldsymbol{1}}^{(\boldsymbol{k})} \otimes \cdots \otimes \boldsymbol{Q}_{\boldsymbol{N}}^{(\boldsymbol{k})} \in \mathbb{C}^{2^N \times 2^N} ,$$

where the tensor factors $\boldsymbol{Q}_{\boldsymbol{j}}^{(\boldsymbol{k})}$ are identities or Pauli matrices.

$$\boldsymbol{\sigma_x} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \boldsymbol{\sigma_y} = \begin{pmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{pmatrix}, \qquad \boldsymbol{\sigma_z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} .$$

Mathematical Model

- The state of an individual spin is described by a unit vector in $\mathbb{C}^2$.
- The state space of a composed system is the **tensor product** of the state spaces of the constituent subsystems.
- The state of the entire system is represented by a vector in $\mathbb{C}^{2^N}$.
- The overall energy of the physical system is described by the Hamiltonian $\boldsymbol{H} \in \mathbb{C}^{2^N \times 2^N}$.
- The Hamiltonian has a representation

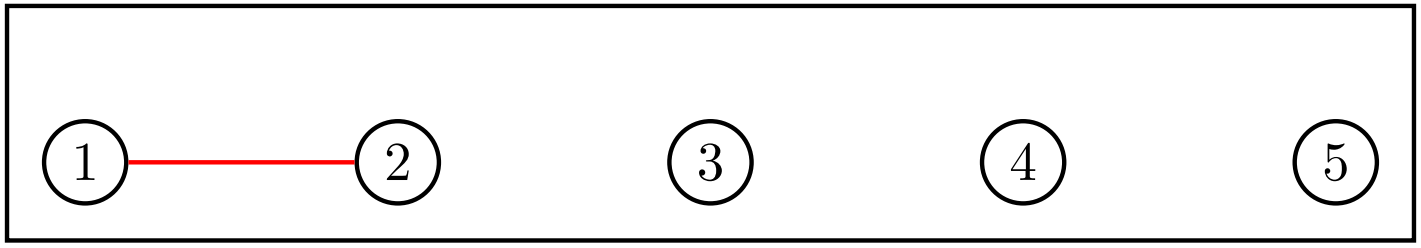
$$\boldsymbol{H} = \sum_{k=1}^{M} \alpha_k \boldsymbol{Q}_{\boldsymbol{1}}^{(\boldsymbol{k})} \otimes \cdots \otimes \boldsymbol{Q}_{\boldsymbol{N}}^{(\boldsymbol{k})} \in \mathbb{C}^{2^N \times 2^N},$$

  where the tensor factors $\boldsymbol{Q}_{\boldsymbol{j}}^{(\boldsymbol{k})}$ are identities or Pauli matrices.

$$\boldsymbol{\sigma_x} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \boldsymbol{\sigma_y} = \begin{pmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{pmatrix}, \qquad \boldsymbol{\sigma_z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

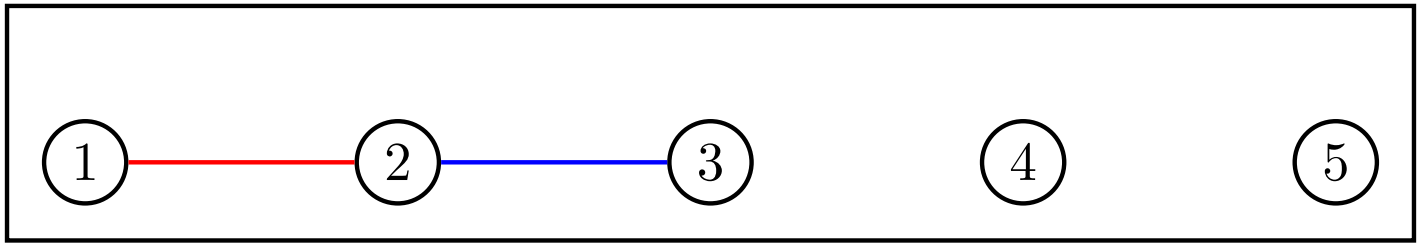
- Then, the ground state corresponds to the **eigenvector related to the smallest eigenvalue**.

Example: Ising-ZZ Hamiltonian



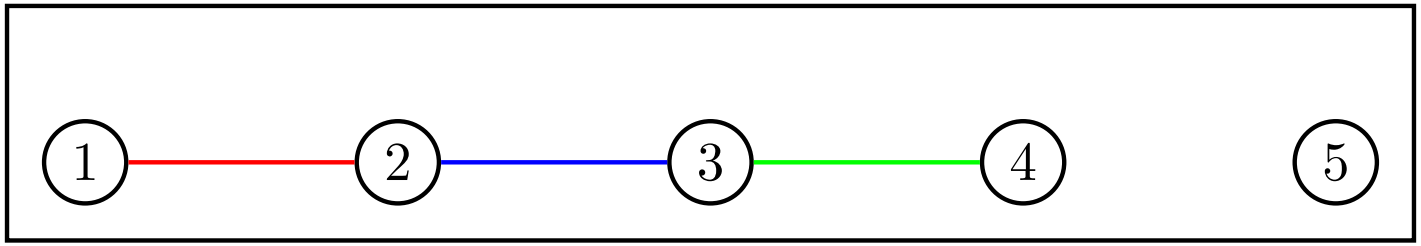
$$H = \sigma_z \otimes \sigma_z \otimes I \otimes I \otimes I$$

Example: Ising-ZZ Hamiltonian



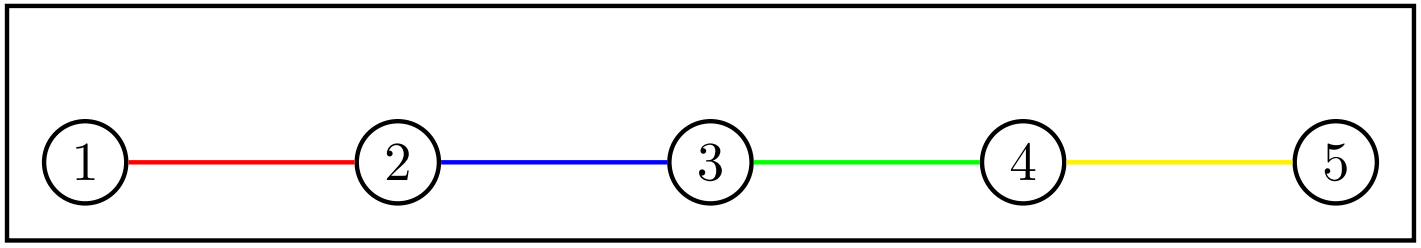
$$
\begin{aligned}
H &= \sigma_z \otimes \sigma_z \otimes I \otimes I \otimes I \\
&+ I \otimes \sigma_z \otimes \sigma_z \otimes I \otimes I
\end{aligned}
$$

Example: Ising-ZZ Hamiltonian



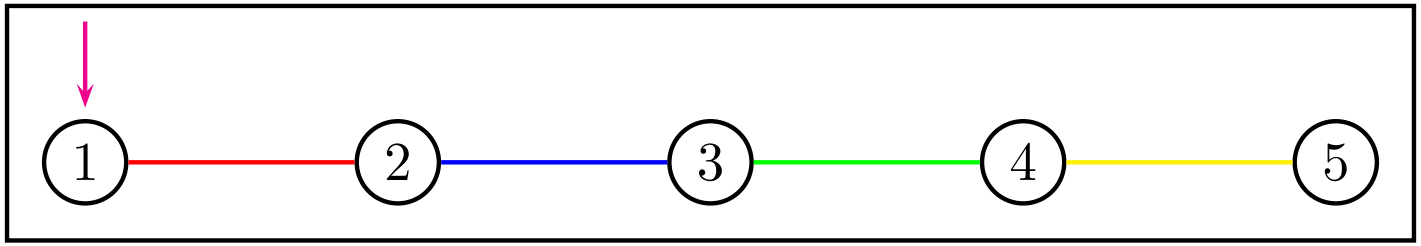
$$
\begin{aligned}
H &= \sigma_z \otimes \sigma_z \otimes I \otimes I \otimes I \\
&+ I \otimes \sigma_z \otimes \sigma_z \otimes I \otimes I \\
&+ I \otimes I \otimes \sigma_z \otimes \sigma_z \otimes I
\end{aligned}
$$

Example: Ising-ZZ Hamiltonian



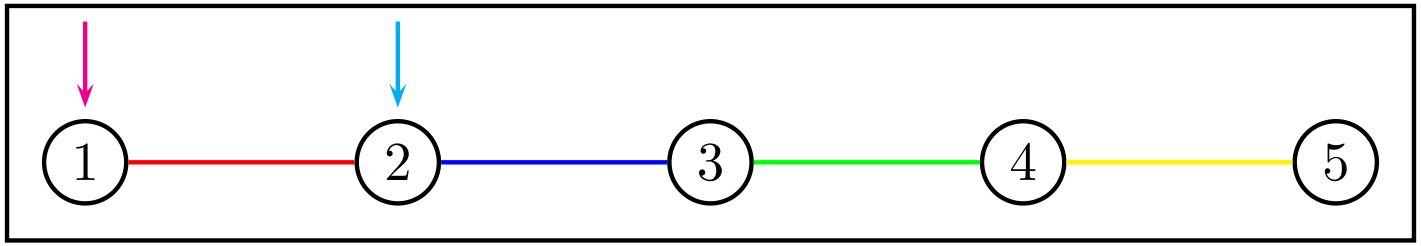
$$
\begin{aligned}
H &= \sigma_z \otimes \sigma_z \otimes I \otimes I \otimes I \\
&+ I \otimes \sigma_z \otimes \sigma_z \otimes I \otimes I \\
&+ I \otimes I \otimes \sigma_z \otimes \sigma_z \otimes I \\
&+ I \otimes I \otimes I \otimes \sigma_z \otimes \sigma_z
\end{aligned}
$$

Example: Ising-ZZ Hamiltonian



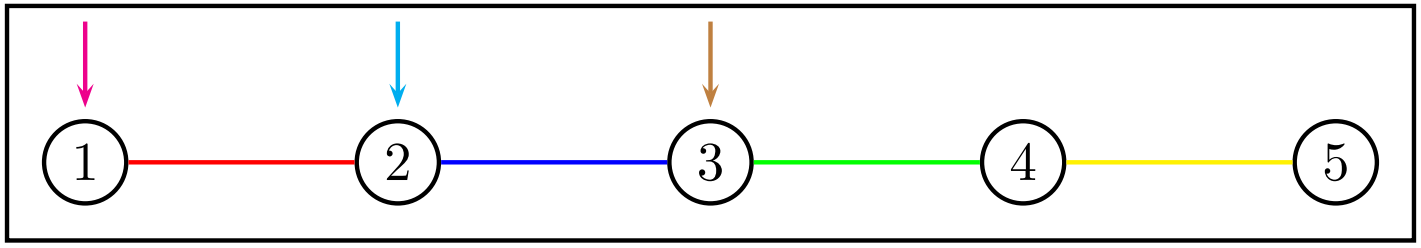
$$
\begin{aligned}
H &= \sigma_z \otimes \sigma_z \otimes I \otimes I \otimes I \\
&+ I \otimes \sigma_z \otimes \sigma_z \otimes I \otimes I \\
&+ I \otimes I \otimes \sigma_z \otimes \sigma_z \otimes I \\
&+ I \otimes I \otimes I \otimes \sigma_z \otimes \sigma_z \\
&+ \sigma_x \otimes I \otimes I \otimes I \otimes I
\end{aligned}
$$

Example: Ising-ZZ Hamiltonian



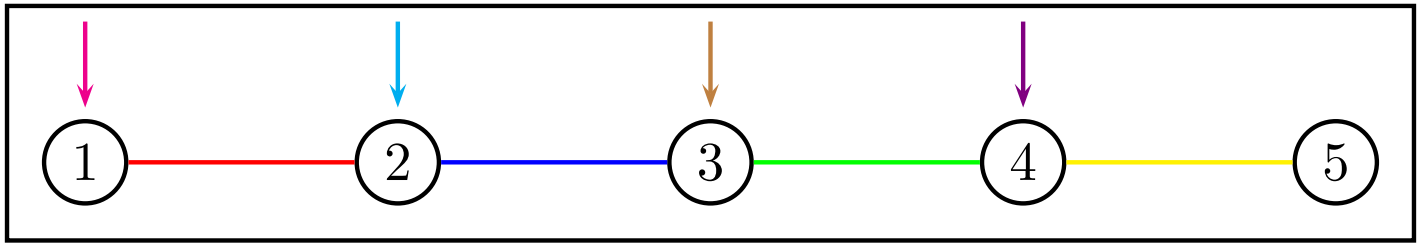
$$
\begin{aligned}
H &= \sigma_z \otimes \sigma_z \otimes I \otimes I \otimes I \\
&+ I \otimes \sigma_z \otimes \sigma_z \otimes I \otimes I \\
&+ I \otimes I \otimes \sigma_z \otimes \sigma_z \otimes I \\
&+ I \otimes I \otimes I \otimes \sigma_z \otimes \sigma_z \\
&+ \sigma_x \otimes I \otimes I \otimes I \otimes I \\
&+ I \otimes \sigma_x \otimes I \otimes I \otimes I
\end{aligned}
$$

Example: Ising-ZZ Hamiltonian



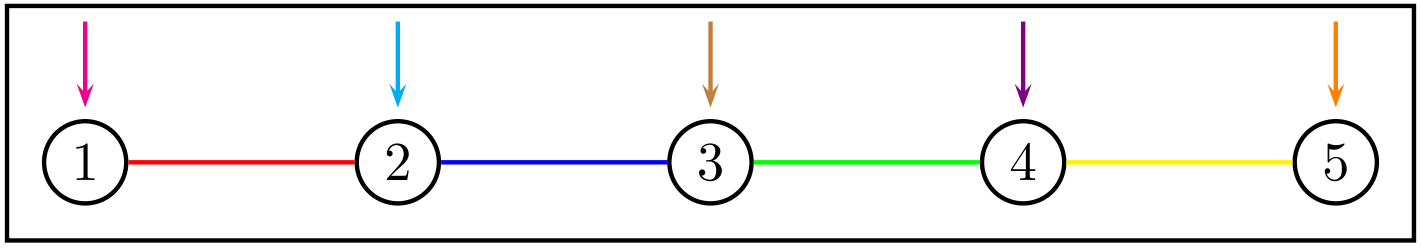
$$
\begin{aligned}
H &= \sigma_z \otimes \sigma_z \otimes I \otimes I \otimes I \\
&+ I \otimes \sigma_z \otimes \sigma_z \otimes I \otimes I \\
&+ I \otimes I \otimes \sigma_z \otimes \sigma_z \otimes I \\
&+ I \otimes I \otimes I \otimes \sigma_z \otimes \sigma_z \\
&+ \sigma_x \otimes I \otimes I \otimes I \otimes I \\
&+ I \otimes \sigma_x \otimes I \otimes I \otimes I \\
&+ I \otimes I \otimes \sigma_x \otimes I \otimes I
\end{aligned}
$$

Example: Ising-ZZ Hamiltonian



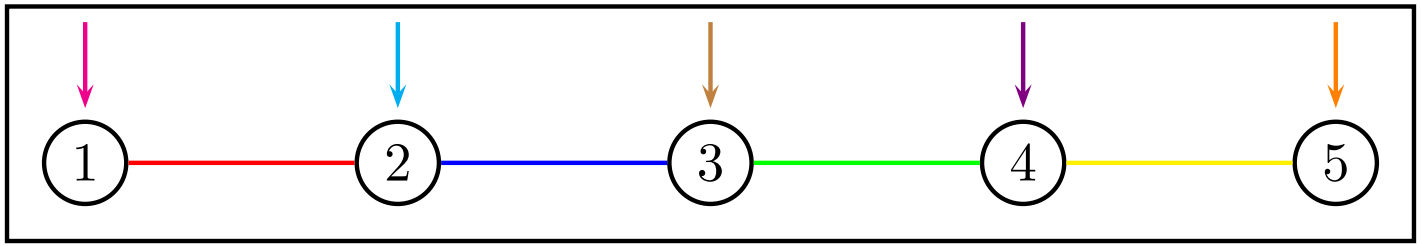
$$
\begin{aligned}
H &= \sigma_z \otimes \sigma_z \otimes I \otimes I \otimes I \\
&+ I \otimes \sigma_z \otimes \sigma_z \otimes I \otimes I \\
&+ I \otimes I \otimes \sigma_z \otimes \sigma_z \otimes I \\
&+ I \otimes I \otimes I \otimes \sigma_z \otimes \sigma_z \\
&+ \sigma_x \otimes I \otimes I \otimes I \otimes I \\
&+ I \otimes \sigma_x \otimes I \otimes I \otimes I \\
&+ I \otimes I \otimes \sigma_x \otimes I \otimes I \\
&+ I \otimes I \otimes I \otimes \sigma_x \otimes I
\end{aligned}
$$

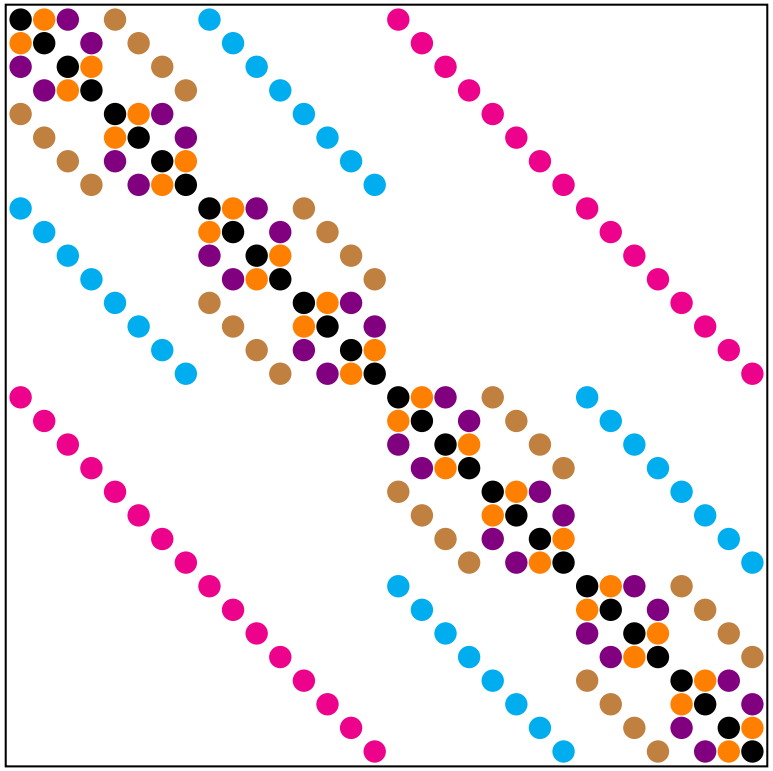
Example: Ising-ZZ Hamiltonian



$$
\begin{aligned}
H &= \sigma_z \otimes \sigma_z \otimes I \otimes I \otimes I \\
&+ I \otimes \sigma_z \otimes \sigma_z \otimes I \otimes I \\
&+ I \otimes I \otimes \sigma_z \otimes \sigma_z \otimes I \\
&+ I \otimes I \otimes I \otimes \sigma_z \otimes \sigma_z \\
&+ \sigma_x \otimes I \otimes I \otimes I \otimes I \\
&+ I \otimes \sigma_x \otimes I \otimes I \otimes I \\
&+ I \otimes I \otimes \sigma_x \otimes I \otimes I \\
&+ I \otimes I \otimes I \otimes \sigma_x \otimes I \\
&+ I \otimes I \otimes I \otimes I \otimes \sigma_x
\end{aligned}
$$

Sparsity Pattern of the Ising-ZZ Hamiltonian





Problem Setting:

What is the smallest eigenvalue of the huge matrix $H \in \mathbb{C}^{2^N \times 2^N}$?

- Minimization problem:

$$\min_{\boldsymbol{x} \neq \mathbf{0}} \frac{\boldsymbol{x}^{\mathrm{H}} \boldsymbol{H} \boldsymbol{x}}{\boldsymbol{x}^{\mathrm{H}} \boldsymbol{x}} \ .$$

- Challenge: the vector space $\mathbb{C}^{2^N}$ grows **exponentially** in N.

Problem Setting:

What is the smallest eigenvalue of the huge matrix $H \in \mathbb{C}^{2^N \times 2^N}$?

- Minimization problem:

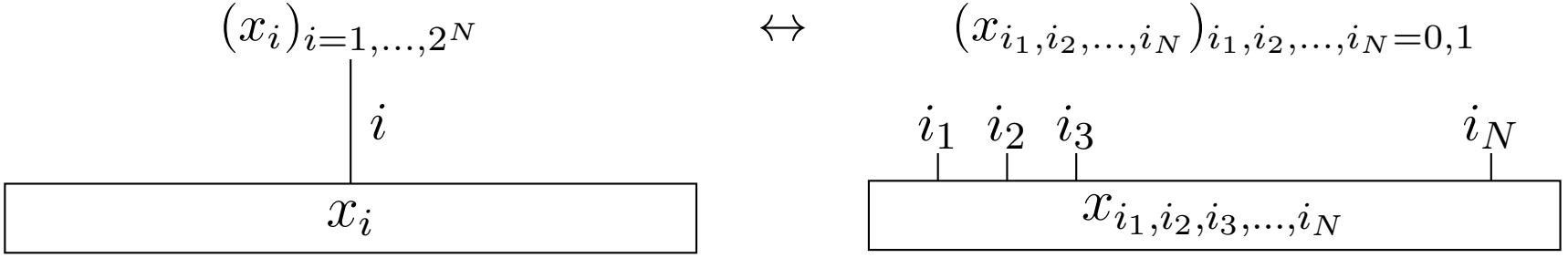
$$\min_{x \neq 0} \frac{x^{\mathrm{H}} H x}{x^{\mathrm{H}} x} .$$

- Challenge: the vector space $\mathbb{C}^{2^N}$ grows **exponentially** in N.
- **Solution: Data-sparse representation format** that
  1. reproduces the underlying physical system,
  2. scales only polynomially in N,
  3. describes the right "corner" $\mathcal{U} \subseteq \mathbb{C}^{2^N}$ of the Hilbert space,
  4. allows for efficient computations of Hx and $x^{\mathrm{H}} y$.

$$\min_{\substack{x \in \mathbb{C}^{2^N} \\ x \neq 0}} \frac{x^{\mathrm{H}} H x}{x^{\mathrm{H}} x} \approx \min_{x \in \mathcal{U}} \frac{x^{\mathrm{H}} H x}{x^{\mathrm{H}} x} = \min_{\substack{y \in \mathbb{C}^{\mathrm{poly}(N)} \\ y \neq 0}} \frac{y^{\mathrm{H}} H_{\mathrm{eff}} y}{y^{\mathrm{H}} N_{\mathrm{eff}} y}$$

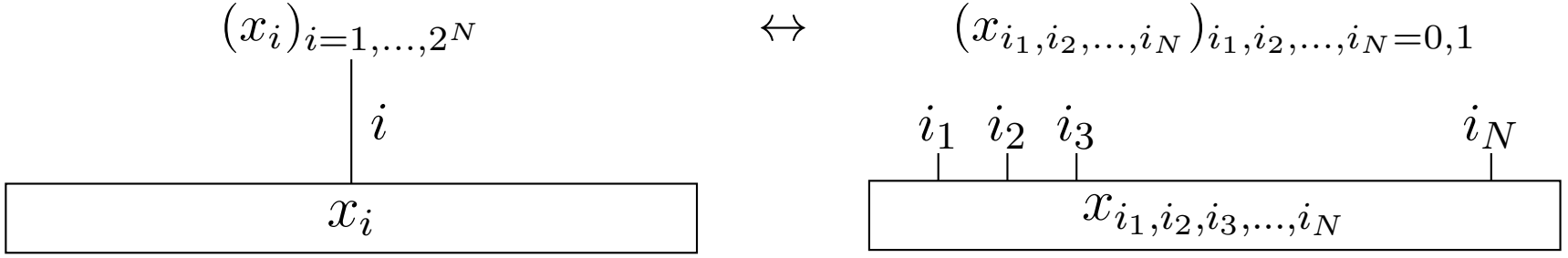
Tensorization of the Vector

- Consider index i in its binary representation $(i_1, i_2, \ldots, i_N)$.
- Reshape 2^N vector into $2 \times 2 \times \cdots \times 2$ tensor:



Tensorization of the Vector

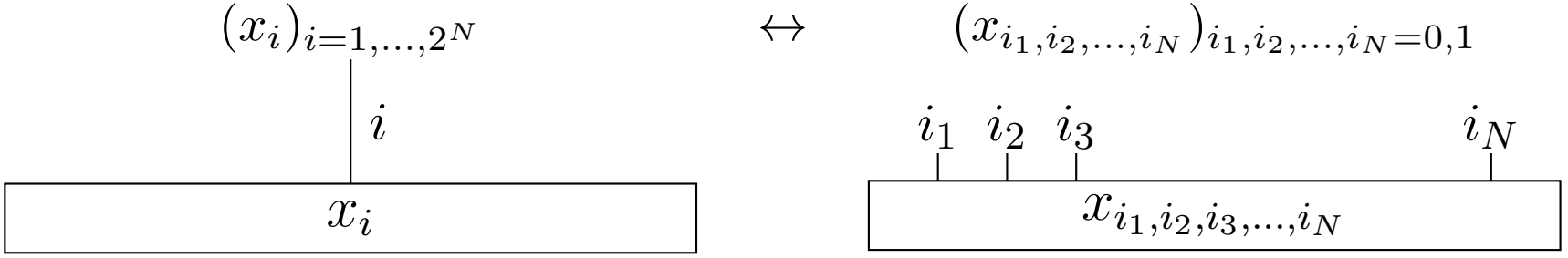
- Consider index i in its binary representation $(i_1, i_2, \ldots, i_N)$.
- Reshape 2^N vector into $2 \times 2 \times \cdots \times 2$ tensor:



Tensor Decomposition Schemes

Tensorization of the Vector

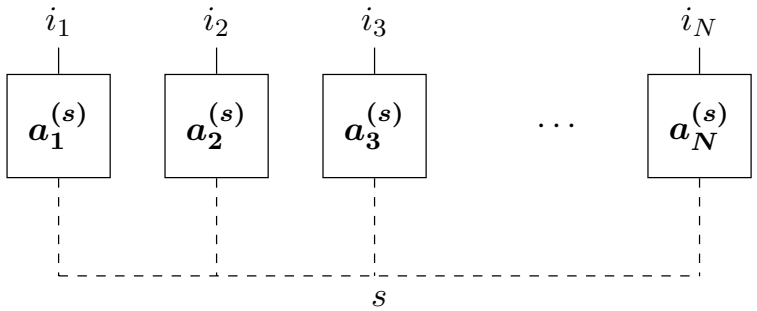
- Consider index i in its binary representation $(i_1, i_2, \ldots, i_N)$.
- Reshape 2^N vector into $2 \times 2 \times \cdots \times 2$ tensor:



Tensor Decomposition Schemes

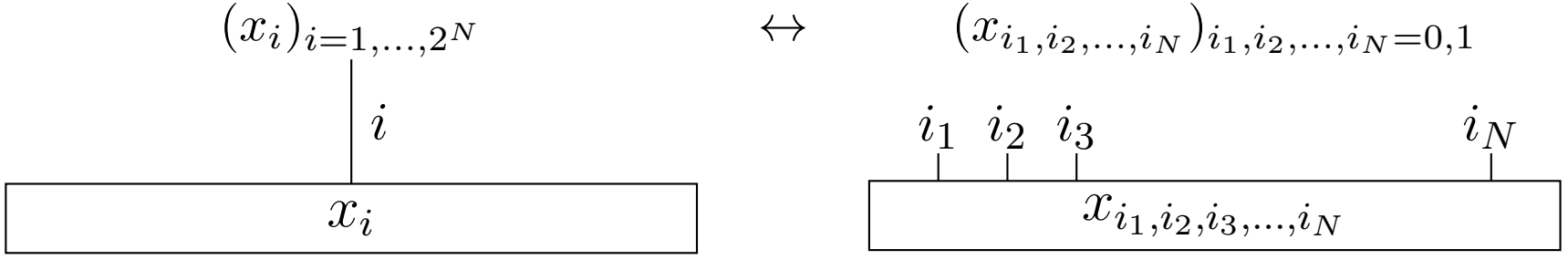
$$\boldsymbol{x} = \sum_{s} \boldsymbol{a_1^{(s)}} \otimes \boldsymbol{a_2^{(s)}} \otimes \cdots \otimes \boldsymbol{a_N^{(s)}}$$

Canonical Format:



Tensorization of the Vector

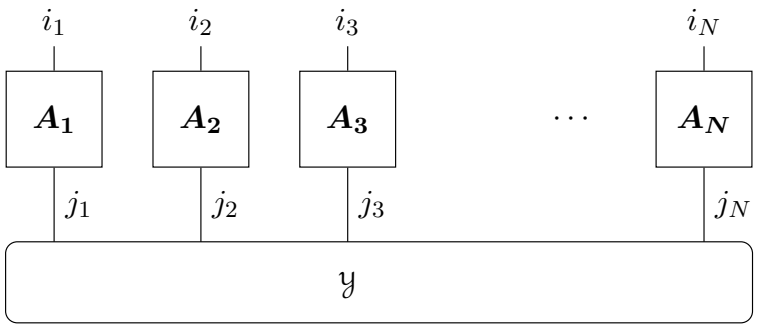
- Consider index i in its binary representation $(i_1, i_2, \ldots, i_N)$.
- Reshape 2^N vector into $2 \times 2 \times \cdots \times 2$ tensor:



Tensor Decomposition Schemes

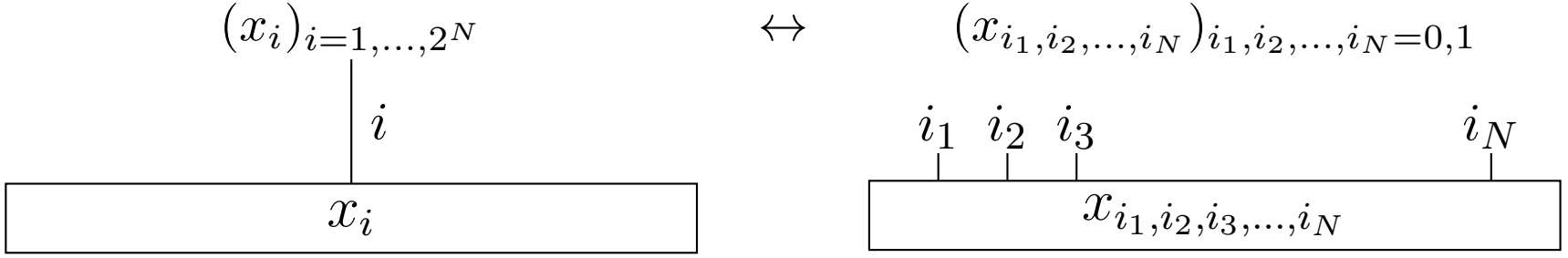
$$\boldsymbol{x} = \sum_{j_1,\ldots,j_N} y_{j_1,\ldots,j_N} \boldsymbol{a_{1;j_1}} \otimes \boldsymbol{a_{2;j_2}} \otimes \cdots \otimes \boldsymbol{a_{N;j_N}}$$

Tucker Format:



Tensorization of the Vector

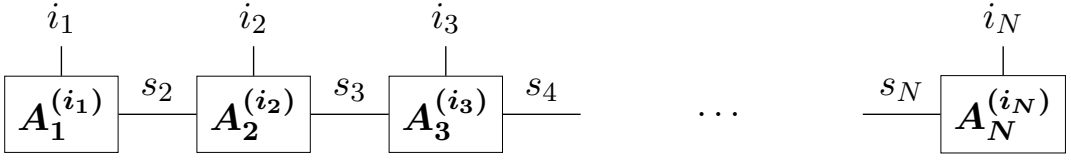
- Consider index i in its binary representation $(i_1, i_2, \ldots, i_N)$.
- Reshape 2^N vector into $2 \times 2 \times \cdots \times 2$ tensor:



Tensor Decomposition Schemes

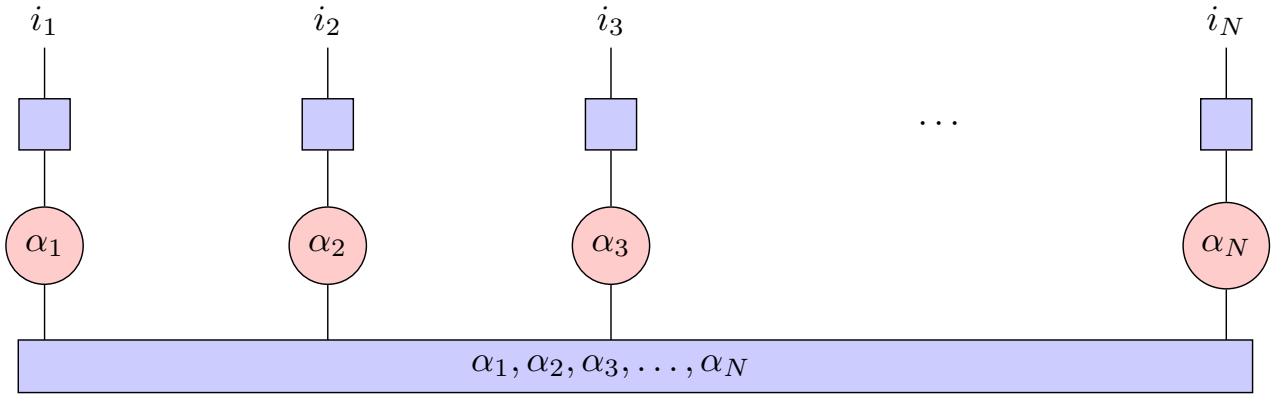
$$\boldsymbol{x} = \sum_{i_1,\ldots,i_N} \left(\boldsymbol{A_1^{(i_1)}} \boldsymbol{A_2^{(i_2)}} \cdots \boldsymbol{A_N^{(i_N)}} \right) \boldsymbol{e_{i_1,\ldots,i_N}}$$

Matrix Product States:

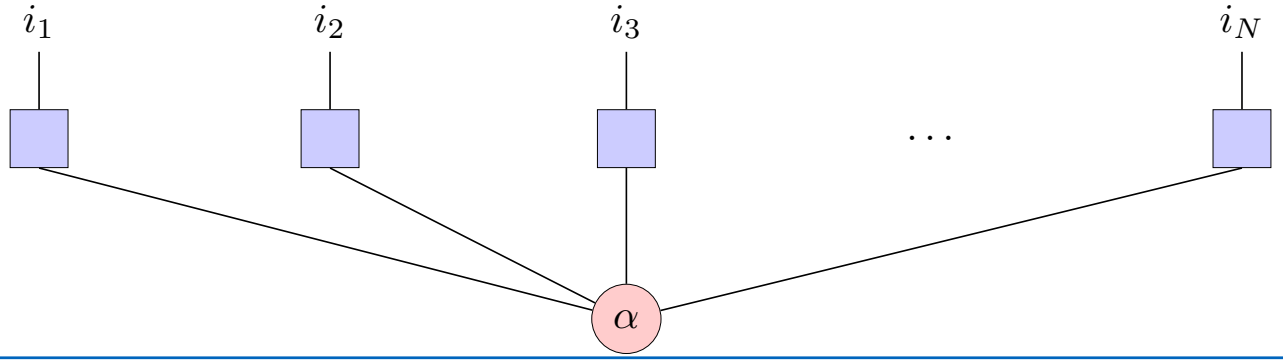


1 ✓

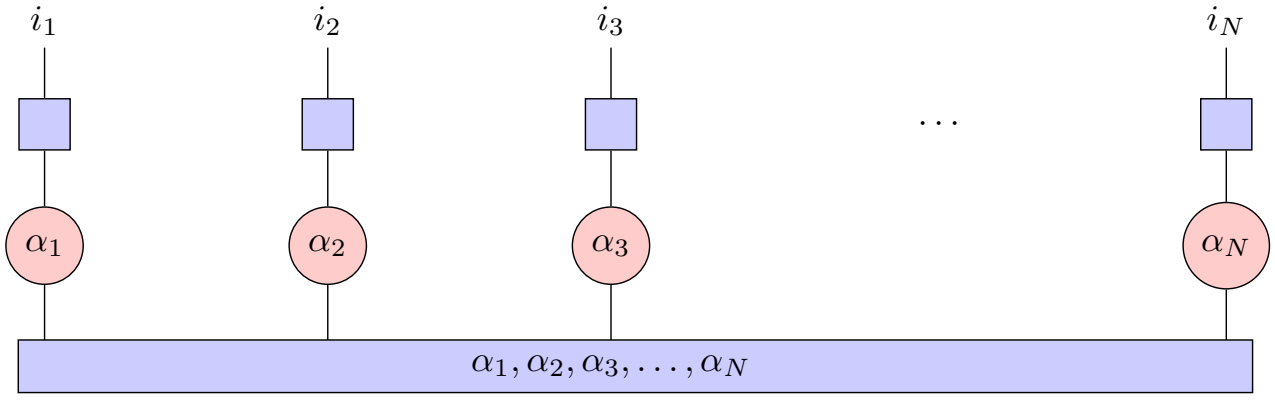
Tucker:


i_1
i_2
i_3
i_N
$\cdots$
α_1
α_2
α_3
α_N
$\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_N$

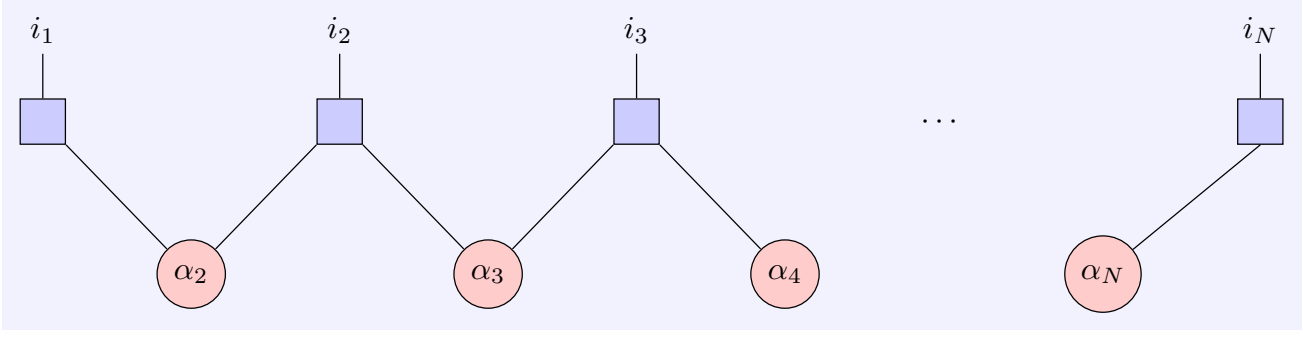
CP:


i_1
i_2
i_3
i_N
$\cdots$
α

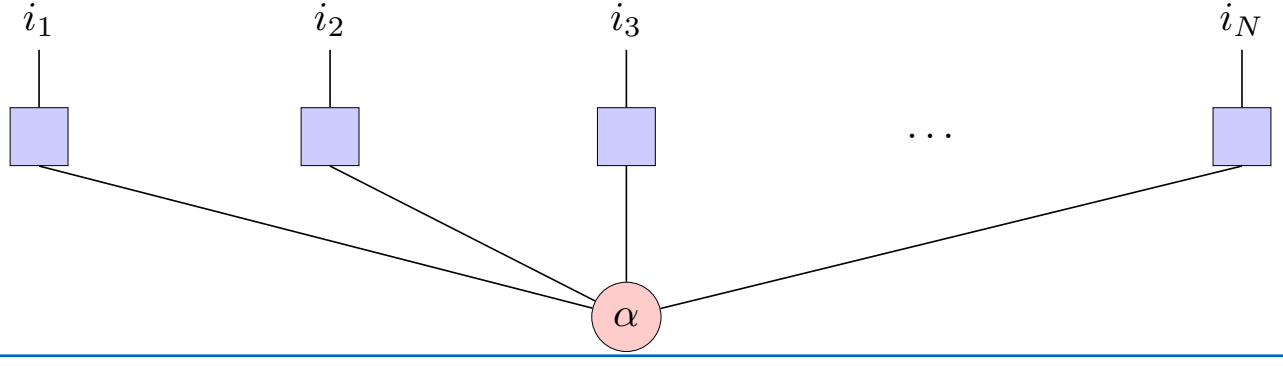
Tucker:



MPS:



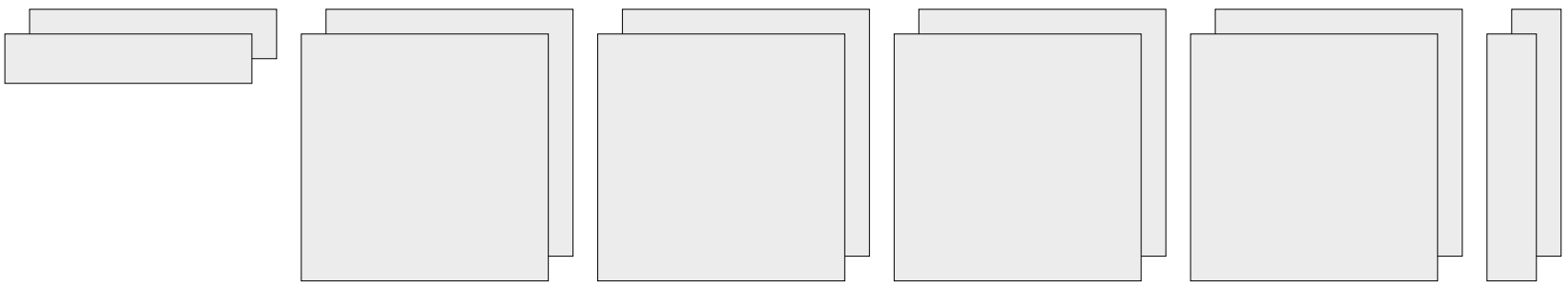
CP:



Matrix Product States: Example

Approximate vector of dimension $2^6 = 64$
by an MPS related to $N = 6$ sites.

$$\boldsymbol{x}_i = \boldsymbol{x}_{i_1,i_2,i_3,i_4,i_5,i_6} = \boldsymbol{A}_1^{(i_1)} \boldsymbol{A}_2^{(i_2)} \boldsymbol{A}_3^{(i_3)} \boldsymbol{A}_4^{(i_4)} \boldsymbol{A}_5^{(i_5)} \boldsymbol{A}_6^{(i_6)}$$

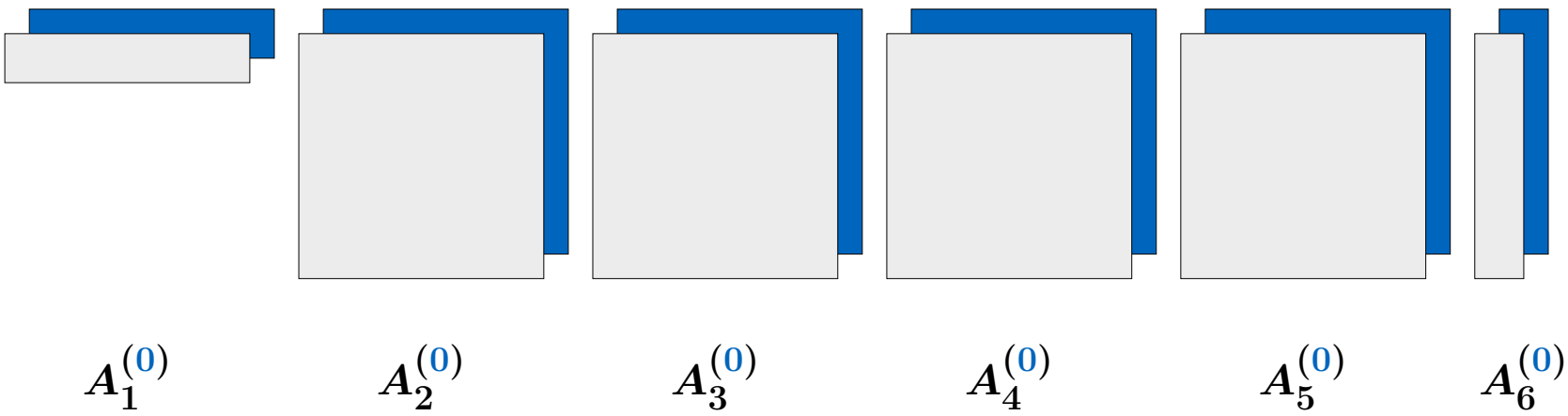


$\boldsymbol{A}_1^{(i_1)}$ $\boldsymbol{A}_2^{(i_2)}$ $\boldsymbol{A}_3^{(i_3)}$ $\boldsymbol{A}_4^{(i_4)}$ $\boldsymbol{A}_5^{(i_5)}$ $\boldsymbol{A}_6^{(i_6)}$

Matrix Product States: Example

Approximate vector of dimension $2^6 = 64$
by an MPS related to $N = 6$ sites.

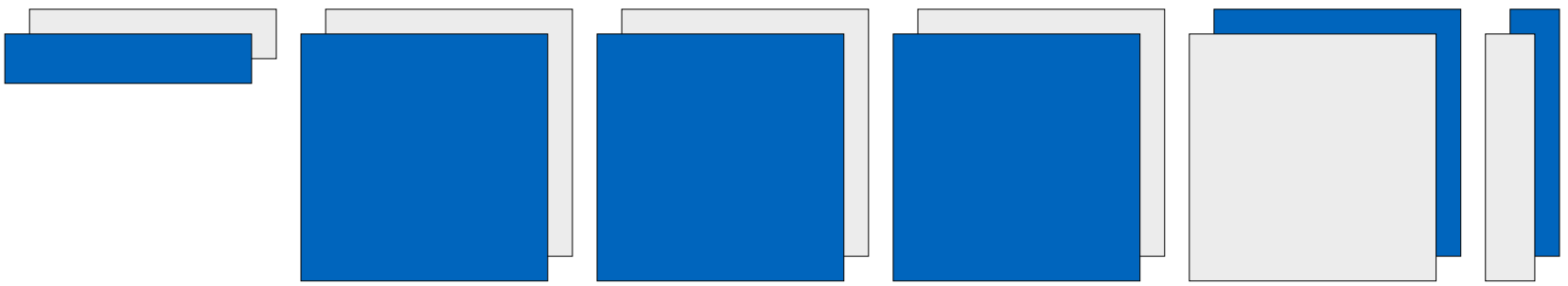
$$\boldsymbol{x}_1 = \boldsymbol{x}_{0,0,0,0,0,0} = \boldsymbol{A}_1^{(0)} \boldsymbol{A}_2^{(0)} \boldsymbol{A}_3^{(0)} \boldsymbol{A}_4^{(0)} \boldsymbol{A}_5^{(0)} \boldsymbol{A}_6^{(0)}$$



Matrix Product States: Example

Approximate vector of dimension $2^6 = 64$
by an MPS related to $N = 6$ sites.

$$\boldsymbol{x}_{60} = \boldsymbol{x}_{1,1,1,1,0,0} = \boldsymbol{A}_{\mathbf{1}}^{(\mathbf{1})} \boldsymbol{A}_{\mathbf{2}}^{(\mathbf{1})} \boldsymbol{A}_{\mathbf{3}}^{(\mathbf{1})} \boldsymbol{A}_{\mathbf{4}}^{(\mathbf{1})} \boldsymbol{A}_{\mathbf{5}}^{(\mathbf{0})} \boldsymbol{A}_{\mathbf{6}}^{(\mathbf{0})}$$



$\boldsymbol{A}_{\mathbf{1}}^{(\mathbf{1})}$ $\boldsymbol{A}_{\mathbf{2}}^{(\mathbf{1})}$ $\boldsymbol{A}_{\mathbf{3}}^{(\mathbf{1})}$ $\boldsymbol{A}_{\mathbf{4}}^{(\mathbf{1})}$ $\boldsymbol{A}_{\mathbf{5}}^{(\mathbf{0})}$ $\boldsymbol{A}_{\mathbf{6}}^{(\mathbf{0})}$

Matrix Product States

$$x = \sum_{i_1,\ldots,i_N} \left(\boldsymbol{A}_{\mathbf{1}}^{(i_1)} \cdot \boldsymbol{A}_{\mathbf{2}}^{(i_2)} \cdots \boldsymbol{A}_{\boldsymbol{N}}^{(i_N)} \right) \boldsymbol{e}_{i_1 i_2 \ldots i_N} \ .$$

- Parameterization uses pair of $D \times D$ matrices along each mode.
- Matrix size D is called bond dimension or rank.
- MPS representation has only $2ND^2$ instead of 2^N entries. [2] ✓

Matrix Product States

$$x = \sum_{i_1,\ldots,i_N} \left(\boldsymbol{A}_1^{(i_1)} \cdot \boldsymbol{A}_2^{(i_2)} \cdots \boldsymbol{A}_N^{(i_N)} \right) \boldsymbol{e}_{i_1 i_2 \ldots i_N} \, .$$

- Parameterization uses pair of $D \times D$ matrices along each mode.
- Matrix size D is called bond dimension or rank.
- MPS representation has only $2ND^2$ instead of 2^N entries. (2) ✓
- Ground states of 1D systems are well described by MPS: D scales only polynomially in N. (3) ✓

Matrix Product States

$$x = \sum_{i_1,\ldots,i_N} \left(\boldsymbol{A}_1^{(i_1)} \cdot \boldsymbol{A}_2^{(i_2)} \cdots \boldsymbol{A}_N^{(i_N)} \right) \boldsymbol{e}_{i_1 i_2 \ldots i_N} \, .$$

- Parameterization uses pair of $D \times D$ matrices along each mode.
- Matrix size D is called bond dimension or rank.
- MPS representation has only $2ND^2$ instead of 2^N entries. (2) ✓
- Ground states of 1D systems are well described by MPS: D scales only polynomially in N. (3) ✓

Ground state of the Ising-ZZ model of $N = 60$ sites:

ϵ	D	DOF
10^{-4}	4	$1,872$
10^{-6}	8	$7,456$
10^{-8}	12	$16,752$
10^{-10}	18	$37,656$
10^{-12}	25	$72,600$

Traditional treatment

$$2^{60} = 1.1529 \times 10^{18}$$

approx. 4.6 Exabyte!!!

Computation of Inner Products

$$
\begin{aligned}
\boldsymbol{x}^{\mathrm{H}}\boldsymbol{y} &= \sum_{i_1,\ldots,i_N} \left(\bar{\boldsymbol{A}}_{\boldsymbol{1}}^{(\boldsymbol{i_1})} \cdots \bar{\boldsymbol{A}}_{\boldsymbol{N}}^{(\boldsymbol{i_N})}\right) \cdot \left(\boldsymbol{B}_{\boldsymbol{1}}^{(\boldsymbol{i_1})} \cdots \boldsymbol{B}_{\boldsymbol{N}}^{(\boldsymbol{i_N})}\right) \\
&= \sum_{i_1,\ldots,i_N} \left(\sum_{k_j} \overline{a_{1;k_1,k_2}^{(i_1)}} \cdots \overline{a_{N;k_N,k_1}^{(i_N)}}\right) \left(\sum_{m_j} b_{1;m_1,m_2}^{(i_1)} \cdots b_{N;m_N,m_1}^{(i_N)}\right)
\end{aligned}
$$

$\Longrightarrow$ Find an efficient ordering of the summations m_j, k_j, i_j.

Requires $\mathcal{O}(ND^3)$ operations.

Computation of the Sum of Two MPS

- Sum of two MPS vectors

$$x = \sum_{i_1,\ldots,i_N} \left(A_1^{(i_1)} A_2^{(i_2)} \ldots A_N^{(i_N)} \right) e_{i_1 \ldots i_N}$$

$$y = \sum_{i_1,\ldots,i_N} \left(B_1^{(i_1)} B_2^{(i_2)} \ldots B_N^{(i_N)} \right) e_{i_1 \ldots i_N}$$

$$x + y =$$

$$= \sum_{i_1,\ldots,i_N} \begin{pmatrix} A_1^{(i_1)} & B_1^{(i_1)} \end{pmatrix} \begin{pmatrix} A_2^{(i_2)} & 0 \\ 0 & B_2^{(i_2)} \end{pmatrix} \cdots \begin{pmatrix} A_N^{(i_N)} \\ B_N^{(i_N)} \end{pmatrix} e_{i_1 \ldots i_N}$$

- Adding MPS leads to an **increase** of the bond dimensions:

$$\boxed{\text{“}MPS_D + MPS_{D'} = MPS_{D+D'}\text{”}}$$

Computation of Matrix-Vector Products

- The MPS concept

$$x_{i_1,i_2,\ldots,i_N} = \left(\boldsymbol{A}_{\boldsymbol{1}}^{(\boldsymbol{i_1})} \boldsymbol{A}_{\boldsymbol{2}}^{(\boldsymbol{i_2})} \cdots \boldsymbol{A}_{\boldsymbol{N}}^{(\boldsymbol{i_N})} \right)$$

can be generalized to operators:

$$O_{\substack{i_1,\ldots,i_N \\ j_1,\ldots,j_N}} = \left(\boldsymbol{W}_{\boldsymbol{1}}^{(\boldsymbol{i_1},\boldsymbol{j_1})} \boldsymbol{W}_{\boldsymbol{2}}^{(\boldsymbol{i_2},\boldsymbol{j_2})} \cdots \boldsymbol{W}_{\boldsymbol{N}}^{(\boldsymbol{i_N},\boldsymbol{j_N})} \right) .$$

- Application of such an MPO to an MPS leads to

$$\left[\left(\sum_{j_1} \boldsymbol{W}_{\boldsymbol{1}}^{(\boldsymbol{i_1},\boldsymbol{j_1})} \otimes \boldsymbol{A}_{\boldsymbol{1}}^{(\boldsymbol{j_1})} \right) \cdots \left(\sum_{j_N} \boldsymbol{W}_{\boldsymbol{N}}^{(\boldsymbol{i_N},\boldsymbol{j_N})} \otimes \boldsymbol{A}_{\boldsymbol{N}}^{(\boldsymbol{j_N})} \right) \right] ,$$

i.e., an MPS of larger bond dimension:

$$\boxed{\text{“}MPO_L \times MPS_D = MPS_{L \cdot D}\text{”}}$$

Variational Ground-State Computations

Variational ansatz:

$$\min_{\boldsymbol{x}\neq\boldsymbol{0}} \frac{\boldsymbol{x}^{\mathrm{H}}\boldsymbol{H}\boldsymbol{x}}{\boldsymbol{x}^{\mathrm{H}}\boldsymbol{x}} \approx \min_{\boldsymbol{x}_{\mathrm{MPS}}} \frac{\boldsymbol{x}_{\mathrm{MPS}}^{\mathrm{H}}\boldsymbol{H}\boldsymbol{x}_{\mathrm{MPS}}}{\boldsymbol{x}_{\mathrm{MPS}}^{\mathrm{H}}\boldsymbol{x}_{\mathrm{MPS}}}$$

Find optimal matrix pairs $\boldsymbol{A}_j^{(i_j)}$ for all sites j:

$$\min_{\boldsymbol{A}_j^{(i_j)}} \frac{\left(\sum_{i_j}\left(\boldsymbol{A}_1^{(i_1)}\boldsymbol{A}_2^{(i_2)}\cdots\boldsymbol{A}_N^{(i_N)}\right)\boldsymbol{e}_i\right)^{\mathrm{H}}\boldsymbol{H}\left(\sum_{i_j}\left(\boldsymbol{A}_1^{(i_1)}\boldsymbol{A}_2^{(i_2)}\cdots\boldsymbol{A}_N^{(i_N)}\right)\boldsymbol{e}_i\right)}{\left(\sum_{i_j}\left(\boldsymbol{A}_1^{(i_1)}\boldsymbol{A}_2^{(i_2)}\cdots\boldsymbol{A}_N^{(i_N)}\right)\boldsymbol{e}_i\right)^{\mathrm{H}}\left(\sum_{i_j}\left(\boldsymbol{A}_1^{(i_1)}\boldsymbol{A}_2^{(i_2)}\cdots\boldsymbol{A}_N^{(i_N)}\right)\boldsymbol{e}_i\right)}$$

Variational Ground-State Computations

Variational ansatz:

$$\min_{\boldsymbol{x}\neq\boldsymbol{0}} \frac{\boldsymbol{x}^{\mathrm{H}}\boldsymbol{H}\boldsymbol{x}}{\boldsymbol{x}^{\mathrm{H}}\boldsymbol{x}} \approx \min_{\boldsymbol{x}_{\mathrm{MPS}}} \frac{\boldsymbol{x}_{\mathrm{MPS}}^{\mathrm{H}}\boldsymbol{H}\boldsymbol{x}_{\mathrm{MPS}}}{\boldsymbol{x}_{\mathrm{MPS}}^{\mathrm{H}}\boldsymbol{x}_{\mathrm{MPS}}}$$

Use alternating least squares, variational MPS (vMPS)

$$\min_{\boldsymbol{X}_1^{(i_1)}} \frac{\left(\sum_{i_j}\left(\boldsymbol{X}_1^{(i_1)}\boldsymbol{A}_2^{(i_2)}\cdots\boldsymbol{A}_N^{(i_N)}\right)\boldsymbol{e}_i\right)^{\mathrm{H}}\boldsymbol{H}\left(\sum_{i_j}\left(\boldsymbol{X}_1^{(i_1)}\boldsymbol{A}_2^{(i_2)}\cdots\boldsymbol{A}_N^{(i_N)}\right)\boldsymbol{e}_i\right)}{\left(\sum_{i_j}\left(\boldsymbol{X}_1^{(i_1)}\boldsymbol{A}_2^{(i_2)}\cdots\boldsymbol{A}_N^{(i_N)}\right)\boldsymbol{e}_i\right)^{\mathrm{H}}\left(\sum_{i_j}\left(\boldsymbol{X}_1^{(i_1)}\boldsymbol{A}_2^{(i_2)}\cdots\boldsymbol{A}_N^{(i_N)}\right)\boldsymbol{e}_i\right)}$$

$$= \min_{\boldsymbol{X}_1} \frac{\boldsymbol{X}_1^{\mathrm{H}}\boldsymbol{H}_1\boldsymbol{X}_1}{\boldsymbol{X}_1^{\mathrm{H}}\boldsymbol{X}_1}$$

Eigenvalue problem of size $(2D^2 \times 2D^2)$.

Variational Ground-State Computations

Variational ansatz:

$$\min_{\boldsymbol{x}\neq\boldsymbol{0}} \frac{\boldsymbol{x}^{\mathrm{H}}\boldsymbol{H}\boldsymbol{x}}{\boldsymbol{x}^{\mathrm{H}}\boldsymbol{x}} \approx \min_{\boldsymbol{x}_{\mathrm{MPS}}} \frac{\boldsymbol{x}_{\mathrm{MPS}}^{\mathrm{H}}\boldsymbol{H}\boldsymbol{x}_{\mathrm{MPS}}}{\boldsymbol{x}_{\mathrm{MPS}}^{\mathrm{H}}\boldsymbol{x}_{\mathrm{MPS}}}$$

Use alternating least squares, variational MPS (vMPS)

$$\min_{\boldsymbol{X}_2^{(i_2)}} \frac{\left(\sum_{i_j}\left(\boldsymbol{A}_1^{(i_1)}\boldsymbol{X}_2^{(i_2)}\cdots\boldsymbol{A}_N^{(i_N)}\right)\boldsymbol{e}_i\right)^{\mathrm{H}}\boldsymbol{H}\left(\sum_{i_j}\left(\boldsymbol{A}_1^{(i_1)}\boldsymbol{X}_2^{(i_2)}\cdots\boldsymbol{A}_N^{(i_N)}\right)\boldsymbol{e}_i\right)}{\left(\sum_{i_j}\left(\boldsymbol{A}_1^{(i_1)}\boldsymbol{X}_2^{(i_2)}\cdots\boldsymbol{A}_N^{(i_N)}\right)\boldsymbol{e}_i\right)^{\mathrm{H}}\left(\sum_{i_j}\left(\boldsymbol{A}_1^{(i_1)}\boldsymbol{X}_2^{(i_2)}\cdots\boldsymbol{A}_N^{(i_N)}\right)\boldsymbol{e}_i\right)}$$

$$= \min_{\boldsymbol{X}_2} \frac{\boldsymbol{X}_2^{\mathrm{H}}\boldsymbol{H}_2\boldsymbol{X}_2}{\boldsymbol{X}_2^{\mathrm{H}}\boldsymbol{X}_2}$$

Eigenvalue problem of size $(2D^2 \times 2D^2)$.

Outline

Matrix product states:

$$x = \sum_{i_1,\ldots,i_N} \left(\boldsymbol{A}_{\mathbf{1}}^{(\boldsymbol{i_1})} \cdot \boldsymbol{A}_{\mathbf{2}}^{(\boldsymbol{i_2})} \cdots \boldsymbol{A}_{\boldsymbol{N}}^{(\boldsymbol{i_N})} \right) \boldsymbol{e}_{\boldsymbol{i_1 i_2 \ldots i_N}} \cdot$$

- MPS as data-sparse representation format to approximate ground states.
- Computations with MPS:
  - inner products,
  - sum of MPS: "$MPS_D + MPS_{D'} = MPS_{D+D'}$"
  - matrix-vector products: "$MPO_L \times MPS_D = MPS_{L \cdot D}$"
- Variational ground-state computations.

Outline

Matrix product states:

$$x = \sum_{i_1,\ldots,i_N} \left(\boldsymbol{A_1^{(i_1)}} \cdot \boldsymbol{A_2^{(i_2)}} \cdots \boldsymbol{A_N^{(i_N)}} \right) \boldsymbol{e_{i_1 i_2 \ldots i_N}} \cdot$$

- MPS as data-sparse representation format to approximate ground states.
- Computations with MPS:
  - inner products,
  - sum of MPS: "$MPS_D + MPS_{D'} = MPS_{D+D'}$"
  - matrix-vector products: "$MPO_L \times MPS_D = MPS_{L \cdot D}$"
- Variational ground-state computations.
- **Two approaches:**
  - Lanczos method in the MPS format,
  - Connections between symmetries and MPS representations.

The Lanczos Algorithm

Algorithm: Lanczos iteration for Hermitian matrix $\boldsymbol{H}$

Choose initial guess $\boldsymbol{r_0} \neq \boldsymbol{0}$
Define $\beta_0 = \|\boldsymbol{r_0}\|$ and $\boldsymbol{q_0} = \boldsymbol{0}$
for $j = 1, 2, \ldots, m$ **do**

$$\boldsymbol{q_j} = \boldsymbol{r_{j-1}} / \beta_{j-1}$$
$$\boldsymbol{u_j} = \boldsymbol{H q_j}$$
$$\alpha_j = \boldsymbol{q_j^H u_j}$$
$$\boldsymbol{r_j} = \boldsymbol{u_j} - \alpha_j \boldsymbol{q_j} - \beta_{j-1} \boldsymbol{q_{j-1}}$$
$$\beta_j = \|\boldsymbol{r_j}\|$$

$$\boldsymbol{T_m} = \mathrm{tridiag}(\boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$
$$(\lambda_{\min}, \boldsymbol{y}) = \mathrm{eig}(\boldsymbol{T_m})$$
$$\boldsymbol{v} = \boldsymbol{Q y} = (\boldsymbol{q_1}, \ldots, \boldsymbol{q_m}) \boldsymbol{y}$$

The Lanczos Algorithm

Algorithm: Lanczos iteration for Hermitian matrix $\boldsymbol{H}$

Choose initial guess $\boldsymbol{r_0} \neq \boldsymbol{0}$
Define $\beta_0 = \|\boldsymbol{r_0}\|$ and $\boldsymbol{q_0} = \boldsymbol{0}$
for $j = 1, 2, \ldots, m$ **do**

$$\boldsymbol{q_j} = \boldsymbol{r_{j-1}} / \beta_{j-1}$$
$$\boldsymbol{u_j} = \boldsymbol{H q_j}$$
$$\alpha_j = \boldsymbol{q_j^H u_j}$$
$$\boldsymbol{r_j} = \boldsymbol{u_j} - \alpha_j \boldsymbol{q_j} - \beta_{j-1} \boldsymbol{q_{j-1}}$$
$$\beta_j = \|\boldsymbol{r_j}\|$$

$$\boldsymbol{T_m} = \mathrm{tridiag}(\boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$
$$(\lambda_{\min}, \boldsymbol{y}) = \mathrm{eig}(\boldsymbol{T_m})$$
$$\boldsymbol{v} = \boldsymbol{Q y} = (\boldsymbol{q_1}, \ldots, \boldsymbol{q_m}) \boldsymbol{y}$$

Computational tasks:

The Lanczos Algorithm

Algorithm: Lanczos iteration for Hermitian matrix $\boldsymbol{H}$

Choose initial guess $\boldsymbol{r_0} \neq \boldsymbol{0}$
Define $\beta_0 = \|\boldsymbol{r_0}\|$ and $\boldsymbol{q_0} = \boldsymbol{0}$
for $j = 1, 2, \ldots, m$ **do**

$$\boldsymbol{q_j} = \boldsymbol{r_{j-1}} / \beta_{j-1}$$
$$\boldsymbol{u_j} = \boldsymbol{H q_j}$$
$$\alpha_j = \boldsymbol{q_j^H u_j}$$
$$\boldsymbol{r_j} = \boldsymbol{u_j} - \alpha_j \boldsymbol{q_j} - \beta_{j-1} \boldsymbol{q_{j-1}}$$
$$\beta_j = \|\boldsymbol{r_j}\|$$

$$\boldsymbol{T_m} = \mathrm{tridiag}(\boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$
$$(\lambda_{\min}, \boldsymbol{y}) = \mathrm{eig}(\boldsymbol{T_m})$$
$$\boldsymbol{v} = \boldsymbol{Q y} = (\boldsymbol{q_1}, \ldots, \boldsymbol{q_m}) \boldsymbol{y}$$

Computational tasks:

1. Application of the matrix to a vector

The Lanczos Algorithm

Algorithm: Lanczos iteration for Hermitian matrix H

Choose initial guess $r_0 \neq 0$
Define $\beta_0 = \|r_0\|$ and $q_0 = 0$
for $j = 1, 2, \ldots, m$ **do**

$$q_j = r_{j-1}/\beta_{j-1}$$
$$u_j = H q_j$$
$$\alpha_j = q_j^H u_j$$
$$r_j = u_j - \alpha_j q_j - \beta_{j-1} q_{j-1}$$
$$\beta_j = \|r_j\|$$

$$T_m = \text{tridiag}(\beta, \alpha, \beta)$$
$$(\lambda_{\min}, y) = \text{eig}(T_m)$$
$$v = Qy = (q_1, \ldots, q_m)y$$

Computational tasks:

1. Application of the matrix to a vector
2. Computation of inner products

The Lanczos Algorithm

Algorithm: Lanczos iteration for Hermitian matrix H

Choose initial guess $r_0 \neq 0$
Define $\beta_0 = \|r_0\|$ and $q_0 = 0$
for $j = 1, 2, \ldots, m$ **do**

$$q_j = r_{j-1}/\beta_{j-1}$$
$$u_j = H q_j$$
$$\alpha_j = q_j^H u_j$$
$$r_j = u_j - \alpha_j q_j - \beta_{j-1} q_{j-1}$$
$$\beta_j = \|r_j\|$$

$$T_m = \mathrm{tridiag}(\beta, \alpha, \beta)$$
$$(\lambda_{\min}, y) = \mathrm{eig}(T_m)$$
$$v = Qy = (q_1, \ldots, q_m)y$$

Computational tasks:

1. Application of the matrix to a vector
2. Computation of inner products
3. Sums and linear combinations of vectors

Augmentation of the bond dimensions

- Increase of the bond dimension during the Lanczos algorithm

Lanczos algorithm	Bond dimension
for $j = 1, 2, \ldots, m$	
$\boldsymbol{q_j} = \boldsymbol{r_{j-1}}/\beta_{j-1}$	$D^{(j)}$
$\boldsymbol{u_j} = \boldsymbol{H}\boldsymbol{q_j}$	$LD^{(j)}$
$\alpha_j = \boldsymbol{q_j^{\mathrm{H}}}\boldsymbol{u_j}$	
$\boldsymbol{r_j} = \boldsymbol{H}\boldsymbol{q_j} - \alpha_j\boldsymbol{q_j} - \beta_{j-1}\boldsymbol{q_{j-1}}$	$LD^{(j)} + D^{(j)} + D^{(j-1)} =$
$\beta_j = \lVert\boldsymbol{r_j}\rVert$	$= D^{(j+1)}$
$\boldsymbol{v} = \boldsymbol{Q}\boldsymbol{y} = (\boldsymbol{q_1}, \ldots, \boldsymbol{q_m})\boldsymbol{y}$	$D^{(1)} + D^{(2)} + \cdots + D^{(m)}$

Augmentation of the bond dimensions

- Increase of the bond dimension during the Lanczos algorithm

Lanczos algorithm	Bond dimension
for $j = 1, 2, \ldots, m$	
$\boldsymbol{q_j} = \boldsymbol{r_{j-1}} / \beta_{j-1}$	$D^{(j)}$
$\boldsymbol{u_j} = \boldsymbol{H}\boldsymbol{q_j}$	$LD^{(j)}$
$\alpha_j = \boldsymbol{q_j^{\mathrm{H}}}\boldsymbol{u_j}$	
$\boldsymbol{r_j} = \boldsymbol{H}\boldsymbol{q_j} - \alpha_j \boldsymbol{q_j} - \beta_{j-1}\boldsymbol{q_{j-1}}$	$LD^{(j)} + D^{(j)} + D^{(j-1)} =$
$\beta_j = \lVert \boldsymbol{r_j} \rVert$	$= D^{(j+1)}$
$\boldsymbol{v} = \boldsymbol{Q}\boldsymbol{y} = (\boldsymbol{q_1}, \ldots, \boldsymbol{q_m})\boldsymbol{y}$	$D^{(1)} + D^{(2)} + \cdots + D^{(m)}$

- Solution: keep the ranks $D^{(j)}$ bounded
  $\Longrightarrow$ Projection required

Compression/Projection of MPS

$$\mathcal{P}_D : \boldsymbol{y}_{MPS_{D'}} \mapsto \underset{\boldsymbol{x}_{MPS_D}}{\arg\min} \left\| \boldsymbol{y}_{MPS_{D'}} - \boldsymbol{x}_{MPS_D} \right\|_2^2 =$$

$$\underset{\boldsymbol{A}_j^{(i_j)}}{\arg\min} \left\| \left[\left(\boldsymbol{B}_1^{(i_1)} \cdots \boldsymbol{B}_N^{(i_N)} \right) - \left(\boldsymbol{A}_1^{(i_1)} \cdots \boldsymbol{A}_N^{(i_N)} \right) \right]_{i_1,\ldots,i_N} \right\|_2^2$$

SVD-based truncation:

- Suppose $\boldsymbol{A_1}, \ldots, \boldsymbol{A_{r-1}}$ are already constructed. $\boldsymbol{B_r}$ is $D \times D'$

$$\begin{pmatrix} \boldsymbol{B}_r^{(0)} \\ \boldsymbol{B}_r^{(1)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{U}_r^{(0)} \\ \boldsymbol{U}_r^{(1)} \end{pmatrix} \boldsymbol{\Sigma}_r \boldsymbol{V}_r^{\mathbf{H}} \overset{\text{trunc.}}{\approx} \begin{pmatrix} \tilde{\boldsymbol{U}}_r^{(0)} \\ \tilde{\boldsymbol{U}}_r^{(1)} \end{pmatrix} \tilde{\boldsymbol{\Sigma}}_r \tilde{\boldsymbol{V}}_r^{\mathbf{H}} .$$

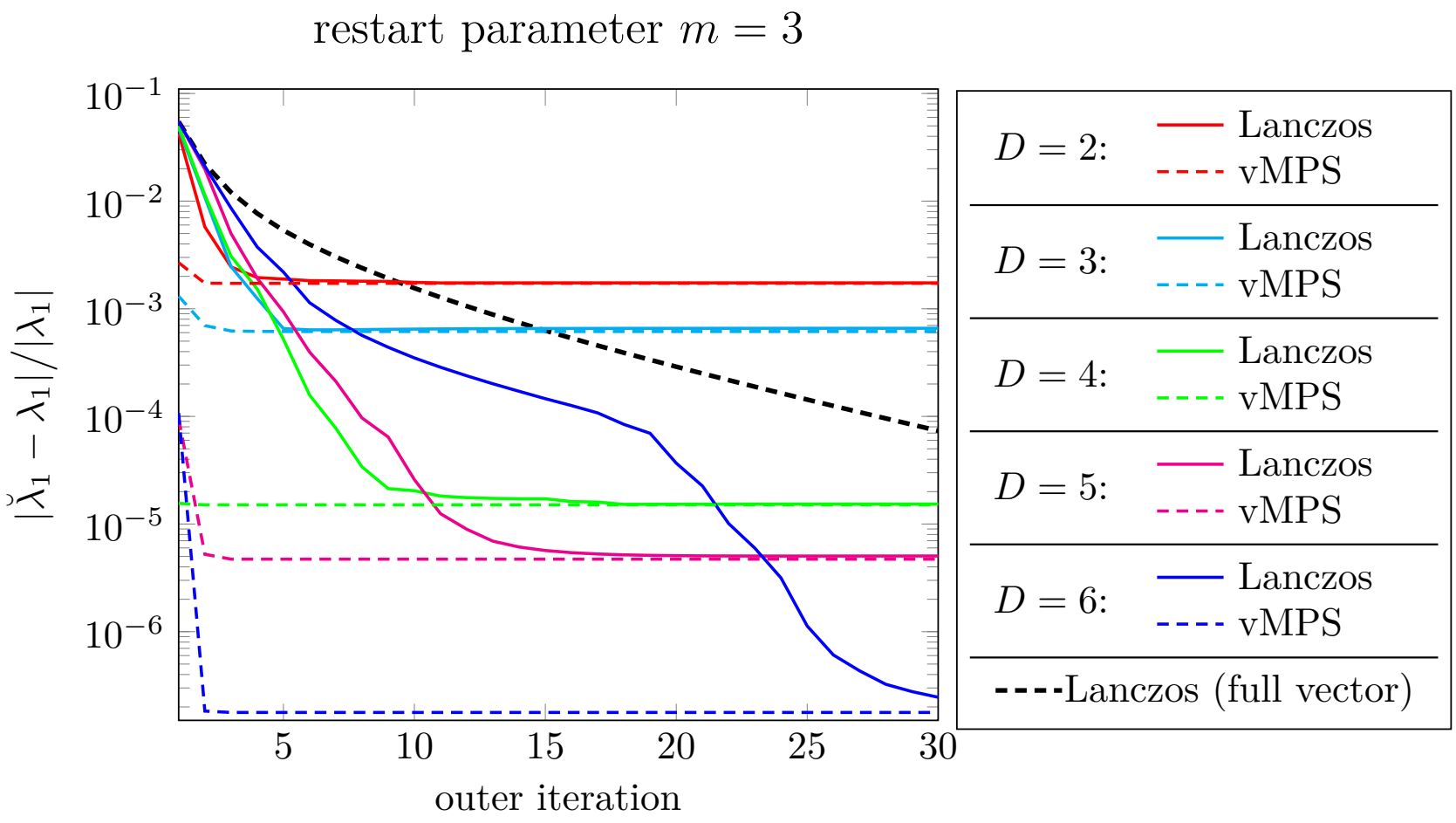
- Define $\boldsymbol{A}_r^{(i_r)} = \tilde{\boldsymbol{U}}_r^{(i_r)}$, a $D \times D$ matrix.
- Proceed with the $D \times D'$ matrix $\tilde{\boldsymbol{\Sigma}}_r \tilde{\boldsymbol{V}}_r^{\mathbf{H}} \boldsymbol{B}_{r+1}^{(i_{r+1})}$.

Restarted Lanczos

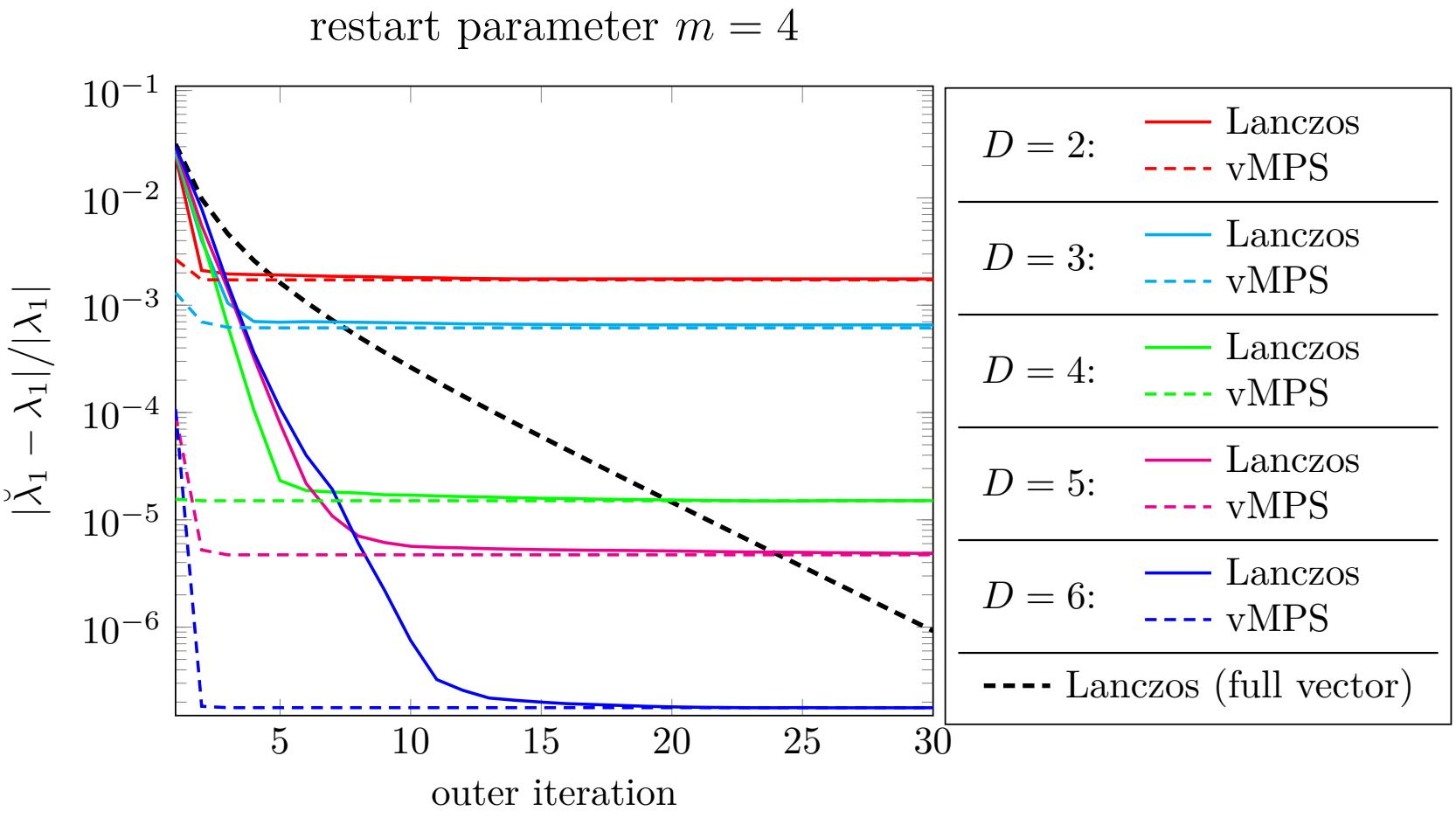
Algorithm: Restarted Lanczos iteration

Choose initial guess $\boldsymbol{r_0} \neq \boldsymbol{0}$;
for $iter = 1, 2, \ldots$ **do**
  Define $\beta_0 = \|\boldsymbol{r_0}\|$ and $\boldsymbol{q_0} = \boldsymbol{0}$;
  for $j = 1, 2, \ldots, m$ **do**
    $\boldsymbol{q_j} = \boldsymbol{r_{j-1}} / \beta_{j-1}$;
    $\boldsymbol{u_j} = \boldsymbol{H q_j}$;
    $\alpha_j = \boldsymbol{q_j}^{\mathrm{H}} \boldsymbol{u_j}$;
    $\boldsymbol{r_j} = \boldsymbol{u_j} - \alpha_j \boldsymbol{q_j} - \beta_{j-1} \boldsymbol{q_{j-1}}$;
    $\beta_j = \|\boldsymbol{r_j}\|$;
  $\boldsymbol{T_m} = \mathrm{tridiag}(\boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\beta})$;
  $(\lambda_{\min}, \boldsymbol{y}) = \mathrm{eig}(\boldsymbol{T_m})$;
  $\boldsymbol{v} = \boldsymbol{Q y} = (\boldsymbol{q_1}, \ldots, \boldsymbol{q_m}) \boldsymbol{y}$;
  $\boldsymbol{r_0} = \mathcal{P}_D(\boldsymbol{v})$

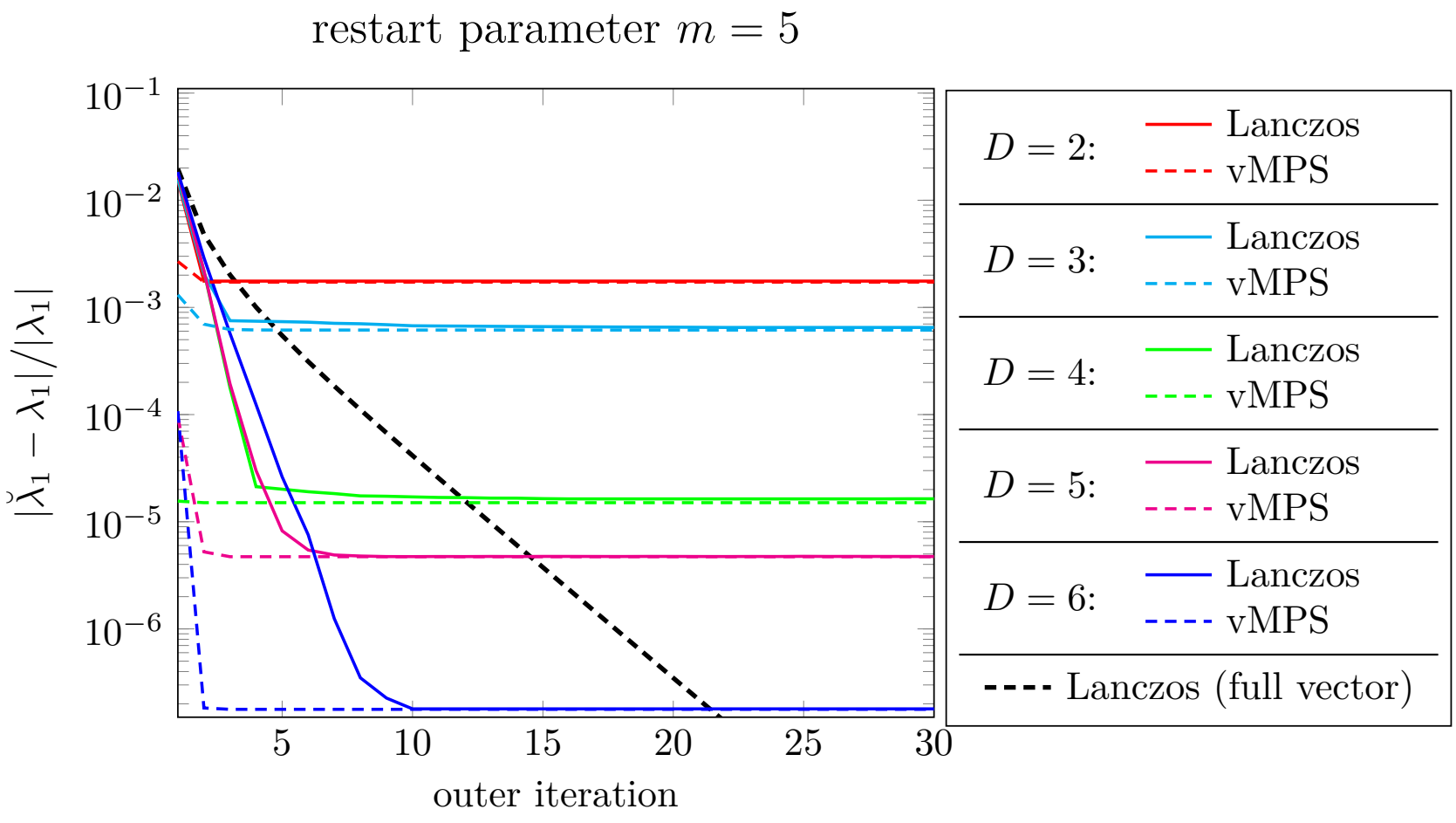
Numerical Results (Ising Hamiltonian $N = 25$)



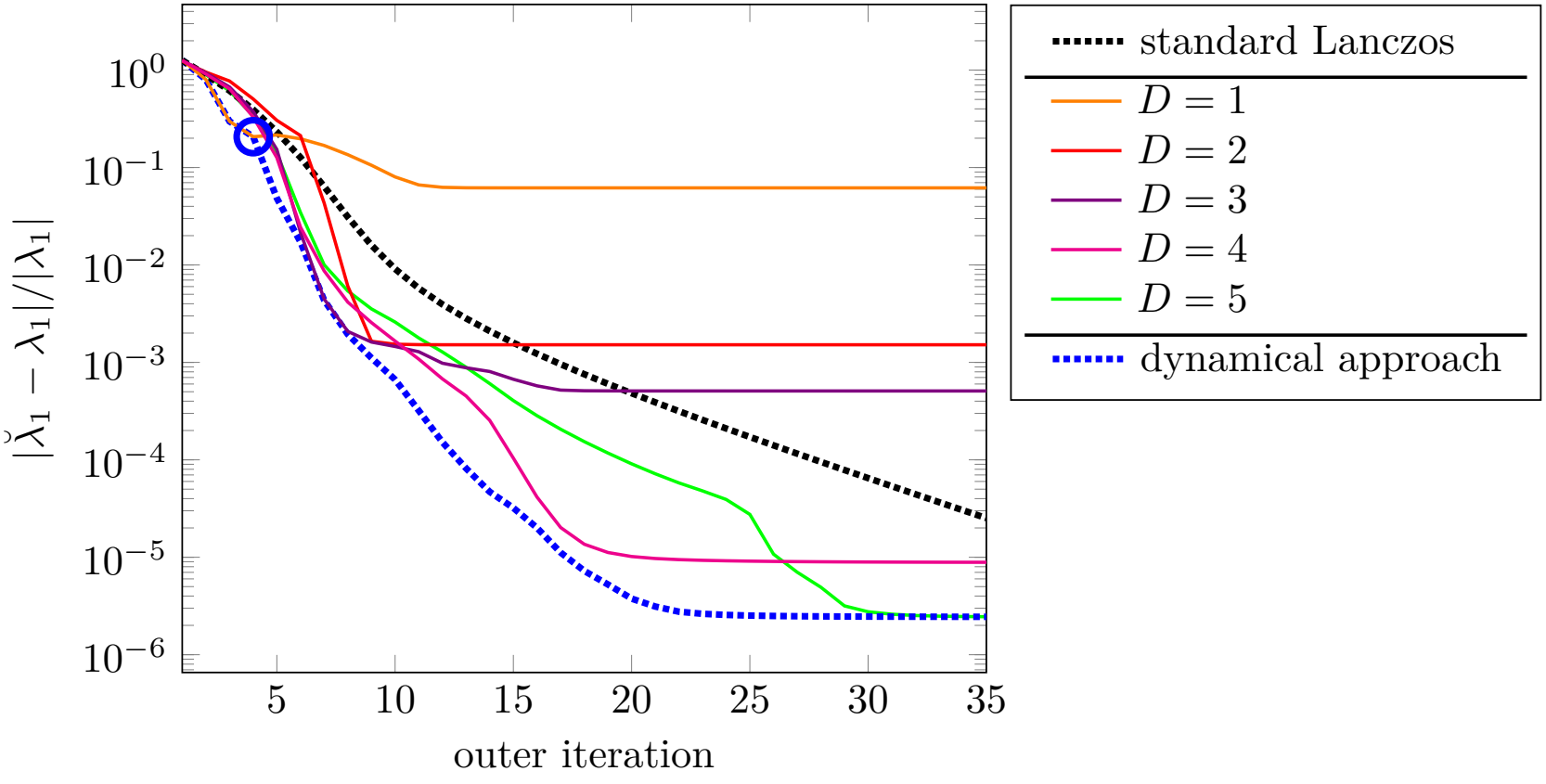
Numerical Results (Ising Hamiltonian $N = 25$)



Numerical Results (Ising Hamiltonian $N = 25$)



Dynamical Adaptation of the Bond Dimension



Matrix and Vector Symmetries

- $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is persymmetric, if $a_{i,j} = a_{n+1-j,n+1-i}$:

$$\boldsymbol{A} = \boldsymbol{J}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{J} = \begin{pmatrix} & & 1 \\ & \iddots & \\ 1 & & \end{pmatrix} \boldsymbol{A}^{\mathrm{T}} \underbrace{\begin{pmatrix} & & 1 \\ & \iddots & \\ 1 & & \end{pmatrix}}_{=:\boldsymbol{J}}.$$

- $\boldsymbol{A}$ is symmetric persymmetric, if $a_{i,j} = a_{j,i} = a_{n+1-i,n+1-j}$.

Matrix and Vector Symmetries

- $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is persymmetric, if $a_{i,j} = a_{n+1-j,n+1-i}$:

$$\boldsymbol{A} = \boldsymbol{J}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{J} = \begin{pmatrix} & & 1 \\ & \iddots & \\ 1 & & \end{pmatrix} \boldsymbol{A}^{\mathrm{T}} \underbrace{\begin{pmatrix} & & 1 \\ & \iddots & \\ 1 & & \end{pmatrix}}_{=:\boldsymbol{J}}.$$

- $\boldsymbol{A}$ is symmetric persymmetric, if $a_{i,j} = a_{j,i} = a_{n+1-i,n+1-j}$.
- A vector $\boldsymbol{v} \in \mathbb{R}^n$ is called symmetric, if $v_i = v_{n+1-i}$:

$$\boldsymbol{v} = \boldsymbol{J}\boldsymbol{v}$$

- A vector $\boldsymbol{v} \in \mathbb{R}^n$ is called skew-symmetric, if $v_i = -v_{n+1-i}$:

$$\boldsymbol{v} = -\boldsymbol{J}\boldsymbol{v}$$

Symmetries of Spin-$\frac{1}{2}$ Hamiltonians

Symmetry of Spin Hamiltonians

Hamiltonians of typical physical models (Ising-ZZ, various Heisenberg models) are real-valued and symmetric persymmetric.

Symmetries of Spin-$\frac{1}{2}$ Hamiltonians

Symmetry of Spin Hamiltonians

Hamiltonians of typical physical models (Ising-ZZ, various Heisenberg models) are real-valued and symmetric persymmetric.

Eigenvectors of Symmetric Persymmetric Matrices ([CB76])

The eigenvectors of a symmetric persymmetric matrix are either symmetric or skew-symmetric.

Symmetries of Spin-$\frac{1}{2}$ Hamiltonians

Symmetry of Spin Hamiltonians

Hamiltonians of typical physical models (Ising-ZZ, various Heisenberg models) are real-valued and symmetric persymmetric.

Eigenvectors of Symmetric Persymmetric Matrices ([CB76])

The eigenvectors of a symmetric persymmetric matrix are either symmetric or skew-symmetric.

Ground States of Spin Hamiltonians

Ground states of typical physical models (Ising-ZZ, various Heisenberg models) are either symmetric or skew-symmetric.

MPS Representation of Symmetric Vectors

$$x_{i_1,i_2,i_3,\ldots,i_{N-1},i_N} = x_{\overline{i_1},\overline{i_2},\overline{i_3},\ldots,\overline{i_{N-1}},\overline{i_N}} \iff \boldsymbol{J}\boldsymbol{x} = \boldsymbol{x} \qquad \text{(BF)}$$

Flip-operator $\bar{i} := 1 - i$.

Example (N=3): $\boldsymbol{x} = \begin{pmatrix} a & b & c & d & d & c & b & a \end{pmatrix}^{\mathrm{T}}$

Theorem: MPS Representation of the Vector Symmetry ([Huckle13])

1. If all MPS matrix pairs $(\boldsymbol{A}_j^{(0)}, \boldsymbol{A}_j^{(1)})$ are connected via

$$\boldsymbol{A}_j^{(i_j)} = \boldsymbol{U}_j \boldsymbol{A}_j^{(\bar{i}_j)} \boldsymbol{U}_{j+1} \ , \ j = 1, \ldots, N \ , \qquad (1)$$

   with involutions $\boldsymbol{U}_j$ ($\boldsymbol{U}_j^2 = \boldsymbol{I}$), the represented vector is symmetric (BF).
2. Any symmetric vector (BF) can be represented by an MPS fulfilling relations (1).

- Reduction factor: ≈ 2

MPS Representation of Symmetric Vectors

Theorem: Uniqueness Result for Symmetric Vectors ([Huckle13])

Assume that the MPS matrices (over $\mathbb{K}$) are related by

$$\boldsymbol{A_j^{(1)}} = \boldsymbol{U_{j-1}} \boldsymbol{A_j^{(0)}} \boldsymbol{V_j} \ ,$$

with square matrices $\boldsymbol{V_j}$ and $\boldsymbol{U_j}$ of appropriate size ($j = 1, \ldots, N$). If any choice of matrices $\boldsymbol{A_j^{(0)}}$ results in the symmetry of the represented vector $\boldsymbol{x}$, i.e.,

$$\boldsymbol{Jx} \ = \ \boldsymbol{x} \ ,$$

then $\boldsymbol{U_j}$ and $\boldsymbol{V_j}$ are—up to a scalar factor—involutions for all j: $\boldsymbol{U_j^2} = u_j \boldsymbol{I}$ and $\boldsymbol{V_j^2} = v_j \boldsymbol{I}$. Furthermore, $\boldsymbol{U_j} = c_j \cdot \boldsymbol{V_j}$, $j = 1, ..., N$ with constants c_j.

- This theorem gives an argument to use involutions as heuristic.
- An involution has as eigenvalues $\pm 1 \Longrightarrow$ choose $\boldsymbol{U_j} = \boldsymbol{J}$.

Exploiting Symmetries in Computations

- Impose symmetry conditions into variational ground-state search
- Suppose that the desired ground-state vector x is symmetric.

$$\boldsymbol{J}\boldsymbol{x} = \boldsymbol{x} \implies \boldsymbol{x}^{\mathrm{H}} \underbrace{(\boldsymbol{x} - \boldsymbol{J}\boldsymbol{x})}_{=\mathbf{0}} = \boldsymbol{x}^{\mathrm{H}} (\boldsymbol{I} - \boldsymbol{J})\, \boldsymbol{x} = 0 \,.$$

- Therefore,

$$\begin{aligned} \lambda_{\min} \leq \frac{\boldsymbol{y}^{\mathrm{H}} \boldsymbol{H} \boldsymbol{y}}{\boldsymbol{y}^{\mathrm{H}} \boldsymbol{y}} &\leq \frac{\boldsymbol{y}^{\mathrm{H}} \boldsymbol{H} \boldsymbol{y}}{\boldsymbol{y}^{\mathrm{H}} \boldsymbol{y}} + \rho \frac{\boldsymbol{y}^{\mathrm{H}} (\boldsymbol{I} - \boldsymbol{J})\, \boldsymbol{y}}{\boldsymbol{y}^{\mathrm{H}} \boldsymbol{y}} \\ &= \frac{\boldsymbol{y}^{\mathrm{H}} (\boldsymbol{H} + \rho(\boldsymbol{I} - \boldsymbol{J}))\, \boldsymbol{y}}{\boldsymbol{y}^{\mathrm{H}} \boldsymbol{y}} \,. \end{aligned}$$

- Apply ground-state algorithm (e.g., vMPS) to the modified Hamiltonian

$$\boldsymbol{H}(\rho) := \boldsymbol{H} + \rho(\boldsymbol{I} - \boldsymbol{J}) \quad \text{for some} \quad \rho > 0 \,.$$

Numerical Results: Number of Iterations

Table : Ising-ZZ model of $N = 60$ spins: Number of ALS sweeps until convergence.

	D								
ρ	2	3	4	5	6	7	8	9	10
0	9	8	6	6	4	4	6	6	3
0.1	7	7	5	6	4	4	6	6	3
0.5	5	6	5	5	4	4	5	6	3
1	5	6	5	5	4	4	5	6	3
2	5	5	5	5	4	4	5	6	3
10	5	4	5	5	4	4	8	5	3
100	5	4	5	5	4	4	9	6	3

Numerical Results: Number of Iterations

Table : Heisenberg-XXX model of $N = 100$ spins: Number of ALS sweeps until convergence.

	D								
ρ	2	3	4	5	6	7	8	9	10
0	66	71	28	13	16	35	13	11	22
0.1	48	24	22	8	12	43	9	8	27
0.5	13	10	14	6	9	31	9	8	17
1	8	10	15	5	8	20	9	8	15
2	6	14	16	5	8	20	9	8	14
10	4	6	46	5	10	9	9	8	9
100	5	5	29	10	9	9	9	9	10

Numerical Studies: Convergence

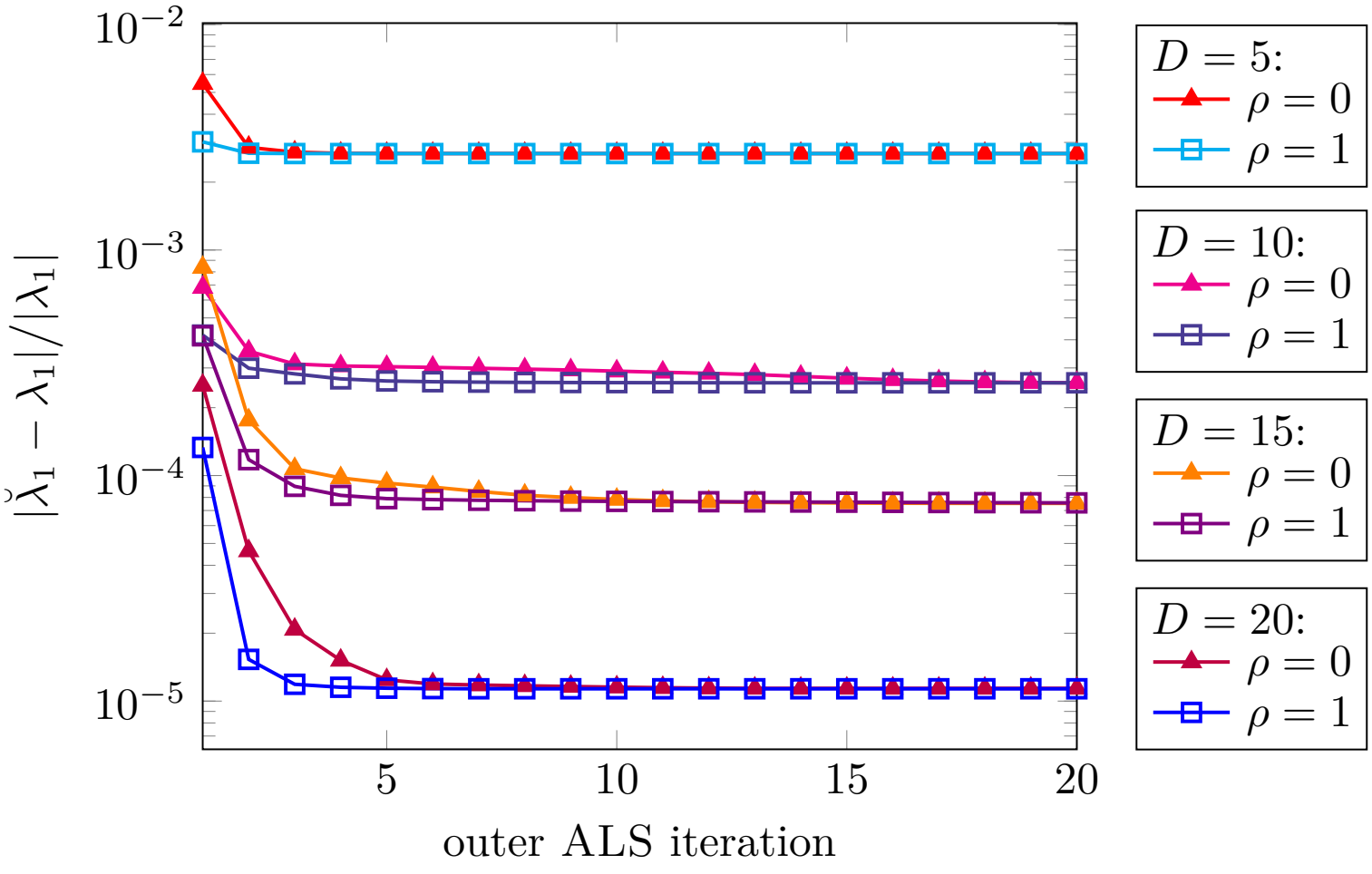


Figure : Heisenberg-XXX model of $N = 100$ spins: Approximation error.

Conclusions

- MPS as data-sparse format to approximate ground states:
  - efficient computations,
  - represent ground states properly,
  - variational methods based on local update schemes.

Conclusions

- MPS as data-sparse format to approximate ground states:
  - efficient computations,
  - represent ground states properly,
  - variational methods based on local update schemes.
- Restarted Lanczos method in the MPS format:
  - convergence against reference solution,
  - projection may even improve the approximation,
  - further information on the desired spectrum.

Conclusions

- MPS as data-sparse format to approximate ground states:
  - efficient computations,
  - represent ground states properly,
  - variational methods based on local update schemes.
- Restarted Lanczos method in the MPS format:
  - convergence against reference solution,
  - projection may even improve the approximation,
  - further information on the desired spectrum.
- Vector symmetries and MPS
  - Connection to MPS representations (via involutions),
  - symmetry-adapted ground-state search improves variational MPS methods,
  - introduces some kind of uniqueness in case of degenerate ground states.

Conclusions

- MPS as data-sparse format to approximate ground states:
  - efficient computations,
  - represent ground states properly,
  - variational methods based on local update schemes.
- Restarted Lanczos method in the MPS format:
  - convergence against reference solution,
  - projection may even improve the approximation,
  - further information on the desired spectrum.
- Vector symmetries and MPS
  - Connection to MPS representations (via involutions),
  - symmetry-adapted ground-state search improves variational MPS methods,
  - introduces some kind of uniqueness in case of degenerate ground states.

Thank you for your attention.


T. Huckle, and K. Waldherr
Subspace iteration methods in terms of matrix product states.
Proc. Appl. Math. Mech., 2012.


T. Huckle, K. Waldherr, et. al.
Computations in quantum tensor networks.
Lin. Alg. Appl., 2013.


T. Huckle, K. Waldherr, et. al.
Exploiting matrix symmetries and physical symmetries in matrix product states and tensor trains.
Linear Multilinear A., 2013.