Low-Rank Tensor Methods for Symmetric Eigenvalue Problems

Michael Steinlechner

MATHICSE

Ecole Polytéchnique Fédérale de Lausanne

Joint work with

Daniel Kressner (EPF Lausanne) and
André Uschmajew (HCM, Universität Bonn)

ProDoc Retreat, Disentis, Switzerland

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Introduction

Solve a *symmetric eigenvalue problem*

$$\mathcal{A}u(x) = \lambda u(x), \quad x \in \Omega \subset \mathbb{R}^d.$$

in an appropriate function space over Ω.

Discretization on a $n_1 \times n_2 \times \cdots \times n_d$ tensor grid yields the matrix eigenvalue problem

$$A\underline{u} = \lambda \underline{u}, \qquad \underline{u} \in \mathbb{R}^N, \text{ with } N = n_1 n_2 \cdots n_d.$$

For d large **we cannot:**

- store and access the grid vector $\underline{u}$.
- perform matrix-vector multiplication $A\underline{u}$.

Introduction

- **Example:** Electronic Schrödinger equation $H\psi = E\psi$ for a system of N electrons and K atoms

$$H = \frac{1}{2}\sum_{i=1}^{N} \Delta_i - \sum_{i=1}^{N} \left(\sum_{I=1}^{K} \frac{Z_I}{\|x_i - x_I\|} + \sum_{j \neq i}^{N} \frac{1}{\|x_i - x_j\|} \right)$$

  operates on antisymmetric functions $\psi \in H^1(\mathbb{R}^{3N})$. Discretization on a tensor grid with n points along each coordinate:

$$\Rightarrow \quad \text{Compute eigenvalues of a matrix with size } n^{3N} \times n^{3N}$$

  So-called **curse of dimensionality**.

- Exploit the low-rank approximability of the solution.

Low-Rank Matrices

Example: The matrix case $d = 2$

- Consider the grid function $\mathbf{u}(i_1, i_2) = u(x_{i_1}, y_{i_2})$ as $n_1 \times n_2$ matrix
- Assume the eigenvector for the smallest eigenvalue has rank $r \ll \min(n_1, n_2)$:

$$\mathrm{vec}(\mathbf{u}) = \mathrm{vec}(UV^{\mathsf{T}}), \qquad U \in \mathbb{R}^{n_1 \times r}, V \in \mathbb{R}^{n_2 \times r}$$

	$\mathbf{u}$	UV^{T}
Storage required:	$n_1 n_2$	$(n_1 + n_2)r$

Low-Rank Matrices

Rewrite the *Rayleigh quotient* for the eigenvalue system in terms of U, V using the Kronecker product:

$$\frac{\mathrm{vec}(\mathbf{u})^\mathsf{T} A\,\mathrm{vec}(\mathbf{u})}{\mathrm{vec}(\mathbf{u})^\mathsf{T}\,\mathrm{vec}(\mathbf{u})} = \frac{\mathrm{vec}(UV^\mathsf{T})^\mathsf{T} A\,\mathrm{vec}(UV^\mathsf{T})}{\mathrm{vec}(UV^\mathsf{T})^\mathsf{T}\,\mathrm{vec}(UV^\mathsf{T})}$$

$$\text{reduced EVP for U} \longrightarrow = \frac{\mathrm{vec}(U)^\mathsf{T}(V \otimes I)^\mathsf{T} A (V \otimes I)\mathrm{vec}(U)}{\mathrm{vec}(U)^\mathsf{T}(V^\mathsf{T}V \otimes I)\mathrm{vec}(U)}$$

$$\text{reduced EVP for V} \longrightarrow = \frac{\mathrm{vec}(V^\mathsf{T})^\mathsf{T}(I \otimes U)^\mathsf{T} A (I \otimes U)\mathrm{vec}(V^\mathsf{T})}{\mathrm{vec}(V^\mathsf{T})^\mathsf{T}(I \otimes U^\mathsf{T}U)\mathrm{vec}(V^\mathsf{T})}$$

Alternate optimization between U and V^T to solve the original eigenvalue problem.

Low-Rank Tensors: TT/MPS format

Matrix Product States (MPS) [Fannes et al. 1992, Östlund/Rommer 1995]
Tensor Train (TT) [Oseledets/Tyrtyshnikov 2009, Oseledets 2011]

$$\mathbf{u}(i_1, i_2, \ldots, i_d) = U_1(i_1)U_2(i_2)\cdots U_d(i_d)$$

with the μth matrix $(\mu = 1, \ldots, d)$:

$$U_\mu(i_\mu) = \mathbb{R}^{r_{\mu-1}\times r_\mu}, \quad r_0 = r_d = 1.$$

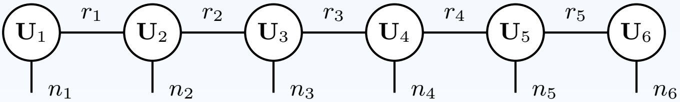
or equivalently in terms of the *core tensors* $\mathbf{U}_\mu \in \mathbb{R}^{r_{\mu-1}\times n_\mu \times r_\mu}$:

$$\mathbf{u}(i_1, \ldots, i_d) = \sum_{j_1=1}^{r_1} \cdots \sum_{j_{d-1}=1}^{r_{d-1}} \mathbf{U}_1(1, i_1, j_1)\mathbf{U}_2(j_1, i_2, j_2)\cdots \mathbf{U}_d(j_{d-1}, i_d, 1).$$

The $U_\mu(i_\mu)$ are slices of the **core tensors**:

$$U_\mu(i_\mu) = \mathbf{U}_\mu(:, i_\mu, :)$$

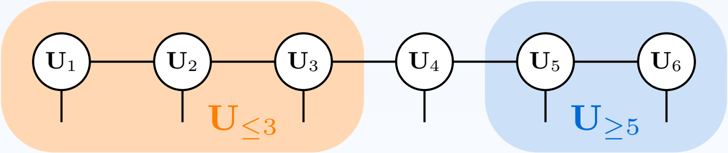
Low-Rank Tensors: TT/MPS format



The minimum ranks r_μ needed for an exact representation are

$$r_\mu = \operatorname{rank}\left(\operatorname{reshape}\left(\mathbf{u}, \mathbb{R}^{(n_1\cdots n_{\mu-1})\times(n_\mu\cdots n_d)}\right)\right)$$

Low-Rank Tensors: TT/MPS format



The minimum ranks r_μ needed for an exact representation are

$$r_\mu = \operatorname{rank}\left(\operatorname{reshape}\left(\mathbf{u}, \mathbb{R}^{(n_1 \cdots n_{\mu-1}) \times (n_\mu \cdots n_d)}\right)\right)$$

For every core $\mathbf{U}_\mu$ we have:

- $\mathbf{u} \in \operatorname{ran}(\mathbf{U}_{\leq \mu-1}) \otimes \mathbb{R}^{n_\mu} \otimes \operatorname{ran}(\mathbf{U}_{\geq \mu+1})$
- $\operatorname{vec}(\mathbf{u}) = \mathcal{U}_{\neq \mu} \operatorname{vec}(\mathbf{U}_\mu), \quad \mathcal{U}_{\neq \mu} = \mathbf{U}_{\leq \mu-1} \otimes I_{n_\mu} \otimes \mathbf{U}_{\geq \mu+1}$

Operator TT format

Low-rank structure of $\mathbf{u}$ is useless unless we can use it in the multiplication with operator A

Operator TT format [Oseledets 2011]

$$A(i_1, \ldots i_d, j_1, \ldots j_d) = A_1(i_1, j_1) A_2(i_2, j_2) \cdots A_d(i_d, j_d),$$

$$\text{with} \quad A_\mu(i_\mu, j_\mu) \in \mathbb{R}^{s_{\mu-1} \times s_\mu}$$

Application to TT tensor:

$$\mathbf{v} = A\mathbf{u} \quad \Leftrightarrow \quad \text{“ } \mathbf{V}_\mu = \mathbf{A}_\mu \mathbf{U}_\mu \text{ ” } \textit{(suitably reshaped)}$$

Ranks multiply!

Operator TT format

Example: d-dimensional discrete Laplacian,

$$L = \sum_{i=1}^{d} I_d \otimes \cdots \otimes L_\mu \otimes \cdots \otimes I_1,$$

with L_μ the one-dim. finite difference matrix, can be represented as TT operator of rank 2: [Kazeev/Khoromskij 2012]

$$A_1(i_1, j_1) = \begin{bmatrix} L_1(i_1, j_1) & I_1(i_1, j_1) \end{bmatrix}, \quad A_d(i_d, j_d) = \begin{bmatrix} I_d(i_d, j_d) \\ L_d(i_d, j_d) \end{bmatrix},$$

and

$$A_\mu(i_\mu, j_\mu) = \begin{bmatrix} I_\mu(i_\mu, j_\mu) & 0 \\ L_\mu(i_\mu, j_\mu) & I_\mu(i_\mu, j_\mu) \end{bmatrix}, \quad \mu = 2, \ldots, d-1.$$

ALS algorithm for smallest EV

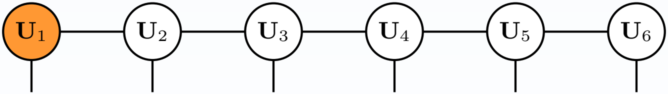
Just like in the matrix case:

$$\frac{\mathrm{vec}(\mathbf{u})^\mathsf{T} A\,\mathrm{vec}(\mathbf{u})}{\mathrm{vec}(\mathbf{u})^\mathsf{T}\,\mathrm{vec}(\mathbf{u})} = \frac{\mathrm{vec}(\mathbf{U}_\mu)^\mathsf{T}\mathcal{U}_{\neq\mu}^\mathsf{T} A\,\mathcal{U}_{\neq\mu}\mathrm{vec}(\mathbf{U}_\mu)}{\mathrm{vec}(\mathbf{U}_\mu)^\mathsf{T}\mathcal{U}_{\neq\mu}^\mathsf{T}\mathcal{U}_{\neq\mu}\mathrm{vec}(\mathbf{U}_\mu)}$$

Note: We can choose $\mathcal{U}_{\neq\mu}$ orthogonal.

Algorithm: Cycle through each core $\mathbf{U}_\mu$ and solve for each the reduced EVP with matrix

$$A_1 = \mathcal{U}_{\neq 1}^\mathsf{T} A\,\mathcal{U}_{\neq 1} \;\in \mathbb{R}^{(r_0 n_1 r_1)\times(r_0 n_1 r_1)}.$$



The ranks r_μ do not change! Need a-priori knowledge?

ALS algorithm for smallest EV

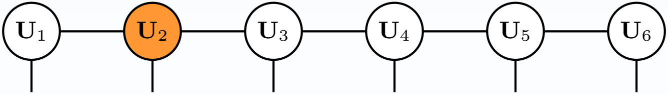
Just like in the matrix case:

$$\frac{\mathrm{vec}(\mathbf{u})^\mathsf{T} A\,\mathrm{vec}(\mathbf{u})}{\mathrm{vec}(\mathbf{u})^\mathsf{T}\,\mathrm{vec}(\mathbf{u})} = \frac{\mathrm{vec}(\mathbf{U}_\mu)^\mathsf{T}\mathcal{U}_{\neq\mu}^\mathsf{T} A\,\mathcal{U}_{\neq\mu}\mathrm{vec}(\mathbf{U}_\mu)}{\mathrm{vec}(\mathbf{U}_\mu)^\mathsf{T}\mathcal{U}_{\neq\mu}^\mathsf{T}\mathcal{U}_{\neq\mu}\mathrm{vec}(\mathbf{U}_\mu)}$$

Note: We can choose $\mathcal{U}_{\neq\mu}$ orthogonal.

Algorithm: Cycle through each core $\mathbf{U}_\mu$ and solve for each the reduced EVP with matrix

$$A_2 = \mathcal{U}_{\neq 2}^\mathsf{T} A\,\mathcal{U}_{\neq 2} \;\in \mathbb{R}^{(r_1 n_2 r_2)\times(r_1 n_2 r_2)}.$$



The ranks r_μ do not change! Need a-priori knowledge?

ALS algorithm for smallest EV

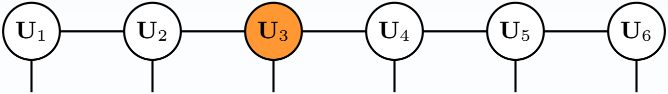
Just like in the matrix case:

$$\frac{\mathrm{vec}(\mathbf{u})^\mathsf{T} A\,\mathrm{vec}(\mathbf{u})}{\mathrm{vec}(\mathbf{u})^\mathsf{T}\,\mathrm{vec}(\mathbf{u})} = \frac{\mathrm{vec}(\mathbf{U}_\mu)^\mathsf{T}\mathcal{U}_{\neq\mu}^\mathsf{T} A\,\mathcal{U}_{\neq\mu}\mathrm{vec}(\mathbf{U}_\mu)}{\mathrm{vec}(\mathbf{U}_\mu)^\mathsf{T}\mathcal{U}_{\neq\mu}^\mathsf{T}\mathcal{U}_{\neq\mu}\mathrm{vec}(\mathbf{U}_\mu)}$$

Note: We can choose $\mathcal{U}_{\neq\mu}$ orthogonal.

Algorithm: Cycle through each core $\mathbf{U}_\mu$ and solve for each the reduced EVP with matrix

$$A_3 = \mathcal{U}_{\neq 3}^\mathsf{T} A\,\mathcal{U}_{\neq 3} \ \in \mathbb{R}^{(r_2 n_3 r_3)\times(r_2 n_3 r_3)}.$$



The ranks r_μ do not change! Need a-priori knowledge?

ALS algorithm for smallest EV

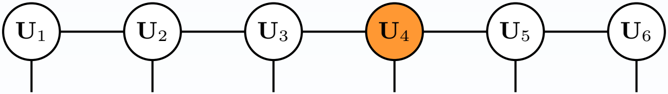
Just like in the matrix case:

$$\frac{\mathrm{vec}(\mathbf{u})^\mathsf{T} A\, \mathrm{vec}(\mathbf{u})}{\mathrm{vec}(\mathbf{u})^\mathsf{T}\, \mathrm{vec}(\mathbf{u})} = \frac{\mathrm{vec}(\mathbf{U}_\mu)^\mathsf{T} \mathcal{U}_{\neq\mu}^\mathsf{T} A\, \mathcal{U}_{\neq\mu} \mathrm{vec}(\mathbf{U}_\mu)}{\mathrm{vec}(\mathbf{U}_\mu)^\mathsf{T} \mathcal{U}_{\neq\mu}^\mathsf{T} \mathcal{U}_{\neq\mu} \mathrm{vec}(\mathbf{U}_\mu)}$$

Note: We can choose $\mathcal{U}_{\neq\mu}$ orthogonal.

Algorithm: Cycle through each core $\mathbf{U}_\mu$ and solve for each the reduced EVP with matrix

$$A_4 = \mathcal{U}_{\neq 4}^\mathsf{T} A\, \mathcal{U}_{\neq 4} \; \in \mathbb{R}^{(r_3 n_4 r_4)\times(r_3 n_4 r_4)}.$$



The ranks r_μ do not change! Need a-priori knowledge?

ALS algorithm for smallest EV

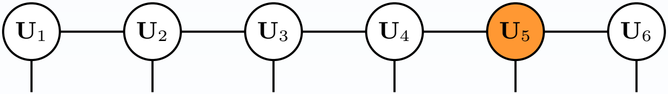
Just like in the matrix case:

$$\frac{\mathrm{vec}(\mathbf{u})^\mathsf{T} A \,\mathrm{vec}(\mathbf{u})}{\mathrm{vec}(\mathbf{u})^\mathsf{T}\,\mathrm{vec}(\mathbf{u})} = \frac{\mathrm{vec}(\mathbf{U}_\mu)^\mathsf{T} \mathcal{U}_{\neq\mu}^\mathsf{T} A \,\mathcal{U}_{\neq\mu} \mathrm{vec}(\mathbf{U}_\mu)}{\mathrm{vec}(\mathbf{U}_\mu)^\mathsf{T} \mathcal{U}_{\neq\mu}^\mathsf{T} \mathcal{U}_{\neq\mu} \mathrm{vec}(\mathbf{U}_\mu)}$$

Note: We can choose $\mathcal{U}_{\neq\mu}$ orthogonal.

Algorithm: Cycle through each core $\mathbf{U}_\mu$ and solve for each the reduced EVP with matrix

$$A_5 = \mathcal{U}_{\neq 5}^\mathsf{T} A \,\mathcal{U}_{\neq 5} \;\in \mathbb{R}^{(r_4 n_5 r_5)\times(r_4 n_5 r_5)}.$$



The ranks r_μ do not change! Need a-priori knowledge?

ALS algorithm for smallest EV

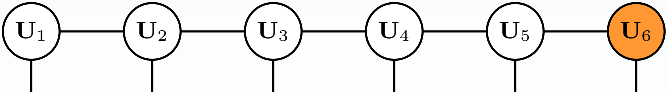
Just like in the matrix case:

$$\frac{\mathrm{vec}(\mathbf{u})^\mathsf{T} A\,\mathrm{vec}(\mathbf{u})}{\mathrm{vec}(\mathbf{u})^\mathsf{T}\,\mathrm{vec}(\mathbf{u})} = \frac{\mathrm{vec}(\mathbf{U}_\mu)^\mathsf{T}\mathcal{U}_{\neq\mu}^\mathsf{T} A\,\mathcal{U}_{\neq\mu}\mathrm{vec}(\mathbf{U}_\mu)}{\mathrm{vec}(\mathbf{U}_\mu)^\mathsf{T}\mathcal{U}_{\neq\mu}^\mathsf{T}\mathcal{U}_{\neq\mu}\mathrm{vec}(\mathbf{U}_\mu)}$$

Note: We can choose $\mathcal{U}_{\neq\mu}$ orthogonal.

Algorithm: Cycle through each core $\mathbf{U}_\mu$ and solve for each the reduced EVP with matrix

$$A_6 = \mathcal{U}_{\neq 6}^\mathsf{T} A\,\mathcal{U}_{\neq 6} \;\in \mathbb{R}^{(r_5 n_6 r_6)\times(r_5 n_6 r_6)}.$$



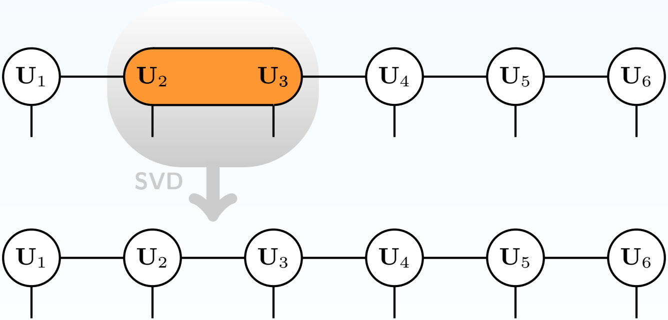
The ranks r_μ do not change! Need a-priori knowledge?

DMRG algorithm for smallest EV

Make the algorithm **rank adaptive** by optimizing neighboring cores together: **(DMRG)** [White 1992]

Solve for each the reduced EVP with matrix

$$A_{\mu,\mu+1} = \mathcal{U}_{\neq\mu,\mu+1}^{\mathsf{T}} A \mathcal{U}_{\neq\mu,\mu+1} \in \mathbb{R}^{(r_{\mu-1}n_\mu n_{\mu+1}r_{\mu+1})\times(r_{\mu-1}n_\mu n_{\mu+1}r_{\mu+1})}.$$



Good, but can easily become prohibitively expensive.

Computing several eigenvectors

Compute p smallest eigenvalues and eigenvectors by **trace minimization:**

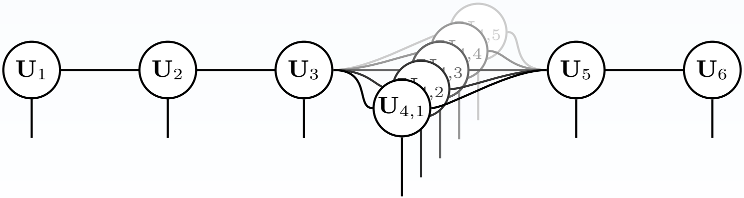
$$\min \operatorname{trace}(\mathbf{U}^T A \mathbf{U}), \quad \mathbf{U}^\mathsf{T}\mathbf{U} = I_p,$$

$$\mathbf{U} = [\operatorname{vec}(\mathbf{u}_1), \operatorname{vec}(\mathbf{u}_2), \ldots, \operatorname{vec}(\mathbf{u}_p)]$$

Use seperate TT-representation for each $\mathbf{u}_\alpha, \alpha = 1, \ldots, p$? Better:

Block-μ TT format [Pižorn/Verstraete 2012, Dolgov et al. 2013]

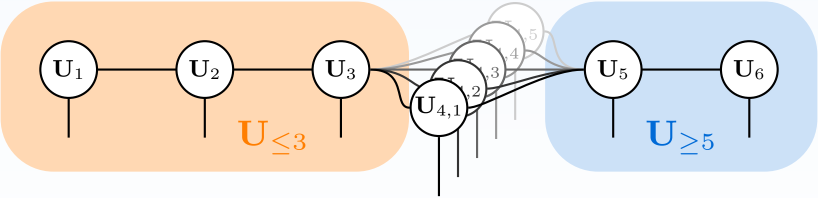
$$\mathbf{u}_\alpha(i_1, i_2, \ldots, i_d) = U_1(i_1) \cdots U_{\mu-1}(i_{\mu-1}) U_{\mu,\alpha}(i_\mu) U_{\mu+1}(i_{\mu+1}) \cdots U_d(i_d)$$



Block-TT format

For $\mathbf{U}$ in **block-μ TT format**, it holds:

- $\mathbf{u}_\alpha \in \operatorname{ran}(\mathbf{U}_{\leq \mu-1}) \otimes \mathbb{R}^{n_\mu} \otimes \operatorname{ran}(\mathbf{U}_{\geq \mu+1})$ independent of $\alpha = 1, \ldots, p$!
- $\mathbf{U} = \mathcal{U}_{\neq\mu} \mathbf{V}_\mu, \quad \mathbf{V}_\mu = [\operatorname{vec}(\mathbf{U}_{\mu,1}), \operatorname{vec}(\mathbf{U}_{\mu,2}), \ldots, \operatorname{vec}(\mathbf{U}_{\mu,p})]$

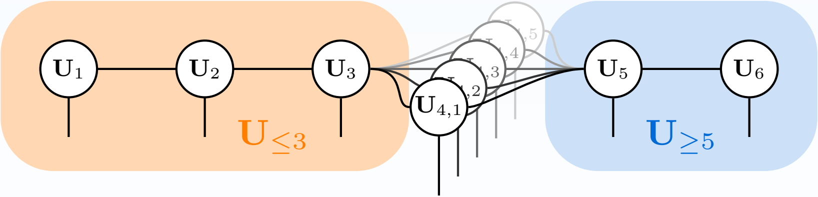


Block-TT format

Reduced problem for the p smallest eigenvalues:

If $\mathbf{U}$ is in block-μ format, and $\mathcal{U}_{\neq\mu}$ is orthogonal:

$$\min \operatorname{trace}(\mathbf{V}_\mu^\mathsf{T} \mathcal{U}_{\neq\mu}^\mathsf{T} A \mathcal{U}_{\neq\mu} \mathbf{V}_\mu), \quad \mathbf{V}_\mu \in \mathbb{R}^{r_{\mu-1} n_\mu r_\mu \times p}, \quad \mathbf{V}_\mu^\mathsf{T} \mathbf{V}_\mu = I_p,$$



Going from block-μ to block-$\mu + 1$

The position of multicore V_μ is *arbitrary* and can be *changed*:

1. Reshape $\mathbf{V}_\mu$ into matrix $\widetilde{\mathbf{V}}_\mu \in \mathbb{R}^{(r_{\mu-1}n_\mu)\times(pr_\mu)}$
2. Minimal-rank decomposition

$$\widetilde{\mathbf{V}}_\mu = Q \begin{bmatrix} P_1 & P_2 & \dots & P_p \end{bmatrix}, \quad Q \in \mathbb{R}^{r_{\mu-1}n_\mu \times s}, \quad P_\alpha \in \mathbb{R}^{s \times r_\mu}$$

3. Update cores

$$\mathbf{U}_\mu \leftarrow Q, \qquad \mathbf{U}_{\mu+1,\alpha} \leftarrow P_\alpha \mathbf{U}_{\mu+1},\ \alpha = 1, \dots, p$$

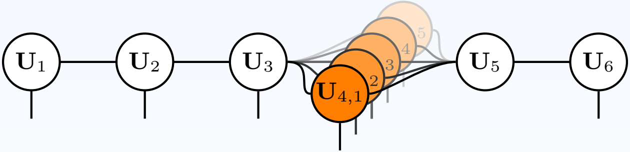
By this shifting procedure, the rank between the two cores has changed

$$r_\mu \leftarrow s$$

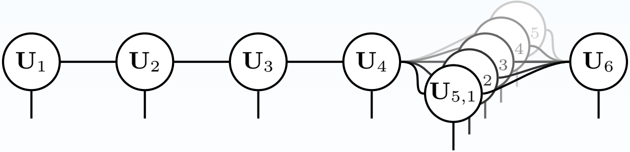
$\Rightarrow$ **Rank adaptivity** if $p > 1$: $\quad 0 < s \leq \min(r_{\mu-1}n_\mu, r_\mu p)$

Block-TT ALS algorithm for p eigenvalues

Optimize



Shift multicore to $\mu + 1$



[Dolgov et al. 2013]

Improving convergence: Subspace correction

Adding gradient information:

$$\mathbf{R} = A\mathbf{U} - \mathbf{U}\Lambda \quad \text{(the global residual)}$$

Add to the current intermediate result to enrich the subspace

$$\tilde{\mathbf{U}} = \mathbf{U} - \mathbf{R}$$

This also increases the rank in all modes:

$$\mathrm{rank}(\tilde{\mathbf{U}}) = \mathrm{rank}(\mathbf{U}) + \mathrm{rank}(\mathbf{R})$$

Too costly to enrich all cores with global gradient information after each step...

AMEn: Local subspace correction

Alternating Minimal Energy (AMEn) method for linear systems
[Dolgov/Savostyanov 2013]

- Start iteration with random TT tensor of rank one
- Add gradients for both acceleration and rank adaptivity of the algorithm
- But: Only inject the gradient information *locally* to cores $\mathbf{U}_\mu$ and $\mathbf{U}_{\mu+1}$.

Use the *projected DMRG (two-core) residual*

$$\mathbf{R}_{\mu,\mu+1} = \mathcal{U}_{\neq\mu,\mu+1}^{\mathsf{T}}(A\mathbf{U} - \mathbf{U}\Lambda)$$

Interpretation: Steepest Descent for DMRG

$$\mathcal{U}_{\neq\mu,\mu+1}^{\mathsf{T}}\tilde{\mathbf{U}} = \mathcal{U}_{\neq\mu,\mu+1}^{\mathsf{T}}\mathbf{U} - \mathbf{R}_{\mu,\mu+1}$$

Local subspace correction

How to augment with the projected DMRG (two-core) residual?

$$\mathbf{R}_{\mu,\mu+1} = \mathcal{U}_{\neq\mu,\mu+1}^{\mathsf{T}}(A\mathbf{U} - \mathbf{U}\Lambda)$$

(We can compute this efficiently)

First decompose into two parts:

$$R_{\mu,\mu+1,\alpha}(i_\mu, i_{\mu+1}) = R_{\mu,\alpha}(i_\mu)R_{\mu+1}(i_{\mu+1})$$

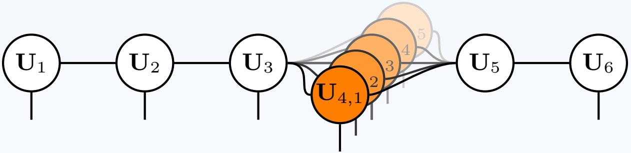
Then augment cores $\mathbf{U}_\mu$ and $\mathbf{U}_{\mu+1}$:

$$\tilde{U}_{\mu,\alpha}(i_\mu) = \begin{bmatrix} U_{\mu,\alpha}(i_\mu) & \\ & R_{\mu,\alpha}(i_\mu) \end{bmatrix}$$

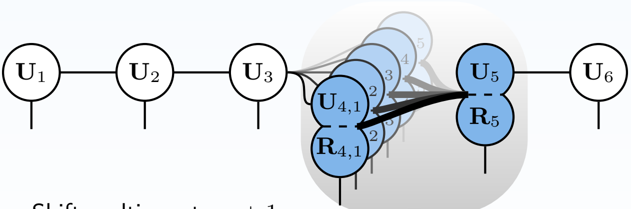
$$\tilde{U}_{\mu+1}(i_{\mu+1}) = \begin{bmatrix} U_{\mu+1}(i_{\mu+1}) & \\ & R_{\mu+1}(i_{\mu+1}) \end{bmatrix}$$

Our algorithm: Block ALS with local enrichment

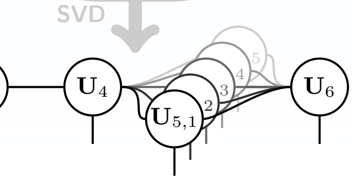
Optimize



Augment with (preconditioned) projected residual



Shift multicore to $\mu + 1$



Preconditioning the residual correction

Inject a preconditioner into the residual correction (think PINVIT)

$$\mathbf{R}_{\mu,\mu+1} = \mathcal{B}_{\mu,\mu+1}^{-1} \mathcal{U}_{\neq\mu,\mu+1}^{\mathsf{T}} (A\mathbf{U} - \mathbf{U}\Lambda)$$

where $\mathcal{B}_{\mu,\mu+1}^{-1}$ is a preconditioner for the local reduced problem.

Deriving a good preconditioner for the local problem can be challenging!

Numerical experiments

Model problem:

$$\begin{aligned} -\Delta u(x) + V(x)u(x) &= \lambda u(x), && \text{for } x \in \Omega = (a,b)^d \\ u(x) &= 0 && \text{for } x \in \delta\Omega \end{aligned}$$

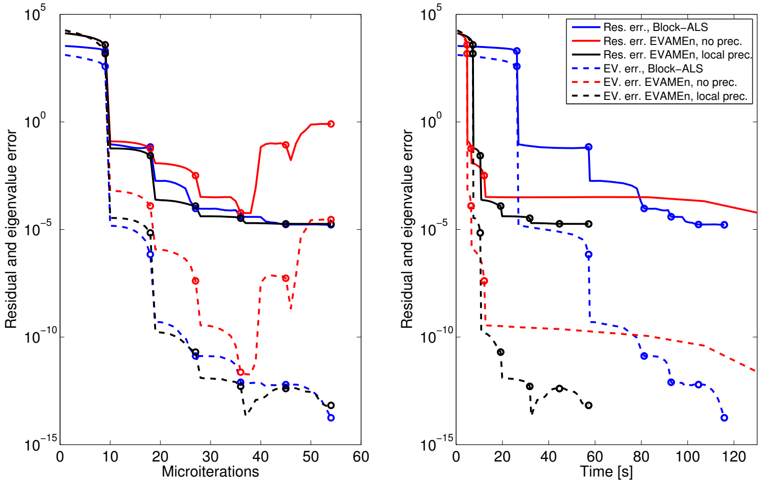
Newton potential:

$$V(x) = \frac{1}{\|x\|}$$

Approximated in low-rank TT format (rank 10) using exponential sums [Hackbusch 2010]

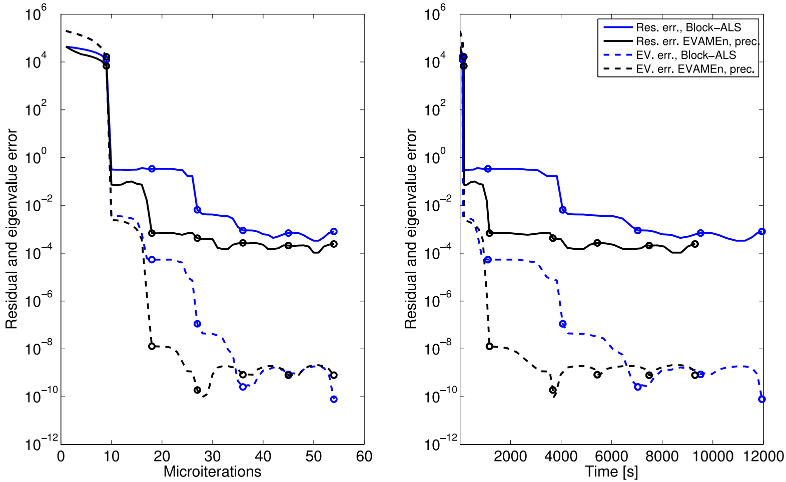
Preconditioner: Approximation of inverse of projected Laplacian by exponential sums (rank 3). [Grasedyk 2004, Hackbusch 2010]

Newton potential: One eigenvalue



$\Omega = (-1, 1)^{10}, n = 128 \rightarrow$ EVP of size 128^{10}. $p = 1$, final ranks 8

Newton potential: 11 eigenvalues



$\Omega = (-1,1)^{10}, n = 128 \ \rightarrow$ EVP of size 128^{10}. $p = 11$,
final ranks 40 (cut-off)

Numerical experiments

Henon-Heiles potential:

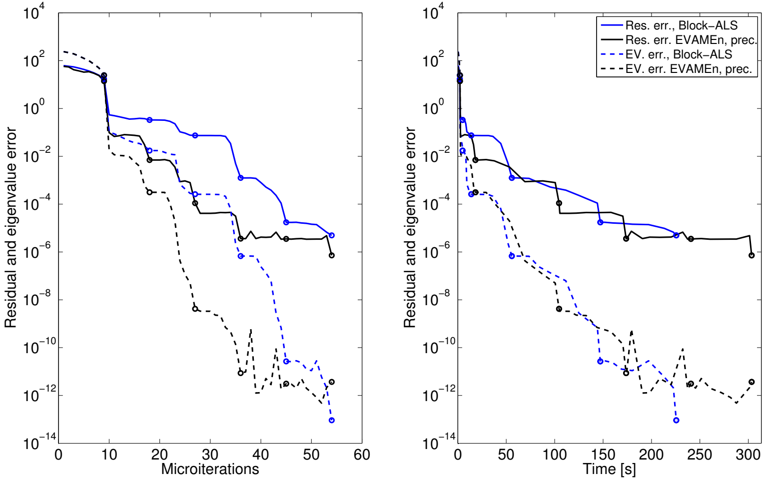
$$V(\mathbf{x}) = \frac{1}{2}\sum_{\mu=1}^{d} x_\mu^2 + \sum_{\mu=1}^{d-1}\left(\sigma_*\left(x_\mu x_{\mu+1}^2 - \frac{1}{3}x_\mu^3\right) + \frac{\sigma_*^2}{16}\left(x_\mu^2 + x_{\mu+1}^2\right)^2\right),$$

modelling a coupled oscillator in quantum physics.

This potential is usually defined on the entire real space. As in [Dolgov et al. 2013] , we apply **spectral collocation**, using a tensor product grid based on the zeros $\xi_1, \ldots, \xi_n$ of the nth Hermite polynomial.

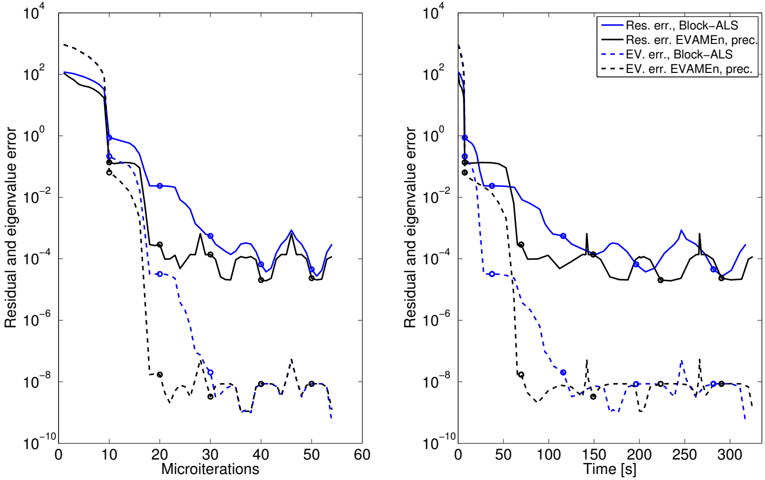
Here: $n = 28$ collocation points in every dimension.

Henon-Heiles potential: 3 eigenvalues



$d = 10, n = 28 \;\rightarrow\;$ EVP of size 28^{10}. $p = 3$

Henon-Heiles potential: 11 eigenvalues



$d = 10, n = 28 \;\rightarrow\;$ EVP of size 28^{10}. $p = 11$

Conclusions

- Tensor techniques, using e.g. the TT format, allow for the solution of extremely high-dimensional eigenvalue problems. Successfully used in physics.
- Need: Low-rank structured operators and low-rank solutions
- (Block)-ALS is extremely efficient and rank-adaptive for $p > 1$
- AMEn injects (preconditioned) residual information to the local problems to speed up convergence and enables rank-adaptivity even for $p = 1$. Reduction in iteration count, but each iteration is more costly.

More details:

D. Kressner, M. Steinlechner, A. Uschmajew:
Low-rank tensor methods with subspace correction for symmetric eigenvalue problems
MATHICSE Technical Report 40.2013, December 2013.
Accepted for publication in SIAM Journal on Scientific Computing.
Available at http://anchp.epfl.ch/publications

Thank you for your attention!