Parallel time-dependent variational principle algorithm for tensor trains

Sergey Dolgov

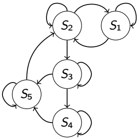
joint work with Paul Secular, Nikita Gourianov, Michael Lubasch, Stephen R. Clark and Dieter Jaksch

UNIVERSITY OF BATH

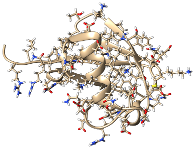
Exploiting Algebraic and Geometric Structure in Time-Integration Methods
Pisa, April 3 2024

High-dimensional time-dependent problems

- Fokker-Planck/Chemical master equations
  - Stochastic mechanics
  - Gene regulation
  - Virus replication



- Schroedinger equation
  - Condensed matter physics
  - Computational chemistry
  - Magnetic resonance



Why tensors?

- Motivation: a multivariate function $u(x^1, \dots, x^d)$
  …discretized **independently** in each variable.

Central object: an array of discrete values $\equiv$ **tensor**:

$$u(i_1, i_2, \dots, i_d). \qquad \begin{array}{l} i_k = 1, \dots, n_k, \\ k = 1, \dots, d. \end{array}$$

- For example $u(i_1, \dots, i_d) = u(x^1_{i_1}, \dots, x^d_{i_d})$.

Curse of dimensionality: $\mathtt{mem} = n^d$.

(think of 10^{80} …)

Large but structured

Our problem of interest is

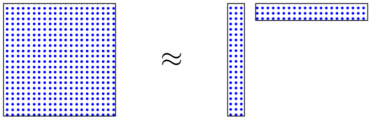
$$\frac{d\vec{u}}{dt} = A\vec{u}$$
$$\vec{u}(0) = \vec{u}_0$$

- $\lambda(A) \in \mathbb{C}_-$. in our app $A = -A^*$
- $\vec{u}(t) \in \mathbb{C}^N$ with $N = n^d \sim 10^{80}$.
- However, $\vec{u}(t)$ can be indexed by $i_1, \dots, i_d$ as $u(i_1, \dots, i_d, t)$.

2 variables: low-rank matrices

- Low-rank matrix decomposition:

$$u(i,j) = \sum_{\alpha=1}^{r} V_\alpha(i) W_\alpha(j) + \mathcal{O}(\varepsilon)$$



- **Rank** $r \ll n$
- $\mathtt{mem}(V) + \mathtt{mem}(W) = 2nr \ll n^2 = \mathtt{mem}(u)$
- **Singular Value Decomposition**: optimal $\varepsilon(r)$ dependence
- Riemannian manifold $\mathcal{M}_r$

Dirac-Frenkel Time-Dependent Variational Principle (TDVP)

Can solve instead

$$\left\| \frac{d\vec{u}}{dt} - A(u) \right\| \to \min \quad \text{over } u(t) \in \mathcal{M}_r$$
$$\vec{u}(0) = \vec{u}_0$$

- Equivalently $\frac{d\vec{u}}{dt} = P_u \cdot A(u)$, where
- P_u is an orthogonal projector on (vectorised)
- $T_u\mathcal{M}_r$, the tangent space of the manifold of rank-r matrices.

Dirac-Frenkel Time-Dependent Variational Principle (TDVP)

Can solve instead

$$\left\| \frac{d\vec{u}}{dt} - A(u) \right\| \to \min \quad \text{over } u(t) \in \mathcal{M}_r$$
$$\vec{u}(0) = \vec{u}_0$$

Dynamical low-rank approximation:[1]

- Let $u = VSW^*$ $\qquad S \in \mathbb{C}^{r\times r}$
- Split the projector

$$P_u = (WW^\dagger) \otimes I - (WW^\dagger) \otimes (VV^\dagger) + I \otimes (VV^\dagger).$$

[1] [Koch, Lubich '07], [Lubich, Oseledets '14]

Dirac-Frenkel Time-Dependent Variational Principle (TDVP)

This gives a convenient linear "KSL" scheme:

- Let $u(0) = V_0 S_0 W_0^*$ with V_0, W_0 orthogonal.

Dirac-Frenkel Time-Dependent Variational Principle (TDVP)

This gives a convenient linear "KSL" scheme:

- Let $u(0) = V_0 S_0 W_0^*$ with V_0, W_0 orthogonal.
- Solve $\frac{dK}{dt} = A(K W_0^*) W_0$ starting from $K(0) = V_0 S_0$. **"K"**

Dirac-Frenkel Time-Dependent Variational Principle (TDVP)

This gives a convenient linear "KSL" scheme:

- Let $u(0) = V_0 S_0 W_0^*$ with V_0, W_0 orthogonal.
- Solve $\frac{dK}{dt} = A(KW_0^*)W_0$ starting from $K(0) = V_0 S_0$. **"K"**
- Factorise $V_1 S_1 = \mathrm{qr}(K(t))$.

Dirac-Frenkel Time-Dependent Variational Principle (TDVP)

This gives a convenient linear "KSL" scheme:

- Let $u(0) = V_0 S_0 W_0^*$ with V_0, W_0 orthogonal.
- Solve $\frac{dK}{dt} = A(K W_0^*) W_0$ starting from $K(0) = V_0 S_0$. **"K"**
- Factorise $V_1 S_1 = \mathrm{qr}(K(t))$.
- Solve $\frac{dS}{dt} = -V_1^* A(V_1 S W_0^*) W_0$ starting from $S(0) = S_1$. **"S"**

Dirac-Frenkel Time-Dependent Variational Principle (TDVP)

This gives a convenient linear "KSL" scheme:

- Let $u(0) = V_0 S_0 W_0^*$ with V_0, W_0 orthogonal.
- Solve $\frac{dK}{dt} = A(KW_0^*)W_0$ starting from $K(0) = V_0 S_0$. **"K"**
- Factorise $V_1 S_1 = \mathrm{qr}(K(t))$.
- Solve $\frac{dS}{dt} = -V_1^* A(V_1 S W_0^*) W_0$ starting from $S(0) = S_1$. **"S"**
- Solve $\frac{dL^*}{dt} = V_1^* A(V_1 L^*)$ starting from $L^*(0) = S(t) W_0^*$. **"L"**

Dirac-Frenkel Time-Dependent Variational Principle (TDVP)

This gives a convenient linear "KSL" scheme:

- Let $u(0) = V_0 S_0 W_0^*$ with V_0, W_0 orthogonal.
- Solve $\frac{dK}{dt} = A(K W_0^*) W_0$ starting from $K(0) = V_0 S_0$. **"K"**
- Factorise $V_1 S_1 = \mathrm{qr}(K(t))$.
- Solve $\frac{dS}{dt} = -V_1^* A(V_1 S W_0^*) W_0$ starting from $S(0) = S_1$. **"S"**
- Solve $\frac{dL^*}{dt} = V_1^* A(V_1 L^*)$ starting from $L^*(0) = S(t) W_0^*$. **"L"**
- Factorise $W_1 S_2^* = \mathrm{qr}(L(t))$.

Dirac-Frenkel Time-Dependent Variational Principle (TDVP)

This gives a convenient linear "KSL" scheme:

- Let $u(0) = V_0 S_0 W_0^*$ with V_0, W_0 orthogonal.
- Solve $\frac{dK}{dt} = A(K W_0^*) W_0$ starting from $K(0) = V_0 S_0$. **"K"**
- Factorise $V_1 S_1 = \mathrm{qr}(K(t))$.
- Solve $\frac{dS}{dt} = -V_1^* A(V_1 S W_0^*) W_0$ starting from $S(0) = S_1$. **"S"**
- Solve $\frac{dL^*}{dt} = V_1^* A(V_1 L^*)$ starting from $L^*(0) = S(t) W_0^*$. **"L"**
- Factorise $W_1 S_2^* = \mathrm{qr}(L(t))$.
- $u(t) = V_1 S_2 W_1^*$.

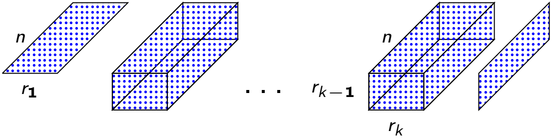
Total error is **linear** in step size, truncation and projection errors, and **independent** of singular values.

[Kieri, Lubich, Walach '16]

Many variables: low-rank tensors

- Matrix Product States/Tensor Train[2]:

$$u(i_1,\dots,i_d) = \sum_{\substack{\alpha_k=1 \\ 0<k<d}}^{r_k} U^1_{\alpha_1}(i_1) U^2_{\alpha_1,\alpha_2}(i_2)\cdots U^d_{\alpha_{d-1}}(i_d).$$



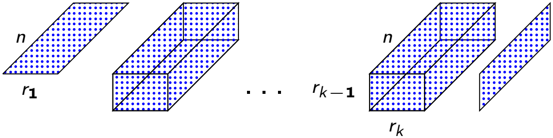
- Or simply

$$u(i_1,\dots,i_d) = U^1(i_1)\cdots U^d(i_d).$$

- Other tensor networks possible (HT, TTN, PEPS, MERA, ...)

[2] Wilson '75, White '93, Verstraete '04, Oseledets '09

Tensor Train and Kronecker products



Another way of writing:

$$\vec{u} = \sum_{\substack{\alpha_k=1 \\ 0<k\leq d}}^{r_k} U^1_{\alpha_1} \otimes U^2_{\alpha_1,\alpha_2} \otimes \cdots \otimes U^d_{\alpha_{d-1}}.$$

- **TT-ranks** $(r_1, \dots, r_{d-1}) \leq (r, \dots, r)$.
- $\mathtt{mem}(U^1) + \cdots + \mathtt{mem}(U^d) = \mathcal{O}(dnr^2) \ll n^d = \mathtt{mem}(u)$.

Tensor Train: algebraic operations

- **Any** data can be decomposed: TT-SVD theorem

$$\varepsilon^2(r_1, \dots, r_{d-1}) \le \sum_{k=1}^{d-1} \varepsilon_k^2(r_k).$$

- Decomposition of **matrices**

$$A = \sum_{\boldsymbol{\beta}} A^1_{\beta_1} \otimes A^2_{\beta_1,\beta_2} \otimes \cdots \otimes A^d_{\beta_{d-1}}.$$

- ... and hence

Tensor Train: distributivity

- ... <u>factorised products</u>

$$\begin{aligned} A(u) &= \sum_{\boldsymbol{\beta}} A^1_{\beta_1} \otimes A^2_{\beta_1,\beta_2} \otimes \cdots \otimes A^d_{\beta_{d-1}} \\ &\cdot \sum_{\boldsymbol{\alpha}} U^1_{\alpha_1} \otimes U^2_{\alpha_1,\alpha_2} \otimes \cdots \otimes U^d_{\alpha_{d-1}} \\ &= \sum_{\boldsymbol{\alpha},\boldsymbol{\beta}} \left(A^1_{\beta_1} U^1_{\alpha_1}\right) \otimes \cdots \otimes \left(A^d_{\beta_{d-1}} U^d_{\alpha_{d-1}}\right). \end{aligned}$$

- Take $A = w^\top$ $\rightarrow$ fast quadratures with $\mathcal{O}(dnr^2)$ cost.

Dynamical Tensor Train approximation

TT decomposition is a recursive matrix decomposition: let

$$\vec{u} = \sum_{\boldsymbol{\alpha}} \underset{VS}{U^1_{\alpha_1}} \otimes \underset{W^*}{\underbrace{U^2_{\alpha_1,\alpha_2} \otimes \cdots \otimes U^d_{\alpha_{d-1}}}_{U^{>1}_{\alpha_1}}}.$$

- "K" and "S" steps are implemented on (small) U^1 directly.
- $L^*(0) = S W_0^*$ reduces to $L^2(i_2) = S U^2(i_2)$.
- Integrate $\frac{dL^2}{dt}$ **only** $\mathcal{O}(nr^2)$ DoFs
- $U^3, \ldots, U^d$ are still **fixed**.

Dynamical <u>**Tensor Train**</u> approximation

General step[3]:

$$u(i_1,\dots,i_d) = \underbrace{U^1(i_1)\cdots U^{k-1}(i_{k-1})}_{U^{<k}(i_{<k})}\underbrace{U^k(i_k)}_{K_{i_k}}\underbrace{U^{k+1}(i_{k+1})}_{W^*_{i_{k+1}}}\underbrace{U^{k+2}(i_{k+2})\cdots U^d(i_d)}_{U^{>k+1}(i_{>k+1})}.$$

- Assume A is also in TT $\Rightarrow A(u)$ is a <u>**factorised product**</u>

[3] [Lubich, Oseledets, Vandereycken '15]

Dynamical Tensor Train approximation

General step[3]:

$$u(i_1,\ldots,i_d) = \underbrace{U^1(i_1)\cdots U^{k-1}(i_{k-1})}_{U^{<k}(i_{<k})}\underbrace{U^k(i_k)}_{K_{i_k}}\underbrace{U^{k+1}(i_{k+1})}_{W^*_{i_{k+1}}}\underbrace{U^{k+2}(i_{k+2})\cdots U^d(i_d)}_{U^{>k+1}(i_{>k+1})}.$$

$$A = \sum_{\boldsymbol{\beta}} A^{<k}_{\beta_{k-1}} \otimes A^k_{\beta_{k-1},\beta_k} \otimes A^{k+1}_{\beta_k,\beta_{k+1}} \otimes A^{>k+1}_{\beta_{k+1}}$$

- Assume A is also in TT $\Rightarrow$ $A(u)$ is a **factorised product**
- $\Rightarrow$ we can multiply $A^{<k}U^{<k}$ and $A^{>k}U^{>k}$ at $\mathcal{O}(dnr^4)$ cost.

[3] [Lubich, Oseledets, Vandereycken '15]

Dynamical **Tensor Train** approximation

General step[3]:

$$\Theta(i_k, i_{k+1}) = \underbrace{U^k(i_k)}_{K_{i_k}} \underbrace{U^{k+1}(i_{k+1})}_{W^*_{i_{k+1}}}.$$

- Assume A is also in TT $\Rightarrow A(u)$ is a **factorised product**
- $\Rightarrow$ we can multiply $A^{<k} U^{<k}$ and $A^{>k} U^{>k}$ at $\mathcal{O}(dnr^4)$ cost.
- This projects $A(u)$ into $A^{k:k+1}(\Theta) \Rightarrow$ **matrix "KSL"**.*

[3] [Lubich, Oseledets, Vandereycken '15]

* cf. DMRG [White '93]

Rank-adaptive dynamical approximation

- Let $P_{<k} = U^{<k}(U^{<k})^*$, $P_{>k} = U^{>k}(U^{>k})^*$.

This TT-KSL scheme is a splitting scheme for $\frac{d\vec{u}}{dt} = P_u \cdot A(u)$ with

$$P_u = \sum_{k=1}^{d} P_{<k} \otimes I \otimes P_{>k} - \sum_{k=1}^{d-1} P_{<k+1} \otimes P_{>k},$$

the orthogonal projector onto the T-space of the TT manifold.

- However, we can define a **2-core projector**:

$$\mathcal{P}_u = \sum_{k=1}^{d-1} P_{<k} \otimes I \otimes I \otimes P_{>k+1} - \sum_{k=1}^{d-2} P_{<k+1} \otimes I \otimes P_{>k+1}$$

Rank-adaptive dynamical approximation

This gives an adaptive integrator similar to DMRG:[4]

- Solve $\frac{d\Theta}{dt} = A^{k:k+1}(\Theta)$ starting from $\Theta_0 = U^k U^{k+1}$.
- Factorise $\Theta(t) \approx VSW^*$ using **truncated** SVD. (new r_k)
- Solve $\frac{dL}{dt} = -V^* A^{k:k+1}(VL)$ starting from $L_0 = SW^*$.
- $U^k = V$, $U^{k+1} = L(t)$.
- Iterate $k \leftarrow (k-1)$ or $(k+1)$

[4] [Haegeman, Lubich, Oseledets, Vandereycken, Verstraete '16]

Rank-adaptive dynamical approximation

This gives an adaptive integrator similar to DMRG:[4]

- Solve $\frac{d\Theta}{dt} = A^{k:k+1}(\Theta)$ starting from $\Theta_0 = U^k U^{k+1}$.
- Factorise $\Theta(t) \approx VSW^*$ using **truncated** SVD. (new r_k)
- Solve $\frac{dL}{dt} = -V^* A^{k:k+1}(VL)$ starting from $L_0 = SW^*$.
- $U^k = V,\ U^{k+1} = L(t)$.
- **Iterate** $k \leftarrow (k-1)$ or $(k+1)$**?**

[4] [Haegeman, Lubich, Oseledets, Vandereycken, Verstraete '16]

Parallelising over TT cores

Can we run steps for different k simultaneously?

- Problem: V and W are orthogonal, while L and K are not $\rightarrow$ different scales of TT cores.

Solution: **inverse gauge** conditions

- $\boxed{VSW^* = (VS)S^{-1}(SW^*)}$ = the "$KS^{-1}L$" scheme!
- In the TT decomposition:

$$u(i_1, \ldots, i_d) = U^1(i_1)S_1^{-1}U^2(i_2)\cdots S_{d-1}^{-1}U^d(i_d).$$

- All $\|U^k\| = \|u\|$ ("centers") $\rightarrow$ parallelisation makes sense.
- Stability of inverting S_k???

Inverse gauge $\Rightarrow$ parallel TT algorithms

Problem: $\mathrm{cond}(S_k) = \frac{1}{\sigma_{r_k}}$. However...

$$\boxed{VSW^* = (VS)S^{-1}(SW^*)}$$

- ...if we use SVD:
  $S_k^{-1} = \mathrm{diag}(1/s_k)$. **Numbers** are perfectly conditioned!

This gives an abstract[5] algorithm:

- Solve over overlapping subset of TT cores in parallel:

Process 0

$$U^1(i_1)\cdots S_{m-1}^{-1} U^m(i_m) \quad S_m^{-1} \quad U^{m+1}(i_{m+1}) S_{m+1}^{-1} \cdots U^d(i_d)$$

Process 1

- Synchronisation: solution/SVD on the overlap $U^m S_m^{-1} U^{m+1}$.

[5] instantiated in par-DMRG [Stoudenmire, White '13], HT-ALS [Etter '16], TT-Cross [D.,Savostyanov '19], and pTDVP discussed

Parallel time-dependent variational principle

Assuming perfect load balance $(d-1) = Pm$:

- Computational cost: perfect scaling

$$\mathcal{O}\Big(\frac{d-1}{P}\Big(\underbrace{n^3r^3}_{\text{Solve } \frac{d\Theta}{dt} \text{ and SVD}} + \underbrace{nr^4}_{\text{projections of } A}\Big)\Big)$$

- Communication volume:

$$\mathcal{O}\Big(\underbrace{r^3}_{\text{projections of } A} + \underbrace{nr^2}_{U^m, S_m, U^{m+1}}\Big)$$

Parallel time-dependent variational principle

Assuming perfect load balance $(d-1) = Pm$:

- Computational cost: perfect scaling

$$\mathcal{O}\left(\frac{d-1}{P}\left(\underbrace{n^3 r^3}_{\text{Solve } \frac{d\Theta}{dt} \text{ and SVD}} + \underbrace{nr^4}_{\text{projections of } A}\right)\right)$$

- Communication volume:

$$\mathcal{O}\left(\underbrace{r^3}_{\text{projections of } A} + \underbrace{nr^2}_{U^m, S_m, U^{m+1}}\right)$$

- What about S_m that are **not** updated in last step?
- Limit the **time step** t to "small enough"

  even the stable KSL may diverge if we step too far off the manifold

Long-range Ising model

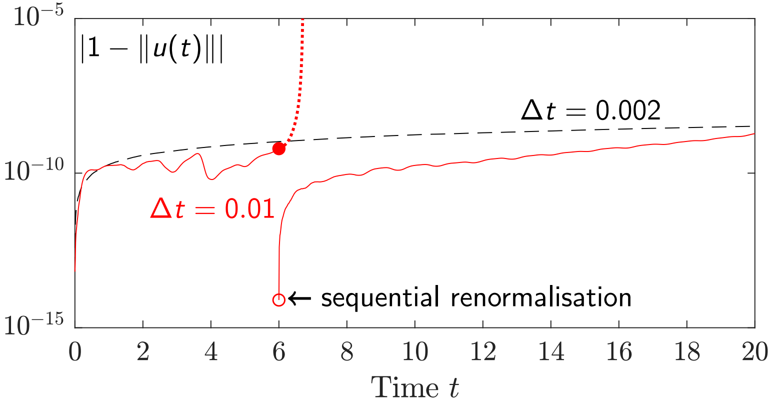
- Let $\sigma^x, \sigma^y, \sigma^z \in \mathbb{C}^{2\times 2}$ be the elementary Pauli matrices.
- Let $\sigma_k^\mu = I \otimes \cdots \otimes I \otimes \sigma^\mu \otimes I \otimes \cdots \otimes I$ be operators of their action on the kth particle ($\mu \in \{x, y, z\}$).
- A Hamiltonian of the Ising chain subject to a magnetic field:

$$A = -\mathrm{i} \sum_{k<m}^{d} \frac{1}{|k-m|^{\alpha}} \sigma_k^z \sigma_m^z - \mathrm{i} B \sum_{k=1}^{d} \sigma_k^x.$$

- α: (non)locality parameter, tunable in trapped ion experiments
- $\alpha = \infty$: nearest-neighbour model, solvable by (parallel) TEBD.
- $\alpha < 3$: TEBD splitting too inaccurate. Here begins the fun...

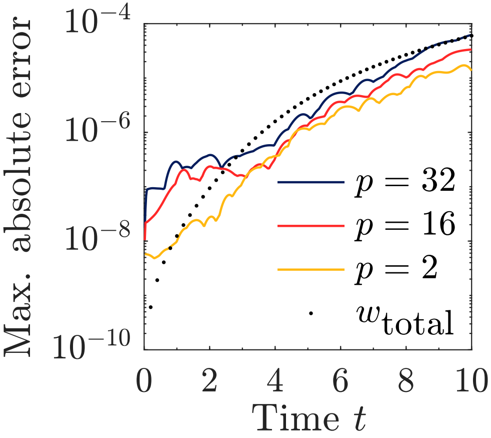
Time step and stability of parallel TDVP

- $\alpha = 2.3,\ B = 0.27$
- $d = 641,\ r \approx 100$

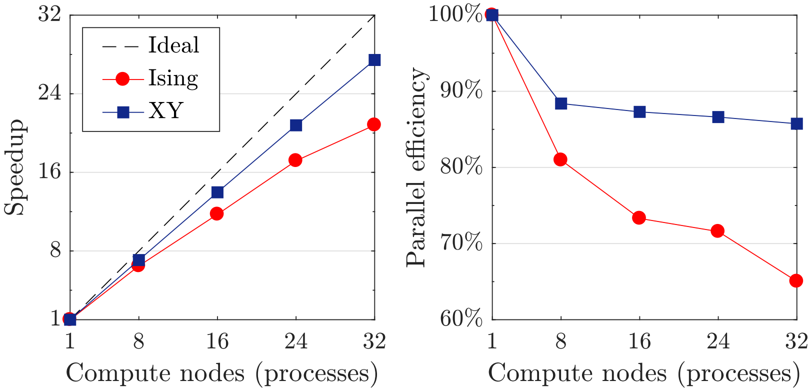


Error scaling

- Compare sequential solution with $p = 2, 16, 32$ processes.
- ω_{total} = cumulative norm of discarded singular values.



Strong time scaling



Conclusion

- Inverse gauge allows parallelisation of various TT algorithms
- Including TDVP where time step is naturally tunable
- $\frac{\text{communications}}{\text{computations}} \to 0$ as $n, r, d \to \infty$ (weak scaling)

- Reference: Phys. Rev. B **101** or arXiv:1912.06127
- See also:

  Ceruti, Kusch, Lubich: parallel DLRA arXiv:2304.05660

Conclusion

- Inverse gauge allows parallelisation of various TT algorithms
- Including TDVP where time step is naturally tunable
- $\frac{\text{communications}}{\text{computations}} \to 0$ as $n, r, d \to \infty$ (weak scaling)

- Reference: Phys. Rev. B **101** or arXiv:1912.06127
- See also:

  Ceruti, Kusch, Lubich: parallel DLRA arXiv:2304.05660

Thank you for your attention!