Numerical simulations using low-rank approximation

Marco Sutti

Postdoctoral fellow at NCTS

國家理論科學研究中心 數學組、博士後

2024 ATAT in HPSC, Hsinchu

March 23, 2024

Overview

Preprint: Implicit low-rank Riemannian schemes for the time integration of stiff partial differential equations, M. Sutti and B. Vandereycken, submitted, arXiv preprint arXiv:2305.11532.

Contributions:

- Preconditioner for the Riemannian trust-region (RTR) method on the manifold of fixed-rank matrices (not in this talk).
- Applications within implicit numerical integration schemes to solve stiff, time-dependent PDEs.

This talk:

I. Motivation for considering the low-rank format.

II. The Allen–Cahn equation.

III. The Fisher–KPP equation.

Motivation for the low-rank format/1

- Often, like in CFD, we need to discretize a problem to represent the continuous solution.
- For high-dimensional problems (e.g., Schrödinger equation, Black–Scholes equation…), a "naive" discretization with n degrees of freedom in each dimension, leads to n^d coefficients.
- For example, if $d = 15$, and $n = 100$ grid points in each dimension, we would need 8 000 TB of memory to store all coefficients in double precision:

$$\frac{100^{15} \times 64 \text{ bits}}{8 \times 10^{12}} = 8 \times 10^{18} = 8\,000 \text{ TB}.$$

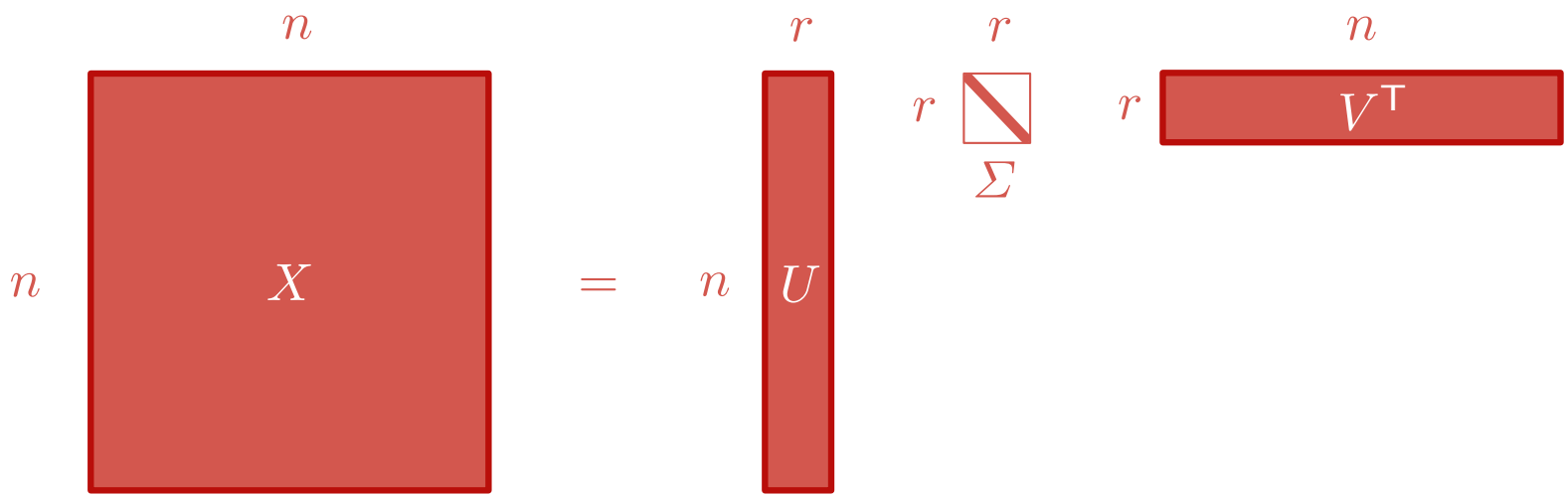
NB: 64 is the number of bits necessary to represent a number in double precision arithmetic.
1 TB = 2^{40} bytes $\approx 10^{12}$ bytes (1 TB is one trillion bytes).

- Since the number of coefficients scales exponentially by d but the accuracy is typically determined by n, this poses a limitation on the size of the problems $\rightsquigarrow$ *Curse of dimensionality*.

Matrix factorization

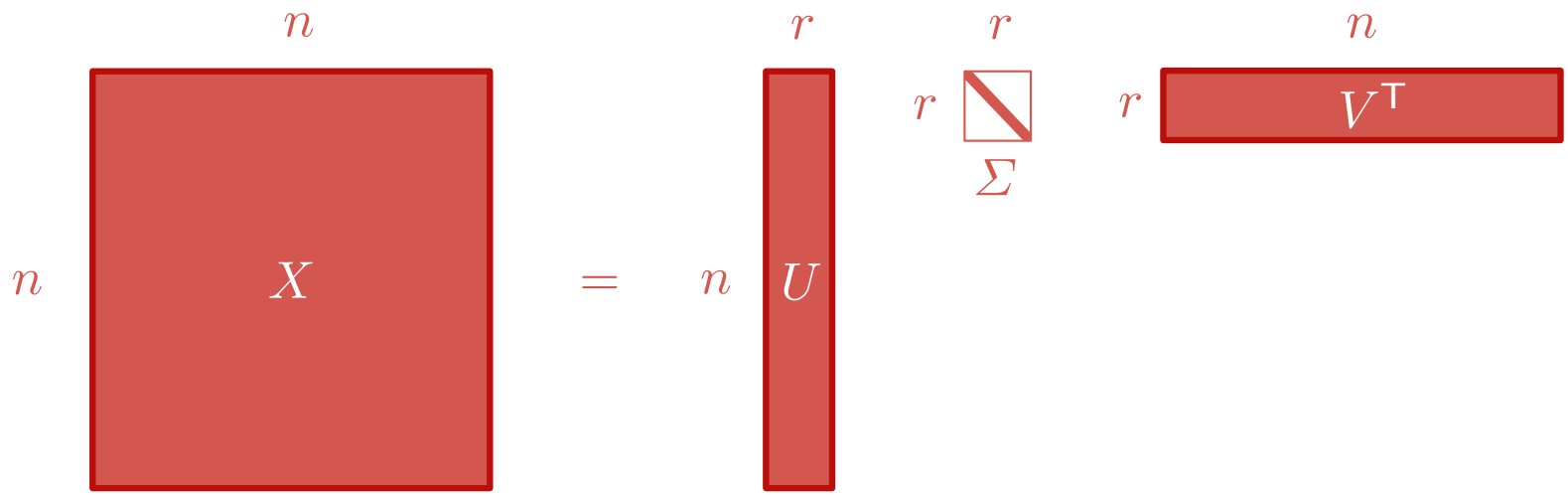
- One of the possible workarounds $\rightsquigarrow$ matrix factorization!

$$X = U\Sigma V^\top: \; U^\top U = I_r, \; V^\top V = I_r, \; \Sigma = \operatorname{diag}(\sigma_i), \; \sigma_1 \geqslant \cdots \geqslant \sigma_r > 0.$$



- Only $2nr + r$ coefficients instead of n^2. If $r \ll n$, then big memory savings.
- Perform the calculations directly in the factorized format.

Motivation for the low-rank format/2



- Storing a dense 5000×5000 matrix in double precision takes $5000^2 \times 8/2^{20} \approx 191$ MB.
  - If it has rank 10 and we store only its factors, it takes $(2 \times 5000 \times 10 + 10) \times 8/2^{20} = 0.76$ kB.
  - If it has rank 100 and we store only its factors, it takes $(2 \times 5000 \times 100 + 100) \times 8/2^{20} = 7.63$ MB.
- For a matrix stored in the dense format, the storage complexity grows as n^2, but if the matrix is stored in low-rank format, then the storage grows as nr.

The manifold of fixed-rank matrices $\mathcal{M}_r$

- We will define an optimization problem over

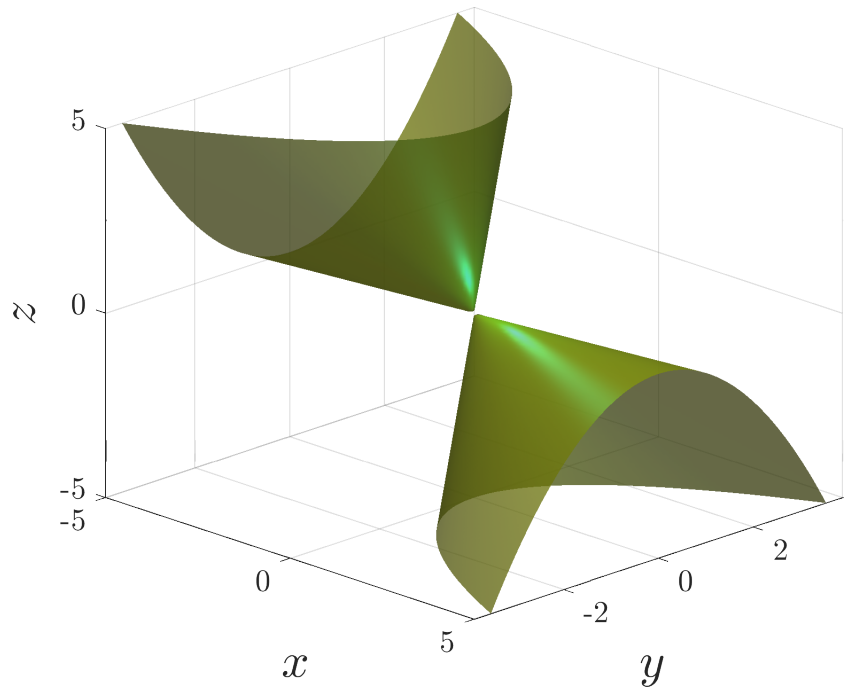
$$\mathcal{M}_r = \{X \in \mathbb{R}^{n\times n} : \ \mathrm{rank}(X) = r\}.$$

$\rightsquigarrow \mathcal{M}_r$ has a smooth structure …

2×2 example:

$$X = \begin{bmatrix} x & -2y \\ y & z \end{bmatrix}.$$

Parametrization:
$\mathrm{rank}(X) = 1 \Leftrightarrow xz = -2y^2$ and $x, z \neq 0$.



- Theorem: $\mathcal{M}_r$ is a smooth Riemannian submanifold embedded in $\mathbb{R}^{n\times n}$ of dimension $r(2n - r)$.

Optimizing on submanifold $\mathcal{M}_r$: [Vandereycken 2013]

The Allen–Cahn equation/1

- Reaction-diffusion equation that models the process of phase separation in multi-component alloy systems.
  - Other applications include mean curvature flows, two-phase incompressible fluids, complex dynamics of dendritic growth, and image segmentation.
- In its simplest form, it reads

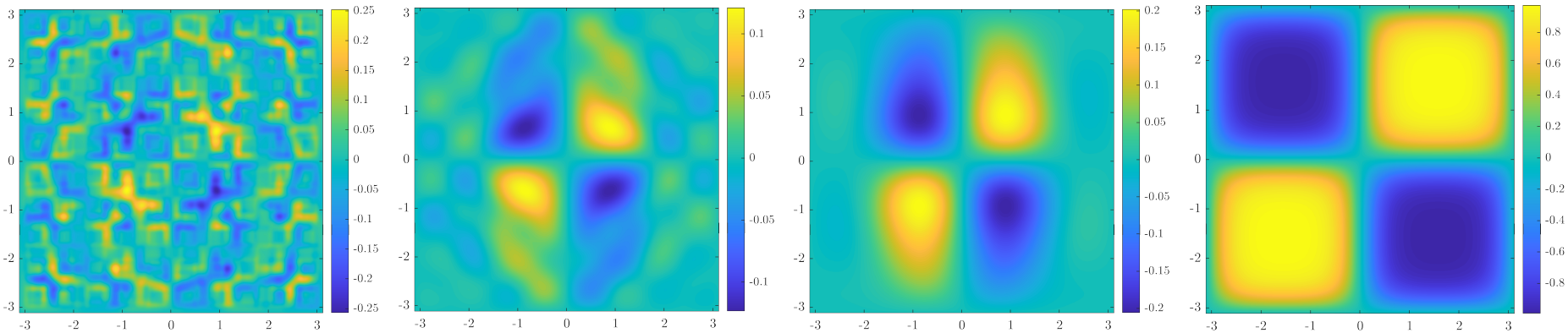
$$\frac{\partial w}{\partial t} = \varepsilon \Delta w + w - w^3,$$

  where $w \equiv w(\boldsymbol{x}, t)$, $\boldsymbol{x} \in \Omega = [-\pi, \pi]^2$, and $t \geq 0$.
- It is a stiff PDE with a low-order polynomial nonlinearity and a diffusion term $\varepsilon \Delta w$.

Allen–Cahn equation: [Allen/Cahn 1972, Allen/Cahn 1973]

The Allen–Cahn equation/2 - “naive” discretization

- Spatial discretization on a uniform grid, 256×256 grid points.
  - Storage of each matrix: $256^2 \times 8/2^{20} \approx 0.5$ MB.
  - The Laplacian Δw is discretized using central finite differences with periodic boundary conditions.
- Numerical time integration with a fourth-order Runge–Kutta method (ERK4), $h = 10^{-4}$, because of the condition for an explicit scheme to be stable (very similar to the CFL condition). Very small time step!

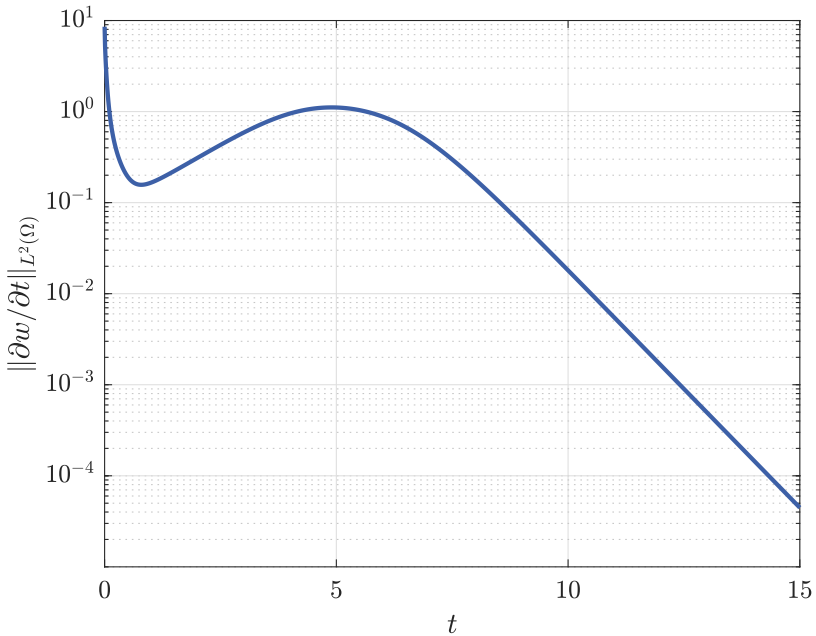


(a) $t = 0$ (b) $t = 0.5$ (c) $t = 2$ (f) $t = 15$

Figure: Time evolution of the solution w to the Allen–Cahn equation, with ERK4, $h = 10^{-4}$.

The Allen–Cahn equation/3 - stationary phase

$$\frac{\partial w}{\partial t} = \varepsilon \Delta w + w - w^3.$$



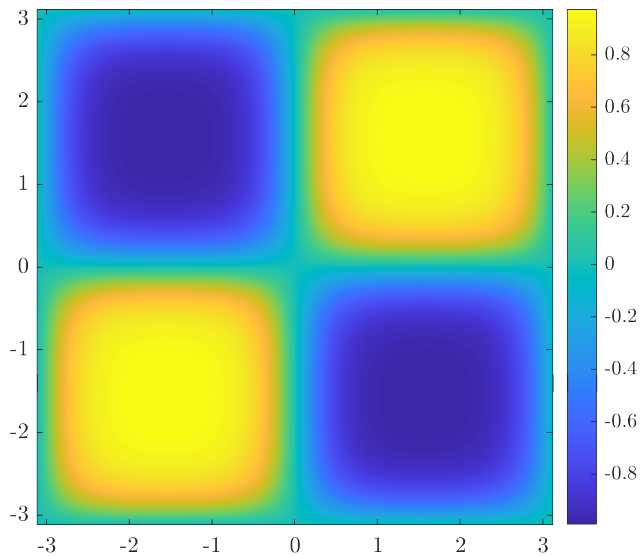


Figure: Left panel: time evolution of the RHS of the Allen–Cahn equation. Right panel: numerical solution w at time $t = 15$, with ERK4, $h = 10^{-4}$.

- For "big enough" t, $\partial w/\partial t \approx 0$, i.e., the solution w enters a steady state.

The Allen–Cahn equation/4 - rank assessment

- Question: is it low rank? Preliminary study on the dense-format solution.

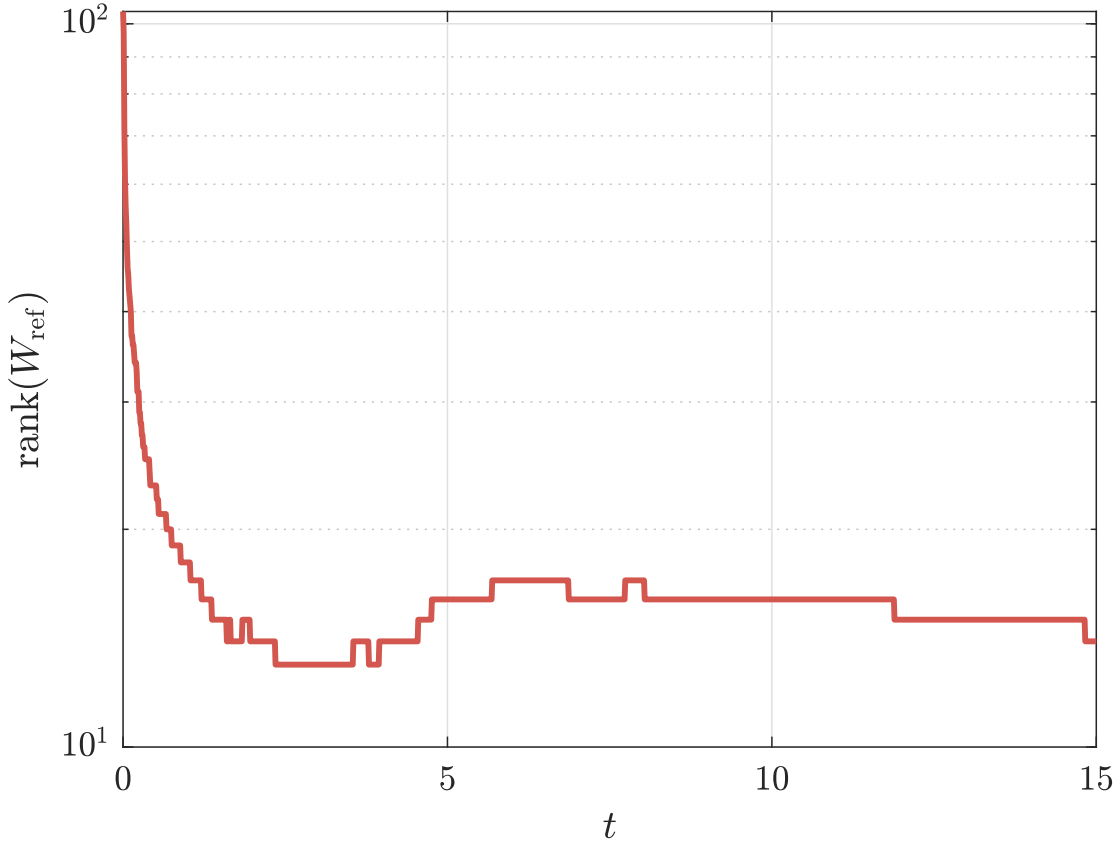


Figure: Time evolution of the rank of the numerical solution W_{ref} to the Allen–Cahn equation, with ERK4, $h = 10^{-4}$.

Implicit numerical integration scheme & low-rank format

- The reference solution in the previous slides is computed with an explicit fourth-order Runge–Kutta method (ERK4), $h = 10^{-4}$. Very small!

  $\rightsquigarrow$ Time to perform the entire simulation until $t = 15$: $\approx$ 36.5 minutes!

- We could use an implicit numerical integration scheme for the time integration.

  ⊕ It allows for a larger time step than its explicit counterpart.

  ⊖ Typically requires the solution of nonlinear equations, which is very expensive.

$\rightsquigarrow$ Idea: Using the low-rank format to reduce the computational cost together with an implicit numerical time integration scheme that avoids the restriction on the time step due to the stability condition!

The Allen–Cahn equation/5 - low-rank simulation

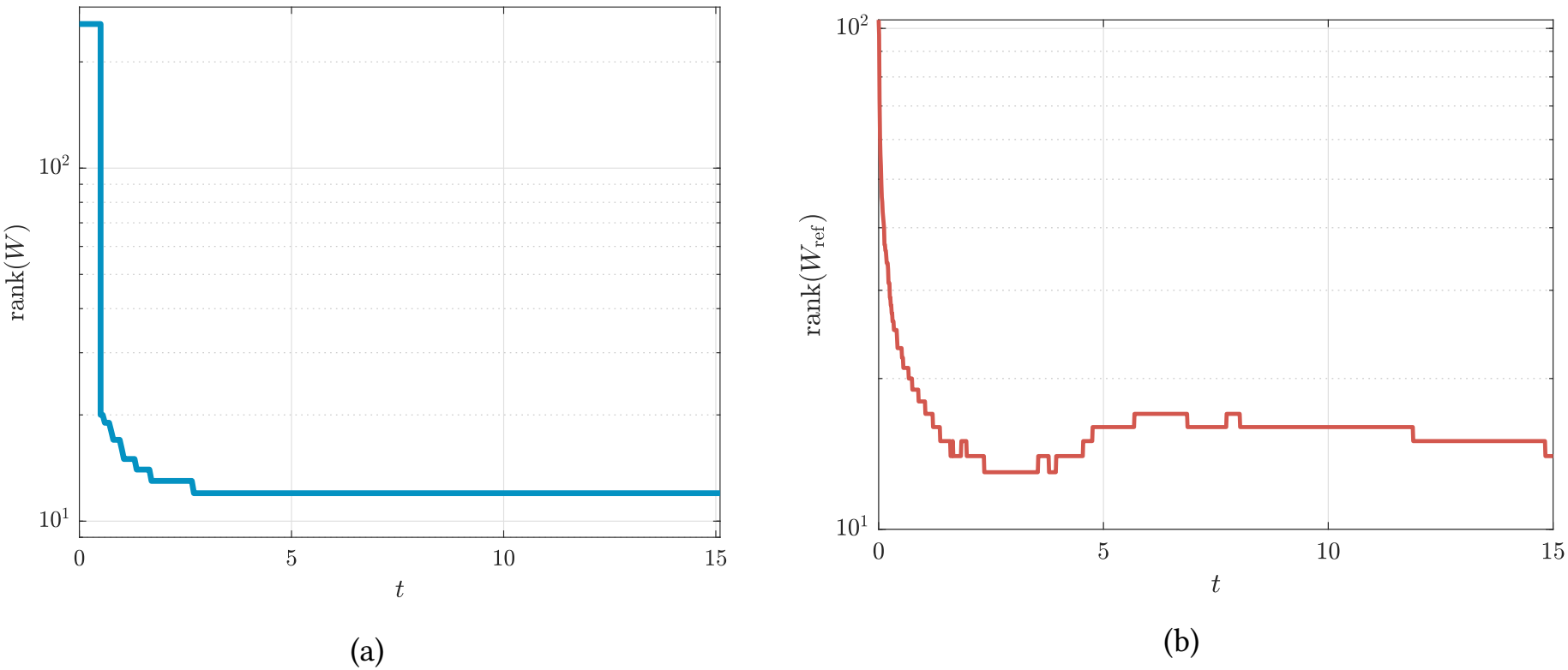


Figure: Panel (a): error versus time for the low-rank evolution of the Allen–Cahn equation. Panel (b): rank evolution of the reference dense-format solution W_{ref}.

The Allen–Cahn equation/6 - low-rank simulation

- We can take very big time steps, and still, the numerical solution at the final time has an acceptable error with respect to the reference solution.
  - Time to perform the simulation until $t = 15$, with $h = 0.05$: $\approx$ 5 minutes.
  - Time to perform the simulation until $t = 15$, with $h = 1.00$: $\approx$ 12.5 seconds.

$\rightsquigarrow$ Compare with the $\approx$ 36.5 minutes for the dense format!

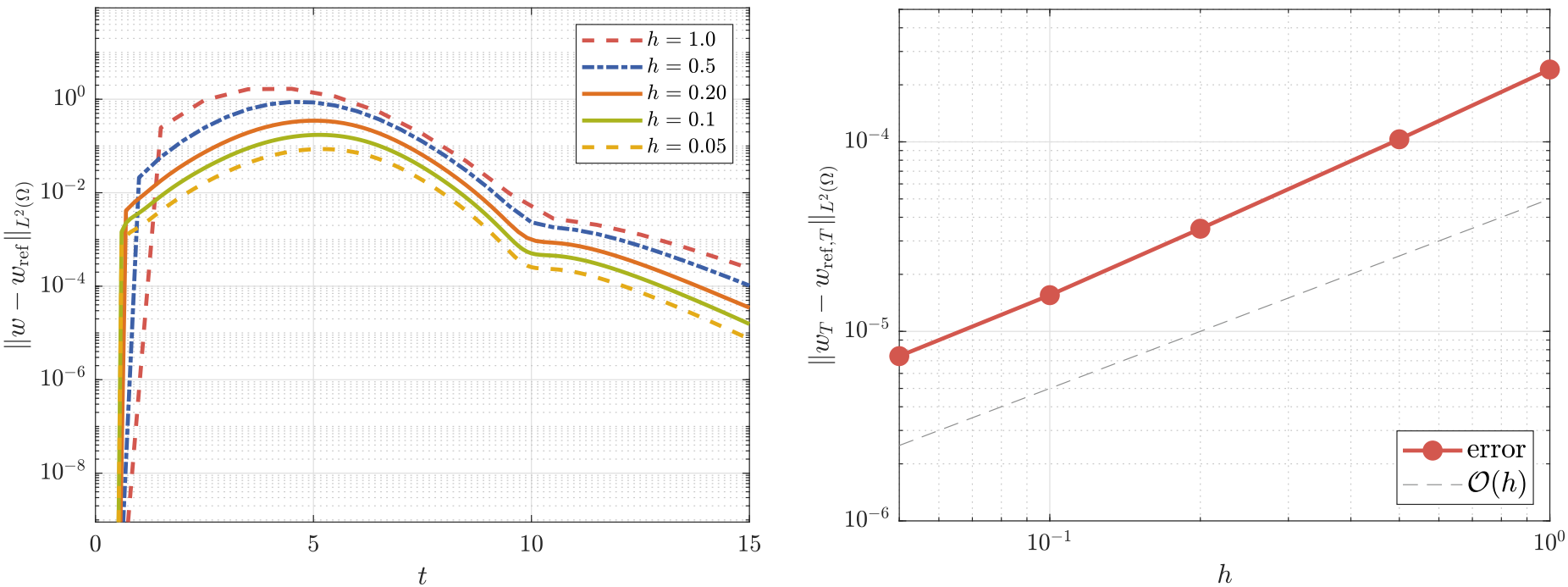


Figure: Panel (a): error versus time for the low-rank evolution of the Allen–Cahn equation. Panel (b): error at $T = 15$ versus time step h.

Fisher–KPP equation/1

- Nonlinear reaction-diffusion equation.
  - Models biological population, chemical reaction dynamics with diffusion, theory of combustion to study flame propagation, nuclear reactors, …
- In its simplest form, it reads

$$\frac{\partial w}{\partial t} = \frac{\partial^2 w}{\partial x^2} + r(\omega)\, w(1-w),$$

  where $w \equiv w(x,t;\omega)$, $r(\omega)$ is a species's reaction rate or growth rate, modeled as a random variable that follows a uniform law $r \sim \mathcal{U}[1/4, 1/2]$.
- Spatial domain: $x \in [0, 40]$, time domain: $t \in [0, 10]$.
- Homogeneous Neumann boundary conditions, i.e.,

$$\forall t \in [0,10], \quad \frac{\partial w}{\partial x}(0,t) = 0, \quad \frac{\partial w}{\partial x}(40,t) = 0.$$

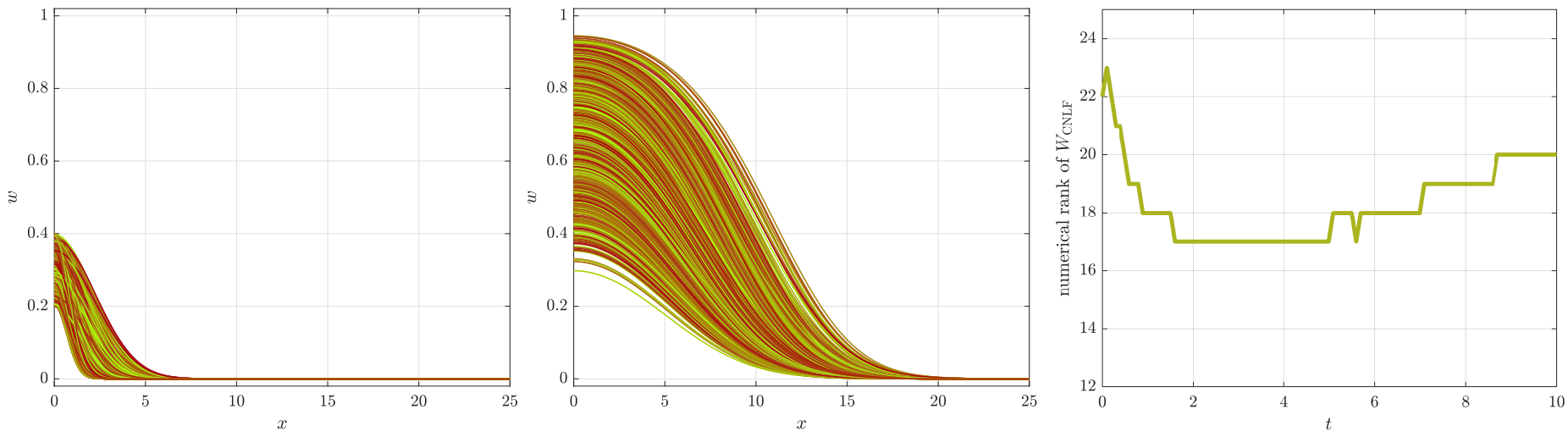
Fisher–KPP equation: [Fisher 1937, Kolmogorov/Petrowsky/Piskunov 1937]

Fisher–KPP equation/2

- The initial condition is of the form

$$w(x, 0; \omega) = a(\omega)\, e^{-b(\omega)\, x^2},$$

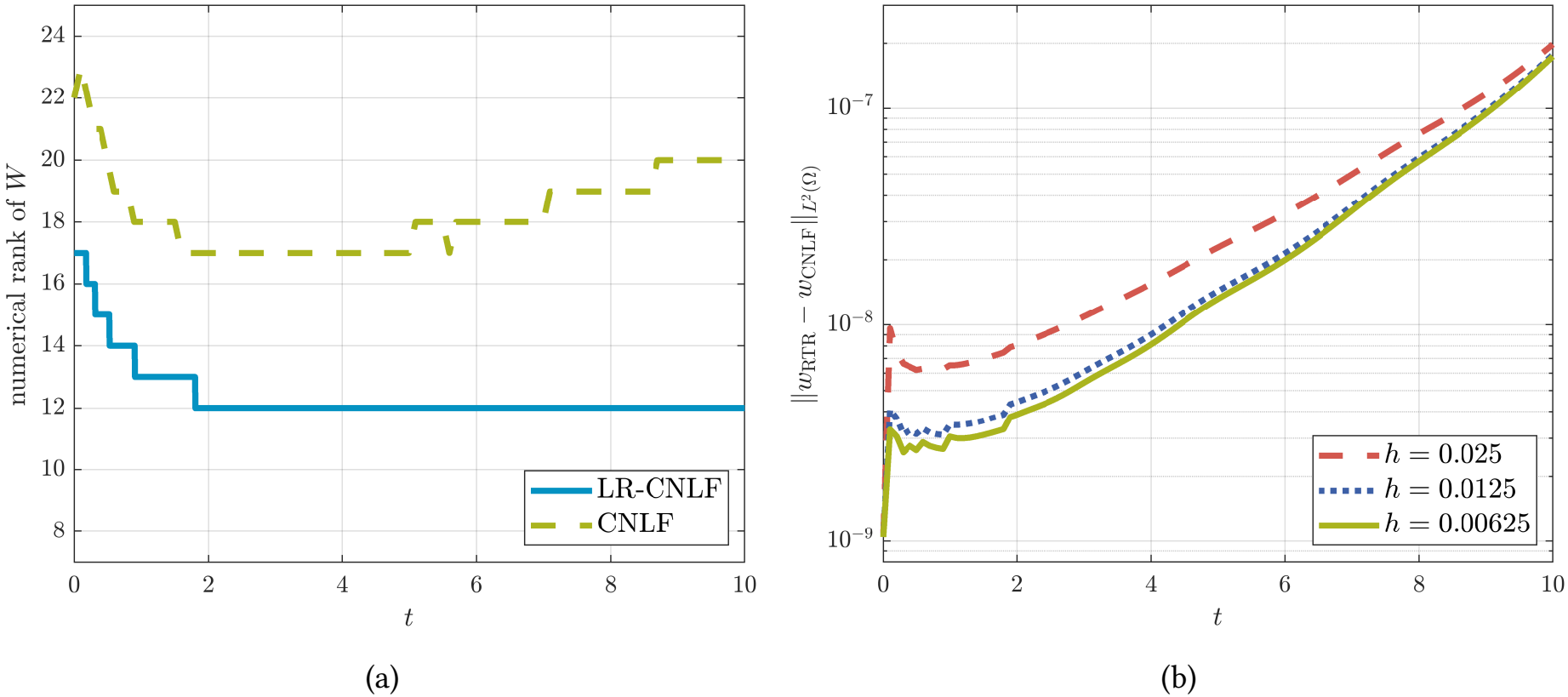
where $a \sim \mathcal{U}[1/5,\ 2/5]$ and $b \sim \mathcal{U}[1/10,\ 11/10]$. The random variables a, b, and r are all independent, and we consider $N_r = 1000$ realizations.



(a) (b) (c)

Figure: Fisher–KPP reference solution computed with an IMEX-CNLF scheme. Panel (a): all the 1000 realizations at $t = 0$. Panel (b): all the 1000 realizations at $t = 10$. Panel (c): numerical rank history.

Fisher–KPP equation/3 - low-rank evolution



(a) (b)

Figure: Panel (a): rank history for the low-rank version (LR-CNLF) compared to the reference solution (CNLF), for $h = 0.00625$. Panel (b): discrete L^2-norm of the error versus time, for several h.

Conclusions

Take-home message: Using a low-rank format allows us to reduce the computational cost and use an implicit numerical time integration scheme that avoids the time step restriction of the stability condition!

Pros and cons:

⊕ Efficient simulations with the low-rank format.

⊕ Solid, well-understood theory behind (not discussed in this talk).

⊖ If the problem does not really admit a low-rank representation, then there is no advantage over using dense matrices.

Outlook:

- Use higher-order numerical integration methods.
- Other applications in mind, e.g., diffusion problems in mathematical biology or problems with low-rank tensor structure.

Thank you for your attention!

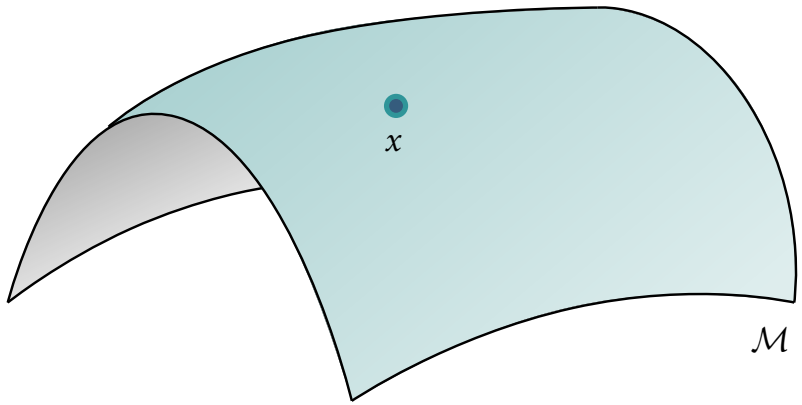
Bonus material

Optimization problems on matrix manifolds

- We can state the optimization problem as

$$\min_{x \in \mathcal{M}} f(x),$$

where $f : \mathcal{M} \to \mathbb{R}$ is the objective function and $\mathcal{M}$ is some matrix manifold.



- Matrix manifold: any manifold that is constructed from $\mathbb{R}^{n \times p}$ by taking either embedded submanifolds or quotient manifolds.
  - Examples of embedded submanifolds: orthogonal Stiefel manifold, manifold of symplectic matrices, manifold of fixed-rank matrices, …
  - Example of quotient manifold: the Grassmann manifold.
- Motivation: by exploiting the underlying geometric structure, only feasible points are considered!

Manifold optimization: [Edelman et al. 1998, Absil et al. 2008, Boumal 2023], …

Problems considered: variational problems

- Variational problem, called "LYAP" herein,

$$\begin{cases} \min\limits_{w} \mathcal{F}(w(x,y)) = \displaystyle\int_{\Omega} \tfrac{1}{2}\|\nabla w(x,y)\|^2 - \gamma(x,y)\, w(x,y)\, \mathrm{d}x\, \mathrm{d}y \\ \quad \text{such that} \quad w = 0 \text{ on } \partial\Omega, \end{cases}$$

  where $\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right)$, $\Omega = [0,1]^2$ and γ is the source term.

- Discretization on a uniform grid: regardless of the specific form of $\mathcal{F}$, we obtain the general formulation

$$\min_{W} F(W) \quad \text{s.t.} \quad W \in \{X \in \mathbb{R}^{n\times n} : \ \operatorname{rank}(X) = r\},$$

  where F denotes the discretization of the functional $\mathcal{F}$.

"LYAP" variational problem: [Henson 2003, Gratton/Sartenaer/Toint 2008, Wen/Goldfarb 2009, S./Vandereycken 2021, …]

Riemannian manifold and gradient

A manifold $\mathcal{M}$ endowed with a smoothly-varying inner product (called Riemannian metric g) is called Riemannian manifold.

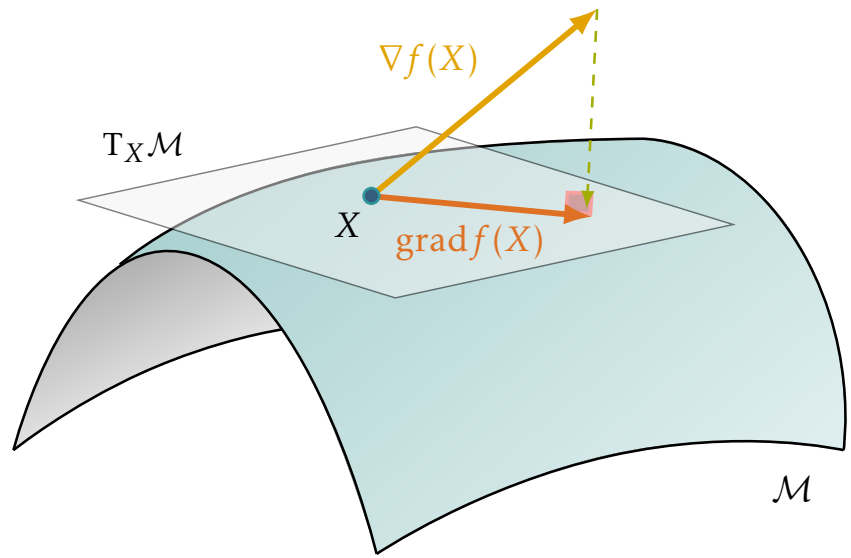
$\rightsquigarrow$ A couple $(\mathcal{M}, g)$, i.e., a manifold with a Riemannian metric on it.

Let $f : \mathcal{M} \to \mathbb{R}$. E.g., the objective function in an optimization problem.

$\rightsquigarrow$ For any embedded submanifold:

- Riemannian gradient: projection onto $\mathrm{T}_X\mathcal{M}$ of the Euclidean gradient

$$\operatorname{grad} f(X) = \mathrm{P}_{\mathrm{T}_X\mathcal{M}}(\nabla f(X)).$$



$\rightsquigarrow$ $\nabla f(X)$ is the Euclidean gradient of $f(X)$.

Matrix and vector calculus: The Matrix Cookbook, www.matrixcalculus.org, ...
Automatic differentiation on low-rank manifolds: [Novikov/Rakhuba/Oseledets 2022]

Riemannian trust-region (RTR) method

Algorithm 1: Riemannian trust-region (RTR)

Given $\bar{\Delta} > 0$, $\Delta_1 \in (0, \bar{\Delta})$
for $i = 1, 2, \ldots$ **do**
  Define the second-order model

$$m_i : \mathrm{T}_{x_i}\mathcal{M} \to \mathbb{R},\ \xi \mapsto f(x_i) + \langle \operatorname{grad} f(x_i), \xi \rangle + \frac{1}{2} \langle \operatorname{Hess} f(x_i)[\xi], \xi \rangle.$$

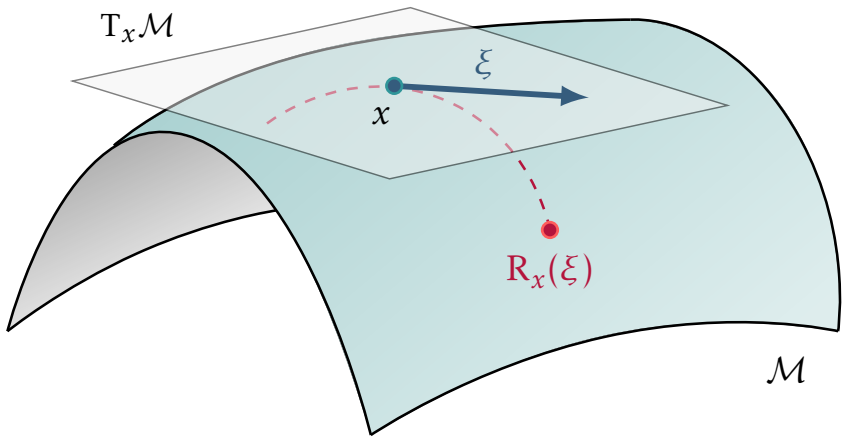
  Trust-region subproblem: compute η_i by solving

$$\eta_i = \operatorname{argmin} m_i(\xi) \quad \text{s.t.} \quad \|\xi\| \leq \Delta_i.$$

  Compute $\rho_i = (\widehat{f}(0) - \widehat{f_i}(\eta_i))/(m_i(0) - m_i(\eta_i))$.
  if $\rho_i \geq 0.05$ **then**
    Accept step and set $x_{i+1} = \mathrm{R}_{x_i}(\eta_i)$.
  else
    Reject step and set $x_{i+1} = x_i$.
  end if
  Radius update: set

$$\Delta_{i+1} = \begin{cases} \min(2\Delta_i, \bar{\Delta}) & \text{if } \rho_i \geq 0.75 \text{ and } \|\eta_i\| = \Delta_i, \\ 0.25\,\|\eta_i\| & \text{if } \rho_i \leq 0.25, \\ \Delta_i & \text{otherwise.} \end{cases}$$

end for



TR method: [Goldfeld/Quandt/Trotter 1966, Sorensen 1982, Fletcher 1980/1987 ...]
RTR method: [Absil/Baker/Gallivan 2007]

The manifold of fixed-rank matrices $\mathcal{M}_r$

- Our optimization problem is defined over

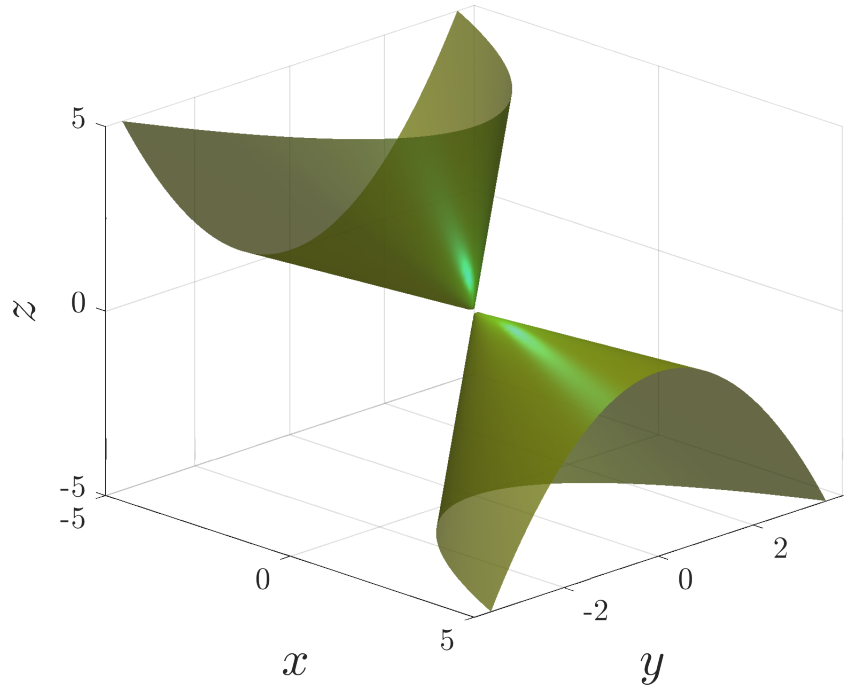
$$\mathcal{M}_r = \{X \in \mathbb{R}^{n\times n} : \ \mathrm{rank}(X) = r\}.$$

$\rightsquigarrow \mathcal{M}_r$ has a smooth structure …

2×2 example:

$$X = \begin{bmatrix} x & -2y \\ y & z \end{bmatrix}.$$

Parametrization:
$\mathrm{rank}(X) = 1 \Leftrightarrow xz = -2y^2$ and $x, z \neq 0$.



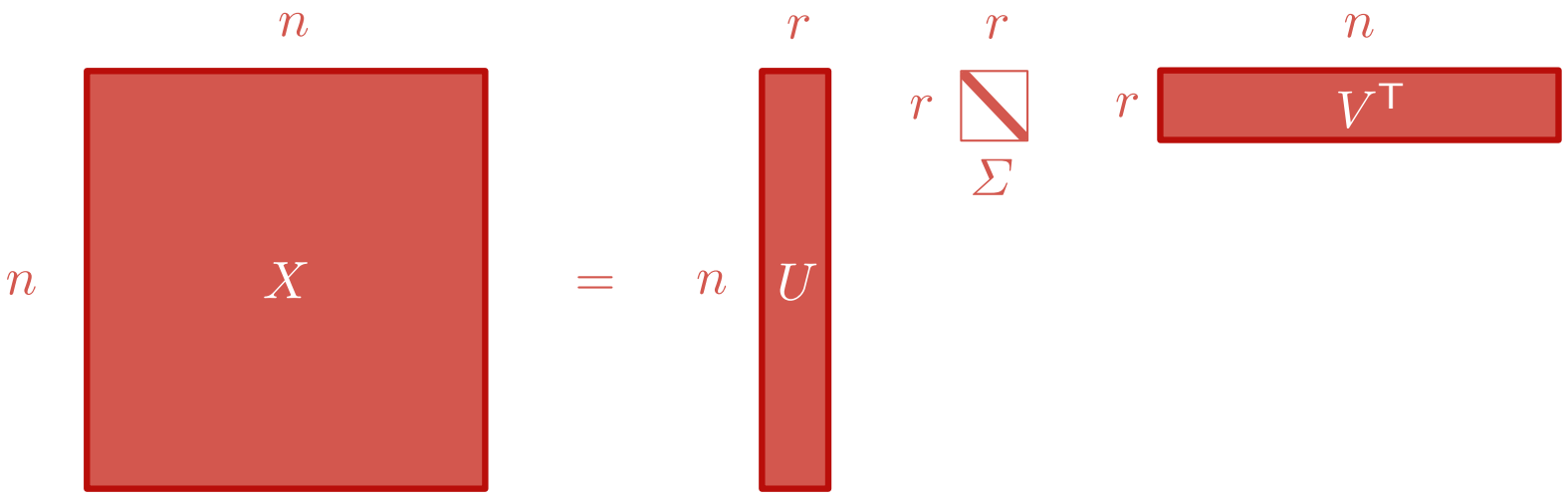
- Theorem: $\mathcal{M}_r$ is a smooth Riemannian submanifold embedded in $\mathbb{R}^{n\times n}$ of dimension $r(2n - r)$.

Optimizing on submanifold $\mathcal{M}_r$: [Vandereycken 2013]

$\mathcal{M}_r$: Alternative characterization

- Using the singular value decomposition (SVD), we have the equivalent characterization

$$\mathcal{M}_r = \{U\Sigma V^\top : \ U^\top U = I_r, \ V^\top V = I_r, \ \Sigma = \mathrm{diag}(\sigma_i), \ \sigma_1 \geqslant \cdots \geqslant \sigma_r > 0\}.$$



- Only $2nr + r$ coefficients instead of n^2. If $r \ll n$, then big memory savings.
- Perform the calculations directly in the factorized format.

$\mathcal{M}_r$: Tangent vectors

- A tangent vector ξ at $X = U\Sigma V^\top$ is represented as

$$\xi = UMV^\top + U_p V^\top + UV_p^\top,$$

$$M \in \mathbb{R}^{r\times r}, \quad U_p \in \mathbb{R}^{n\times r}, \quad U_p^\top U = 0, \quad V_p \in \mathbb{R}^{n\times r}, \quad V_p^\top V = 0.$$

- We can rewrite it as

$$\xi = (UM + U_p)V^\top + UV_p^\top.$$

$\rightsquigarrow$ ξ is a rank-$2r$ bounded matrix. Useful in implementation.

$\mathcal{M}_r$: Metric, projection, gradient, retraction

- The Riemannian metric is

$$g_X(\xi, \eta) = \langle \xi, \eta \rangle = \mathrm{Tr}(\xi^\top \eta), \quad \text{with} \quad X \in \mathcal{M}_r \quad \text{and} \quad \xi, \eta \in \mathrm{T}_X \mathcal{M}_r,$$

  where ξ, η are seen as matrices in the ambient space $\mathbb{R}^{n \times n}$.

- Orthogonal projection onto the tangent space at X is

$$\mathrm{P}_{\mathrm{T}_X \mathcal{M}_r} : \mathbb{R}^{n \times n} \to \mathrm{T}_X \mathcal{M}_r, \qquad Z \to \mathrm{P}_U Z \mathrm{P}_V + \mathrm{P}_U^\perp Z \mathrm{P}_V + \mathrm{P}_U Z \mathrm{P}_V^\perp.$$

- Riemannian gradient: projection onto $\mathrm{T}_X \mathcal{M}_r$ of the Euclidean gradient

$$\mathrm{grad}\, f(X) = \mathrm{P}_{\mathrm{T}_X \mathcal{M}_r}(\nabla f(X)).$$

- Retraction $\mathrm{R}_X : \mathrm{T}_X \mathcal{M}_r \to \mathcal{M}_r$. Typical: truncated SVD.

Many retractions for $\mathcal{M}_r$: [Absil/Oseledets 2015]

Riemannian Hessian and preconditioning/1

- In the case of Riemannian submanifolds, the full Riemannian Hessian of f at $x \in \mathcal{M}$ is given by the projected Euclidean Hessian plus the curvature part

$$\operatorname{Hess} f(x)[\xi] = P_x \nabla^2 f(x) P_x + P_x(\text{“curvature terms”}) P_x.$$

  $\rightsquigarrow$ Use $P_x \nabla^2 f(x) P_x$ as a preconditioner in RTR.

- For LYAP, we can get the symmetric n^2-by-n^2 matrix

$$H_X = P_X(A \otimes I + I \otimes A) P_X.$$

- Inverse of $H_X \rightsquigarrow$ good candidate for a preconditioner.

  ⚠ Not inverted directly, since this would cost $\mathcal{O}(n^6)$.

- A good preconditioner should reduce the number of iterations of the inner trust-region solver. It has to be effective and cheap to compute.

Symmetric positive semidefinite matrices with fixed rank: [Vandereycken/Vandewalle 2010]

Riemannian Hessian and preconditioning/2

- Applying the preconditioner in $X \in \mathcal{M}_r$ means solving for $\xi \in \mathrm{T}_X\mathcal{M}$ the system

$$H_X \operatorname{vec}(\xi) = \operatorname{vec}(\eta),$$

where $\eta \in \mathrm{T}_X\mathcal{M}$ is a known tangent vector.

- This is equivalent to

$$\mathrm{P}_X(A\xi + \xi A) = \eta.$$

- Using the definition of the orthogonal projector onto $\mathrm{T}_X\mathcal{M}_r$, we obtain

$$P_U(A\xi + \xi A)P_V + P_U^\perp(A\xi + \xi A)P_V + P_U(A\xi + \xi A)P_V^\perp = \eta,$$

which is equivalent to the system

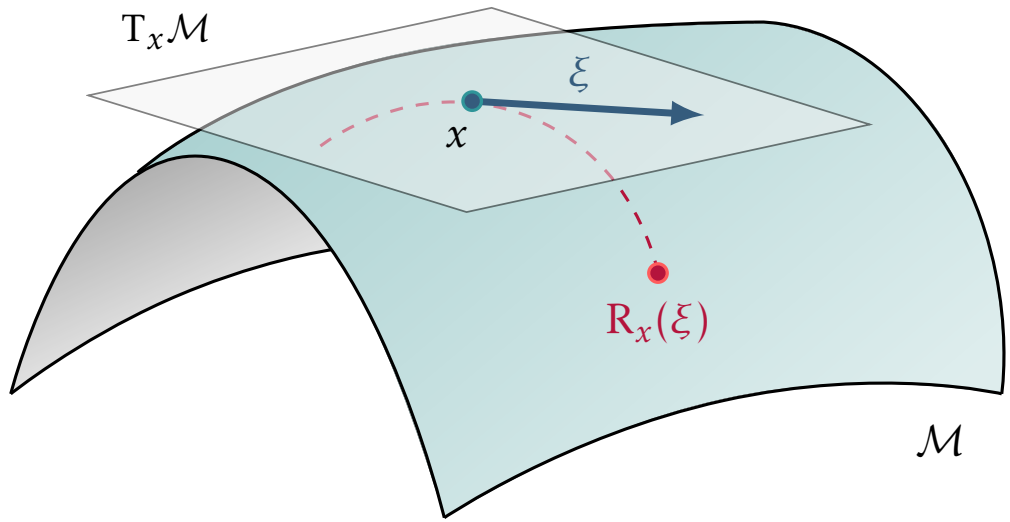
$$\begin{cases} P_U(A\xi + \xi A)P_V = P_U\eta P_V, \\ P_U^\perp(A\xi + \xi A)P_V = P_U^\perp\eta P_V, \\ P_U(A\xi + \xi A)P_V^\perp = P_U\eta P_V^\perp. \end{cases}$$

$$\vdots$$

$\rightsquigarrow$ Many (boring) calculations, but the numerical results are quite striking!

Retractions

- Move in the direction of ξ while remaining constrained to $\mathcal{M}$.
- Smooth mapping $\mathrm{R}_x : \mathrm{T}_x\mathcal{M} \to \mathcal{M}$ with a local condition that preserves gradients at x.



- The Riemannian exponential mapping is also a retraction, but it is not computationally efficient.
- Retractions: first-order approximation of the Riemannian exponential!

Constructing retractions: [Absil/Malick 2012]

An example of factorized gradient

- "LYAP" functional: $\mathcal{F}(w(x,y)) = \int_\Omega \frac{1}{2}\|\nabla w(x,y)\|^2 - \gamma(x,y)\, w(x,y)\,\mathrm{d}x\,\mathrm{d}y$.
- The gradient of $\mathcal{F}$ is the variational derivative $\frac{\delta \mathcal{F}}{\delta w} = -\Delta w - \gamma$.
- The discretized Euclidean gradient in matrix form is given by

$$G = AW + WA - \Gamma.$$

  with A is the second-order periodic finite difference differentiation matrix.
- The first-order optimality condition $G = AW + WA - \Gamma = 0$ is a Lyapunov (or Sylvester) equation.

$\rightsquigarrow$ Factorized Euclidean gradient:

$$G = \begin{bmatrix} AU & U & U_\gamma \end{bmatrix} \operatorname{blkdiag}\left(\Sigma, \Sigma, \Sigma_\gamma\right) \begin{bmatrix} V & AV & V_\gamma \end{bmatrix}^\top.$$