lrAA: Low-Rank Anderson Acceleration

Yingda Cheng, Virginia Tech

Joint work with **Daniel Appelö** (VT)
Supported by DOE MMICC center DE-SC0023164 and ASCR Quantum Computing DE-SC0025424

Ref: D. Appelo & Y. Cheng, https://arxiv.org/abs/2503.03909

Outline:

- **Motivation**
- Low-rank solution to nonlinear matrix differential equations
- Cross-DEIM
- Numerical experiments

Motivation

Low-rank compression is one of the big ideas in applied math.

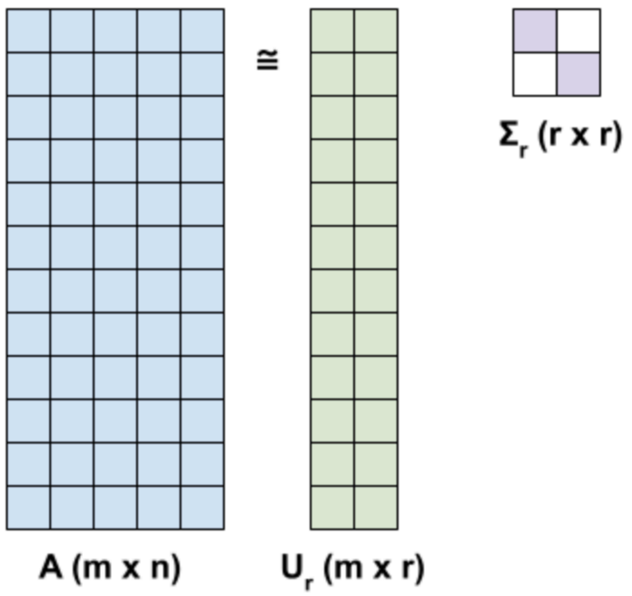
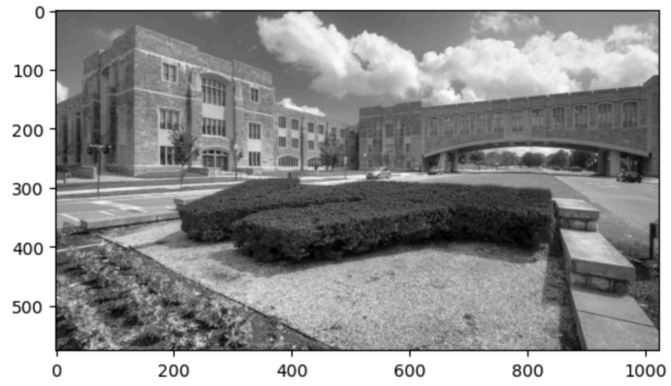
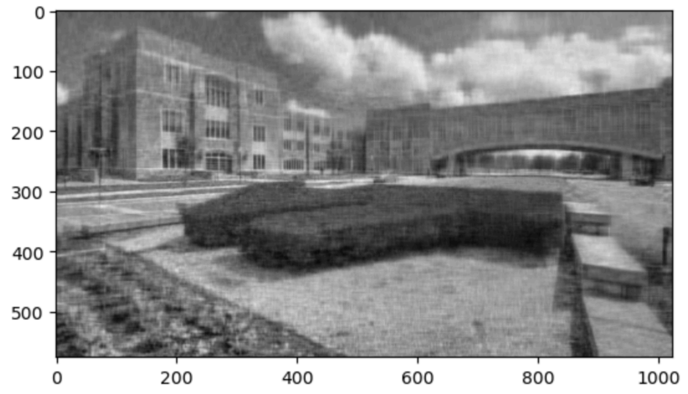


Figure from tensorflow.org



Original 576x1024



Rank 40 approximation

Motivation

Low-rank matrix/tensor have been widely adopted in data science, quantum mechanics …

[PDF] **Lora: Low-rank** adaptation of large language models.

EJ Hu, Y Shen, P Wallis, Z Allen-Zhu, Y Li, S Wang… - ICLR, 2022 - arxiv.org

… **Low-Rank** Adaptation, or **LoRA**, which freezes the pre-trained model weights and injects trainable **rank** … For GPT-3, **LoRA** can reduce the number of trainable parameters by 10,000 …

☆ Save 99 Cite Cited by 12112 Related articles All 12 versions

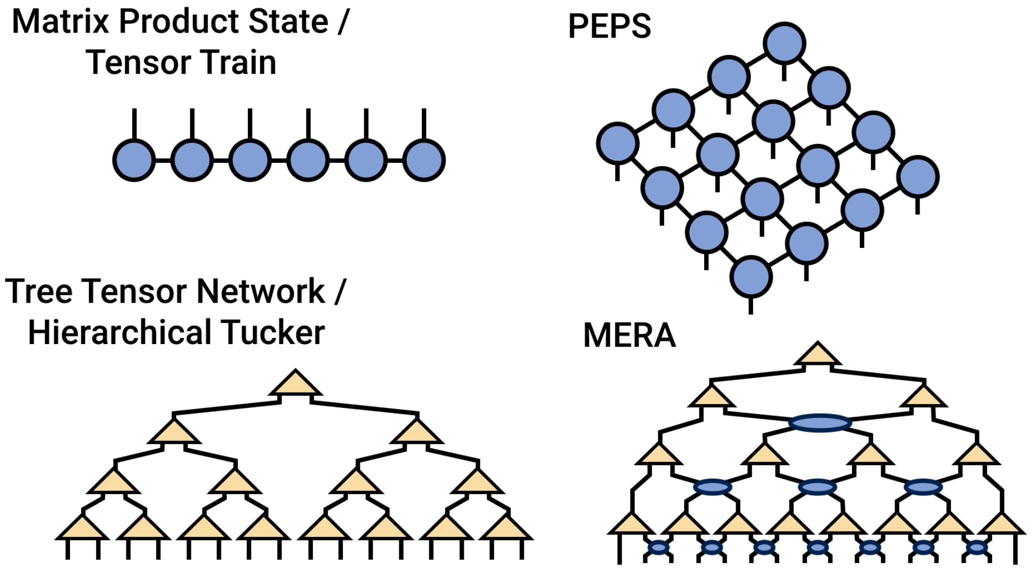


Figure from https://tensornetwork.org/

Low-rank methods to help numerical PDE

Baseline:

We store the unknowns of PDE solution as

$$\text{matrix } \{X_{i,j}\} \text{ or tensor } \{X_{i,j,k\ldots}\}.$$

If they have low-rank property, we modify traditional PDE discretizations to incorporate this compression.

VIRGINIA TECH

Questions:

(1) What is low-rank? - Separation of variables.

- Suppose the PDE solution on 2D $X(x, y)$ is separable, i.e. it can be expressed as $X(x, y) = f(x)g(y)$, Then $X(x, y)$ represented by 1D functions $f(x)$ and $g(y)$.
- Of course, we are not that lucky, but we hope $X(x, y) \approx \sum_{i=1}^{r} f_i(x)g_i(y)$. Then we only need $2r$ 1D functions.

Questions:

(2) If it holds, how do we compute the low-rank factors directly?

- If we can compute $f_i(x), g_i(y)$ directly, then we are "solving" 1D problems.

Low-rank in physical applications

What is low-rank in physical applications?

* Stochastic/parametric problems. Reduced order models are constructed Based on POD.

* Note: the difference with ROM is here we don't have the offline phase. Everything is online.

Low-rank in physical applications

- Kinetic problem $f(x, v) \approx \rho(x)M(v)$, e.g. particles in equilibrium. Meso $\to$ Macro.
- Many body problem $f(x_1, v_1, x_2, v_2) \approx f_1(x_1, v_1)f_2(x_2, v_2)$. Independent particles.
- 'Smoother' is better.
- And quantum applications...

What is low-rank in physical applications?
—- It's a measurement of complexity.

VIRGINIA TECH

Numerical approaches to obtain low-rank solutions

- Time-independent PDE
  - Iterative scheme with truncation
  - Optimization based approaches (ALS)
  - Greedy, PGD
- Time-dependent PDE
  - Dynamic low-rank approximation
  - Step and truncate
  - Space time

See *Bachmayr, Low-rank tensor methods for partial differential equations, Acta Numerica 2023.*

This work belongs to `Iterative scheme with truncation'

—- The key is to control intermediate rank and iteration number

VIRGINIA TECH

This talk will focus on how to obtain low-rank solution of nonlinear PDE formulated as nonlinear matrix equation

$$G(X) = X \quad \text{or} \quad F(X) = 0$$

where X is approximately a low-rank matrix, so 2D case for now

Example: Bratu problem

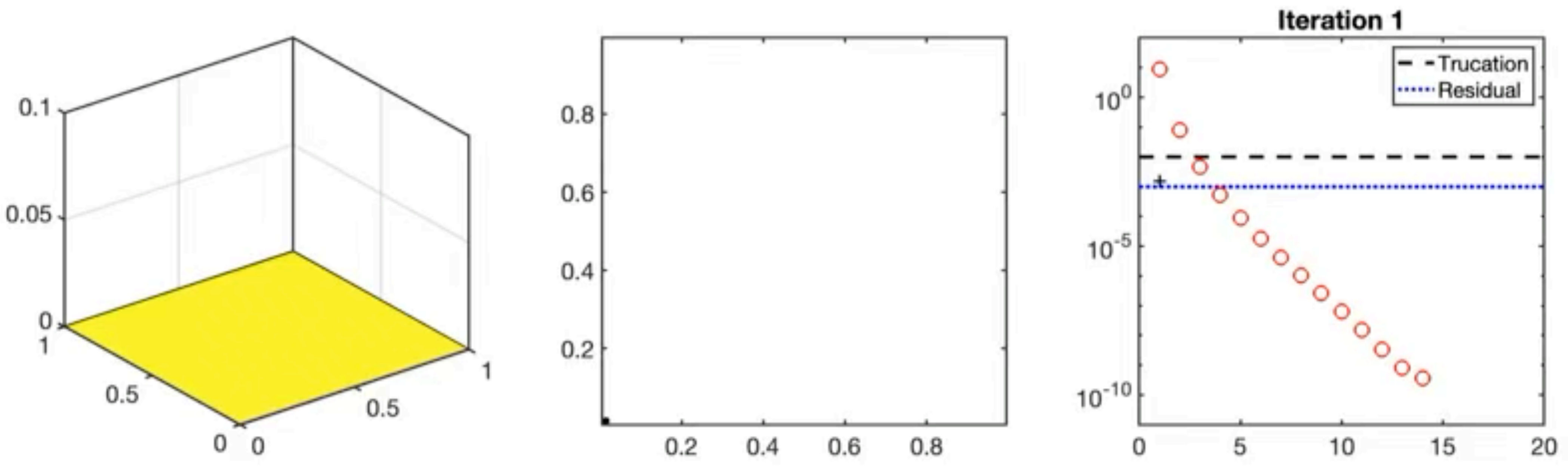
$$u_{xx} + u_{yy} + \lambda e^{u} = 0$$

2nd order FD

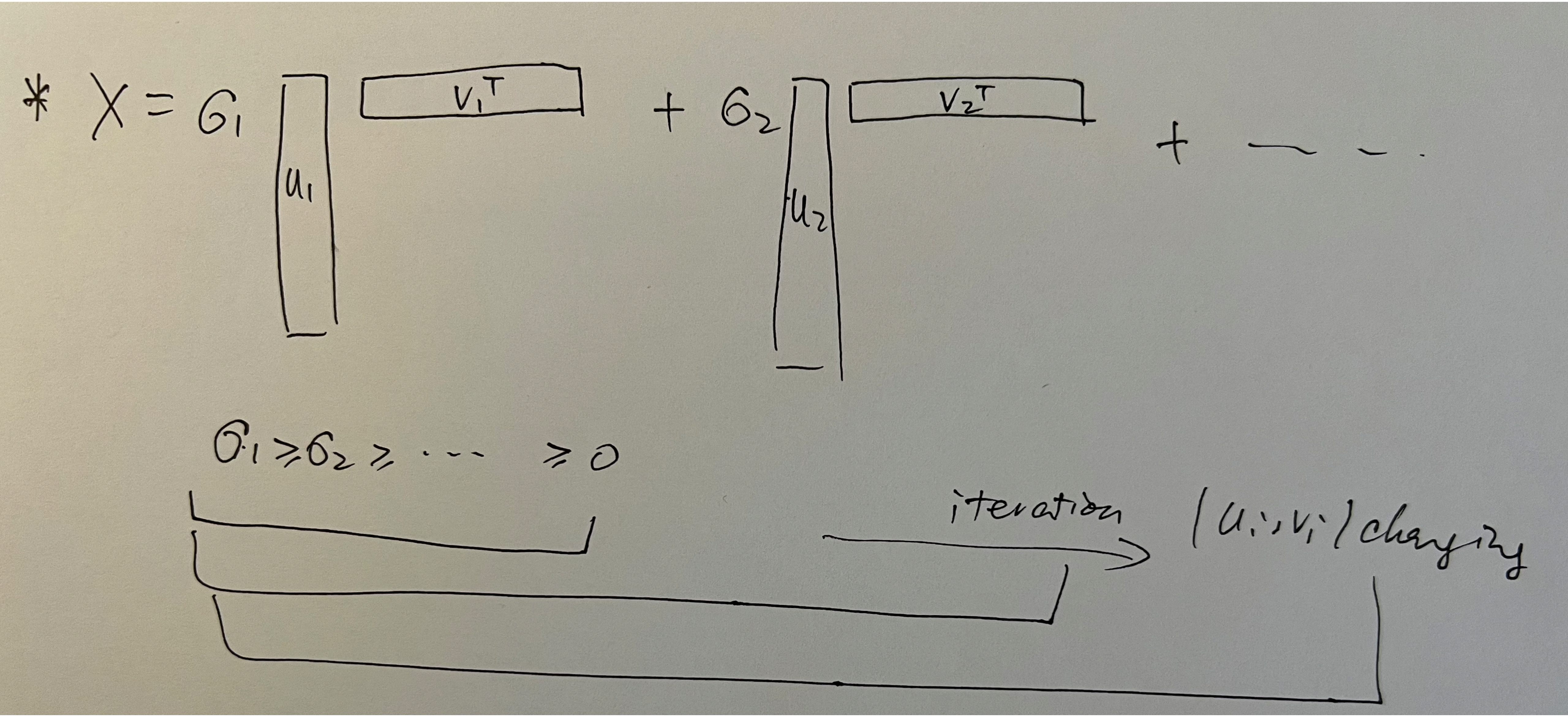
$$\begin{aligned} F_{\mathrm{B}}(i,j;X) &= \frac{1}{h_x^2}\left(X(i+1,j) - 2X(i,j) + X(i-1,j)\right) \\ &+ \frac{1}{h_y^2}\left(X(i,j+1) - 2X(i,j) + X(i,j-1)\right) + \lambda e^{X(i,j)} = 0. \end{aligned} \tag{1}$$

Goal: design iterative scheme to obtain the low-rank factors of X, given a desired Tolerance

Bratu problem



200X200 mesh



VIRGINIA TECH

Outline:

- Motivation
- **Low-rank solution to nonlinear matrix differential equations**
- Cross-DEIM
- Numerical experiments

Low-rank numerical methods for matrix differential equations

- **Linear matrix equations**
  - A well-studied topic in numerical linear algebra, see Simonicini, SIAM review, 2016

- **Nonlinear matrix equations**
  - less studied so far
  - Newton + (low-rank)TT-GMRES, Adak et al, 2024, Rogers, Venturi, 2024
  - Riemannian optimization, Sutti, Vandereycken, 2024
  - Sparse residual collocation, Naderi, Akhavan, Babaee, 2024

Anderson Acceleration is a natural and great candidate for low-rank!

- AA is a popular approach for accelerating fixed point

$$X^{n+1} = G(X^n)$$

- We will see fixed point iteration is great for low rank! (In later slides)
- Finite window size gives us rank control! (Critical point for iterative low-rank methods)
- And more…

This work is motivated by low-rank GMRES for linear problems.

VIRGINIA TECH

Anderson Acceleration (Anderson, 1965)

Algorithm Unconstrained variant of Anderson acceleration in $\mathbb{R}^n$.

Input: $x_0 \in \mathbb{R}^n$, memory parameter $\hat{m} \geq 1$.
Output: $x_k \in \mathbb{R}^n$ as an approximate solution to $x = g(x)$.
$x_1 = g(x_0)$.
for $k = 1, 2, \ldots$ until convergence **do**
  $\hat{m}_k = \min(\hat{m}, k)$.
  Set $D_k = (\Delta f_{k-\hat{m}_k}, \ldots, \Delta f_{k-1})$, where $\Delta f_i = f_{i+1} - f_i$ and $f_i = g(x_i) - x_i$.
  Solve $\gamma^{(k)} = \operatorname{argmin}_{v \in \mathbb{R}^{\hat{m}_k}} \|D_k v - f_k\|, \qquad \gamma^{(k)} = (\gamma_0^{(k)}, \ldots, \gamma_{\hat{m}_k-1}^{(k)})^T$.
  $x_{k+1} = g(x_k) - \sum_{i=0}^{\hat{m}_k-1} \gamma_i^{(k)} \left[g(x_{k-\hat{m}_k+i+1}) - g(x_{k-\hat{m}_k+i}) \right]$.
end for

See *Saad, Acceleration methods for fixed point iterations, Acta Numerica 2025. C. T. Kelley, SIAM book, 2022.*

VIRGINIA TECH

Anderson Acceleration - make it low rank

Algorithm Unconstrained variant of Anderson acceleration in $\mathbb{R}^n$.

Input: $x_0 \in \mathbb{R}^n$, memory parameter $\hat{m} \geq 1$.
Output: $x_k \in \mathbb{R}^n$ as an approximate solution to $x = g(x)$.
$x_1 = g(x_0)$.
for $k = 1, 2, \ldots$ until convergence **do**
  $\hat{m}_k = \min(\hat{m}, k)$.
  Set $D_k = (\Delta f_{k-\hat{m}_k}, \ldots, \Delta f_{k-1})$, where $\Delta f_i = f_{i+1} - f_i$ and $f_i = g(x_i) - x_i$.
  Solve $\gamma^{(k)} = \operatorname{argmin}_{v \in \mathbb{R}^{\hat{m}_k}} \|D_k v - f_k\|$, $\gamma^{(k)} = (\gamma_0^{(k)}, \ldots, \gamma_{\hat{m}_k-1}^{(k)})^T$.
  $x_{k+1} = g(x_k) - \sum_{i=0}^{\hat{m}_k-1} \gamma_i^{(k)} [g(x_{k-\hat{m}_k+i+1}) - g(x_{k-\hat{m}_k+i})]$.
end for

Switch all iterates, e.g. $X_k, G(X_k)$ and everything else into their SVD form

All operations are performed on SVD form, e.g.

Algorithm 2.4 Computing the least squares solution minimizing $\| \sum_{j=1}^{s} \gamma_j U_j S_j V_j^T - U_B S_B V_B^T \|$

Input: low rank matrices in the form $U_j S_j V_j^T, j = 1, \ldots, s$, right hand side $U_B S_B V_B^T$
Output: $\gamma_j, j = 1, \ldots s$
Let $U = [U_1, \ldots, U_s]$, $V = [V_1, \ldots, V_s]$
Perform column pivoted QR: $[Q_1, R_1, \Pi_1] = \mathsf{qr}(U)$, $[Q_2, R_2, \Pi_2] = \mathsf{qr}(V)$
Set $b = \mathsf{vec}(Q_1^T U_B S_B V_B^T Q_2)$.
Find the least squares γ that minimizes the small problem $\|A\gamma - b\|$ where the kth column of A is $a_k = \mathsf{vec}(R_1 \Pi_1^T D_k \Pi_2 R_2^T)$, and $D_k = \mathsf{diag}(0, \ldots, 0, S_k, 0, \ldots, 0)$.

VIRGINIA TECH

For low-rank, we need to truncate! And truncate on nonlinear function!

Algorithm lrAA for nonlinear matrix equation $G(X) = X$.

Input: $X_0 = U_0 S_0 (V_0)^T$, memory parameter $\hat{m} \geq 1$, scheduling parameter $\theta \in (0,1)$, tolerance TOL.
Output: Approximate solution X_k to the fixed point problem $G(X) = X$ in its SVD form.

$\epsilon_G = 10^{-2}$ # Choose ϵ_G so that G_0 has low rank.
$X_1 = G_0 =$ **Cross-DEIM**$(G(X_0), U_0, V_0, \epsilon_G, r_{\max})$.
$\rho_0 = \|X_1 - G_0\|$.
for $k = 1, 2, \ldots$ **do**
  $G_k =$ **Cross-DEIM**$(G(X_k), U_k, V_k, \epsilon_G, r_{\max})$.
  $\rho_k = \|G_k - X_k\|$.
  $\hat{m}_k = \min(\hat{m}, k)$.
  Solve least square problem to get γ^k
  $X_{k+1} =$ **Cross-DEIM**$(G_k - \sum_{i=0}^{\hat{m}_k - 1} \gamma_i^{(k)} \left[G_{k-\hat{m}_k+i+1} - G_{k-\hat{m}_k+i}\right], U_k, V_k, \epsilon_G, r_{\max})$.
  Set $\epsilon_G = \theta \rho_k$.
  if $\rho_k < \text{TOL}$ **then**
    Exit and return X_{k+1}.
  end if
end for

VIRGINIA TECH

Algorithm lrAA for nonlinear matrix equation $G(X) = X$.

Input: $X_0 = U_0 S_0 (V_0)^T$, memory parameter $\hat{m} \geq 1$, scheduling parameter $\theta \in (0,1)$, tolerance TOL.
Output: Approximate solution X_k to the fixed point problem $G(X) = X$ in its SVD form.

$\epsilon_G = 10^{-2}$ # Choose ϵ_G so that G_0 has low rank.

$X_1 = G_0 =$ **Cross-DEIM**$(G(X_0), U_0, V_0, \epsilon_G, r_{\max})$.

$\rho_0 = \|X_1 - G_0\|$.

for $k = 1, 2, \dots$ **do**

  $G_k =$ **Cross-DEIM**$(G(X_k), U_k, V_k, \epsilon_G, r_{\max})$.

  $\rho_k = \|G_k - X_k\|$.

  $\hat{m}_k = \min(\hat{m}, k)$.

  Solve least square problem to get γ^k

  $X_{k+1} =$ **Cross-DEIM**$\left(G_k - \sum_{i=0}^{\hat{m}_k - 1} \gamma_i^{(k)} \left[G_{k-\hat{m}_k+i+1} - G_{k-\hat{m}_k+i}\right], U_k, V_k, \epsilon_G, r_{\max}\right)$.

  Set $\epsilon_G = \theta \rho_k$.

  if $\rho_k <$ TOL **then**

    Exit and return X_{k+1}.

  end if

end for

Rank truncating operations
With warm start

Scheduling the truncation

VIRGINIA TECH

Outline:

- Motivation
- Low-rank solution to nonlinear matrix differential equations
- **Cross-DEIM**
- Numerical experiments

VIRGINIA TECH

Given a rank r matrix X of size $m \times n$, a new matrix G defined through $G_{ij} = G(X_{ij})$.

G may not be low-rank, but suppose it is, how do we get its SVD, without accessing all the entries in X

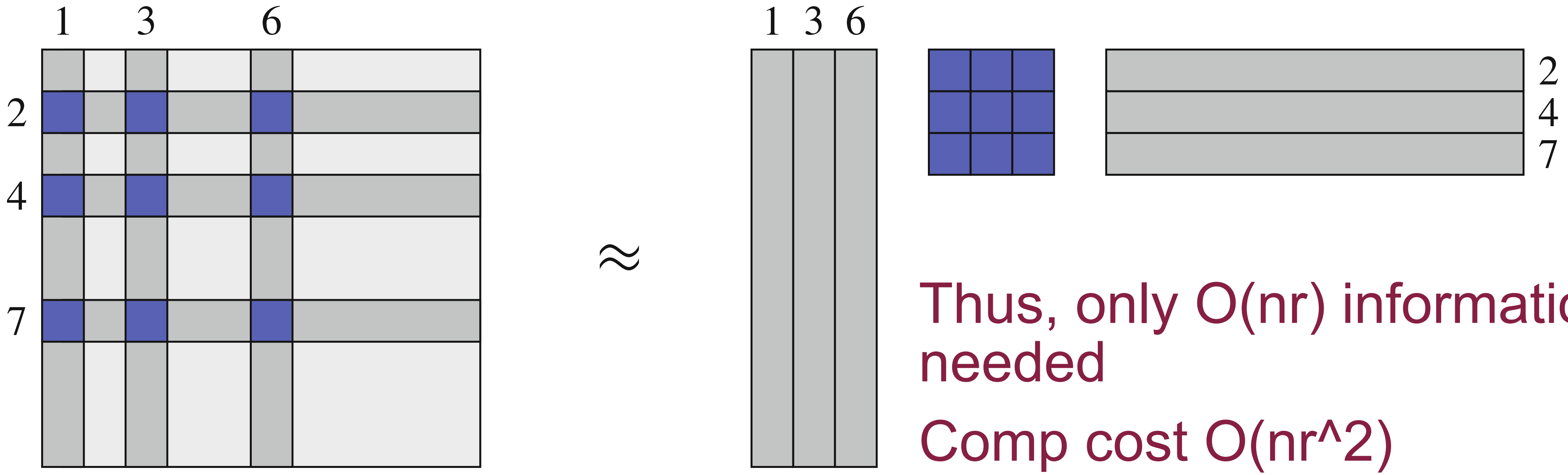
i.e. We would like sublinear algorithms to get truncated SVD of G

Cross approximation

With stabilization

$$G \approx G(:,\mathcal{J})G(\mathcal{I},\mathcal{J})^+G(\mathcal{I},:) = Q_1R_1G(\mathcal{I},\mathcal{J})^+R_2^TQ_2^T = USV^T$$

How to chose rows $\mathcal{I}$ and columns $\mathcal{J}$?



Thus, only O(nr) information is needed

Comp cost O(nr^2)

Figure from J. Ballani and D. Kressner Matrices with Hierarchical Low-Rank Structures

Index selection for cross approximation

- It was shown **maxvol** index selection is quasi-optimal
- We use **DEIM** (discrete empirical interpolation method) based selection
- DEIM is a well-known method in model reduction, and can be used for CUR matrix approximation
- It is based on singular vectors, and give better results than the leverage score based selection.

See *Chaturantabut, Sorenson, SISC, 2010, Sorenson, Embree, SISC, 2016.*

DEIM index selection

- Goal: To approximate matrix G
- Given U, V leading left and right singular vectors (size mxr, nxr)
- DEIM(U) -> row index set I, DEIM(V) -> column index set J

- Error bound exists and we use it to design stopping criteria (Donello et al 2023)

Chicken and egg problem: need singular vectors to get index

Fixing the ‘chicken and egg’ problem

- We can do an iteration (similar to maxvol iteration)
- Start with some index ->Cross -> SVD-> update singular vector->update index
- To have adaptive rank method (rather than fixed rank), we merge the old and new index. Prune at the end.
- We can **warm start** the iteration

$$X^{n+1} = G(X^n)$$

Singular vectors from previous iterate are close
Use as Warm Start

Donella et al 2023, Dektor 2024 use Warm start for time-dependent prob.

VIRGINIA TECH

Cross-DEIM

Algorithm `[U,S,V] = Cross-DEIM(G,U0,V0,`ϵ`)`
Adaptive Cross-DEIM approximation to $G \in \mathbb{R}^{m\times n}$

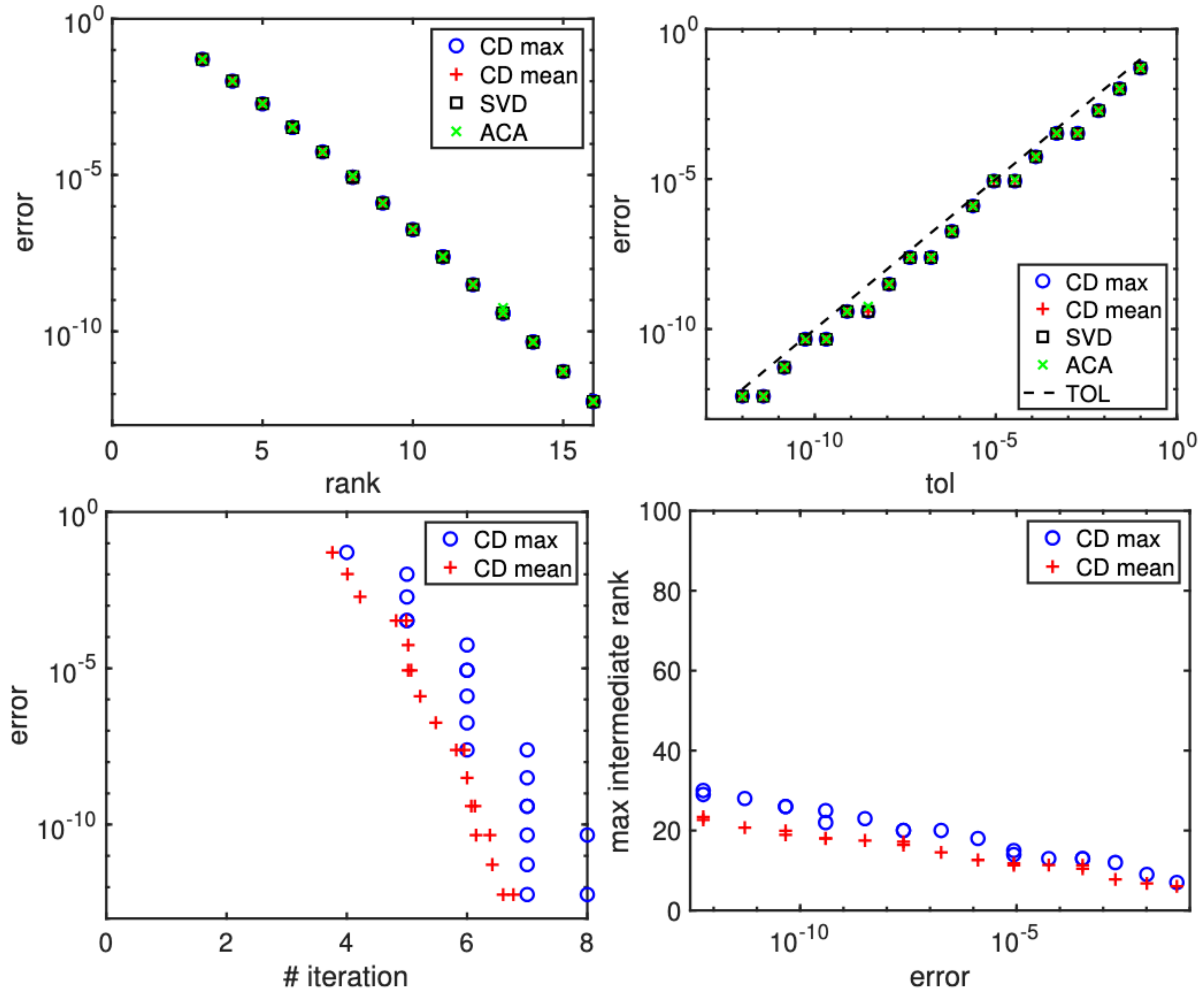
1: **Input:** Matrix $G \in \mathbb{R}^{m\times n}$, initial guess for the singular vector matrix $U_0 \in \mathbb{R}^{m\times r}$, $V_0 \in \mathbb{R}^{n\times r}$, tol. ϵ.
2: **Output:** Approximate SVD of G, $U \in \mathbb{R}^{m\times r}$, $S \in \mathbb{R}^{r\times r}$, $V \in \mathbb{R}^{r\times n}$.

- In general, we can start with random vector U_0, V_0
- In lrAA, we use U_n, V_n to warm start

Outline:
- Motivation
- Low-rank solution to nonlinear matrix differential equations
- Cross-DEIM
- **Numerical experiments**

Cross-DEIM

100X100 Hilbert matrix

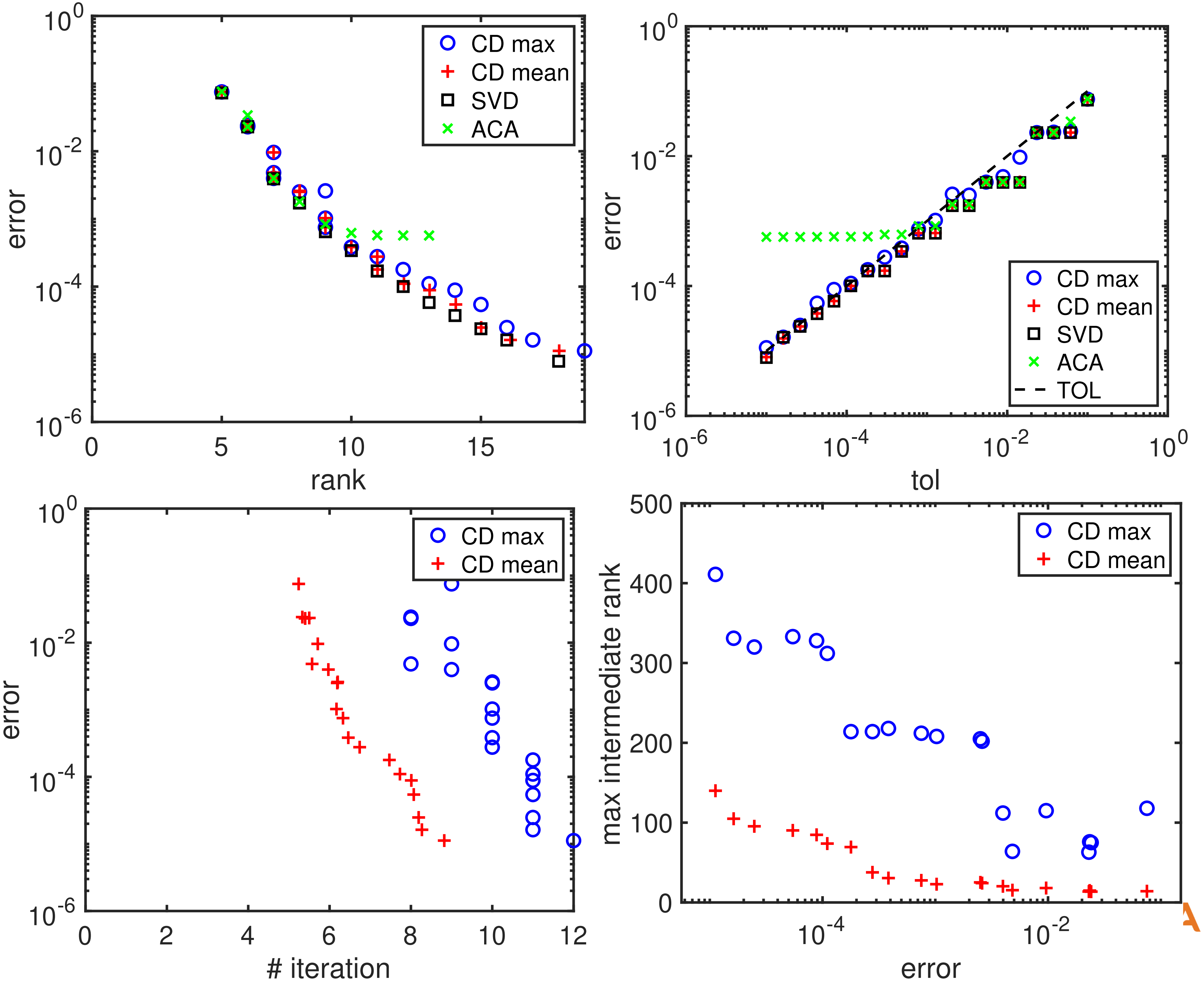


VIRGINIA TECH

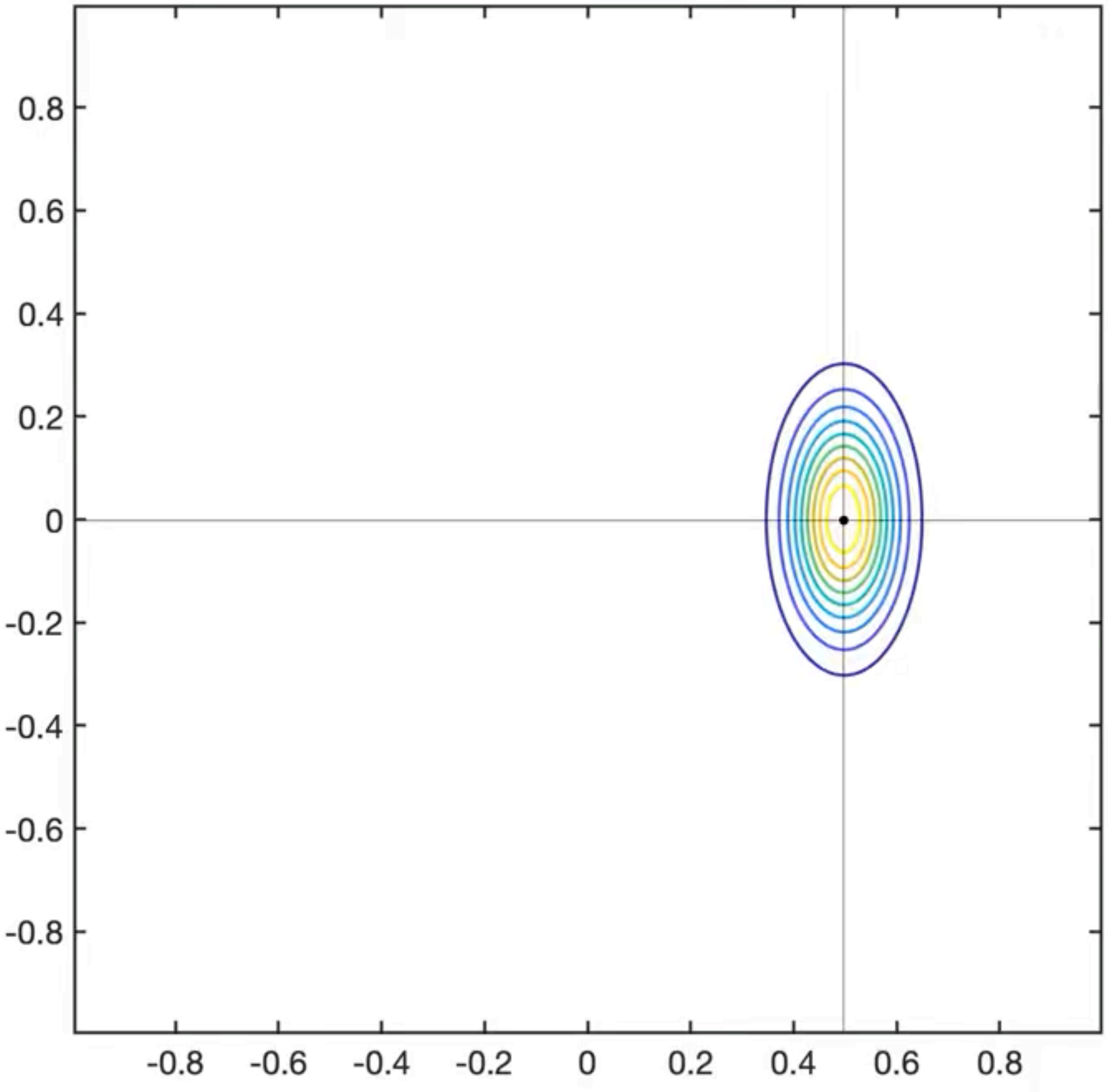
Cross-DEIM

$$G_2(i,j) = \left(\frac{|x_i + y_j|}{2}\right)^5$$

500X500

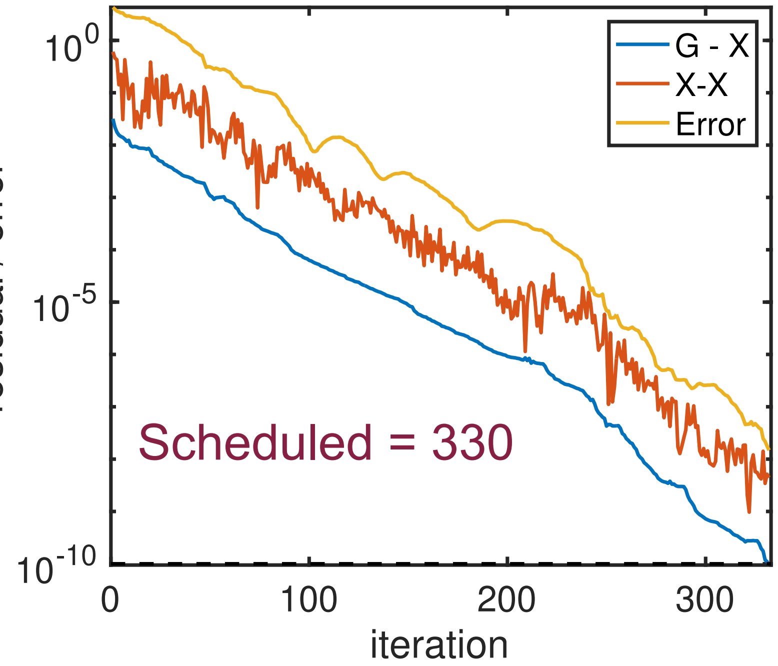


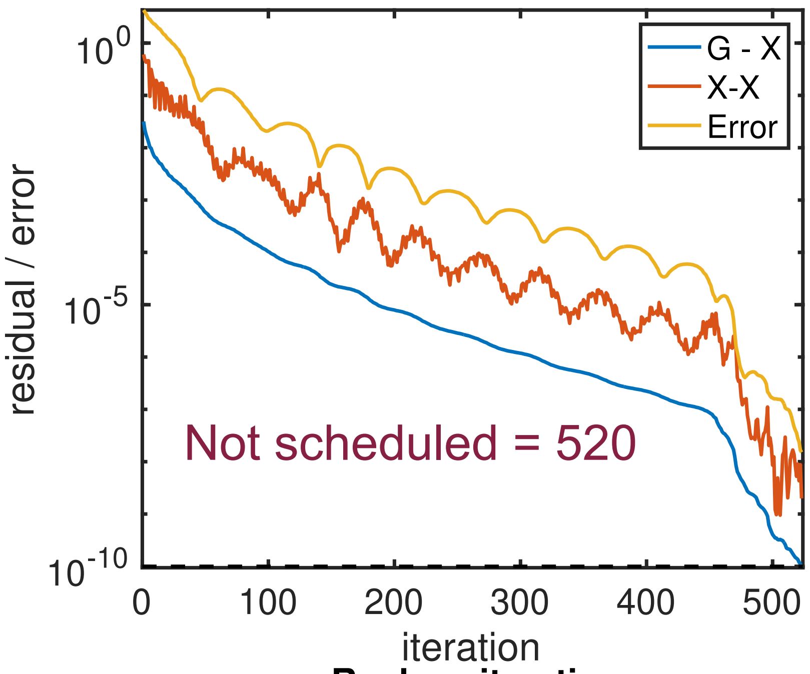
Cross-DEIM for parametric matrix approximation

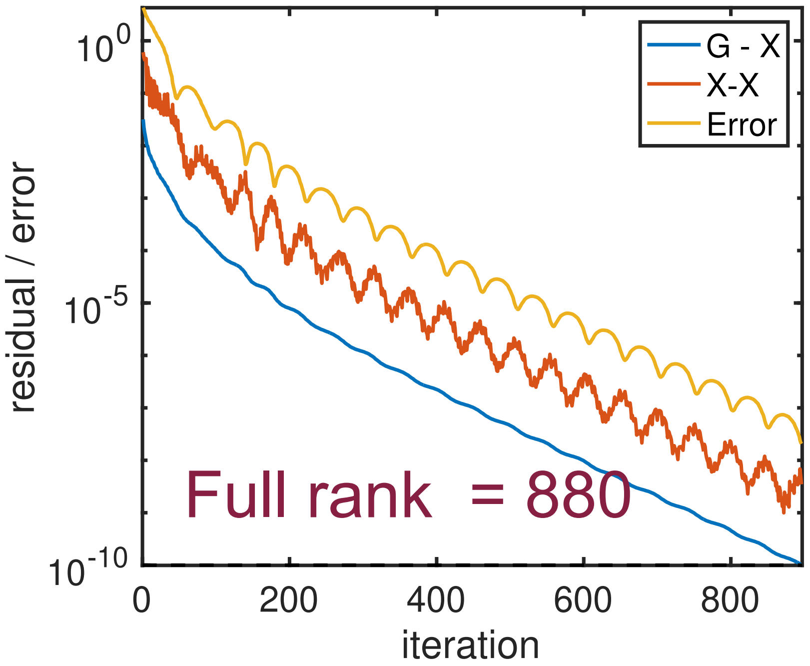


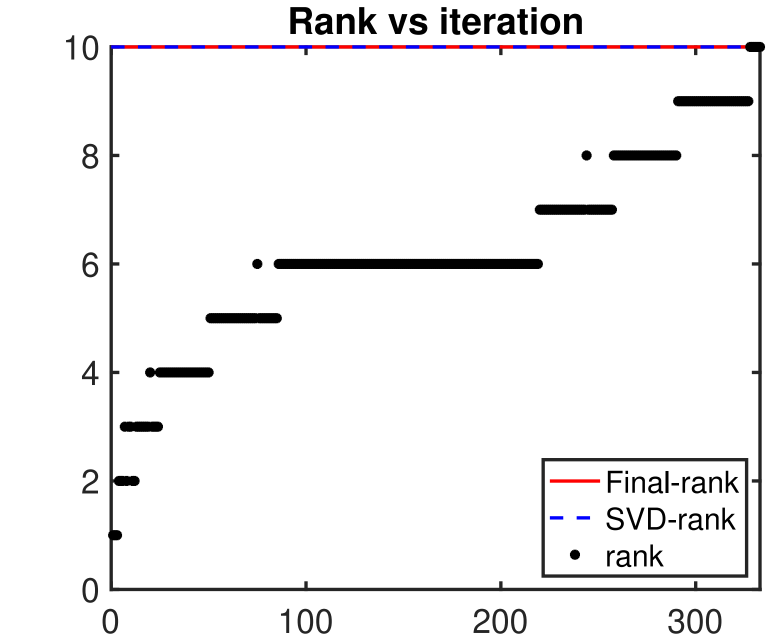
VIRGINIA TECH.

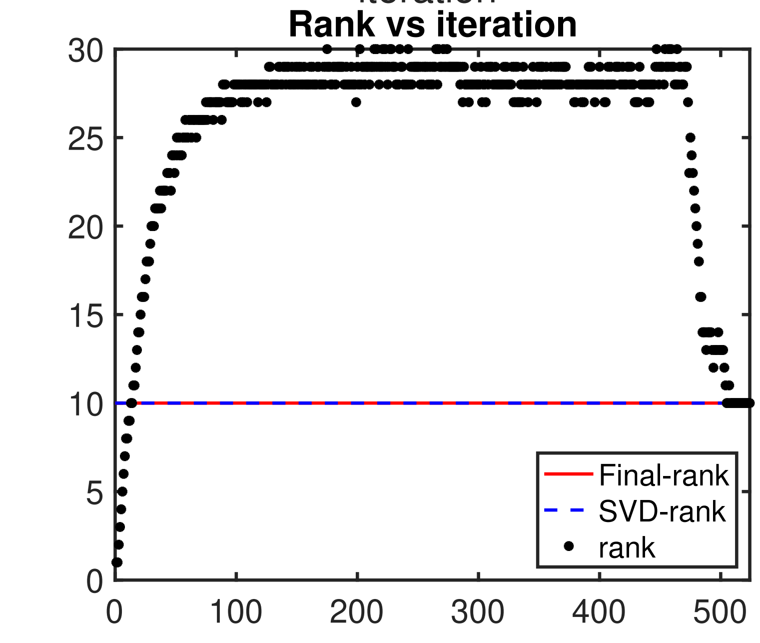
IrAA: Laplace's equation







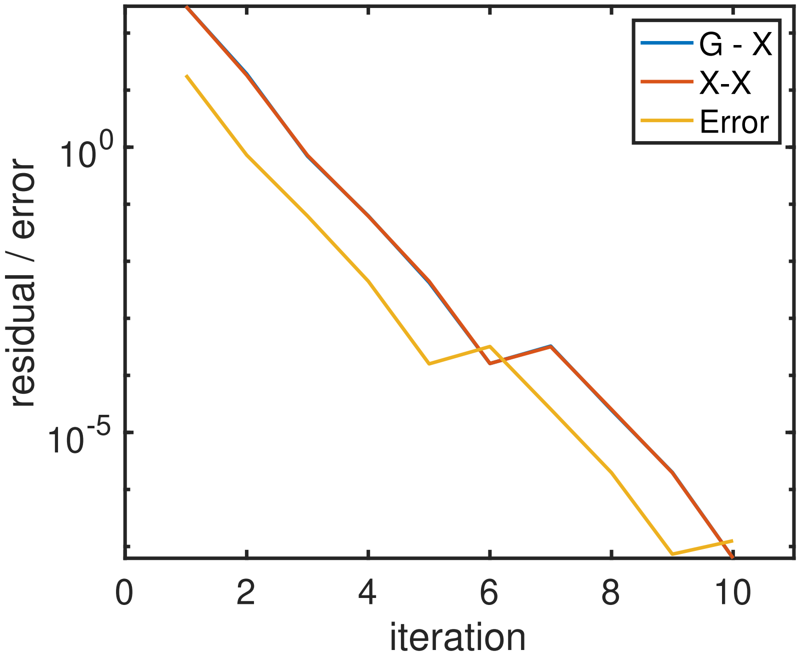


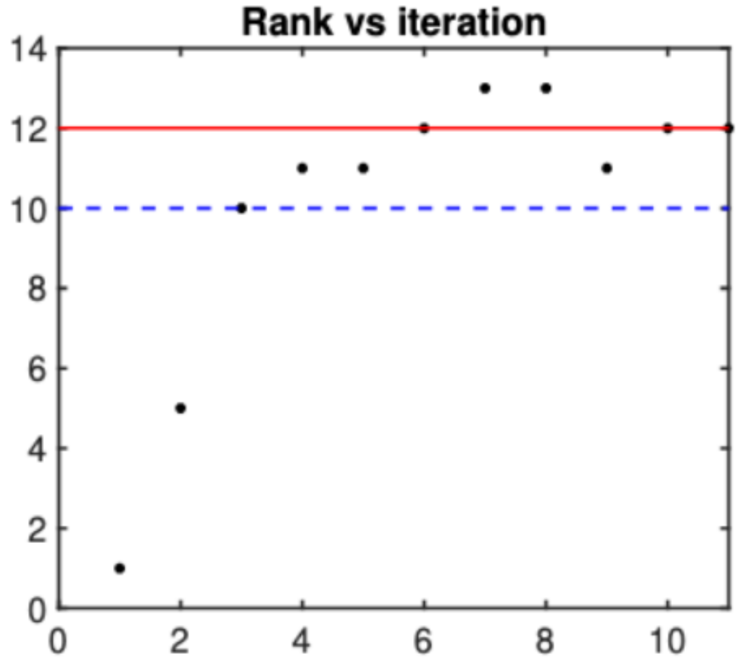


Scheduling keeps the rank controlled

often monotonic

Laplace's equation





$$M(G_\Delta(X^k) - F) = M(U_R S_R V_R^T)$$

$$= -\sum_{k=1}^{n_{\mathrm{ES}}} \alpha_k (e^{\beta_k D_{xx}} U_R) S_R (e^{\beta_k D_{yy}} V_R)^T$$

Exponential sum preconditioner

1024X1024, rank 12

D. Braess and W. Hackbusch, *Approximation of* $1/x$ *by exponential sums in* $[1, \infty)$

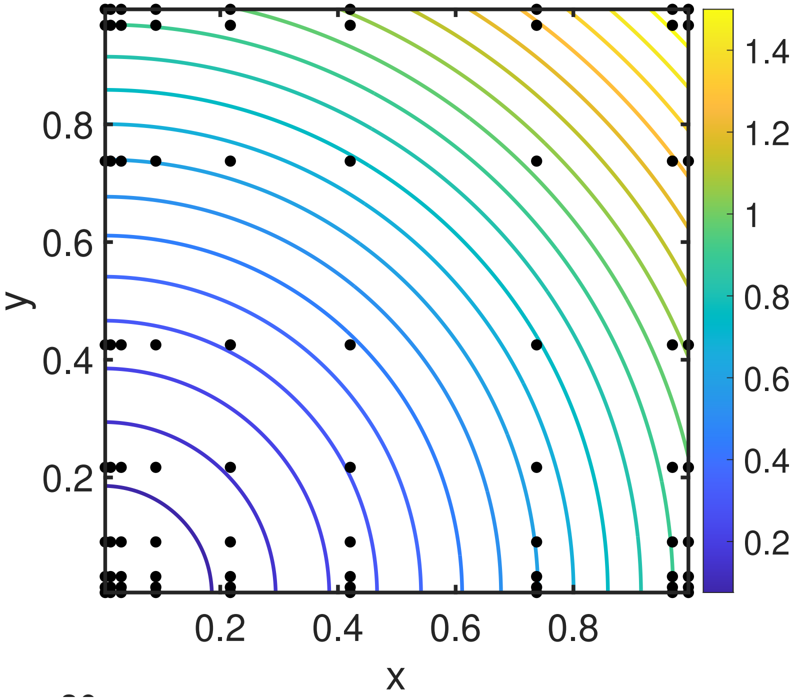
VIRGINIA TECH

Monge-Ampere

Discretized with “method 1” from [6]

$$\frac{\partial^2 u}{\partial x^2}\frac{\partial^2 u}{\partial y^2} - \left(\frac{\partial^2 u}{\partial x \partial y}\right)^2 = f(x,y),$$

$$u = \frac{2\sqrt{2}}{3}(x^2+y^2)^{\frac{3}{4}}$$



$n=m$	lrAA iterations	iterations reported in[6]	final rank
21	109 (7)	1083	13 (4)
61	287 (8)	8967	18 (7)
101	443 (13)	23849	20 (7)
221	675 (11)	107388	26 (8)

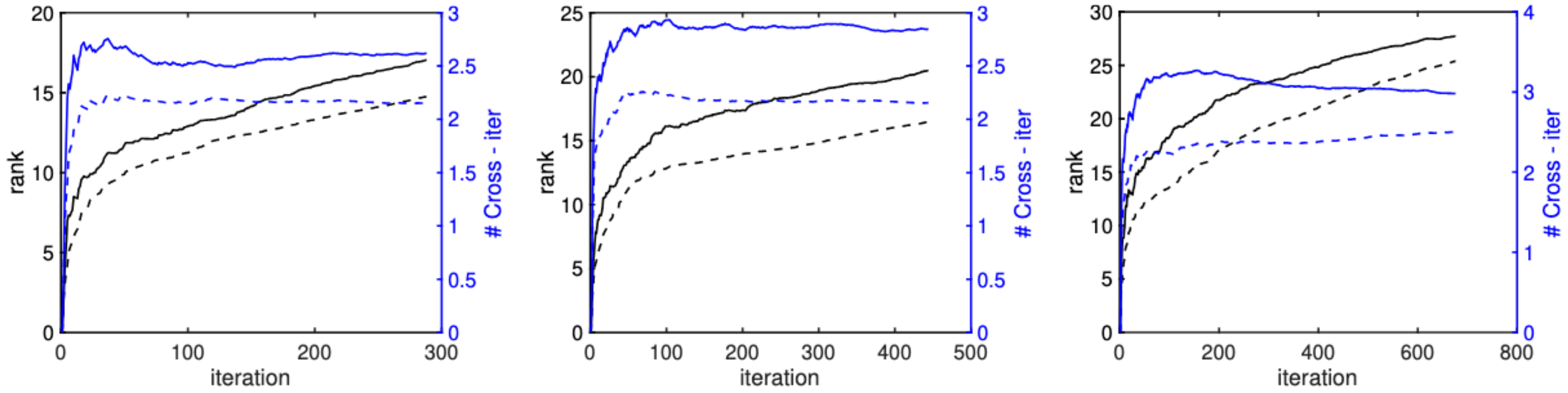
Again the scheduling gives much lower number of iterations.

It is highly beneficial to incorporate local truncation error in rounding / truncation

[6] Benamou, Jean-David, Brittany D. Froese, and Adam M. Oberman. "Two Numerical Methods for the elliptic Monge-Ampère equation." *ESAIM: Mathematical Modelling and Numerical Analysis* 44, no. 4 (2010): 737-758.

VIRGINIA TECH

Monge-Ampere: performance of Cross-DEIM

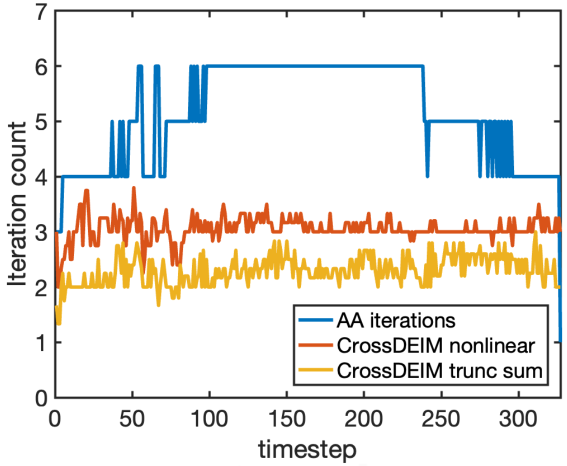


mesh: 61x61, 101x101, 221x221

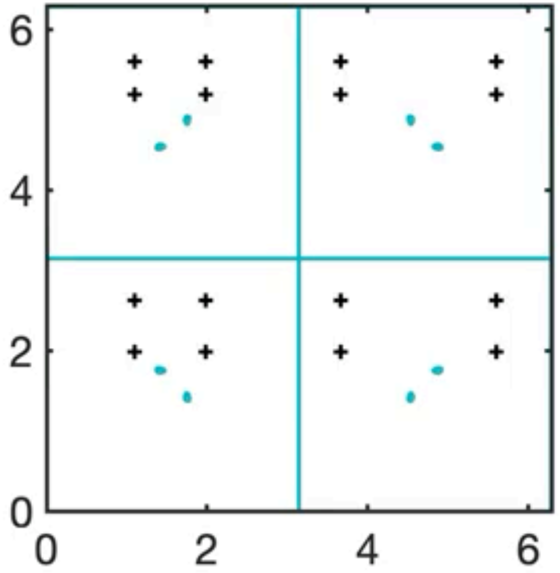
Allen-Cahn

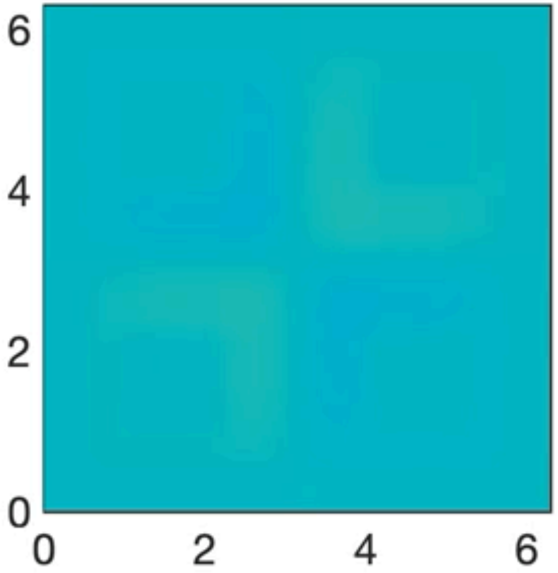
Here we solve Allen-Cahn $u_t = \frac{1}{100}\Delta u + u - u^3$, $\qquad X(i,j) \approx u(x_i, y_j)$.
We use a 1024×1024 mesh.

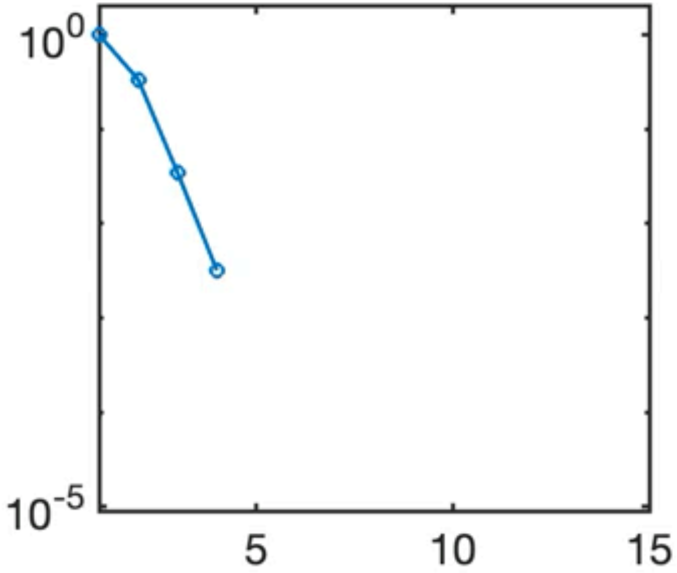
$$u(x,y) = \frac{\left[e^{-\tan^2(x)} + e^{-\tan^2(y)}\right]\sin(x)\sin(y)}{1 + e^{|\csc(-x/2)|} + e^{|\csc(-y/2)|}}.$$



Exponential sum preconditioner







Summary:

- lrAA (low-rank Anderson acceleration): a new approach for computing low-rank solution to nonlinear problem.
- Cross-DEIM: adaptive iterative cross approximation with a warm-start strategy.

Future work:

- Generalizing to tensor.
- Application.
- Improve and analyze the method.

Adaptive Cross DEIM approximation to $G \in \mathbb{R}^{m \times n}$

Input: Matrix $G \in \mathbb{R}^{m\times n}$, initial rank r guess to the singular vector matrix $U_0 \in \mathbb{R}^{m\times r}, V_0 \in \mathbb{R}^{n\times r}$, tolerance ϵ, maximum output rank $r_{\max}$, maximum index set cardinality $\aleph_{\max}$, maximum number of iterations `maxiter`.
Output: Approximate SVD of G, $U \in \mathbb{R}^{m\times r}$, $S \in \mathbb{R}^{r\times r}$, $V \in \mathbb{R}^{r\times n}$.
Set $\mathcal{I}_0 = \mathcal{J}_0 = \emptyset$.
for $k = 1, 2, \ldots,$ `maxiter` **do**
  $\mathcal{I}_k^* =$ `QDEIM`(U_{k-1})
  $\mathcal{J}_k^* =$ `QDEIM`(V_{k-1}) # `QDEIM` can be replaced by `DEIM`
  $\mathcal{I}_k = \mathcal{I}_k^* \cup \mathcal{I}_{k-1}, \mathcal{J}_k \leftarrow \mathcal{J}_k^* \cup \mathcal{J}_{k-1}$ # Note that the index sets are ordered by `QDEIM`.
  if $|\mathcal{I}_k| = |\mathcal{I}_{k-1}|$ or $k = 1$ **then**   # Make sure that that the index set increase by one
    $\mathcal{I}_k = \mathcal{I}_k^* \cup \{i_{\text{rand}} \in \complement(\mathcal{I}_k^*)\}$ # using a random i_{rand} from the complement of $\mathcal{I}_k^*$.
  end if
  if $|\mathcal{J}_k| = |\mathcal{J}_{k-1}|$ or $k = 1$ **then**
    $\mathcal{J}_k \leftarrow \mathcal{J}_k^* \cup \{j_{\text{rand}} \in \complement(\mathcal{J}_k^*)\}$
  end if
  if $|\mathcal{I}_k| > \aleph_{\max}$ **then**
    $\mathcal{I}_k \leftarrow \mathcal{I}_k(1 : \aleph_{\max})$ # Keep the $\aleph_{\max}$ most important indices.
  end if
  if $|\mathcal{J}_k| > \aleph_{\max}$ **then**
    $\mathcal{J}_k \leftarrow \mathcal{J}_k(1 : \aleph_{\max})$
  end if
  $[U_k, S_k, V_k, r_{\mathrm{C}}, r_{\mathrm{R}}] =$ `scross`$(G, \mathcal{I}_k, \mathcal{J}_k)$
  for $l = 1, 2, \ldots, |\mathcal{I}_k|$ **do**
    if $|(r_{\mathrm{R}})_l| < 10^{-12}$ **then**
      Remove element l from $\mathcal{I}_k$ # Remove redundant rows in $R = G(\mathcal{I}_k, :)$.
    end if
  end for
  for $l = 1, 2, \ldots, |\mathcal{I}_k|$ **do**
    if $|(r_{\mathrm{C}})_l| < 10^{-12}$ **then**
      Remove element l from $\mathcal{J}_k$ # Remove redundant columns in $C = G(:, \mathcal{J}_k)$.
    end if
  end for
  $\rho = \|U_k S_k V_k^T - U_{k-1} S_{k-1} V_{k-1}^T\|, \quad S_{\min} = \min(\mathsf{diag}(S_k))$
  $\eta_1 = \|(I(:, \mathcal{I}_k))^T U_k\|_2^{-1}, \quad \eta_2 = \|V_k^T I(\mathcal{J}_k, :)\|_2^{-1}$
  if $\max(\rho, \min(\eta_1(1+\eta_2), \eta_2(1+\eta_1))S_{\min}) < \epsilon$ **then**
    Break out of for loop   # Above $S_{\min}$ is the smallest s.v. in the kth approx.
  end if
end for
Find r^* so that $\sum_{l=r^*+1}^{\min(m,n)} S_l^2 < \epsilon^2$
Set $r = \max(\min(r^*, r_{\max}), 1)$
Return $U_k(:, 1:r), S_k(1:r, 1:r), V_k(:, 1:r)$