A Sampling-based Adaptive-Rank Method for Nonlinear Kinetic Models

Jingmei Qiu

Department of Mathematical Sciences
University of Delaware, USA.

Supported by: AFOSR (MURI: Tensor Network) and DOE (CHARMNET)
Collaborators: A. Christlieb (MSU), D. Hayes, W. Sands, N. Zheng (UDel)

Advances in Mathematical Developments for Computational Plasma Physics
Enabling Fusion Relevant Applications

Multi-Fidelity Methods for Fusion Energy,
IPAM, April 2026.

Kinetic theory

Plasma dynamics may be described by the kinetic model

$$\frac{\partial f_s}{\partial t} + \mathbf{v} \cdot \nabla_{\mathbf{x}} f_s + \frac{q_s}{m_s}(\mathbf{E} + \mathbf{v} \times \boldsymbol{B}) \cdot \nabla_{\mathbf{v}} f_s = C(f). \qquad (1)$$

with the electromagnetic fields $\mathbf{E}$ and $\boldsymbol{B}$ satisfying the Maxwell equations,

$$\frac{\partial \boldsymbol{B}}{\partial t} = -\nabla \times \mathbf{E}, \quad \frac{\partial \mathbf{E}}{\partial t} = \nabla \times \boldsymbol{B} - \mathbf{J},$$
$$\nabla \cdot \mathbf{E} = \rho, \quad \nabla \cdot \boldsymbol{B} = 0,$$

with the charge density $\rho = \sum_s q_s \int f_s d\mathbf{v}$ the current density $\mathbf{J} = \sum_s q_s \int f_s \mathbf{v} d\mathbf{v}$.

- $f_s(t, v, x)$ is the distribution function of particle of species s.
- 6D + time nonlinear dynamical systems.

Curse of dimensionality

Sparse grid and ROM

- Sparse grid methods, *Smolyak (63'), Zenger (91'), Griebel (98'), ...*
- Reduced order modeling, *Benner, S. Gugercin, and K. Willcox (15'), ...*

Neural Network and Tensor Network

- Neural Network: *Han, Jentzen, E (18'), Raissi, Perdikaris, Karniadakis (19'), ...*
- Tensor Network: factorization of high order tensors, into a hierarchical decomposition of smaller tensors that can be characterized by tensor networks (typically a tree topology).
  - Tucker, Hierarchical Tucker and tensor train.

    Kolda, Bader 09', Hackbusch 09', Grasedyck 10', Oseledets 11', Grasedyck, Kressner and Tobler 13', Bigoni, Engsig-Karup, Marzouk 16', Tang, Ying, et. al. 24', ...
  - Parametric and Stochastic Elliptic PDEs.

    Khoromskij and Schwab 11', Kazeev, Reichmann, Schwab 13', ...
  - Time-dependent Matrix/Tensor differential equation

    Koch, Lubich 07', Lubich, Oseledets 14', Dektor, Venturi 20', ...
  - Kinetic models

    Kormann 15', Einkemmer, Lubich 18', Einkemmer and Joseph 21', ...

Tensor evolution of time-dependent (TD) solutions

- Dynamical Low Rank (DLR) approach: derive a set of differential equations for the low-rank factors by projecting the update onto the tangent space.
- Step and Truncate (SAT) approach: low rank truncation of a full rank high order solver.
  - Eulerian adaptive rank tensor integrations via low rank approximations to RHS terms (linear or nonlinear) and HOSVD truncations
- Sampling-based Adaptive Rank (SAR) approach:
  - Nonlinear Vlasov: semi-Lagrangian adaptive rank (SLAR) approach
  - Multi-scale BGK model: SLAR + adaptive cross approximation from function handles

DLR :	spatial discret. $\rightarrow$ low rank projection $\rightarrow$ evolve
SAT (Galerkin) :	spatial + temporal discret. $\rightarrow$ adaptive rank projection
SAR (Collocation) :	spatial + temporal discret. $\rightarrow$ adaptive rank sampling

Galerkin vs. Collocation

	Galerkin	Collocation
Traditional full rank	▶ Given bases functions ▶ find TD coefficients	▶ Given collocation points ▶ find solution point values on a tensor product of grids
Adaptive rank	▶ Adaptively find TD bases functions ▶ Then find TD coefficients on a reduced system	▶ Adaptively find important columns and rows as collocation points (sampling based) ▶ Find solution values on selected rows and columns; for the rest, can be obtained by matrix interpolation
Challenges	▶ Nonlinear functional of tensor solutions	▶ Effective, robust and accurate tensor fiber search and decomposition tools are needed.*

*Our effort: An Efficient and Robust Projection Enhanced Interpolation Based Tensor Train Decomposition, Hayes, Q. and Shi, arXiv 2602.07653.

Outline

- A Semi-Lagrangian Adaptive-Rank (SLAR) Method: Vlasov-Poisson system
  - linear transport:
    - local SL solver
    - adaptive cross approximation with greedy search
    - SVD truncation
  - Nonlinear VP system:
    - Nonlinear characteristics tracing
    - LoMaC correction with preservation of macroscopic conservation laws
- Tensor network approximation of high D Vlasov solutions
- Summary and future

A Semi-Lagrangian Adaptive-Rank (SLAR) Approach for the nonlinear Vlasov-Poisson system

A full rank semi-Lagrangian finite difference scheme

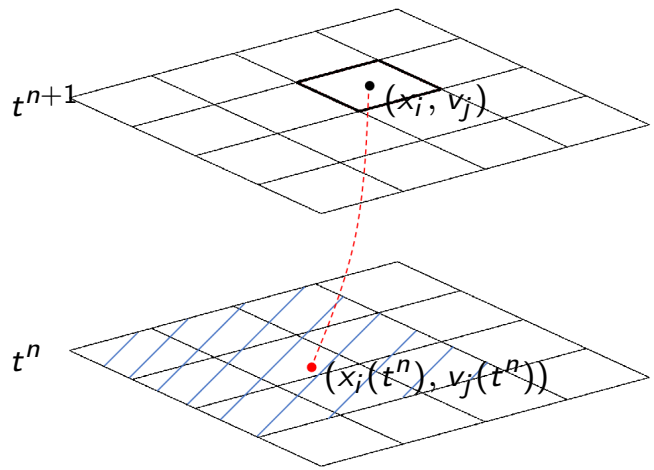
A linear advection equation:

$$f_t + a(x, y, t) f_x + b(x, y, t) f_y = 0, \quad x \in [x_L, x_R], \quad y \in [y_B, y_T] \quad (2)$$

SL scheme traces characteristics backward in time: for all (x_i, y_j)

- Track characteristics backward in time to (x_i^*, v_j^*)
- Identify the neighborhood for reconstruction (i^*, j^*)
- Perform a local polynomial reconstruction based on a 9-point stencil in a least square WENO fashion

$$f(x_i, v_j, t^{n+1}) = f(x_{ij}^*, v_{ij}^*, t^n)$$

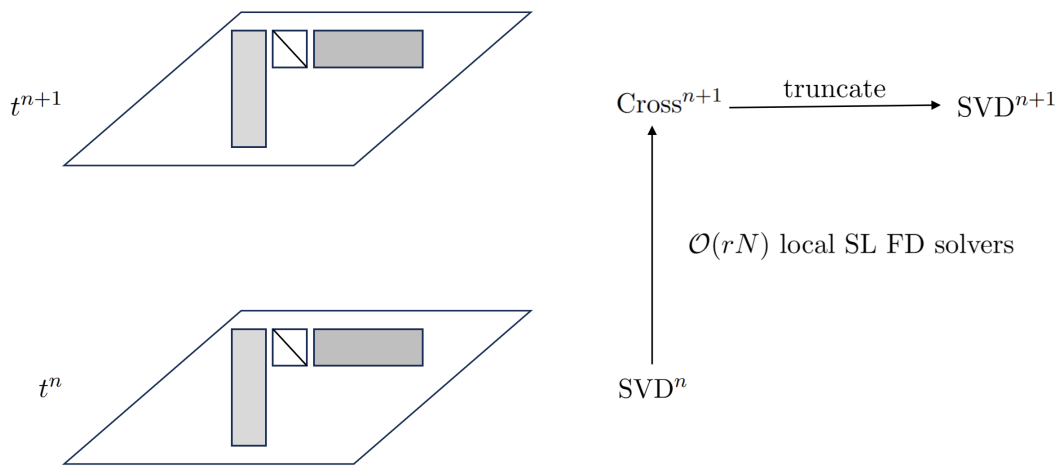


Computational complexity: $\mathcal{O}(N^d)$ per time step.

Semi-Lagrangian Adaptive Rank (SLAR) approach

1. Adaptive rank representation of the solution at t^n via SVD: $U\Sigma V^T$.
2. (Over)sampling via adaptive cross approximation of the solution matrix: via a semi-Lagrangian finite difference update.
3. An SVD truncation for numerical stability.

Computational complexity: $\mathcal{O}(dNr)$ per time step.



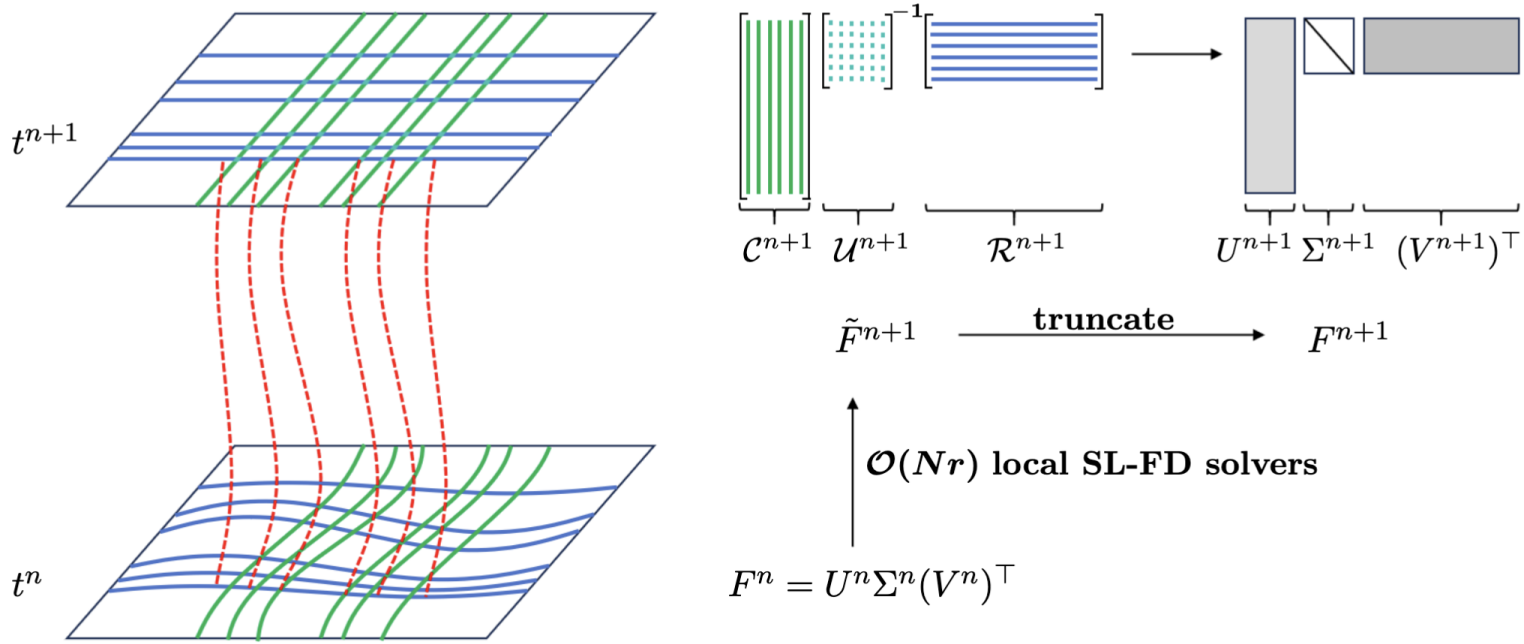
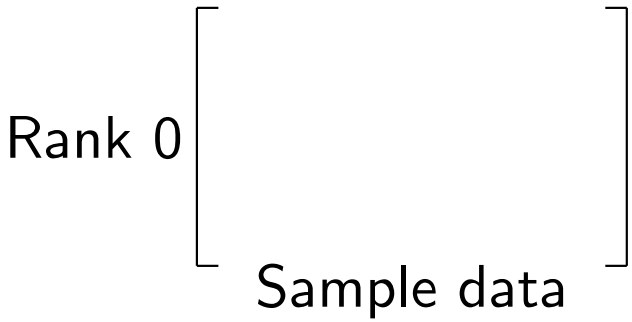
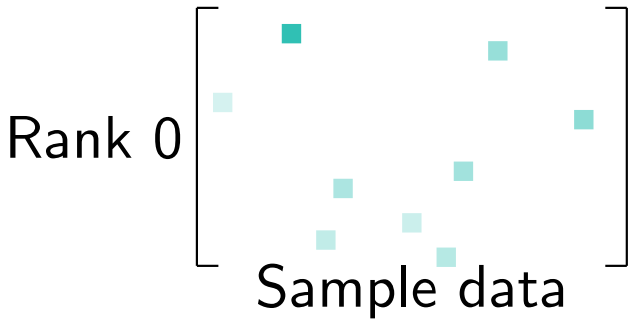


Figure 1.1: The SLAR method for linear advection equations.

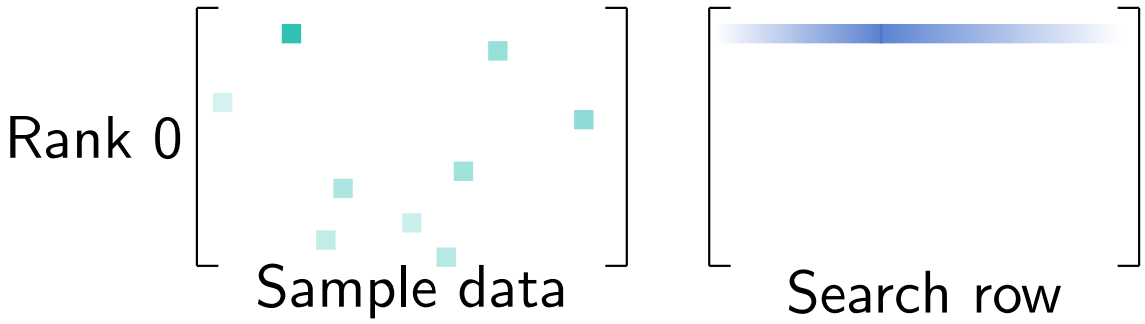
Greedy ACA for row and column identification



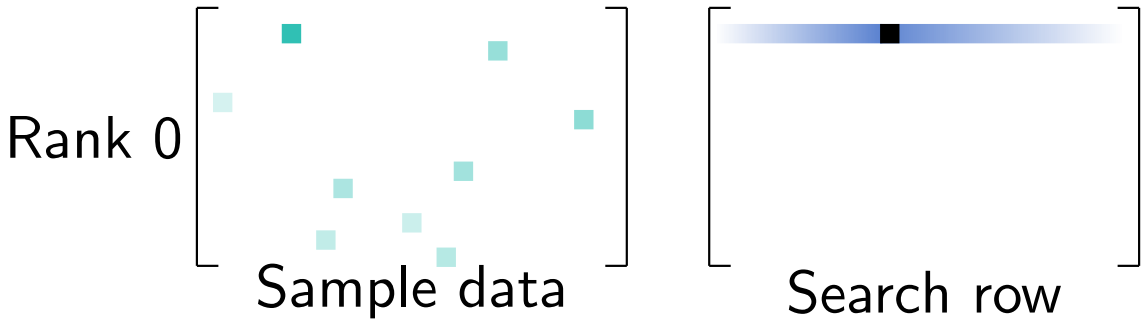
Greedy ACA for row and column identification



Greedy ACA for row and column identification



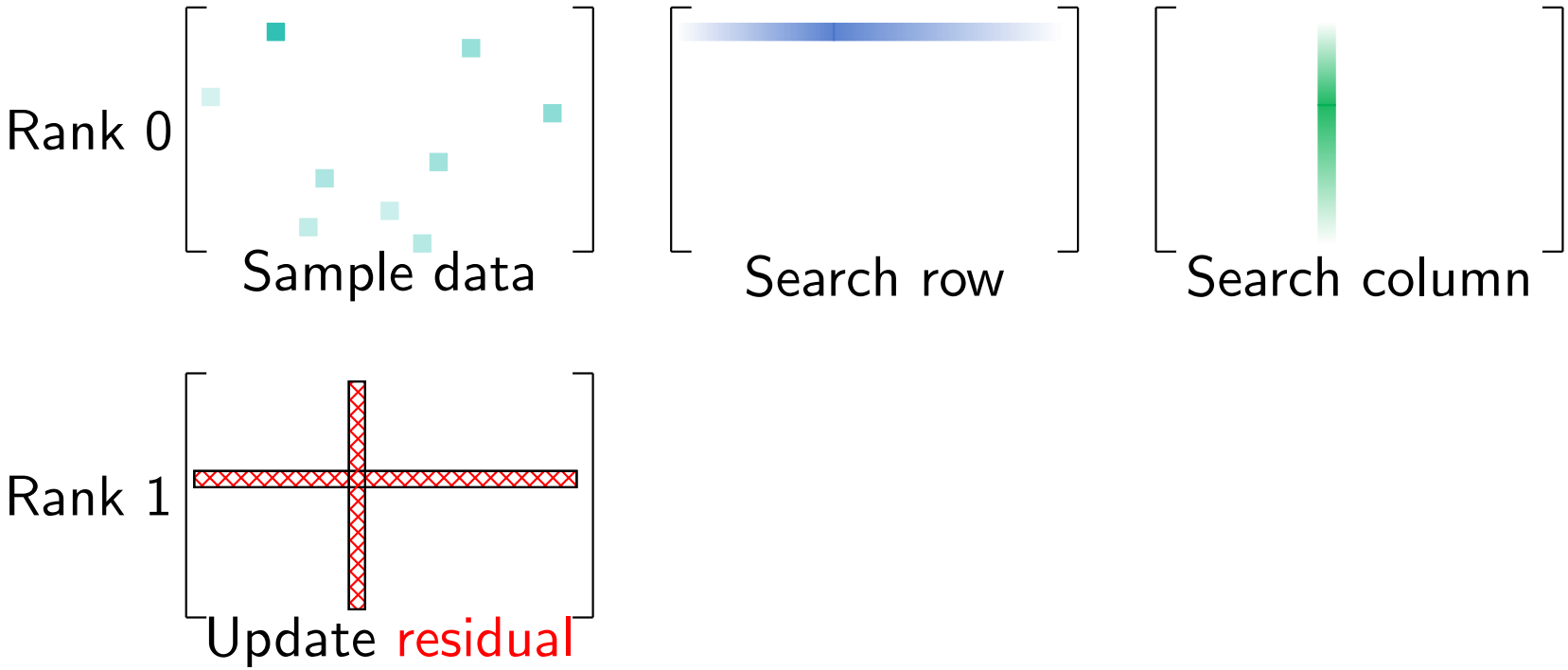
Greedy ACA for row and column identification



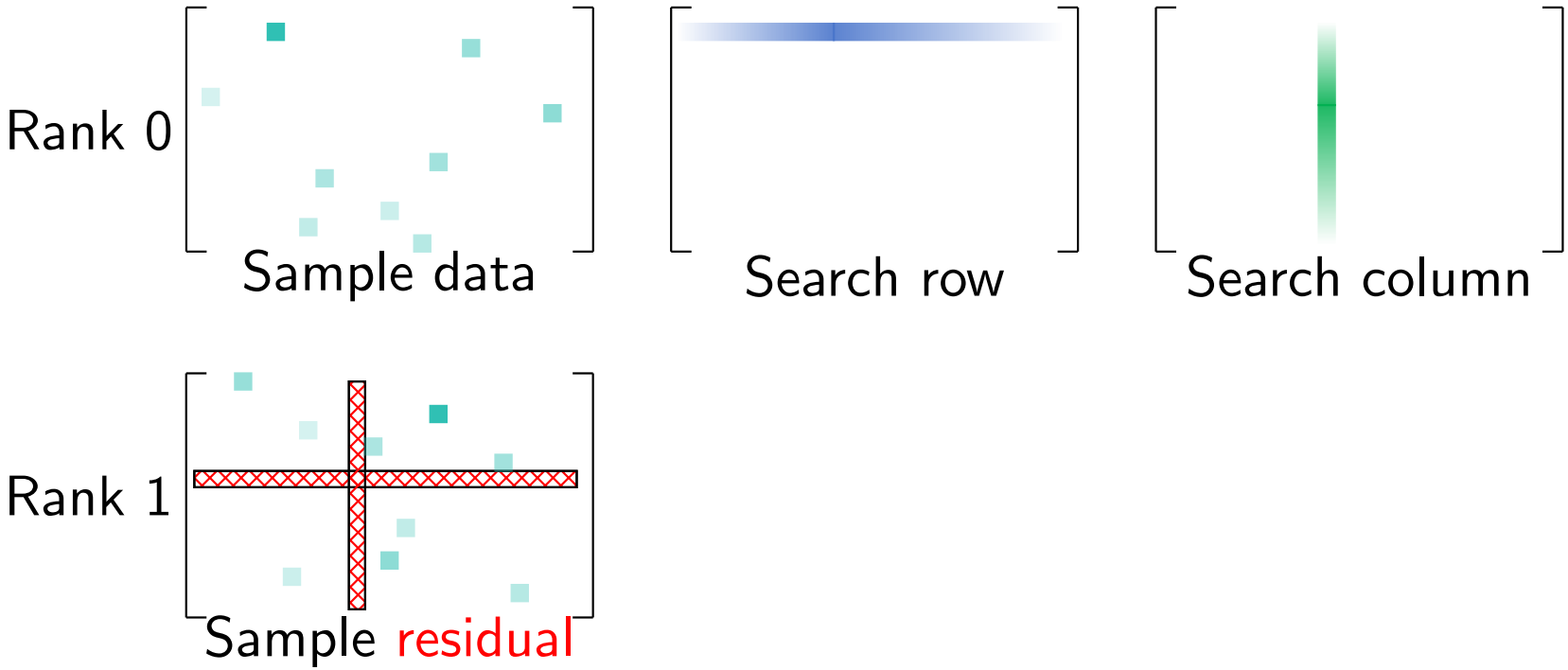
Greedy ACA for row and column identification

Greedy ACA for row and column identification

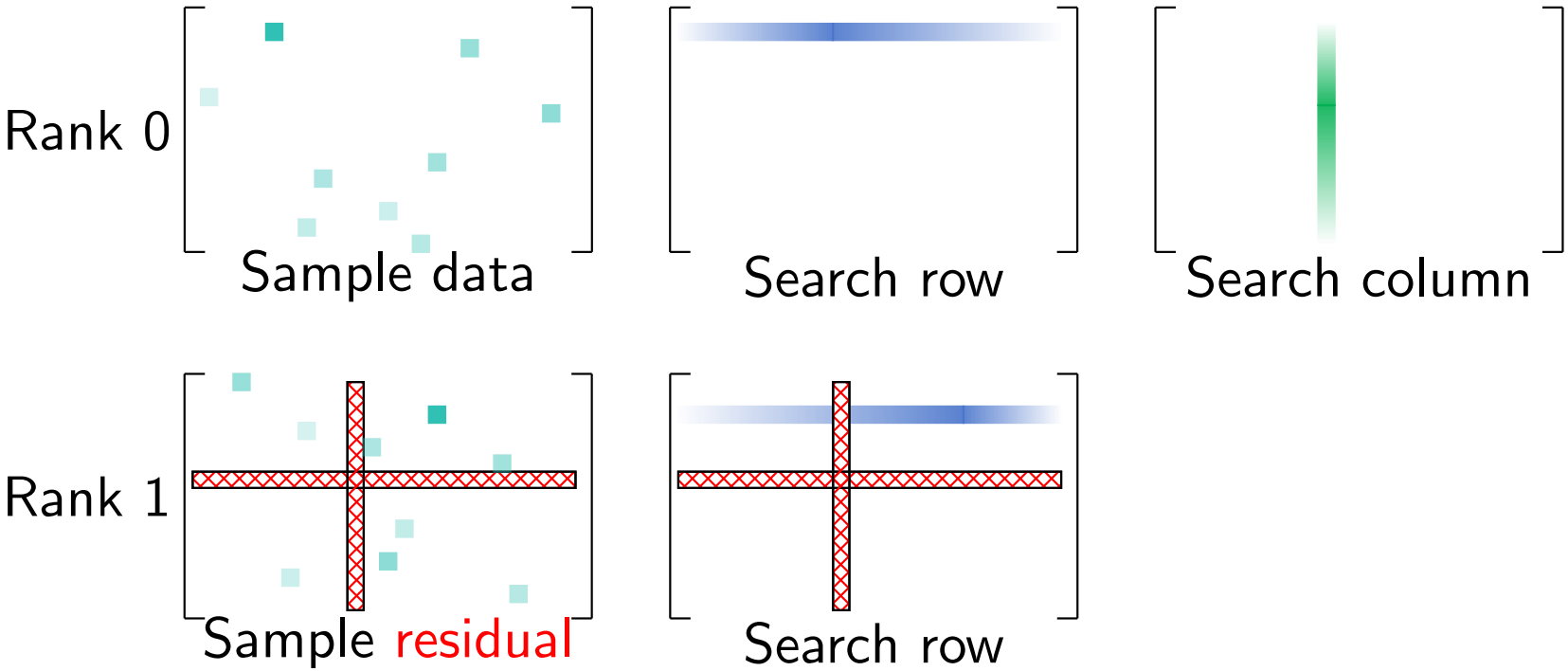
Greedy ACA for row and column identification



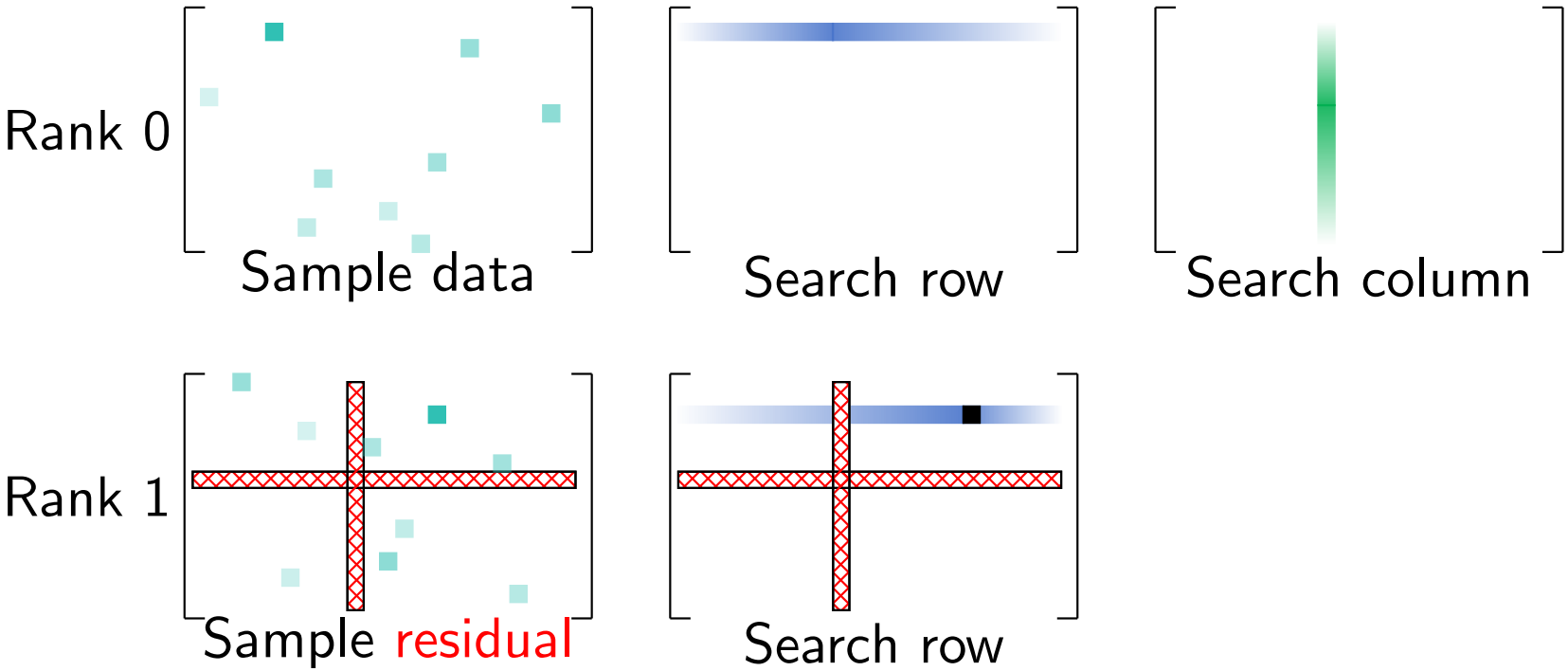
Greedy ACA for row and column identification



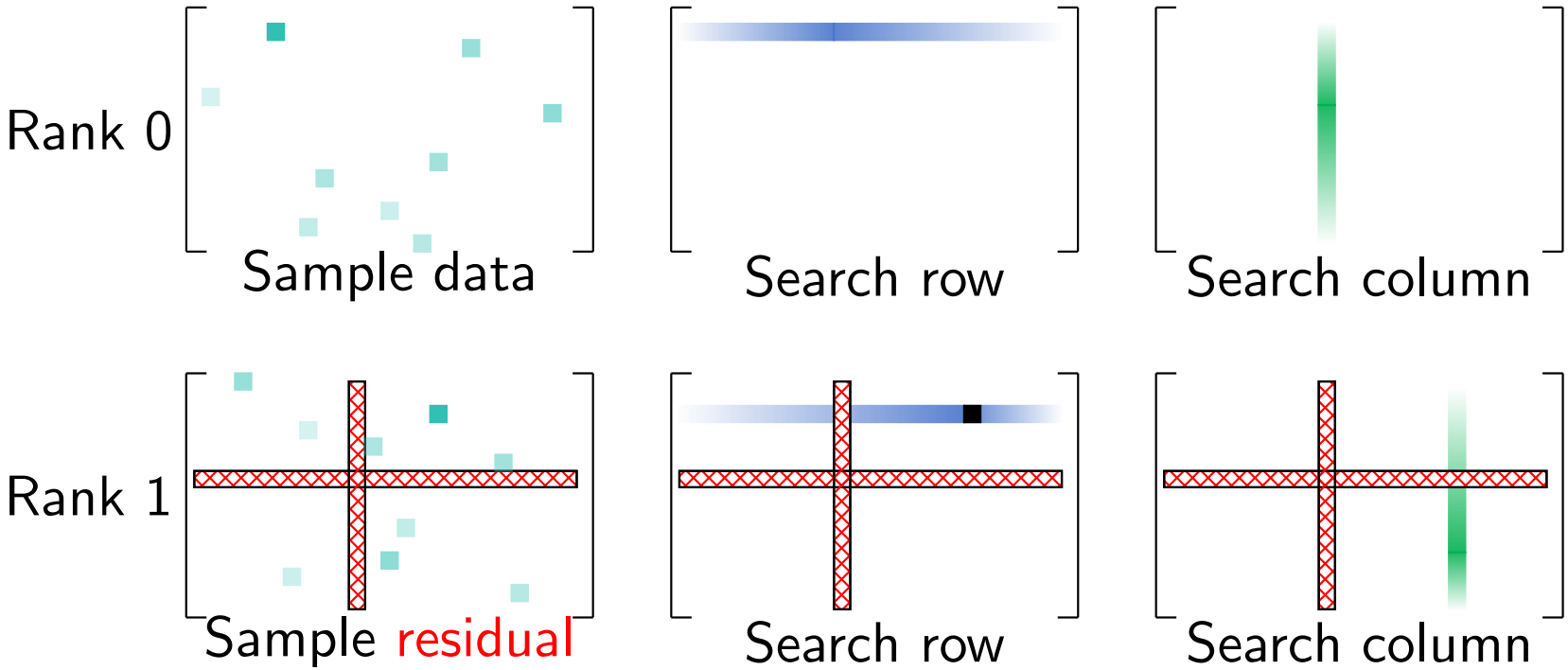
Greedy ACA for row and column identification



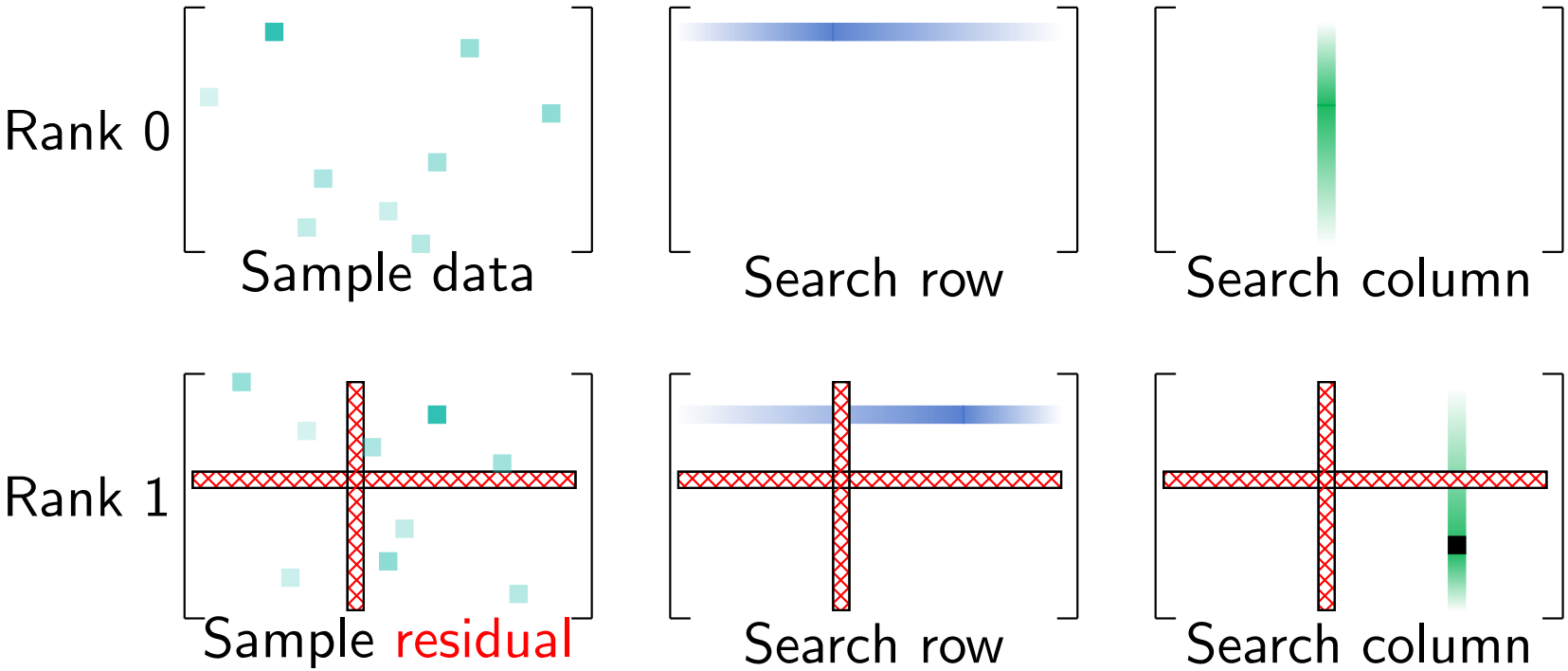
Greedy ACA for row and column identification



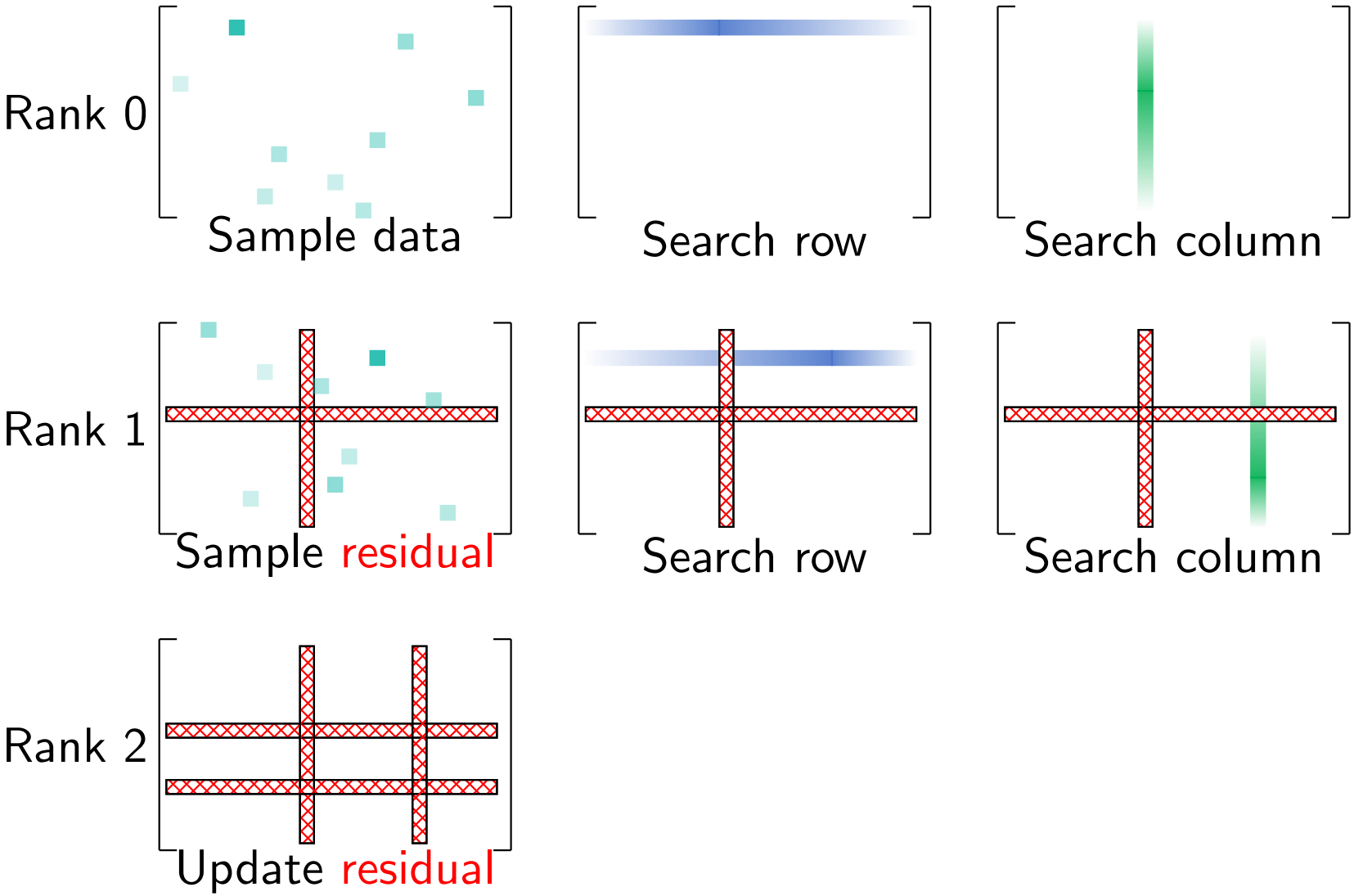
Greedy ACA for row and column identification



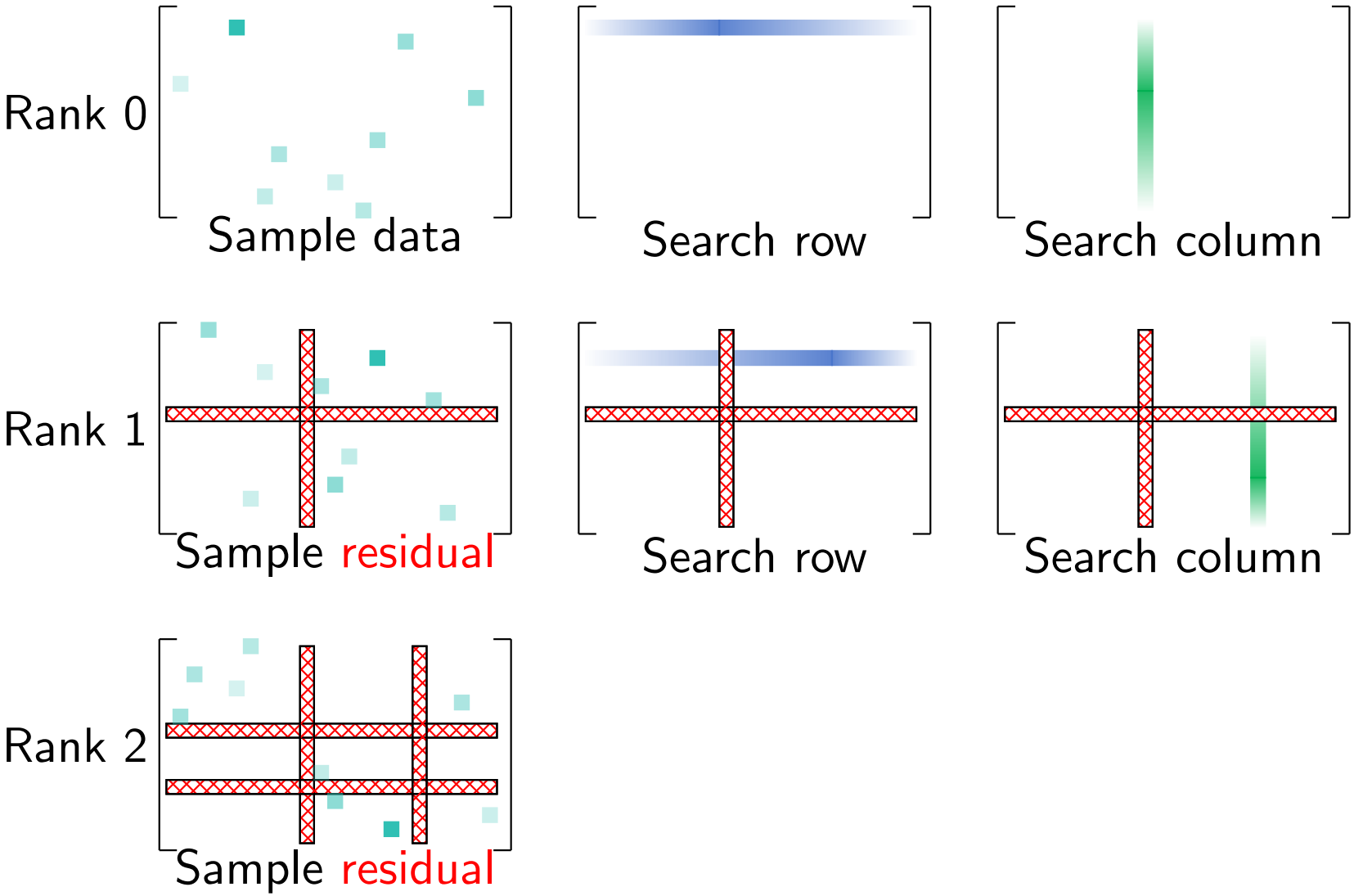
Greedy ACA for row and column identification



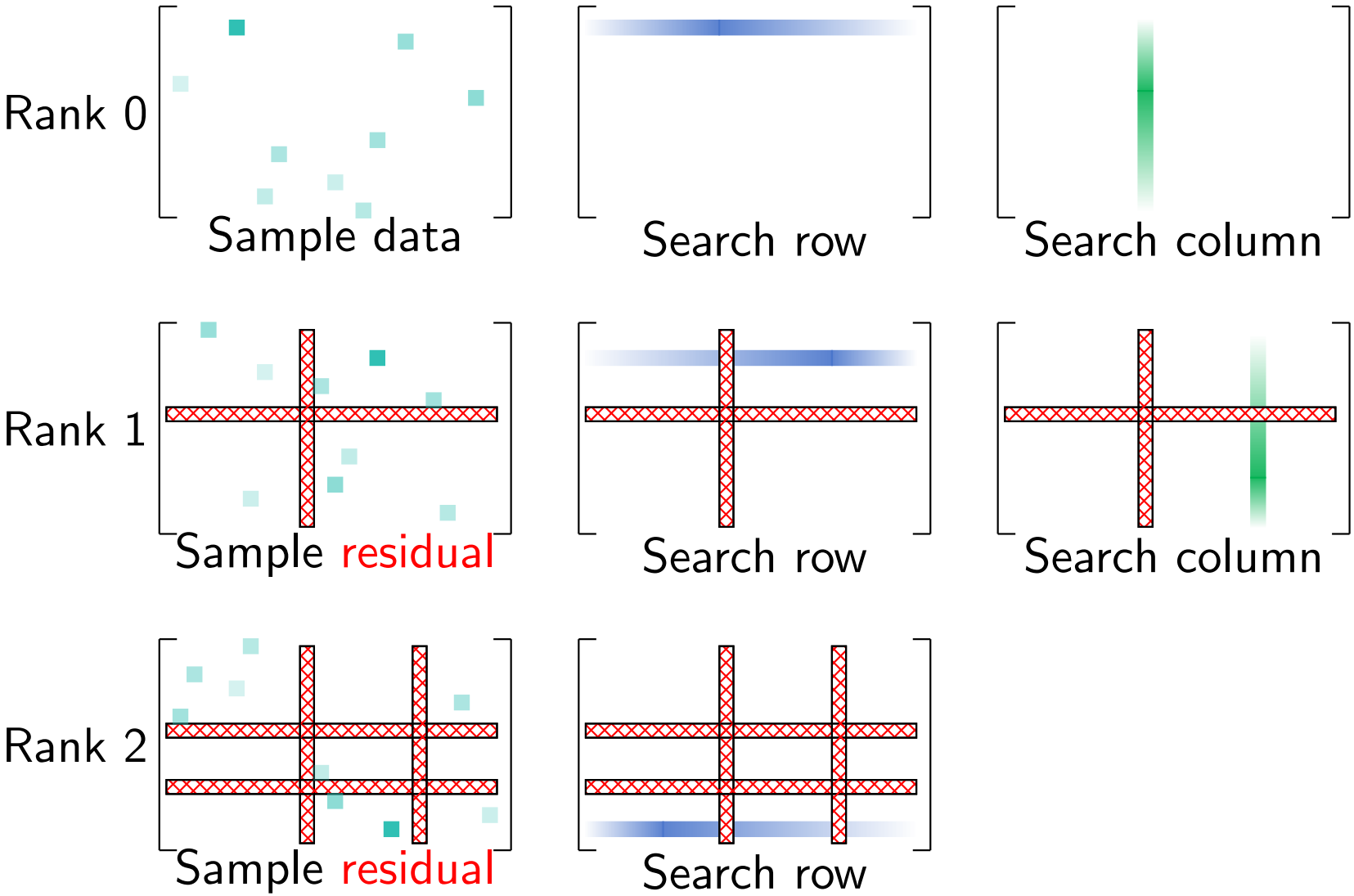
Greedy ACA for row and column identification



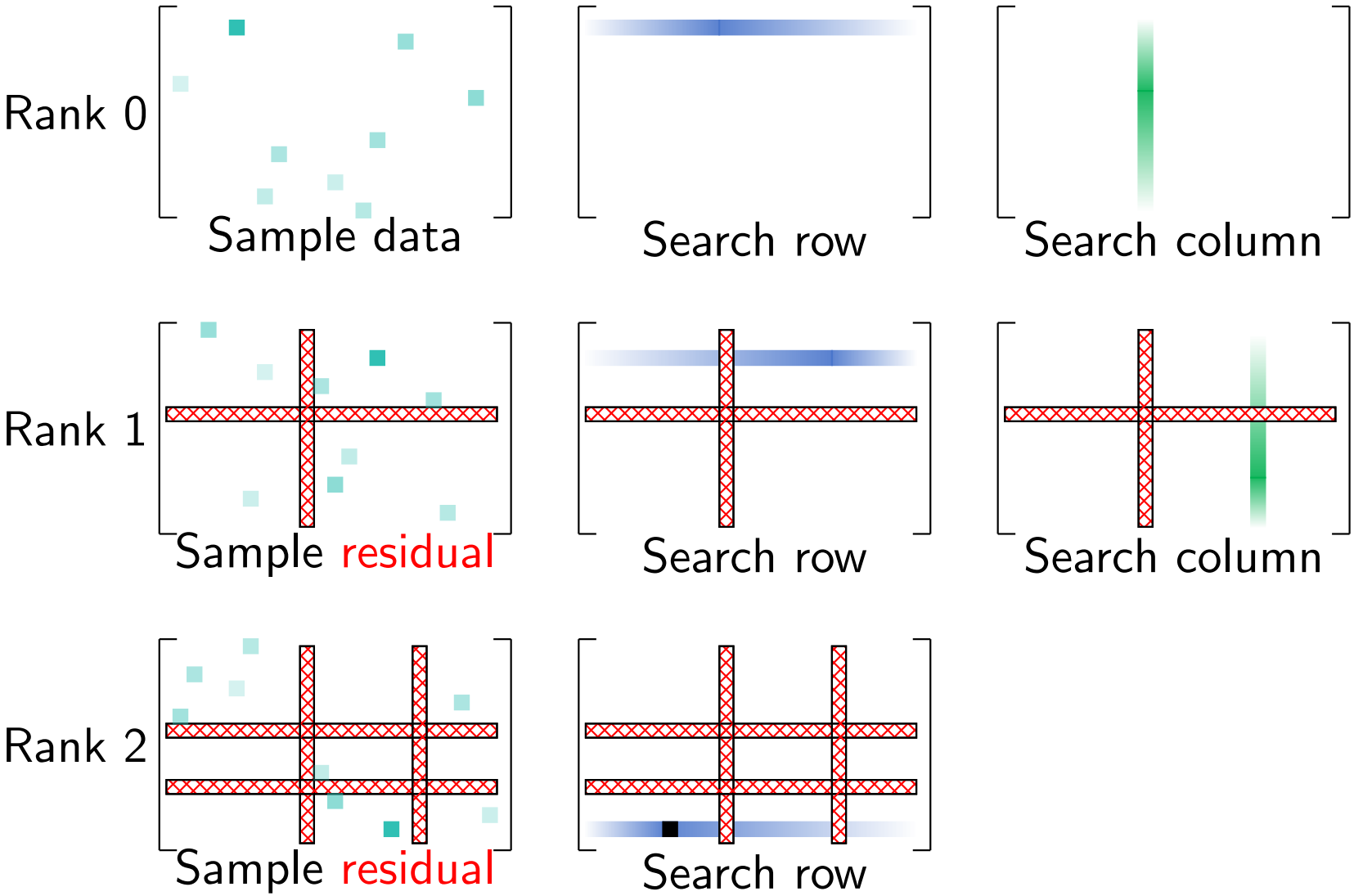
Greedy ACA for row and column identification



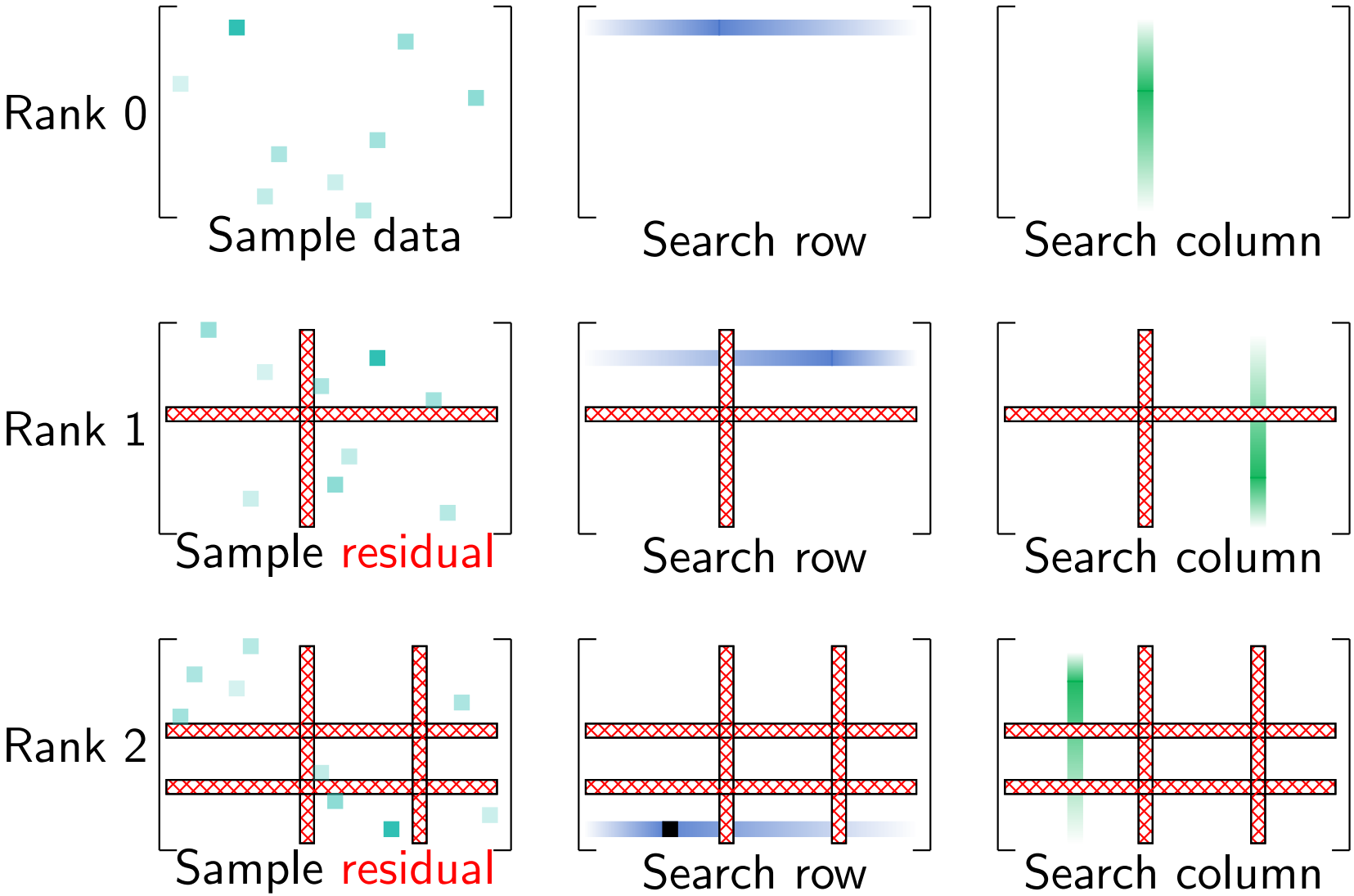
Greedy ACA for row and column identification



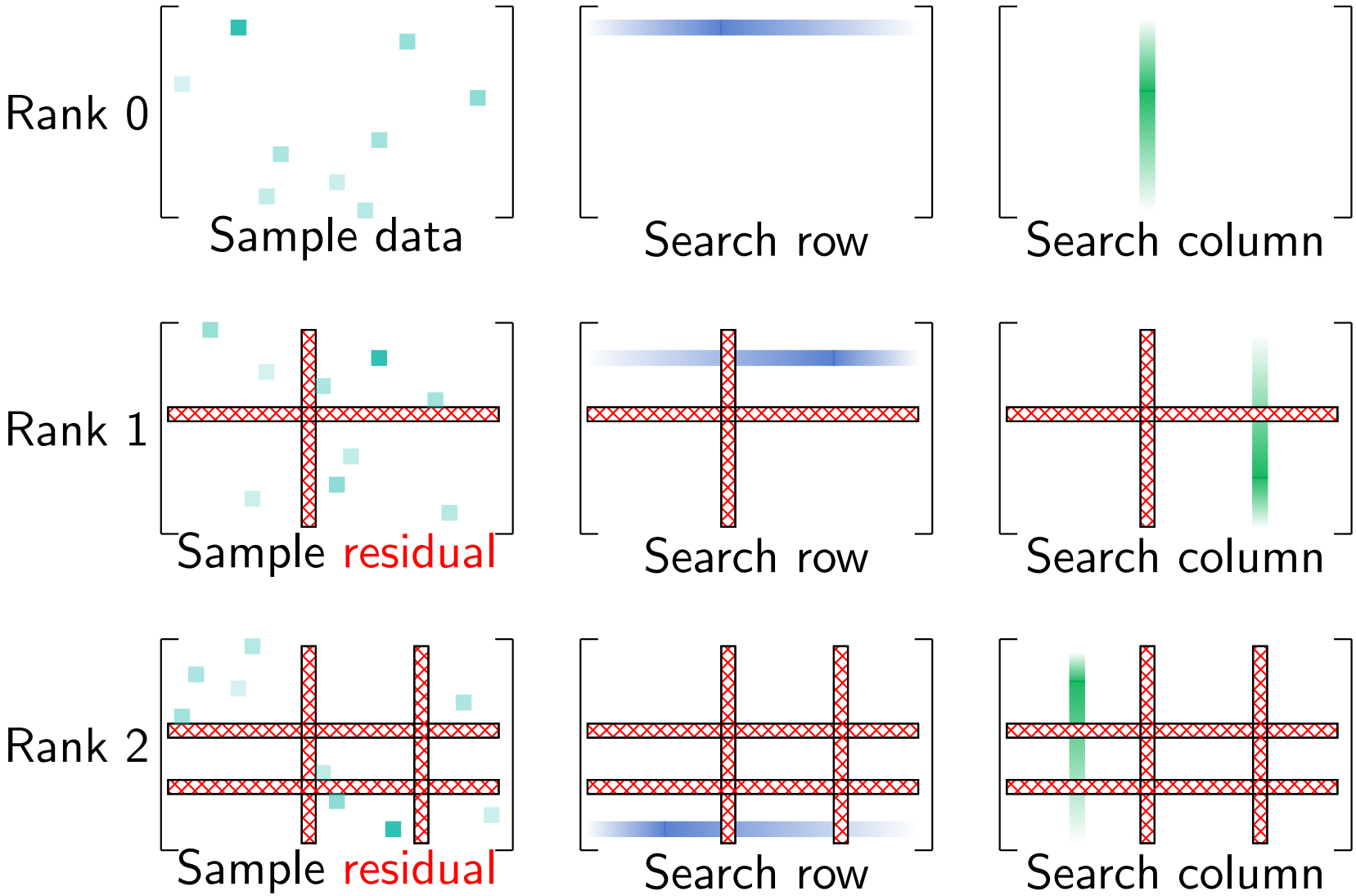
Greedy ACA for row and column identification



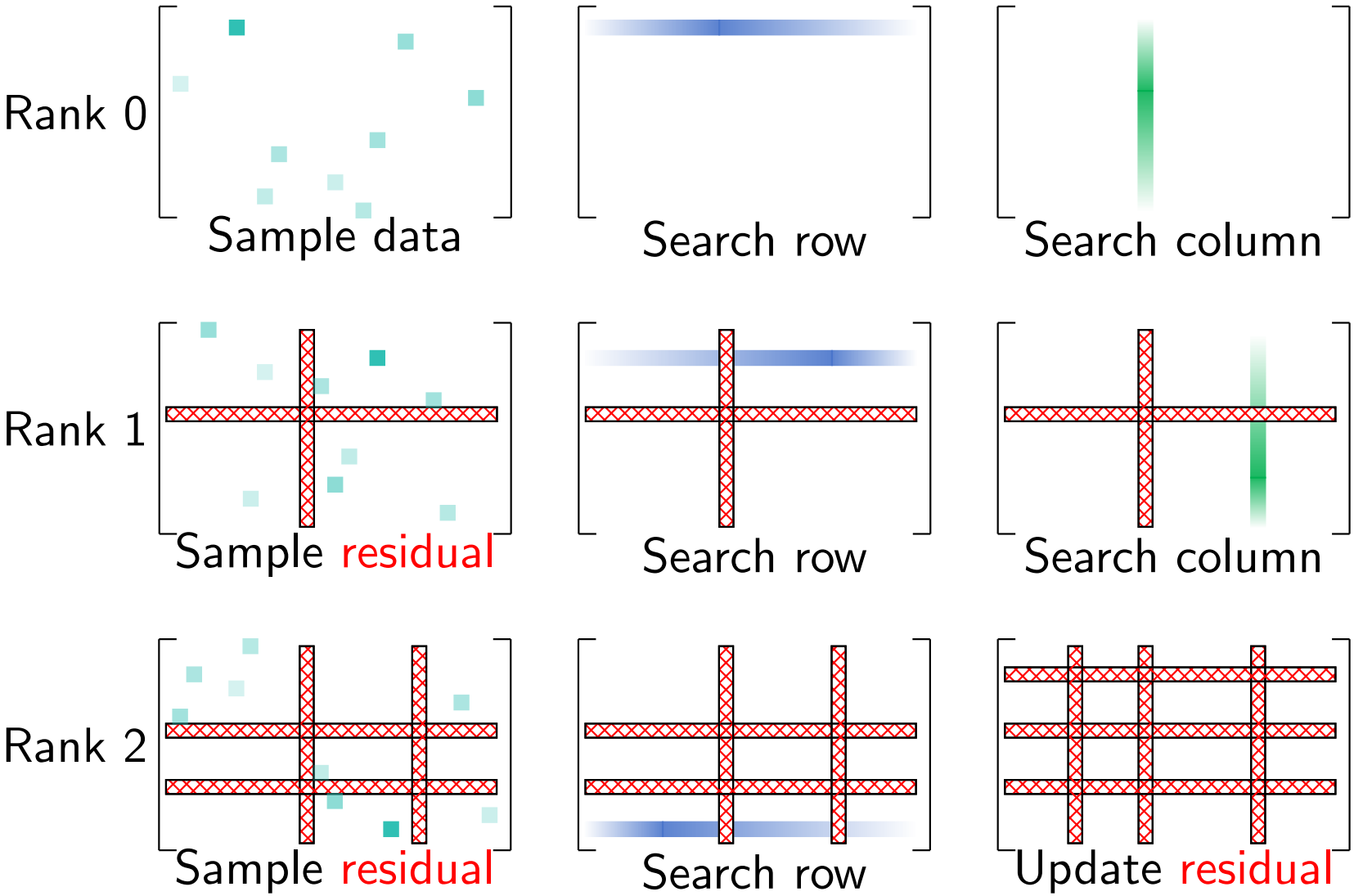
Greedy ACA for row and column identification



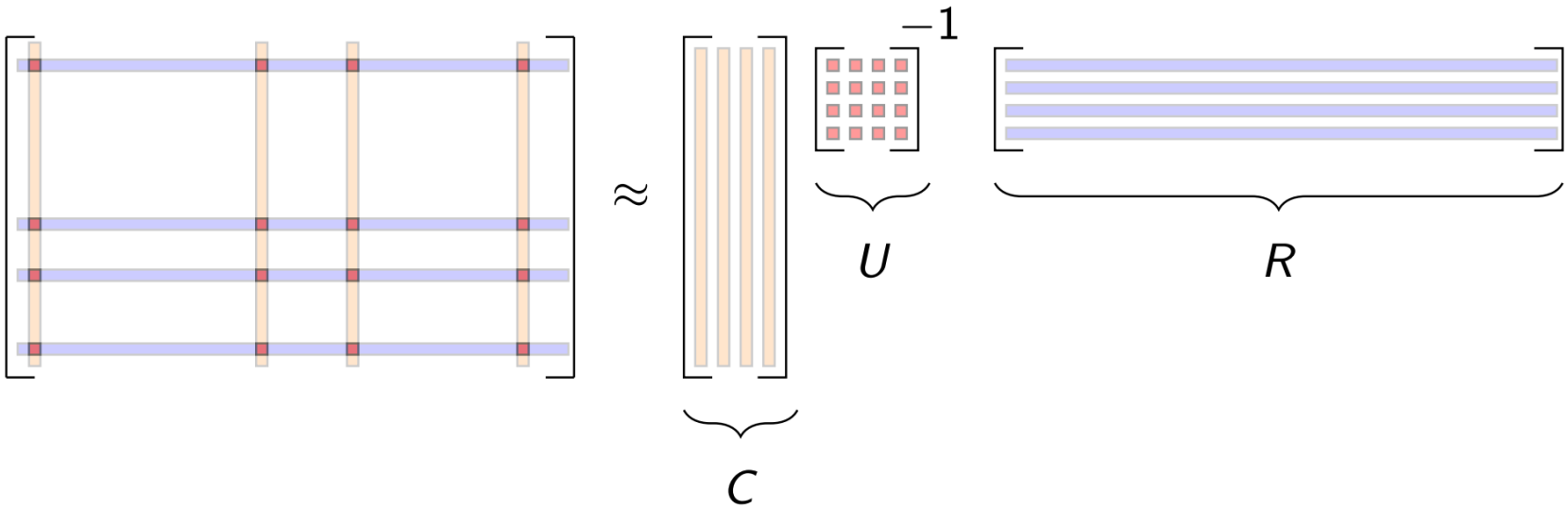
Greedy ACA for row and column identification



Greedy ACA for row and column identification



Adaptive cross sampling of SL solution update



- The intersection of the columns that make C and the rows that make R form a skeleton in the original matrix, which are used to form U.
- If $k \ll \min\{m, n\}$: significant computational savings.
- Recursive update:
  $A_k = A_{k-1} + \frac{1}{(E_{k-1})_{i,j}} (E_{k-1})_{:,j} (E_{k-1})_{i,:}, \quad E_{k-1} = A - A_{k-1}.$

Adaptive rank cross approximation of matrices[†]

1. Initialization: $A_0 = O$; $\mathcal{I} = \mathcal{J} = \emptyset$.
2. For k^{th} iteration:
   - 2.1 Initial sampling: pick p samples $\mathcal{L} = \{(i_l, j_l)\}_{l=1}^{p}$ randomly with $i_l \in \mathbb{I}_x \setminus \mathcal{I}$ and $j_l \in \mathbb{I}_y \setminus \mathcal{J}$,

$$(i^*, j^*) \leftarrow \arg \max_{(i,j)\in\mathcal{L}} |A_{i,j} - (A_{k-1})_{i,j}|.$$

   - 2.2 Pivoting strategy in selecting rows and columns

$$i_k^* \leftarrow \arg \max_{i\in\mathbb{I}_x\setminus\mathcal{I}} |A_{i,j^*} - (A_{k-1})_{i,j^*}|, \qquad j_k^* \leftarrow \arg \max_{j\in\mathbb{I}_y\setminus\mathcal{J}} |A_{i_k^*,j} - (A_{k-1})_{i_k^*,j}|.$$

$$\mathcal{I} \leftarrow \mathcal{I} \cup \{i_k^*\}, \quad \mathcal{J} \leftarrow \mathcal{J} \cup \{j_k^*\}.$$

   - 2.3 Update rank-k approximation:

$$A_k \leftarrow A_{k-1} + \frac{1}{(E_{k-1})_{i_k^\star, j_k^\star}} (E_{k-1})_{:,j_k^\star} (E_{k-1})_{i_k^\star,:}.$$

   - 2.4 If $\| \frac{1}{(E_{k-1})_{i_k^\star, j_k^\star}} (E_{k-1})_{:,j_k^\star} (E_{k-1})_{i_k^\star,:} \|_F < \varepsilon_C$ or $|(E_{k-1})_{i_k^\star, j_k^\star}| < 10^{-14}$, we obtain the cross decomposition $\tilde{A} = A_k.$ as a sum of rank 1 matrices

$$\tilde{A} = \sum_{k=1}^{r} \frac{1}{(E_{k-1})_{i_k^\star, j_k^\star}} (E_{k-1})_{:,j_k^\star} (E_{k-1})_{i_k^\star,:} := E_J D E_I.$$

[†] Tensor train extensions with MPI: Shi et. al. SISC 2025

1D1V Vlasov-Poisson system

Consider the 1D1V Vlasov Poisson system:

$$f_t + vf_x - E(x,t)f_v = 0, \quad x \in \Omega_x, \quad v \in \mathbb{R}, \tag{3}$$

$$E(x,t) = -\phi_x, \quad -\phi_{xx}(x,t) = \rho_0 - \rho(x,t), \tag{4}$$

where

- $f(x,v,t)$ is the probability distribution function of finding a particle at position x with velocity v at time t.
- Update solution $f^{n+1,\star}$ by the sampling-based adaptive rank approach.
  - Nonlinear Characteristics tracing coupled with SLAR, Cai, Boscarino & Q., JCP (2021).
  - Adaptive cross approximation of solution matrices.

Macroscopic conservation laws: $\boldsymbol{U}_t + \nabla_{\mathbf{x}} \cdot \mathbf{F} = \mathbf{S}$

$$\rho_t + \nabla_{\mathbf{x}} \cdot \mathbf{J} = 0$$
$$\partial_t \mathbf{J} + \nabla_{\mathbf{x}} \cdot \sigma = (\rho_0 - \rho)\mathbf{E}$$
$$\partial_t e + \nabla_{\mathbf{x}} \cdot (\mathbf{Q} + \mathbf{F}_{e_p}) = 0$$

- particle density: $\rho(\mathbf{x}, t) = \rho_0 - \int_{\Omega_{\boldsymbol{v}}} f(\mathbf{x}, \boldsymbol{v}, t) d\boldsymbol{v}$
- current density: $\mathbf{J}(\mathbf{x}, t) = -\int_{\Omega_{\boldsymbol{v}}} \boldsymbol{v} f(\mathbf{x}, \boldsymbol{v}, t) d\boldsymbol{v}$
- kinetic energy density: $\kappa(\mathbf{x}, t) = \frac{1}{2} \int_{\Omega_{\boldsymbol{v}}} \boldsymbol{v}^2 f(\mathbf{x}, \boldsymbol{v}, t) d\boldsymbol{v}$
- energy density: $e(\mathbf{x}, t) = \kappa(\mathbf{x}, t) + \frac{1}{2}|\mathbf{E}|^2$
- electric field: $\boldsymbol{E} = -\nabla_{\boldsymbol{x}} \phi$, ϕ is electrostatic potential $-\Delta_{\boldsymbol{x}} \phi = \rho$.
- fluxes:
  - $\sigma(\mathbf{x}, t) = -\int_{\Omega_{\boldsymbol{v}}} (\boldsymbol{v} \otimes \boldsymbol{v}) f(\mathbf{x}, \boldsymbol{v}, t) d\boldsymbol{v}$
  - $\mathbf{Q}(\mathbf{x}, t) = \frac{1}{2} \int_{\Omega_{\boldsymbol{v}}} \boldsymbol{v} |\boldsymbol{v}|^2 f(\mathbf{x}, \boldsymbol{v}, t) d\boldsymbol{v}$
  - $\boldsymbol{F}_{e_p} = -\phi \nabla_{\boldsymbol{x}} (\partial_t \phi) + \phi \mathbf{J}$
- $\boldsymbol{U} = (\rho, \mathbf{J}, e)'$, $\mathbf{F} = (\mathbf{J}, \sigma, \mathbf{Q} + \mathbf{F}_{e_p})'$ and $\mathbf{S} = (0, (\rho_0 - \rho)\mathbf{E}, 0)'$.

However, the mass, momentum and energy conservation of $f^{n+1,\star}$ is lost in sampling and SVD truncation steps.

LoMaC [‡]

$$\partial_t f + \boldsymbol{v}\cdot\nabla_{\boldsymbol{x}} f + \boldsymbol{E}\cdot\nabla_{\boldsymbol{v}} f = 0$$

Kinetic flux →

← **Macroscopic densities**

$$\begin{aligned} \rho_t + \nabla_{\boldsymbol{x}}\cdot\boldsymbol{J} &= 0 \\ \partial_t \boldsymbol{J} + \nabla_{\boldsymbol{x}}\cdot\boldsymbol{\sigma} &= \rho\,\boldsymbol{E} \\ \partial_t e + \nabla_{\boldsymbol{x}}\cdot(\boldsymbol{Q} + \boldsymbol{F}_{e_p}) &= 0. \end{aligned}$$

→ Kinetic flux: enabling a conservative method with kinetic flux vector splitting (KFVS) to update $\boldsymbol{U}^{n+1}$:

$$\nabla_x \cdot \begin{pmatrix} \mathbf{J} \\ \sigma \\ \mathbf{Q} \end{pmatrix} = \nabla_x \cdot \left[\int f v \begin{pmatrix} 1 \\ v \\ \frac{1}{2} v^2 \end{pmatrix} dv \right]$$

← Macroscopic densities. Project the adaptive rank solution $\mathbf{f}^{n+1,\star}$ to ensure consistency with local conservation laws $\boldsymbol{U}^{n+1}$,

$$\mathbf{f}^{n+1,\star} = \mathbf{f}_1^{\star} + \mathbf{f}_2 = \mathcal{P}_{\mathcal{N}}(\mathbf{f}^{n+1}) + (\mathcal{I} - \mathcal{P}_{\mathcal{N}})(\mathbf{f}^{n+1}),$$

where $\mathcal{P}_{\mathcal{N}}$ denote the orthogonal projection onto $\mathcal{N}$ with respect to $\langle\cdot,\cdot\rangle_{\omega^{-1}}$, with

$$L^2_{\omega^{-1}}: \quad \langle g, h\rangle_{\omega^{-1}} = \int_{\Omega_{\boldsymbol{v}}} g(\boldsymbol{v})\, h(\boldsymbol{v})\, \omega(\boldsymbol{v})^{-1}\, d\boldsymbol{v},$$

$$\mathcal{N}(\omega) = \operatorname{span}\Big\{\omega(\boldsymbol{v}),\ v^{(1)}\,\omega(\boldsymbol{v}), \ldots, v^{(d_v)}\,\omega(\boldsymbol{v}),\ |\boldsymbol{v}|^2\,\omega(\boldsymbol{v})\Big\}.$$

[‡] Guo & Q., JSC (2024)

LoMaC

- From the adaptive rank kinetic solution $\mathbf{f}^{n+1,\star}$ and its moments $\boldsymbol{U}^{n+1,\star}$
  - $$\mathbf{f}_1^{\star} = \omega \star (c_1 \otimes \mathbf{1}_v + c_2 \otimes \mathbf{v} + c_3 \otimes (\boldsymbol{v}^2 - c)),$$

    with c_1, c_2, c_3 and c from $\boldsymbol{U}^{n+1,\star}$. .
  - Perform the weighted SVD truncation on

$$\mathcal{T}_\omega(\mathbf{f}_2), \qquad \text{with} \quad \mathbf{f}_2 = \mathbf{f}^{*,(n+1)} - \mathbf{f}_1^{\star}.$$

- From the conservation laws, $\boldsymbol{U}^{n+1}$,

$$\mathbf{f}_1^{n+1} = \omega \star (c_1 \otimes \mathbf{1}_v + c_2 \otimes \mathbf{v} + c_3 \otimes (\boldsymbol{v}^2 - c)),$$

  with c_1, c_2, c_3 and c from $\boldsymbol{U}^{n+1}$.
- $\mathbf{f}^{n+1} = \mathbf{f}_1^{n+1} + \mathcal{T}_\omega(\mathbf{f}_2)$.

However, that LoMaC

- was an explicit correction procedure/postprocessing procedure for an explicit step-and-truncate adaptive rank scheme.
- used an explicit weight function $\omega(\boldsymbol{v}) = exp(-\frac{|\boldsymbol{v}|^2}{2})$, which is not optimal for Vlasov simulations and requires ad hoc tuning of Maxwellian shape.

With SLAR, we need a new version of LoMaC, as

- an implicit macroscopic solver is desirable allowing for large time stepping sizes as in a SL scheme
- adaptive and robust choice of weight function without ad hoc parameter tuning.

An implicit LoMaC with adaptive choice of the weight function $\omega(\boldsymbol{v})$.

An implicit solver of the macroscopic system

A BE discretization of the macroscopic system yields

$$\boldsymbol{U}^{n+1} = \boldsymbol{U}^{n} - \Delta t \nabla_{\mathbf{x}} \cdot \mathbf{F}(\boldsymbol{U}^{n+1}) + \Delta t\, \mathbf{S}^{n+1},$$

$$F(U^{n+1}) := \underbrace{\left\langle \left[(f^{n+1,\star} - \mathcal{W}_{U(f^{n+1,\star})} \right] \begin{pmatrix} v \\ v^2 \\ \frac{1}{2} v^3 \end{pmatrix} \right\rangle_v}_{\text{This term is explicit from adaptive rank update!}} + \left\langle \mathcal{W}_{U^{n+1}} \begin{pmatrix} v \\ v^2 \\ \frac{1}{2} v^3 \end{pmatrix} \right\rangle_v$$

- $f(\mathbf{U}) = f^{n+1,\star} - \mathcal{W}_{\mathbf{U}(f^{n+1,\star})} + \mathcal{W}_{\mathbf{U}}$
- $\mathcal{W}$ is taken to be Maxwellian in the collisional simulations; but in the Vlasov setting $\mathcal{W}$ is taken to be the dominant singular vector in v dimension [§].
- In the fully-discrete setting, we will have a nonlinear system of algebraic equations of the form

$$\mathcal{G}\left(\mathbf{U}^{n+1}\right) = 0, \quad \mathbf{U}^{n+1} \in \mathbf{R}^{3N_x}$$

[§] By Perron–Frobenius theorem, a real square matrix with positive entries has a unique eigenvalue of largest magnitude and that eigenvalue is real

Newton-Krylov (NK) Solver[¶]

To solve $\mathcal{G}(\mathbf{U}) = 0$, we use a NK method with moments of the SLAR solution as an initial guess.

- Newton's iteration (outer loop): we solve a linearized problem

$$J^{(k)}\delta\mathbf{U}^{(k)} = -\mathcal{G}\left(\mathbf{U}^{(k)}\right), \quad \mathbf{U}^{(k+1)} := \mathbf{U}^{(k)} + \delta\mathbf{U}^{(k)}, \quad \text{for } k = 1, 2, \cdots$$

  where $J^{(k)} = \mathcal{G}'\left(\mathbf{U}^{(k)}\right)$ is the Jacobian matrix.

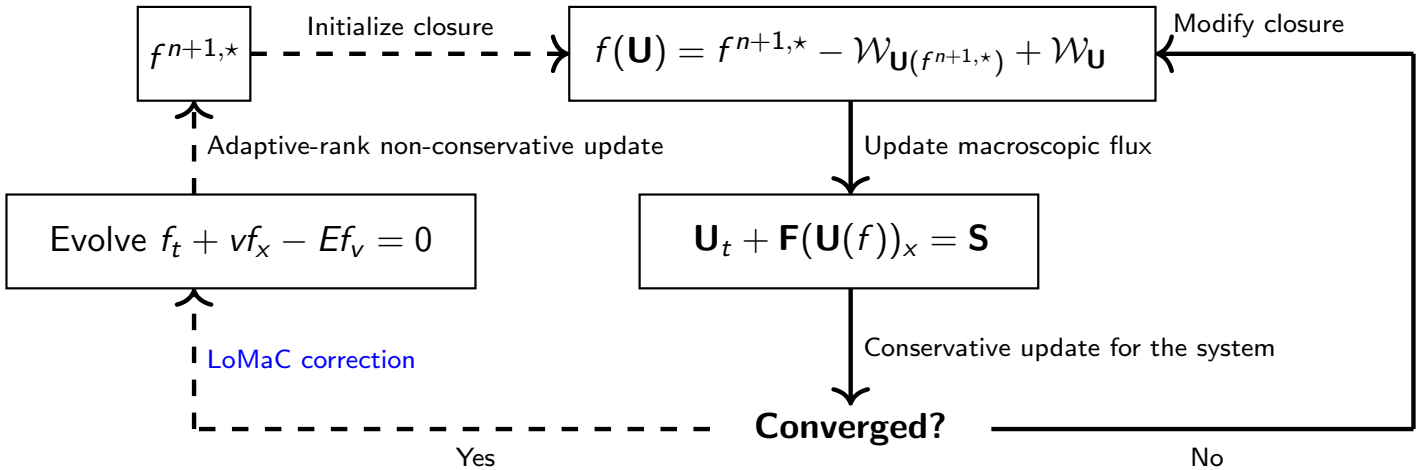
- Krylov iteration (inner loop): Krylov methods only require the action of the Jacobian. For example, consider the action of $J^{(k)}$ on an arbitrary vector $\mathbf{V}$. It can be approximated as

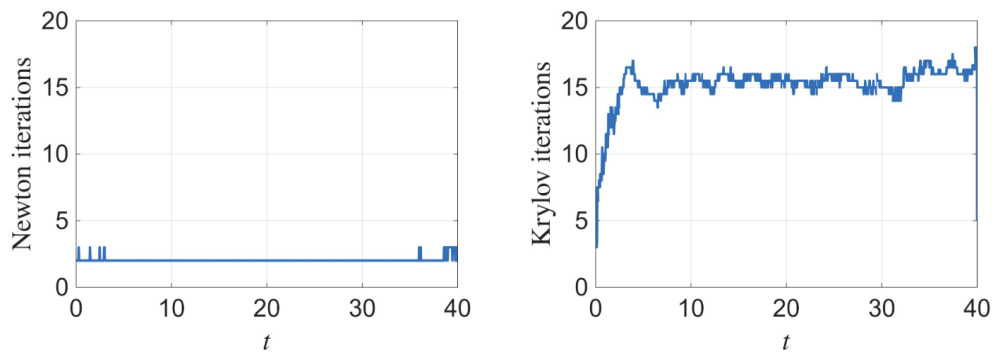
$$J^{(k)}\mathbf{V} = \frac{\mathcal{G}\left(\mathbf{U}^{(k)} + \epsilon\mathbf{V}\right) - \mathcal{G}\left(\mathbf{U}^{(k)}\right)}{\epsilon} + \mathcal{O}(\epsilon).$$

Solve for $\mathbf{U}^{n+1}$ from moment system.

[¶]Knoll and Keyes, 2004

A flow chart of LoMaC correction ‖

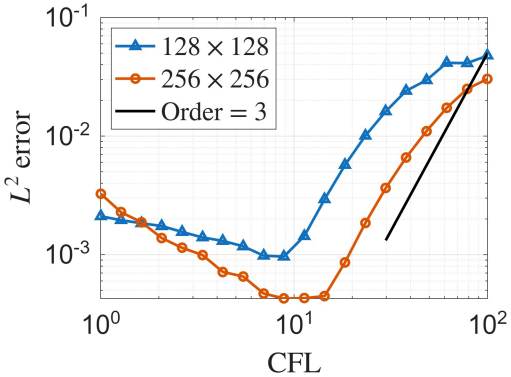


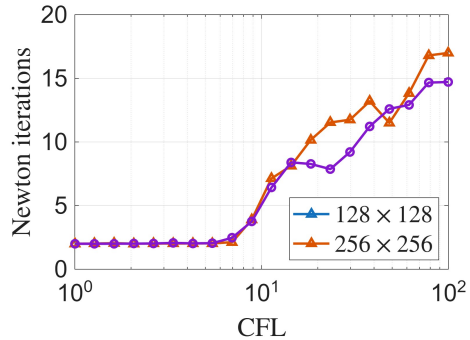


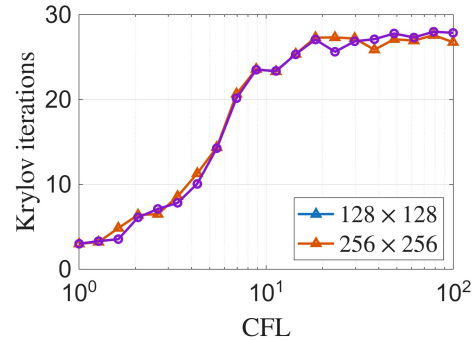
Average number of iterations per stage over time: strong Landau damping, mesh 256×256, $CFL = 5$.

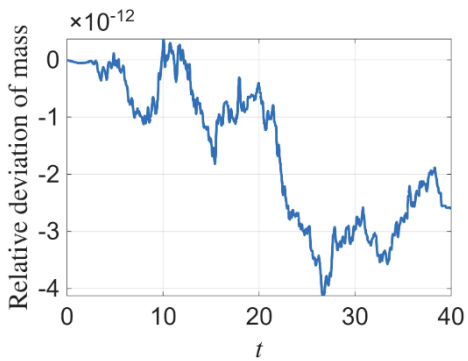
‖ Similar to our previous LoMaC for BGK but with adaptive weight: JCP, 2026, arXiv: 2505.17191

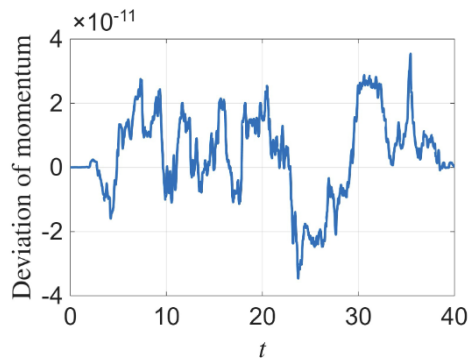
Vlasov-Poisson System: strong Landau damping

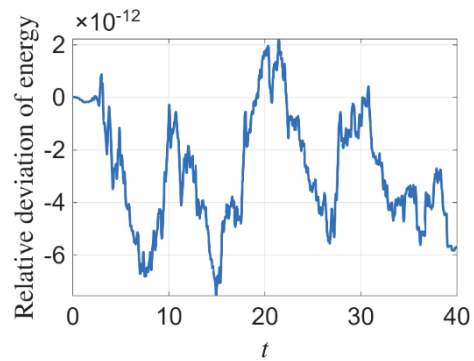








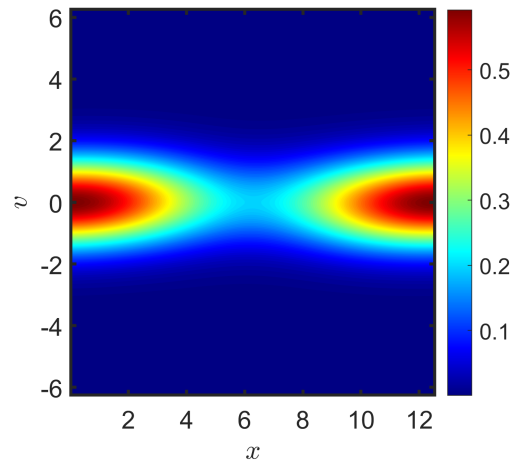


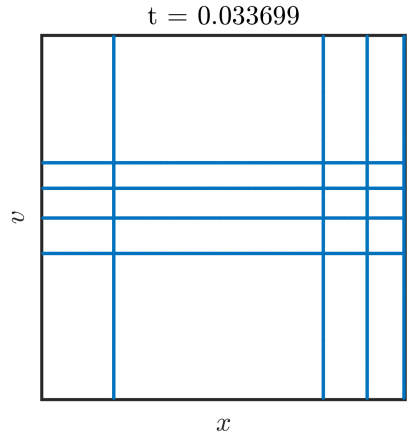


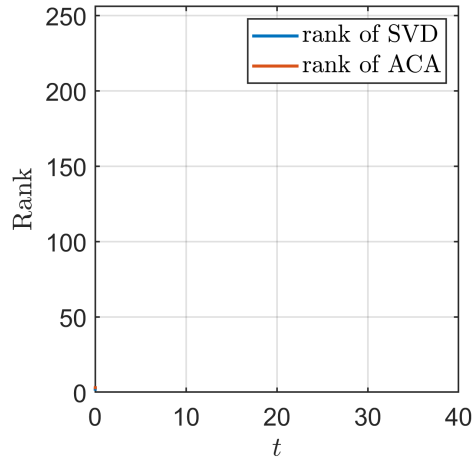
Strong Landau damping. A mesh of 256×256 for $t = 5$. Truncation threshold $\varepsilon_{\text{Trunc}} = 10^{-4}$. Better accuracy, capture of adaptive rank basis, and compression ratio with mesh refinement, which can be done efficiently due to linear complexity in N.

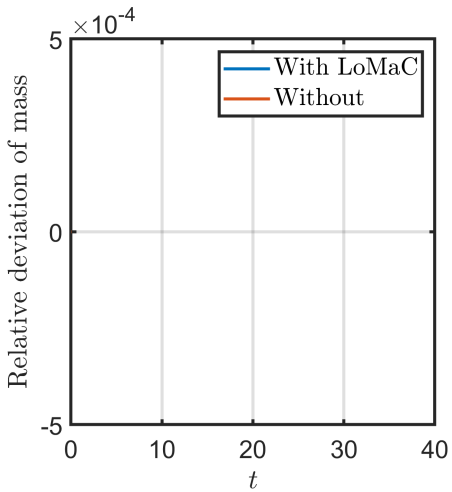
Strong Landau damping

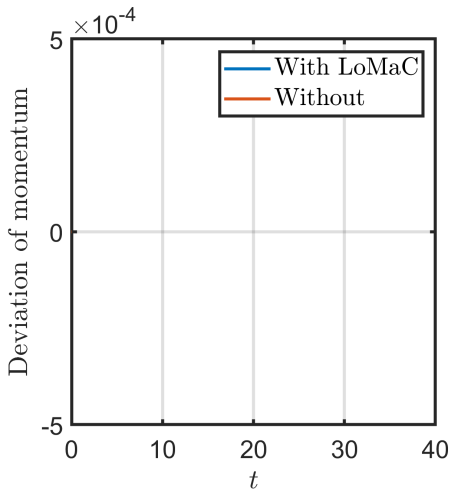
Truncation threshold $\varepsilon_{\text{Trunc}} = 10^{-4}$. $N_x \times N_v = 256 \times 256$.

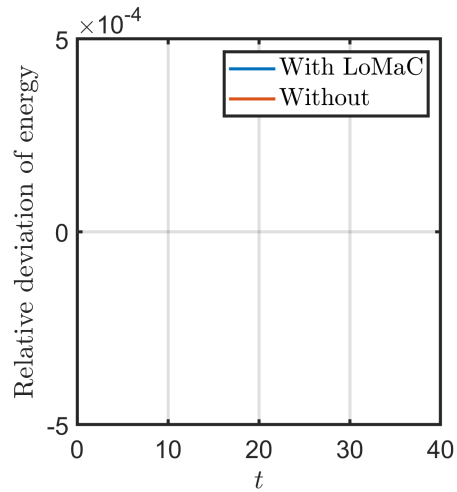






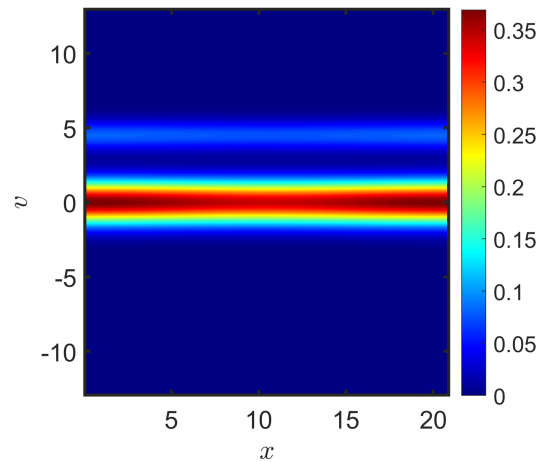


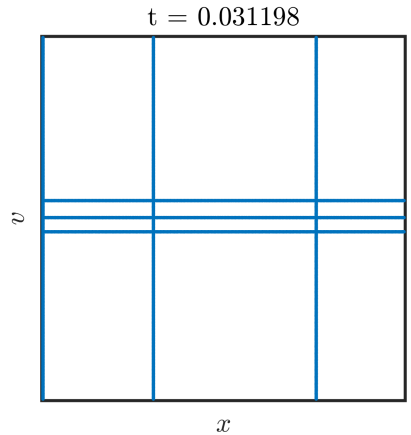


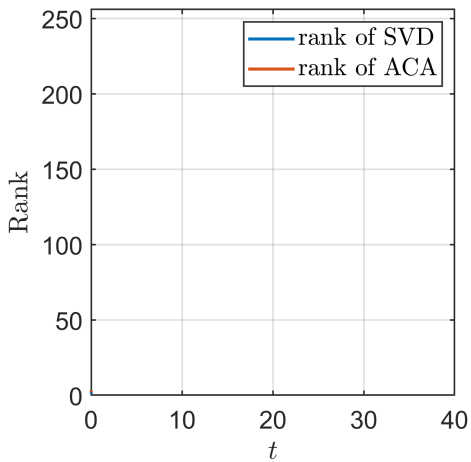


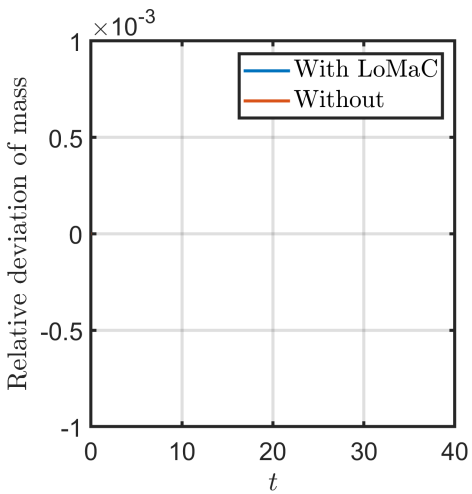
Bump-on-tail instability

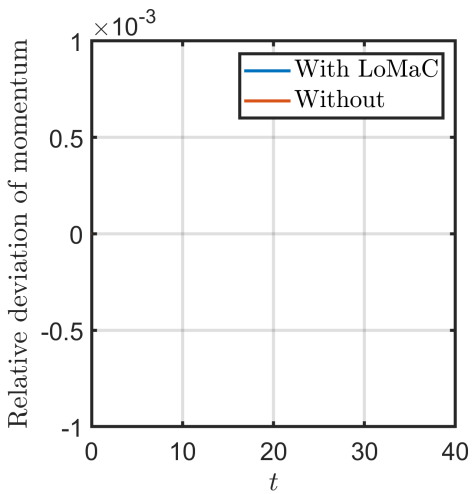
Truncation threshold $\varepsilon_{\text{Trunc}} = 10^{-4}$. $N_x \times N_v = 256 \times 256$.

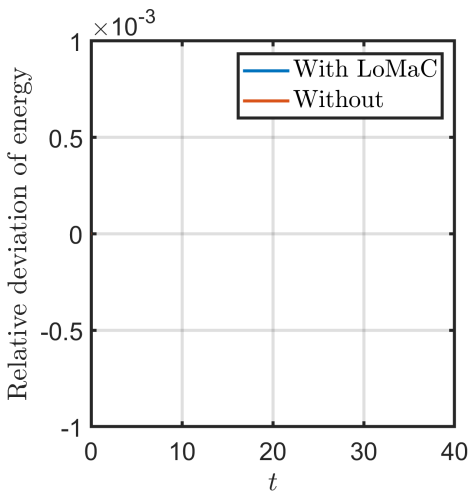












Tensor Network Approximation of High Dimensional PDE Solutions

Low-rank tensor approximation of functions

- Huge amount of literature. Textbook by [Hachbusch 2012].
- The canonical polyadic (CP) tensor decomposition represents a multivariate function as a sum of rank-one separable functions [Hitchcock 1927], [Carroll, Chang 1970], [Kolda, Bader 2009], ...).
- Tensor networks for sparse data tensor decomposition: the hierarchical Tucker (HT) format ([Hackbusch, Kühn 2009], [Grasedyck 2010], ...) and the tensor train (TT) format ([Oseledets 2011], ...). Storage complexity: linear scaling with the dimension.

Tensor

Algebraically, a tensor is a multilinear array (this is a very simplified setting).

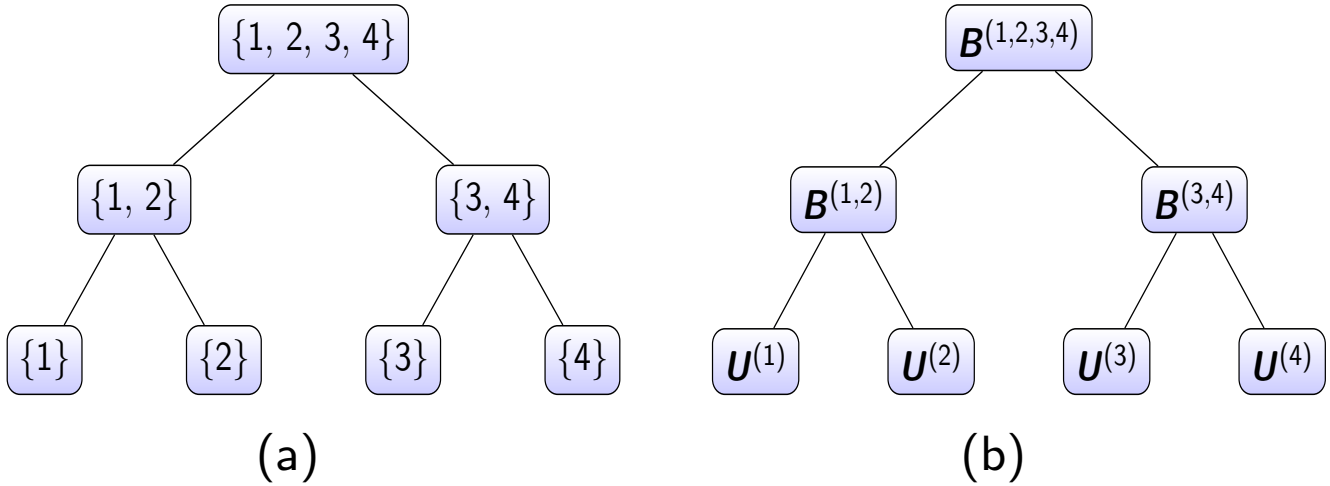
$$\mathbf{a}[\mathbf{i}] = \mathbf{a}[i_1, i_2, \ldots, i_d] \in \mathbb{R},$$

$i_j \in I_j = \{1, 2, \ldots, n_j\}$, $\mathbf{I} = \otimes_{j=1}^{n} I_j$. $\mathbf{a} \in \mathbb{R}^{\mathbf{I}}$ is called a d-th order tensor.
Example

- $d = 1$, vector
- $d = 2$, matrix
- grid function based on the direct discretization of a multivariate function $f(\mathbf{x})$ on product grids, $\mathbf{a}[i_1, i_2, \ldots, i_d] = f(i_1 h, i_2 h, \cdots, i_d h)$. h: the mesh size.
  - Storage cost: $\prod_{j=1}^{d} n_j$, suffers the curse of dimensionality.

Hierarchical Tucker (HT) format

To represent a four-order tensor in HT format, we define a dimension tree.

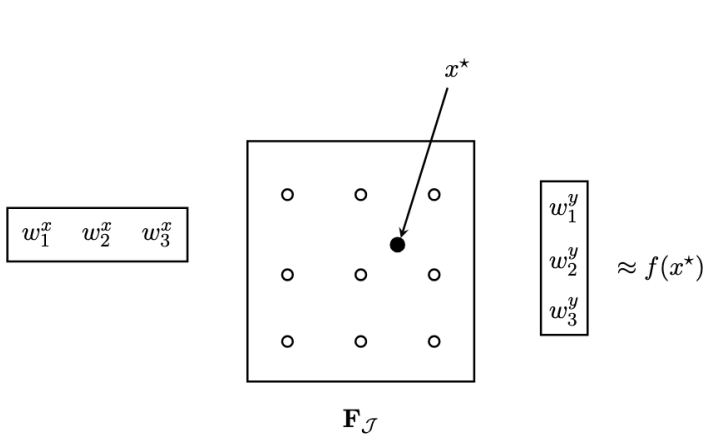


- The storage complexity of the HT format scales like $O(ndr + dr^3)$, where $r = \max_\alpha\{r_\alpha \in \mathbf{r}_{HT}\}$, avoiding the exponential scaling on d.
  - leaf node $\{j\}$, the basis $u^{(j)}$ of U_j is explicitly stored
  - at a non-leaf node α, the *transfer* tensor $\boldsymbol{B}^{(\alpha)}$ is stored
- a tensor already in the HT format can be truncated to smaller HT rank with a quasi-optimal error bound, and the cost only scales like $O(dnr^2 + dr^4)$, i.e., linear scaling with d.

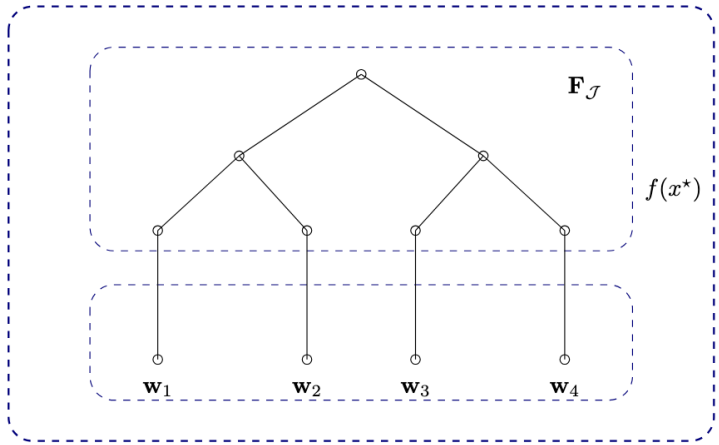
New components for the high D extension with HT format

- High D polynomial approximation of $f(\mathbf{x}^\star)$
  - A high D tensor contraction
- High D sampling scheme:
  - Hierarchical Tucker Adaptive Cross Approximation (HTACA)
- High D LoMaC extension

High dimensional approximation of $f(\mathbf{x}^\star)$



(a) Local tensor-product interpolation.



(b) Tree-structured contraction of tensors.

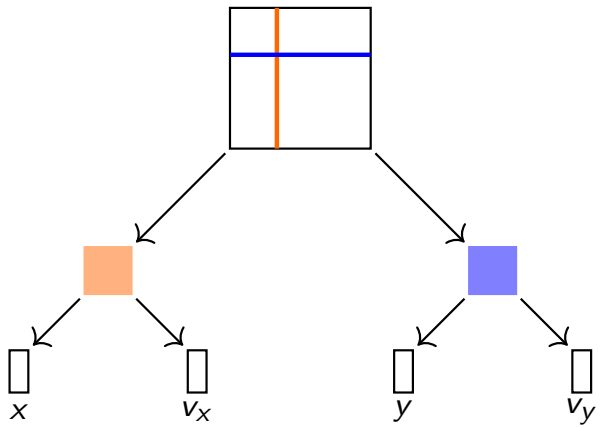
$f(\mathbf{x}^\star)$ needs to be approximated at the foot of characteristics. A tensor contraction from f at a local stencil (tensor product of grids) in a HT format. Left: 2D reconstruction; Right: extensions to 4D on hierarchical Tucker

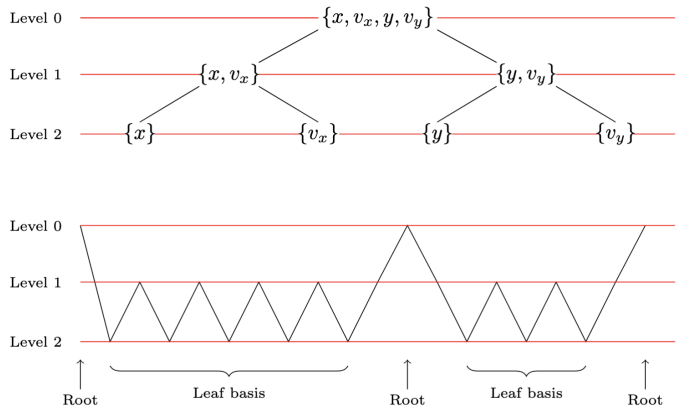
- Unconditional ℓ^2-stability on uniform tensor product of grids for linear high D problems

$$\|\mathbf{F}^{n+1}\|_{\ell^2} \leq \|\mathbf{F}^n\|_{\ell^2}, \qquad \forall n \geq 0. \tag{5}$$

- The dominant complexity scales as $\mathcal{O}(d\, r^3)$, r is the maximum rank in a HT tree.

Hierarchical Tucker Adaptive Cross Approximation (HTACA)

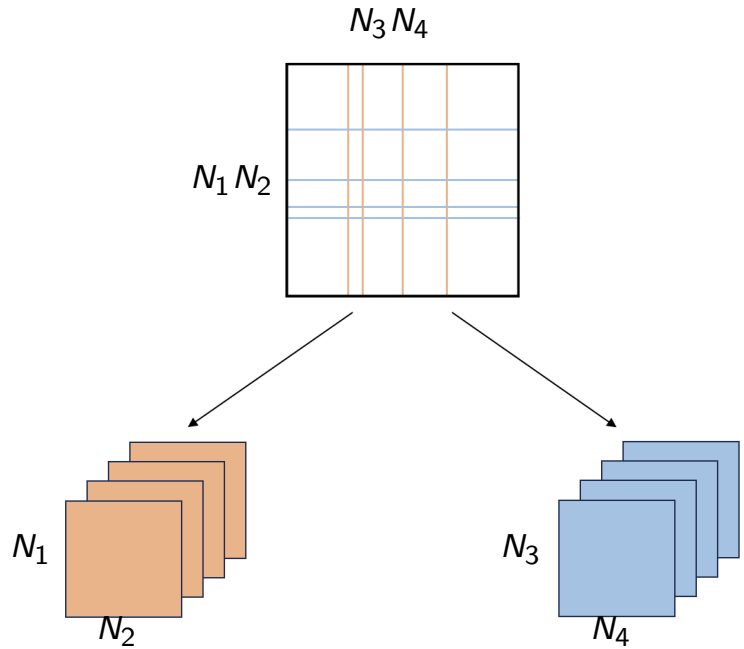




A 2D2V example. Left: HT dimension tree with matrization of a 4D tensor $\mathcal{X}$. Right: a recursive pivot search from root to leaf nodes. The pivot search starts from the root node down the tree: to search the pivot along a column, i.e. the orange matrix, the search iterates between sibling nodes x and v_x; this returns a row pivot $(i_x^\star, i_{v_x}^\star)$ back to the root node; and then a search in the corresponding row, i.e. the blue matrix of $\mathcal{X}(i_x^\star, i_{v_x}^\star, :, :)$, is done a similar recursive fashion.

HTACA construction

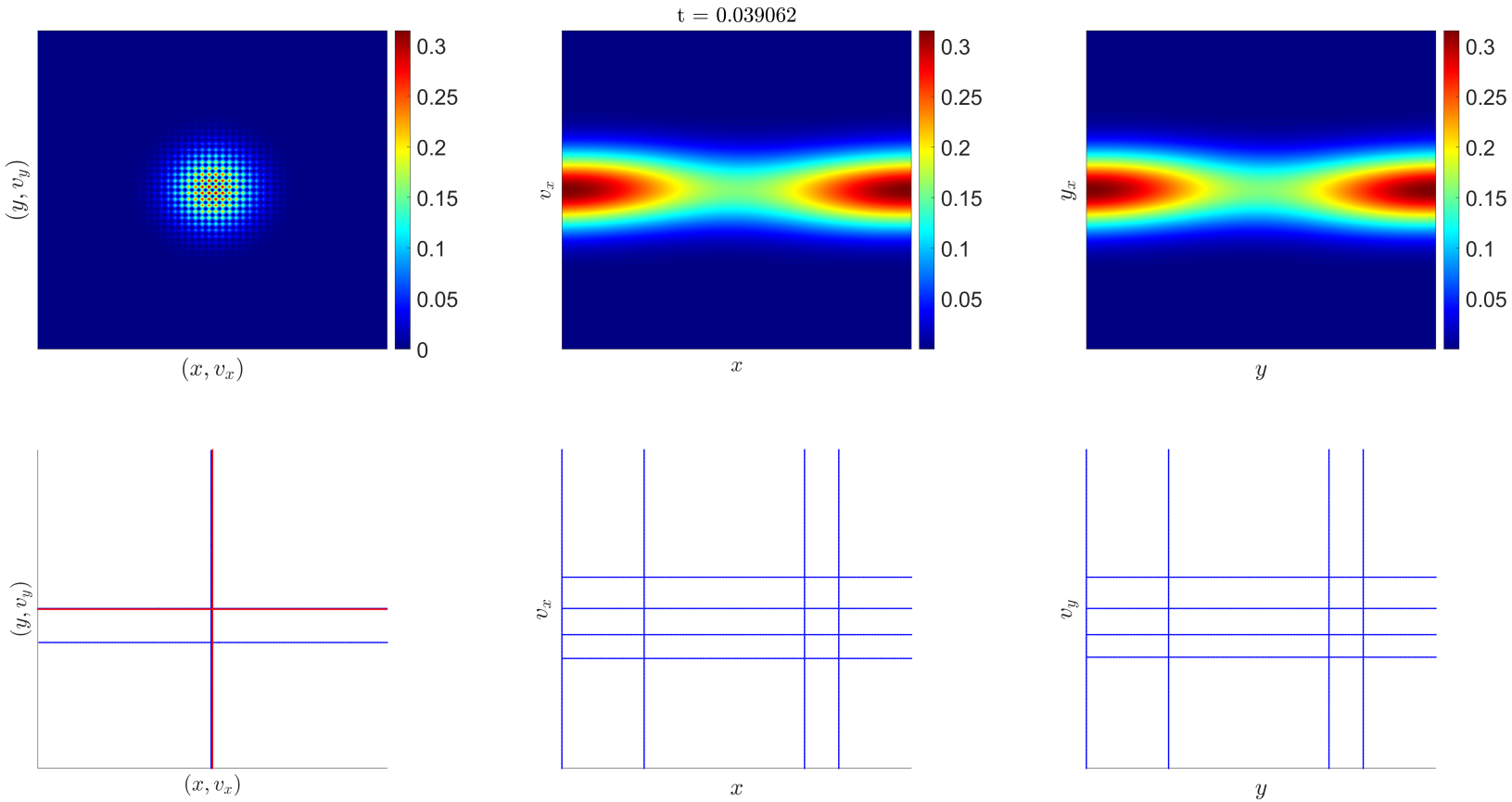
- Starting from the root node, apply matrix ACA: recursive rank 1 update of residual, followed by a truncation.
- Recursively apply HTACA construction on children nodes until leaf nodes.



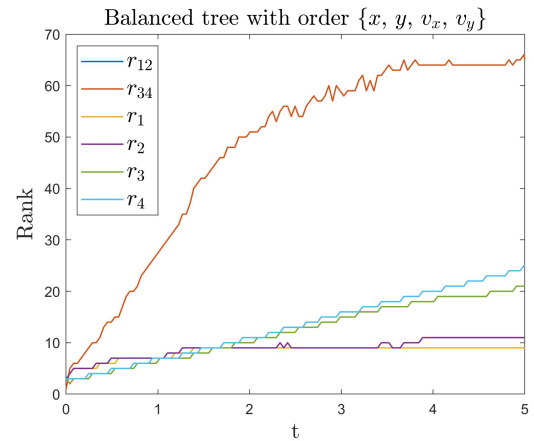
A recursive HTACA construction: a 2D2V example.

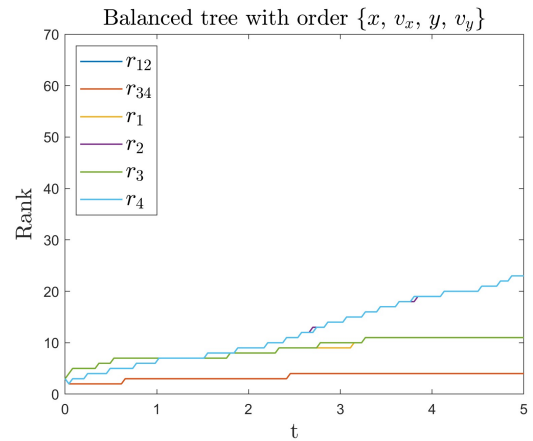
2D2V strong Landau damping

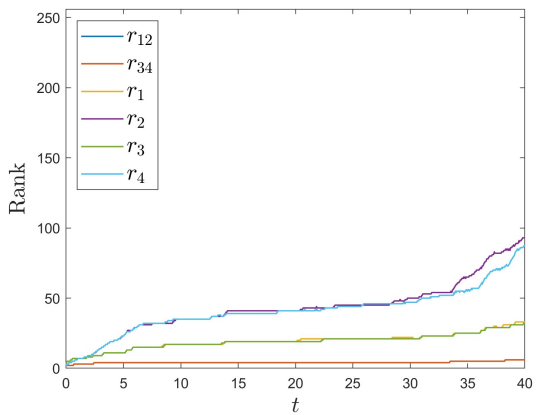
Truncation threshold $\varepsilon_{\text{Base}} = 1e-3$. Mesh is $256^2 \times 256^2$.

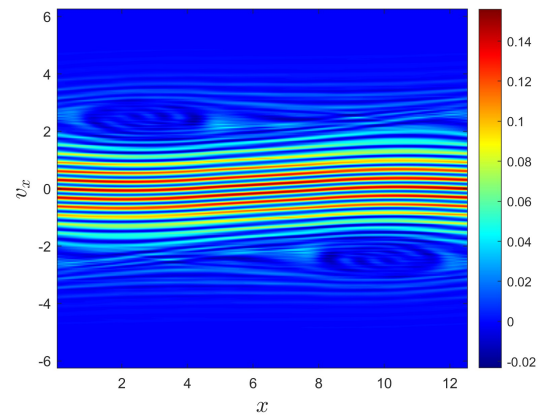


2D2V strong Landau damping



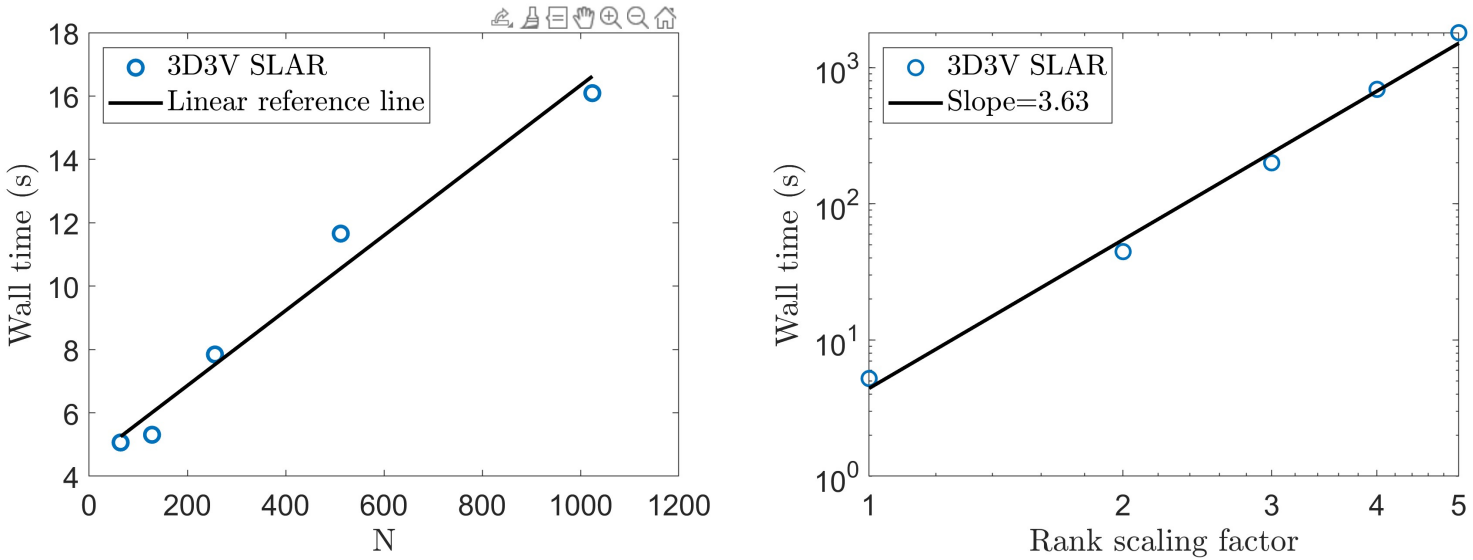






Upper plots: comparison of two different tensor trees. Lower left: rank history; right: contour plot of the f at $(y, v_y) = (2\pi, 0)$ and $t = 40$.
Simulation settings: mesh $= 256^4$, CFL $= 10$. $v_{\text{max}} = 2\pi$. $\epsilon_{\text{Trunc}} = 10^{-3}$.

Computing time vs. N and r

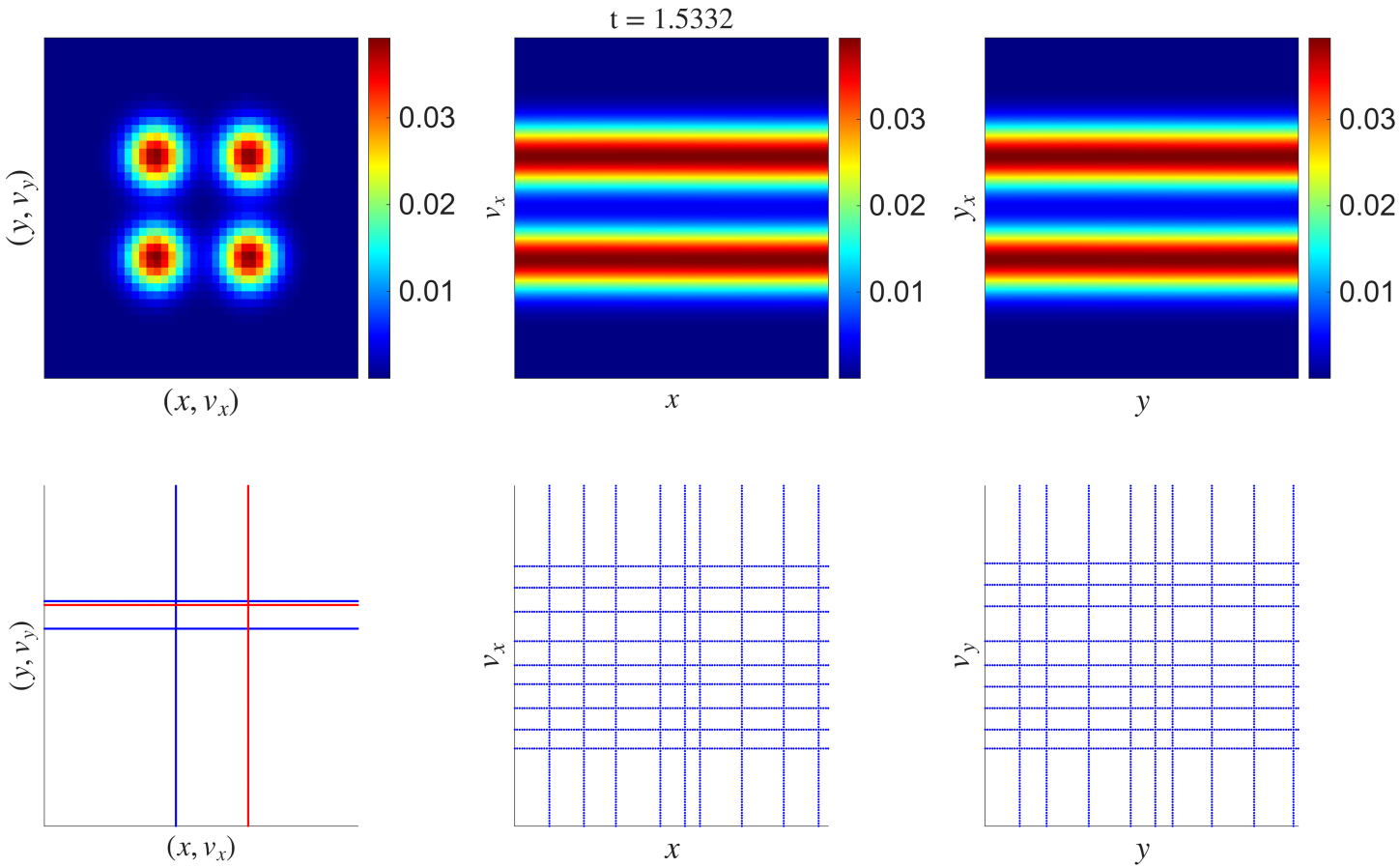


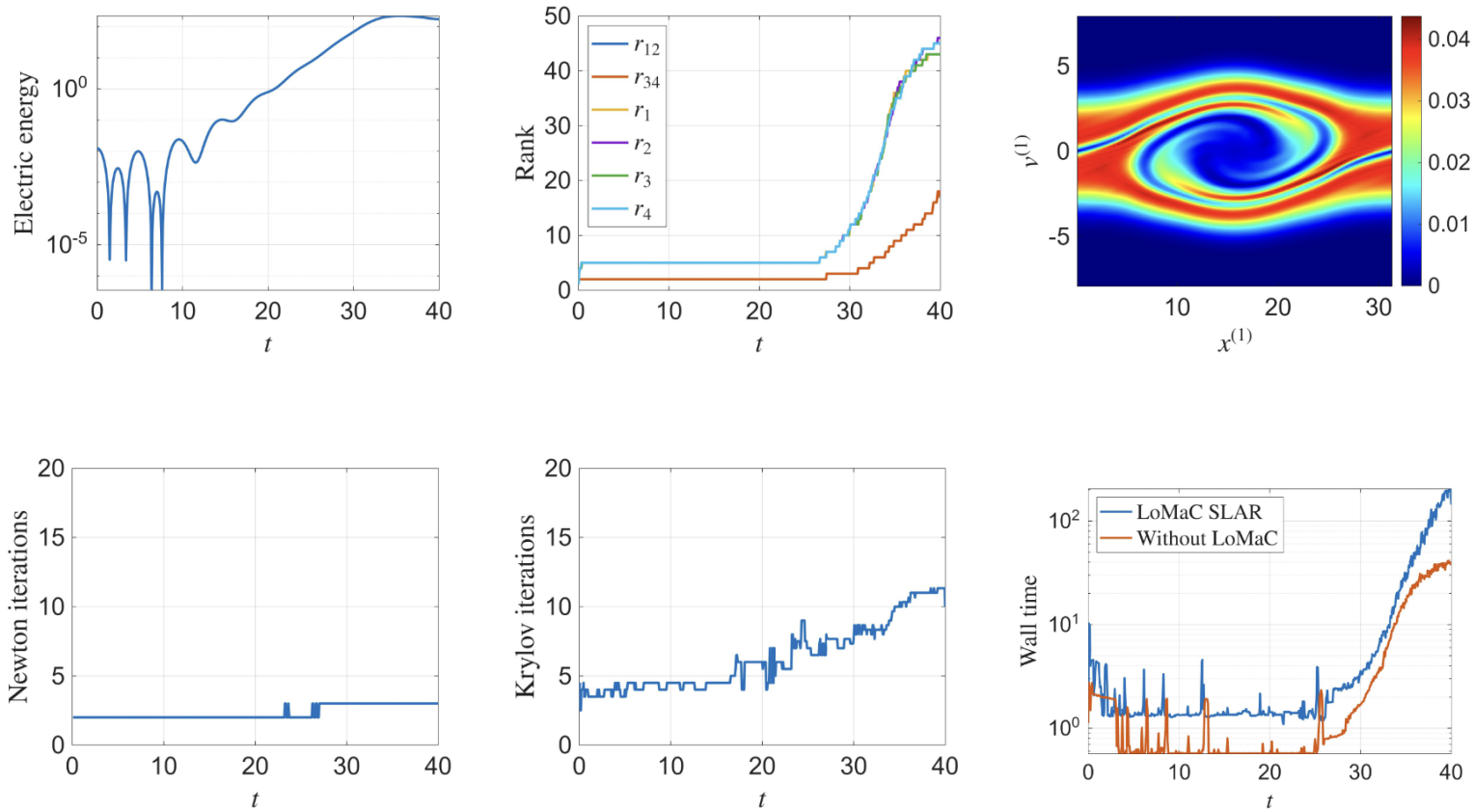
Wall clock time for a single time step update for 3D3V strong Landau damping. Left: computing time vs. N; right: computing time vs. rank r with leaf ranks $3r$, non-leaf ranks $2r$ with $N = 64$. Shared settings: $v_{\max} = 2\pi$. Complexity of the algorithm: $\mathcal{O}(d^4 N r^{3+\lceil \log_2 d \rceil})$.

Two stream instability

$$\frac{1}{(2\sqrt{2\pi})^2}\left(1+\alpha\sum_{\mu=1}^{2}\cos(kx^{(\mu)})\right)\prod_{\mu=1}^{2}\left[\exp\left(-\frac{(v^{(\mu)}-v_0)^2}{2}\right)+\exp\left(-\frac{(v^{(\mu)}+v_0)^2}{2}\right)\right],$$

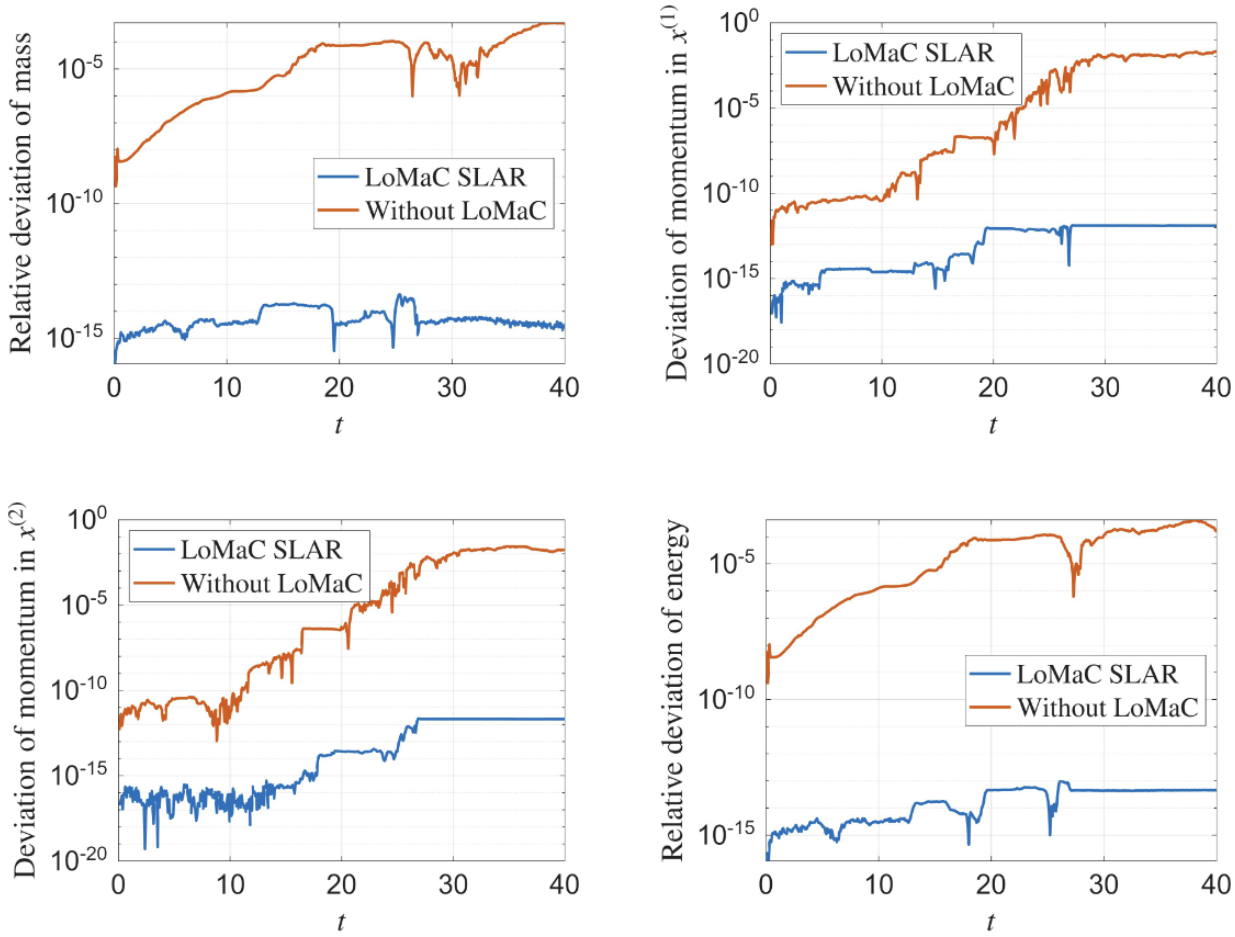
Mesh is $128^2 \times 128^2$, with truncation threshold $\varepsilon_{\text{Base}} = 5e-4$.





Two stream instability: 2D2V. The computations use a mesh of 128^4 with a CFL number of 5 and final time $T = 40$, with the HTACA tolerance set to $\varepsilon_{\text{Base}} = 5 \times 10^{-4}$.

Two stream instability



Macroscopic conservation: SLAR with vs. without LoMaC

Summary and future work

- Accuracy: High order accuracy in spatial and temporal discretizations.
- Efficiency: Low rank complexity in storage and computing time.
- Robustness: Large time stepping sizes with numerical stability.
- Structure preservation: Consistency with local conservation laws.

Compared with Galerkin approach: collocation approach offers more flexibility as only *element-wise function handle* is needed in nonlinear term evaluations and a Maxwellian LoMaC corrections.

Remaining questions/challenges to be addressed:

- robust and efficient sampling scheme
- positivity preservation with efficiency
- extra cost associated with LoMaC due to rank increase

References

1. Hayes, Q. and Shi (2026), An Efficient and Robust Projection Enhanced Interpolation Based Tensor Train Decomposition, arXiv: 2602.07653.
2. Zheng, Sands, Hayes, Christlieb and Q. (2025), A Semi-Lagrangian Adaptive Rank (SLAR) Method for High-Dimensional Vlasov Dynamics, arXiv:2510.24861.
3. Christlieb, Gong, Q. and Zheng (2025), A Sampling-Based Adaptive Rank Approach to the Wigner-Poisson System, SISC.
4. Sands, Q., Hayes, Zheng (2025), An Adaptive-rank Approach with Greedy Sampling for Multi-scale BGK Equations, JCP.
5. Shi, Hayes and Q. (2025) Distributed Memory Parallel Adaptive Tensor-Train Cross Approximation, SISC.
6. Einkemmer, Kormann, Kusch, McClarren and Q. (2025), A Review of Low-rank Methods for Time-Dependent Kinetic Simulations, JCP.
7. El Kahza, Q., Chacón, Taitano (2025), Sylvester-Preconditioned Adaptive-Rank Implicit Time Integrators for Advection-Diffusion Equations with Inhomogeneous Coefficients, JCP.
8. Sands, Guo, Q. Xiong (2025), High-order Adaptive Rank Integrators for Multi-scale Linear Kinetic Transport Equations in the Hierarchical Tucker Format, SISC.
9. Zheng, Hayes, Christlieb, Q. (2025), A Semi-Lagrangian Adaptive-Rank (SLAR) Method for Linear Advection and Nonlinear Vlasov-Poisson System, JCP.
10. Nakao, Q.,Einkemmer (2025), Reduced Augmentation Implicit Low-rank (RAIL) integrators for advection-diffusion and Fokker-Planck models, SISC.
11. Guo and Q. (2024), A Local Macroscopic Conservative (LoMaC) low rank tensor method for the Vlasov dynamics, JSC.
12. El Kahza, Taitano, Q., Chacón (2024), Krylov-based Adaptive-Rank Implicit Time Integrators for Stiff Problems with Application to Nonlinear Fokker-Planck Kinetic Models, JCP.
13. Guo and Q. (2024), A conservative low rank tensor method for the Vlasov dynamics, SISC.
14. Guo, Ema and Q. (2024) A Local Macroscopic Conservative (LoMaC) low rank tensor method with the discontinuous Galerkin method for the Vlasov dynamics, CAMC.
15. Guo and Q. (2022) A Low Rank Tensor Representation of Linear Transport and Nonlinear Vlasov Solutions and Their Associated Flow Maps, JCP.

Questions? Thank you!