Low-rank approximation for nonlinear kinetic problems
Part 1: Introduction and the dynamical low-rank algorithm

Lukas Einkemmer
University of Innsbruck

Summer School 'Wave Phenomena: Analysis and Numerics', KIT, 2021

Link to slides: `http://www.einkemmer.net/training.html`

Newton's law

Newton's law: $F = ma$ with acceleration $a = \ddot{x}$, mass m, and force F.

Multi-particle system:

$$\dot{x}_i(t) = v_i(t), \qquad \dot{v}_i(t) = F(x_i(t))/m_i, \qquad x(t) \in \mathbb{R}^N,\ v(t) \in \mathbb{R}^N.$$

The ith particle is described by position x_i, velocity v_i, and mass m_i.

For practical systems N is extremely large (order of **Avogadro constant** $\approx \mathbf{10^{23}}$).

- Even if we could track the position of each particle this is usually not interesting.
- Macroscopic quantities (density, momentum density, ...) much more important.

Kinetic description

We introduce a **particle-density** $f(t,x,v)$ such that

$$\int_{x_1}^{x_2}\int_{v_1}^{v_2} f(t,x,v)\,\mathrm{d}(x,v) = \text{number of particles with } x\in[x_1,x_2] \text{ and } v\in[v_1,v_2].$$

Number of particles is conserved

$$\partial_t f(t,x,v) + \nabla_{x,v}\cdot\left(f(t,x,v)\begin{bmatrix} v \\ a \end{bmatrix}\right) = 0.$$

Acceleration $a = F/m$ determined by Newton's law. Yields **kinetic equation**

$$\partial_t f(t,x,v) + v\cdot\nabla_x f(t,x,v) + \frac{F}{m}\cdot\nabla_v f(t,x,v) = 0.$$

Often F self-consistenctly couples to f (i.e. F depends on f).

Kinetic description

The force field F is not very useful to model collisions. **Boltzmann equation:**

$$\partial_t f + v \cdot \nabla_x f + \frac{F}{m} \cdot \nabla_v f = C(f).$$

First order hyperbolic equation with

$$\begin{aligned} \text{Transport:} &\qquad f(t,x,v) = f(0, x - vt, v) \\ \text{Acceleration:} &\qquad f(t,x,v) = f(0, x, v - tF/m) \\ \text{Collision:} &\qquad \text{usually only acts in } v. \end{aligned}$$

For **collisionless problems** (i.e. $C(f) = 0$) particles travel along the **characteristics given by Newton's law**.

Fluid models, such as the Euler equation and Navier–Stokes equation, can be derived by assuming $f(t,x,v) \propto \rho \exp(-(v-u)^2/(2T))$, i.e. thermodynamic equilibrium.

Vlasov–Poisson equation

Interest from applications such as Tokamak devices (fusion energy), astrophysical plasmas (space weather, magnetosphere, star formation), radiative transfer, ion thrusters, laser plasma interaction, etc.

- Many large scale codes: GYSELA5D, Vlasiator, ...

In a plasma electromagnetic effects are important: $F = qE/m$.

Vlasov equation in dimensionless form

$$\partial_t f + v \cdot \nabla_x f - E \cdot \nabla_v f = 0$$

coupled to a **Poisson problem** (Gauss's law)

$$E = -\nabla \phi \qquad \text{with} \qquad -\Delta \phi = 1 - \int f \, dv.$$

Vlasov–Maxwell equations include magnetic effects.

Landau damping

Initial value

$$f(0,x,v) = (1+\alpha\cos(kx))\frac{e^{-v^2/2}}{(2\pi)^{d_v/2}}, \qquad d_v \text{ number of dimensions in the } v\text{-direction.}$$

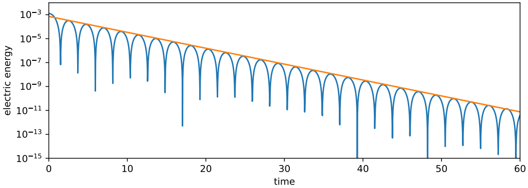
For $\alpha = 0$ we have an equilibrium.

Quantity of interest is the **electric energy** given by $\frac{1}{2}\int E^2\,\mathrm{d}x$.

- Exponential decay is **not** expected for a **hyperbolic problem**.

First described by **Landau in 1946** using linearization.

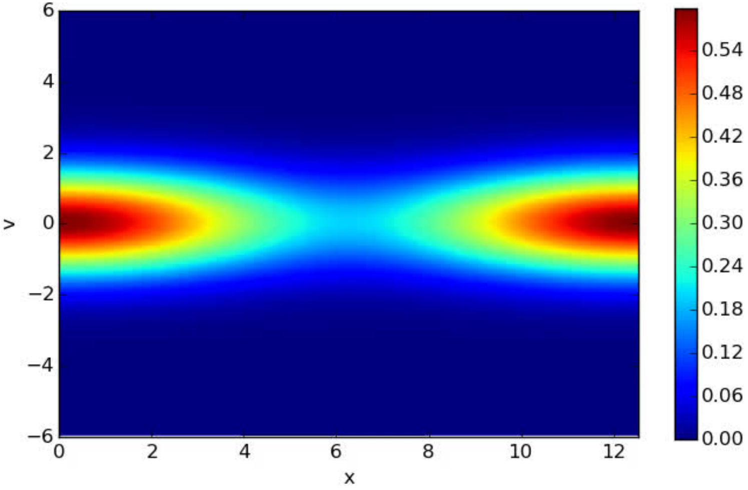
Fields Medal 2010 (Cédric Villani) for proving Landau damping and convergence to equilibrium (for α small).



L. Landau. J. Phys. (USSR) 10 (1946).
C. Mouhot, C. Villani. Acta Math. 207:1 (2011).

Nonlinear Landau damping

To study kinetic dynamics (in almost all situations) requires numerical simulation.



Numerical challenges

The phase space is **up to six-dimensional**.

- $n = 50$ 250 GB memory (workstation)
- $n = 100$ 16 TB memory (local cluster)
- $n = 200$ 1024 TB memory (largest supercomputer)

Small scale structures force a sufficiently fine space discretization.

We need a numerical method

- for which stability is not dictated by $v\tau < h$
- **that does not introduce additional memory requirements**
- **that is scalable to large HPC systems**
- that does not introduce too much numerical diffusion

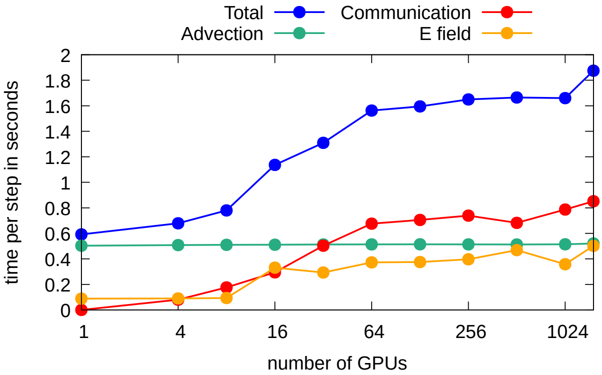
High-performance computing

To obtain results for **five or six-dimensional** problems requires the **largest supercomputers currently available** (perhaps more than that).

Simulation using ≈ 1500 GPUs and $72^3 144^3$ grid points.

JUWELS Booster:

- 2×24 AMD EPYC 7402 cores and $4\times$ NVIDIA A100 GPUs per node.
- Total of 150 TB of GPU memory.
- $4\times$ Mellanox HDR200 InfiniBand ConnectX 6 (200 Gbit/s each).



The wave perspective

Summary of **elementary plasma waves**

EM character	oscillating species	conditions	dispersion relation	name
electrostatic	electrons	$\vec{B}_0 = 0$ or $\vec{k} \parallel \vec{B}_0$	$\omega^2 = \omega_p^2 + 3k^2 v_{th}^2$	plasma oscillation (or Langmuir wave)
		$\vec{k} \perp \vec{B}_0$	$\omega^2 = \omega_p^2 + \omega_c^2 = \omega_h^2$	upper hybrid oscillation
	ions	$\vec{B}_0 = 0$ or $\vec{k} \parallel \vec{B}_0$	$\omega^2 = k^2 v_s^2 = k^2 \frac{\gamma_e K T_e + \gamma_i K T_i}{M}$	ion acoustic wave
		$\vec{k} \perp \vec{B}_0$ (nearly)	$\omega^2 = \Omega_c^2 + k^2 v_s^2$	electrostatic ion cyclotron wave
		$\vec{k} \perp \vec{B}_0$ (exactly)	$\omega^2 = [(\Omega_c \omega_c)^{-1} + \omega_i^{-2}]^{-1}$	lower hybrid oscillation
electromagnetic	electrons	$\vec{B}_0 = 0$	$\omega^2 = \omega_p^2 + k^2 c^2$	light wave
		$\vec{k} \perp \vec{B}_0,\ \vec{E}_1 \parallel \vec{B}_0$	$\frac{c^2 k^2}{\omega^2} = 1 - \frac{\omega_p^2}{\omega^2}$	O wave
		$\vec{k} \perp \vec{B}_0,\ \vec{E}_1 \perp \vec{B}_0$	$\frac{c^2 k^2}{\omega^2} = 1 - \frac{\omega_p^2}{\omega^2} \frac{\omega^2 - \omega_p^2}{\omega^2 - \omega_h^2}$	X wave
		$\vec{k} \parallel \vec{B}_0$ (right circ. pol.)	$\frac{c^2 k^2}{\omega^2} = 1 - \frac{\omega_p^2/\omega^2}{1 - (\omega_c/\omega)}$	R wave (whistler mode)
		$\vec{k} \parallel \vec{B}_0$ (left circ. pol.)	$\frac{c^2 k^2}{\omega^2} = 1 - \frac{\omega_p^2/\omega^2}{1 + (\omega_c/\omega)}$	L wave
	ions	$\vec{B}_0 = 0$		none
		$\vec{k} \parallel \vec{B}_0$	$\omega^2 = k^2 v_A^2$	Alfvén wave
		$\vec{k} \perp \vec{B}_0$	$\frac{\omega^2}{k^2} = c^2 \frac{v_s^2 + v_A^2}{c^2 + v_A^2}$	magnetosonic wave

Landau damping is an example of wave-particle interaction.

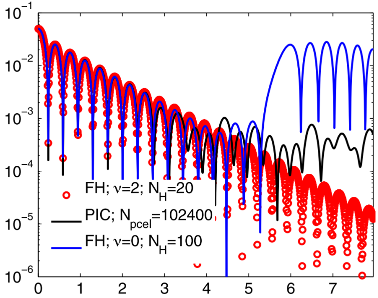
- Particles with matching velocity interact strongly with an electromagnetic wave.

https://en.wikipedia.org/wiki/Waves_in_plasmas

Dimension reduction for the Vlasov equation

Fundamental problem of Eulerian Vlasov solvers is that effort scales as $\mathcal{O}(n^{d_x+d_v})$.

- **Curse of dimensionality**



Particle methods have been employed extensively.

- Only x is discretized.
- Particles push and field solves are alternated.

Sparse grids

- Have problems resolving Gaussians.
- regularity is an issue as $\|\partial_v^m f(t,\cdot,\cdot))\| \propto t^m$.

E. Camporeale et al., 198, Comput. Phys. Commun., 2016.
K. Kormann, E. Sonnendrücker, Sparse Grids and Applications. 2016.

Dynamical low-rank approximation

Singular value decomposition

Singular value decomposition for a matrix $A_{ij} = g(x_i, v_j)$ is given by

$$A = VSW^T \in \mathbb{R}^{n \times m}$$

with $V \in \mathbb{R}^{n \times r}$, $S \in \mathbb{R}^{r \times r}$, $W \in \mathbb{R}^{m \times r}$, and r the rank of A.

Low-rank approximation

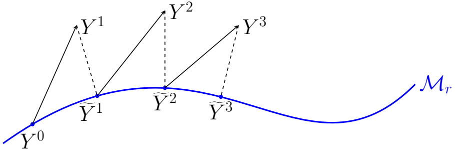
$$g(x, v) \approx \sum_{ij} X_i(x) S_{ij} V_j(v).$$

Orthogonality constraints: $\langle X_i, X_j \rangle_x = \delta_{ij}$, $\langle V_i, V_j \rangle_v = \delta_{ij}$.

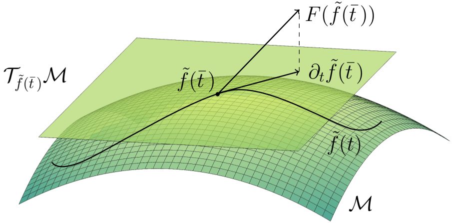
Why do we think that low-rank works any better?

Dynamical low-rank approximation

Traditional approach



Dynamical low-rank approximation

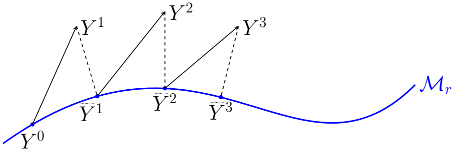


K. Kormann, SIAM J. Sci. Comput., 37(4), 2015.

O. Koch, C. Lubich. SIAM J. Matrix Anal. Appl., 29(2), 2007.

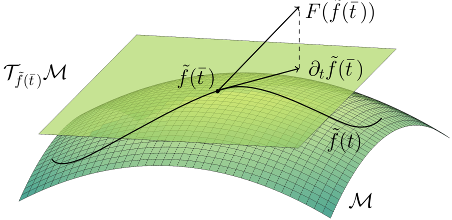
Dynamical low-rank approximation

Traditional approach



Discretize first.
Then low-rank.

Dynamical low-rank approximation



Low-rank first.
Then discretize.

K. Kormann, SIAM J. Sci. Comput., 37(4), 2015.
O. Koch, C. Lubich. SIAM J. Matrix Anal. Appl., 29(2), 2007.

Dynamical low-rank approximation

Dynamical low-rank approximation

$$f(t,x,v) = \sum_{ij} X_i(t,x) S_{ij}(t) V_j(t,v).$$

Low-rank functions (with fixed r) form a **manifold** with functions in the tangent space represented as

$$\dot{f} = \sum_{ij} \left(\dot{X}_i S_{ij} V_j + X_i \dot{S}_{ij} V_j + X_i S_{ij} \dot{V}_j \right).$$

This representation is **not unique**. For example,

$$\dot{X}_i = X_i,\ \dot{S}_{ij} = 0 \qquad \text{and} \qquad \dot{X}_i = 0,\ \dot{S}_{ij} = S_{ij}$$

gives the same vector in the tangent space.

Gauge conditions

We impose the **Gauge conditions** $\langle X_i, \dot{X}_j \rangle_x = 0$ and $\langle V_i, \dot{V}_j \rangle_v = 0$.

Equation for S

$$
\begin{aligned}
\langle X_k V_l, \dot{f} \rangle &= \sum_{ij} \langle X_k V_l, \dot{X}_i S_{ij} V_j + X_i \dot{S}_{ij} V_j + X_i S_{ij} \dot{V}_j \rangle_{xv} \\
&= \sum_{ij} \langle X_k, \dot{X}_i \rangle_x S_{ij} \langle V_l, V_j \rangle_v + \sum_{ij} \langle X_k, X_i \rangle_x \dot{S}_{ij} \langle V_l, V_j \rangle_v + \sum_{ij} \langle X_k, X_i \rangle_x S_{ij} \langle V_l, \dot{V}_j \rangle_v \\
&= \dot{S}_{kl}
\end{aligned}
$$

Equation for X

$$
\begin{aligned}
\langle V_l, \dot{f} \rangle &= \sum_{ij} \langle V_l, \dot{X}_i S_{ij} V_j + X_i \dot{S}_{ij} V_j + X_i S_{ij} \dot{V}_j \rangle_{xv} \\
&= \sum_{ij} \dot{X}_i S_{ij} \langle V_l, V_j \rangle_v + \sum_{ij} X_i \dot{S}_{ij} \langle V_l, V_j \rangle_v + \sum_{ij} X_i S_{ij} \langle V_l, \dot{V}_j \rangle_v \\
&= \sum_i \dot{X}_i S_{il} + \sum_i X_i \dot{S}_{il}
\end{aligned}
$$

Dynamical low-rank approximation

Equations of motion

$$\begin{aligned}
\partial_t S_{ij} &= \langle X_i V_j, \mathrm{RHS} \rangle, && \text{ODE} \\
\sum_i S_{ij}(\partial_t X_i) &= \langle V_j, \mathrm{RHS} \rangle - \sum_i X_i(\partial_t S_{ij}), && \text{x dependent PDE} \\
\sum_j S_{ij}(\partial_t V_j) &= \langle X_i, \mathrm{RHS} \rangle - \sum_j (\partial_t S_{ij}) V_j. && \text{v dependent PDE}
\end{aligned}$$

In principle we can substitute

$$\mathrm{RHS} = -v \cdot \nabla_x f + E(f) \cdot \nabla_v f.$$

But

- The equations couple S, X, and V;
- To obtain equations in X_i and V_j we have to invert S and S^{T}.

Robustness

Back to the SVD

$$A \approx VSW^T.$$

Approximation by truncation

$$S = \begin{bmatrix} \mu_1 & 0 & 0 & 0 & 0 \\ 0 & \mu_2 & 0 & 0 & 0 \\ 0 & 0 & \mu_3 & 0 & 0 \\ 0 & 0 & 0 & \mu_4 & 0 \\ 0 & 0 & 0 & 0 & \mu_5 \end{bmatrix} \approx \begin{bmatrix} \mu_1 & 0 & 0 \\ 0 & \mu_2 & 0 \\ 0 & 0 & \mu_3 \end{bmatrix}.$$

Error vs condition number

- If μ_4 is large than the error is large.
- If μ_3 is small than inverting S is ill-conditioned.

Projector splitting integrator

Vlasov–Poisson equation **constrained to the low-rank manifold**

$$\partial_t f = P(f)\mathrm{RHS} = P(f)\left(-v \cdot \nabla_x f + E(f) \cdot \nabla_v f\right),$$

where $P(f)$ is the orthogonal projector onto the tangent space.

We have

$$\begin{aligned}
P(f)\mathrm{RHS} &= \sum_{ij} \left(\partial_t X_i S_{ij} V_j + X_i \partial_t S_{ij} V_j + X_i S_{ij} \partial_t V_j\right) \\
&= \sum_{ij} \left(\partial_t (X_i S_{ij}) V_j - X_i \partial_t S_{ij} V_j + X_i \partial_t (S_{ij} V_j)\right) \\
&= \sum_j \langle V_j, \mathrm{RHS}\rangle_x V_j - \sum_{ij} X_i \langle X_i V_j, \mathrm{RHS}\rangle_{xv} V_j + \sum_i X_i \langle X_i, \mathrm{RHS}\rangle_v
\end{aligned}$$

Projector splitting integrator

We can write

$$P(f)g = P_{\overline{V}}g - P_{\overline{V}}P_{\overline{X}}g + P_{\overline{X}}g,$$

where $P_{\overline{X}}$ and $P_{\overline{V}}$ are the orthogonal projectors on $\overline{X} = \mathrm{span}\{X_i : i = 1 \ldots r\}$ and $\overline{V} = \mathrm{span}\{V_j : j = 1 \ldots r\}$.

This suggests a **splitting**.

C. Lubich and I.V. Oseledets. BIT Numer. Math. 54(1) 2014.

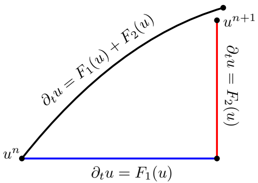
Splitting

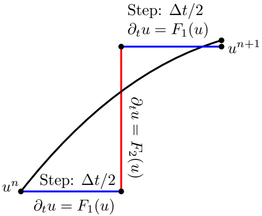
We consider

$$\partial_t u(t) = F_1(u(t)) + F_2(u(t)), \qquad u(0) = u^0,$$

where F_1 and F_2 could describe **different physics**, **different timescales**, **different coordinate axis, . . .**

$$u^{n+1} = \varphi^{F_2}_{\Delta t} \circ \varphi^{F_1}_{\Delta t}(u^n) \textbf{ (Lie)} \qquad u^{n+1} = \varphi^{F_1}_{\Delta t/2} \circ \varphi^{F_2}_{\Delta t} \circ \varphi^{F_1}_{\Delta t/2}(u^n) \textbf{ (Strang)}$$





Fundamental idea of splitting is that **only subflows have to be solved**.

K step

Our goal is to solve

$$\partial_t f = P_{\overline{V}}\left(-v \cdot \nabla_x f + E(f) \cdot \nabla_v f\right).$$

We rewrite the solution using K_j as follows

$$f(t,x,v) = \sum_j K_j(t,x) V_j(t,v), \qquad \text{with} \quad K_j(t,x) = \sum_i X_i(t,x) S_{ij}(t).$$

This yields

$$\begin{aligned}\sum_j \partial_t K_j(t,x) V_j(t,v) &+ \sum_j K_j(t,x) \partial_t V_j(t,v) \\ &= \sum_j \left\langle V_j(t,\cdot), v \mapsto -v \cdot \nabla_x f(t,x,v) + E(f)(t,x) \cdot \nabla_v f(t,x,v) \right\rangle_v V_j(t,v).\end{aligned}$$

K step

The solution is given by $V_j(t,v) = V_j(0,v)$ and

$$\begin{aligned}\partial_t K_j(t,x) &= \left\langle V_j, v \mapsto v \cdot \nabla_x f(t,x,v) + E(f)(t,x) \cdot \nabla_v f(t,x,v) \right\rangle_v \\ &= -\sum_l \langle V_j v V_l \rangle_v \cdot \nabla_x K_l(t,x) + \sum_l E \cdot \langle V_j \nabla_v V_l \rangle_v K_l(t,x)\end{aligned}$$

For the **first subflow of the projector splitting algorithm** we thus obtain

$$\partial_t K_j(t,x) = -\sum_l c^1_{jl} \cdot \nabla_x K_l(t,x) + \sum_l c^2_{jl} \cdot E(K)(t,x) K_l(t,x),$$

The coefficients are determined as follows ($V = V^0$)

$$c^1_{jl} = \int_{\Omega_v} v V^0_j V^0_l \,\mathrm{d}v, \quad c^2_{jl} = \int_{\Omega_v} V^0_j (\nabla_v V^0_l) \,\mathrm{d}v.$$

Do not neglect the cost of computing the coefficients.

K step

The equation is formulated with K and V (neither X nor S are explicitly involved).

To proceed with the next step in the algorithm we have to obtain X and S.

- **Why is this approach then advantageous?**

The X and S are recovered from K by a **QR decomposition** as

$$K_j = \sum_i X_i S_{ij}$$

Well defined even for singular $K = [K_1, \ldots, K_r]$ and gives automatically the (almost correct) orthogonality relation for the X_i.

- Result is a robust approximation even if the rank r is chosen *too large*.

Note that S is not necessarily diagonal.

S step

Our goal is to solve

$$\partial_t f = -P_{\overline{V}} P_{\overline{X}} \left(-v \cdot \nabla_x f + E(f) \cdot \nabla_v f \right).$$

The solution is $X_i(t,x) = X_i(0,x)$, $V_j(t,v) = V_j(0,v)$, and

$$\begin{aligned} \partial_t S_{ij} &= \left\langle X_i^1 V_j^0, (x,v) \mapsto \left(v \cdot \nabla_x - E(S)(t,x) \cdot \nabla_v \right) \sum_{kl} X_k^1(x) S_{kl}(t) V_l^0(v) \right\rangle_{xv} \\ &= \sum_{kl} \left(c_{jl}^1 \cdot d_{ik}^2 - c_{jl}^2 \cdot d_{ik}^1[E(S(t))] \right) S_{kl}(t) \end{aligned}$$

with

$$d_{ik}^1[E] = \int_{\Omega_x} X_i^1 E X_k^1 \,\mathrm{d}x, \qquad d_{ik}^2 = \int_{\Omega_x} X_i^1 (\nabla_x X_k^1) \,\mathrm{d}x.$$

The S step integrates backward in time.

Electric field (S step)

The electric field $E(S(t))$ is computed from

$$-\Delta\phi = 1 - \sum_{ij} X_i^1(x) S_{ij}(t) \int V_j^0 \, \mathrm{d}v, \qquad\qquad E = -\nabla\phi.$$

In practice we usually approximate E by E^n (first order) or $E^{n+1/2}$ (second order).

- $E^{n+1/2}$ has to be approximated (to first order) in an actual implementation.

L step

Our goal is to solve

$$\partial_t f = P_{\overline{X}}\left(-v \cdot \nabla_x f + E(f) \cdot \nabla_v f\right).$$

We define

$$f(t,x,v) = \sum_i X_i(t,x) L_i(t,v), \qquad \text{with} \quad L_i(t,v) = \sum_j S_{ij}(t) V_j(t,v).$$

The solution is $X_i(t,x) = X_i(0,x)$ and

$$\begin{aligned} \partial_t L_i(t,v) &= \left\langle X_j^1, x \mapsto \left(-v \cdot \nabla_x + E(L)(t,x) \cdot \nabla_v\right) \sum_k X_k^1 L_k(t,v) \right\rangle_x \\ &= \sum_k d_{ik}^1 [E(L(t,\cdot))] \cdot \nabla_v L_k(t,v) - \sum_k (d_{ik}^2 \cdot v) L_k(t,v). \end{aligned}$$

Then S and V are recovered from L by a **QR decomposition**.

Dynamical low-rank algorithm

First order Lie splitting

1. Solve $\partial_t K_j = -\sum_l c^1_{jl} \cdot \nabla_x K_l + \sum_l c^2_{jl} \cdot E(K) K_l$ with initial value $\sum_i X^0_i S^0_{ij}$ up to time Δt to obtain K^1_j.
2. Perform a QR decomposition of K^1_j to obtain X^1_i and S^1_{ij}.
3. Solve $\partial_t S_{ij} = \sum_{kl} \left(c^1_{jl} \cdot d^2_{ik} - c^2_{jl} \cdot d^1_{ik} \right) S_{kl}$ with initial value S^1_{ij} up to time Δt to obtain S^2_{ij}.
4. Solve $\partial_t L_i = \sum_k d^1_{ik} \cdot \nabla_v L_k - \sum_k (d^2_{ik} \cdot v) L_k$ equation with initial value $\sum_j S^2_{ij} V^0_j$ up to time Δt to obtain L^1_i.
5. Perform a QR decomposition of L^1_i to obtain V^1_j and S^3_{ij}.

Spectral and **semi-Lagrangian** methods can still be employed.

L.E., C. Lubich, SIAM J. Sci. Comput. 40(5), 2018.

Computational complexity

Computational complexity: $\mathcal{O}(r^2 n^d)$ instead of $\mathcal{O}(n^{d_x+d_v})$.

- Limited by computation of the coefficients and solution of evolution equations.

Memory usage: $\mathcal{O}(rn^d)$ instead of $\mathcal{O}(n^{d_x+d_v})$.

- Limited by storage of X_i and V_j.

Coefficients:	$c^1_{jl} = \int_{\Omega_v} v V^0_j V^0_l \,\mathrm{d}v$	Storage: $\mathcal{O}(r^2)$	Effort: $\mathcal{O}(r^2 n^{d_v})$
Integration:	$\partial_t K_j = \ldots$	Storage: $\mathcal{O}(rn^{d_x})$	Effort: $\mathcal{O}(r^2 n^{d_x})$

$d = \max(d_x, d_v)$.

Implementation

Discretized system

$$f = XSV^{\mathsf{T}}$$

with

$$f_{kl} = f(t, x_k, v_l), \qquad X_{ki} = X_i(t, x_k), \qquad V_{lj} = V_j(t, v_l).$$

In **matrix form**

$$X(t) = \begin{bmatrix} X_1(t, x_1) & \cdots & X_r(t, x_1) \\ \vdots & \ddots & \vdots \\ X_1(t, x_n) & \cdots & X_r(t, x_n) \end{bmatrix}, \qquad V(t) = \begin{bmatrix} V_1(t, v_1) & \cdots & V_r(t, v_1) \\ \vdots & \ddots & \vdots \\ V_1(t, v_m) & \cdots & V_r(t, v_m) \end{bmatrix}.$$

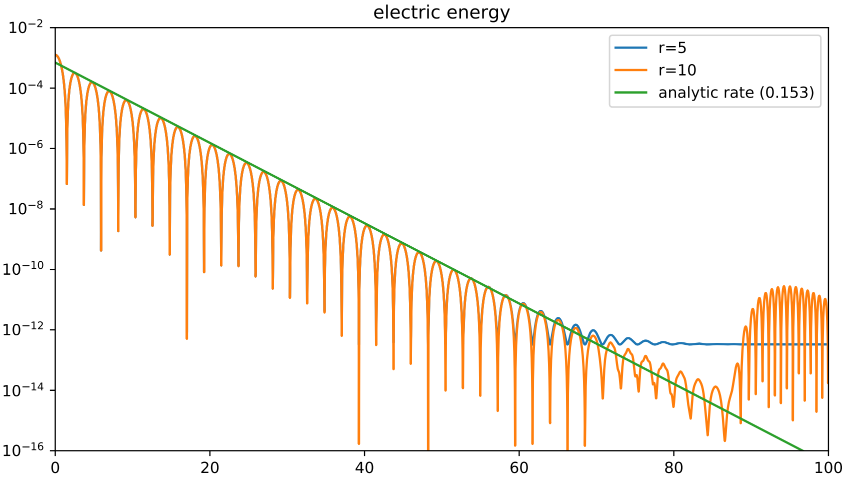
K step (one-dimensional case)

$$\partial_t K = -A_{\partial_x} K (c^1)^T + \mathrm{diag}(E^n) K (c^2)^T,$$

where A_{∂_x} is the discretization of the spatial derivative.

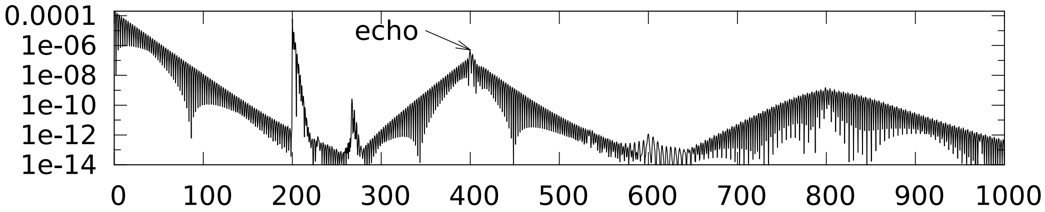
Linear Landau damping

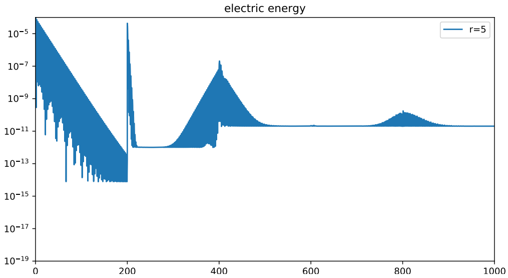
Low-rank approximation with 256 grid points in each direction.

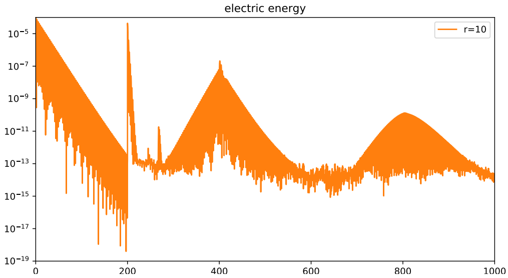


Plasma echo

Plasma echo with 512×4096 grid points.

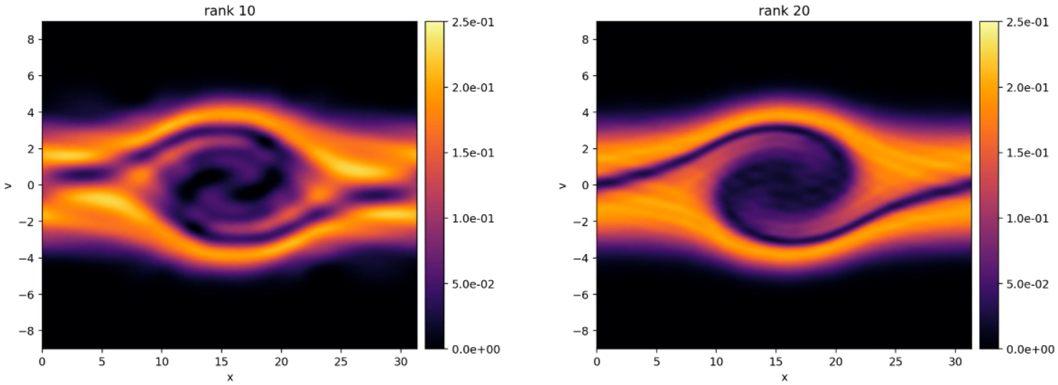






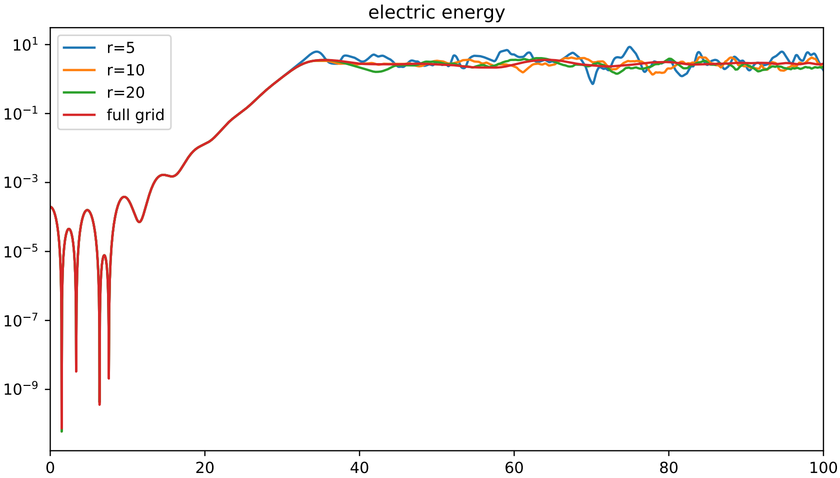
Two-stream instability

Low-rank approximation with 512 grid points per direction ($r = 10$ left, $r = 20$ right).



Two-stream instability

Time evolution of the electric energy.



Why does dynamical low-rank work?

Linear regime

Why does low-rank work so well in the linear regime?

We consider a small perturbation around the equilibrium $f^{(0)}(v)$

$$f(t,x,v) = f^{(0)}(v) + f^{(1)}(t,x,v), \qquad E(t,x) = 0 + E^{(1)}(t,x).$$

This results in the **linearized Vlasov equation**

$$\partial_t f^{(1)}(t,x,v) + v \cdot \nabla_x f^{(1)}(t,x,v) + E^{(1)}(x) \cdot \nabla_v f^{(0)}(v) = 0,$$

where we have dropped the second order term $E^{(1)}(x) \cdot \nabla_v f^{(1)}(v)$.

Fourier transforme (in x) the **Vlasov–Poisson** equation

$$\partial_t \hat{f}_k^{(1)}(t,v) + iv \cdot k \hat{f}_k^{(1)}(t,v) + \hat{E}_k^{(1)} \cdot \nabla_v f^{(0)}(v) = 0,$$

$$\hat{E}_k^{(1)} = -\frac{k}{k^2} \int \hat{f}_k^{(1)}(t,v)\,\mathrm{d}v, \qquad k \neq 0.$$

Linear regime

Now let us assume

$$f(0,x,v) = f^{(0)}(v) + \sum_{k=1}^{m} \hat{f}_{k_i}^{(1)}(0,v)\mathrm{e}^{ik_i x}.$$

E.g. Landau damping with $m = 2$ (rank 1).

Since the linear problem does not excite any new Fourier modes

$$f(t,x,v) = f^{(0)}(v) + \sum_{k=1}^{m} \hat{f}_{k_i}^{(1)}(t,v)\mathrm{e}^{ik_i x}$$

which is at most rank $m + 1$.

Our low-rank algorithm is **more general than the previous analysis suggests** (i.e. in general $X_i(t,x) \neq \mathrm{e}^{ikx}$).

- The low-rank algorithm **captures saturation** perfectly.

L.E., A. Ostermann, C. Piazzola, J. Comput. Phys. 403, 2020.

Smoothness

The low-rank algorithm is able to **resolve filamentation**. Consider

$$\partial_t f(t,x,v) + v \cdot \nabla_x f(t,x,v) = 0, \qquad f(0,x,v) = \mathrm{e}^{ikx}\mathrm{e}^{-v^2}.$$

Then

$$f(t,x,v) = \mathrm{e}^{ik(x-vt)}\mathrm{e}^{-v^2} = \mathrm{e}^{ikx}\mathrm{e}^{-ikvt}\mathrm{e}^{-v^2}.$$

This is still rank 1.

Thus, **smoothness in v is not necessary** for low-rank approximations.

Literature

Literature

[O. Koch, C. Lubich. SIAM J. Matrix Anal. Appl., 29(2), 2007]

- Dynamical low-rank algorithm for ODEs in the matrix case.

[C. Lubich, I.V. Oseledets. BIT Numer. Math. 54(1) 2014]

- Projector splitting to obtain a robust integrators (ODE case).

[L.E., C. Lubich, SIAM J. Sci. Comput. 40(5), 2018]

- Projector splitting based dynamical low-rank algorithm for Vlasov–Poisson.
- Probably the best starting point to get more details.

[L.E., A. Ostermann, C. Piazzola, J. Comput. Phys. 403, 2020]

- Extension to Vlasov–Maxwell which respects the divergence constraint.
- Low-rank structure for the linear Vlasov–Maxwell equations.