NuFI: The numerical flow iteration for the Vlasov–Poisson equation

Rostislav-Paul Wilhelm
Matthias Kirchhart

December 12, 2022

Applied and Computational Mathematics | RWTH AACHEN UNIVERSITY

1. Vlasov–Poisson equation
2. NuFI
3. Numerical experiments
4. Conclusion

Vlasov–Poisson equation

The Vlasov–Poisson equation

Kinetic model for collective electron behaviour

$$\partial_t f + v \cdot \nabla_x f - \underbrace{E \cdot \nabla_v f}_{\text{Neglect magnetic field.}} = \underbrace{0}_{\substack{\text{Neglect collisions:}\\ Q(f,f)=0.}}$$

The Poisson equation for E reads:

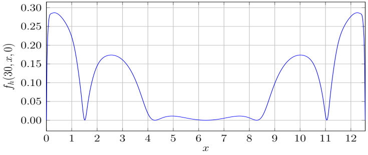
$$\begin{aligned} E(t,x) &= -\nabla_x \varphi(t,x), \\ \Delta_x \varphi(t,x) &= -\rho(t,x) \\ \rho(t,x) &= 1 - \int_{\mathbb{R}^d} f(t,x,v)\mathrm{d}v. \end{aligned}$$

Problems with solving Vlasov equation numerically

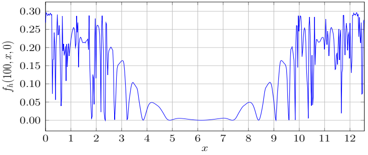
$$\partial_t f + v \cdot \nabla_x f - E \cdot \nabla_v f = 0$$

$$f : \mathbb{R}_{\geq 0} \times \mathbb{T}^d \times \mathbb{R}^d \to \mathbb{R}_{\geq 0}$$

- **High-dimensional** equation: Full Vlasov equation for 1 species is a 7-dimensional PDE.
  - ➢ Grid-based solvers suffer from extensive memory-usage.



$t = 30$

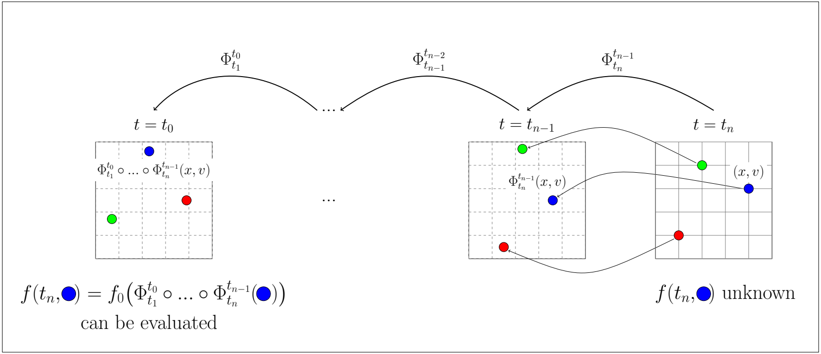


$t = 100$

- **Lack of diffusion** in the model leads to strong filamentation in the solution for long-times.
  - ➢ Steep gradients ⇒ Practically impossible to resolve.
  - ➢ DSMC/SPH/PIC require high resolution and remeshing.

NuFI

Motivation


$\Phi_{t_1}^{t_0}$
$\Phi_{t_{n-1}}^{t_{n-2}}$
$\Phi_{t_n}^{t_{n-1}}$
...
$t = t_0$
$t = t_{n-1}$
$t = t_n$
$\Phi_{t_1}^{t_0} \circ \ldots \circ \Phi_{t_n}^{t_{n-1}}(x,v)$
...
$\Phi_{t_n}^{t_{n-1}}(x,v)$
(x,v)
$f(t_n,$●$) = f_0\big(\Phi_{t_1}^{t_0} \circ \ldots \circ \Phi_{t_n}^{t_{n-1}}($●$)\big)$
can be evaluated
$f(t_n,$●$)$ unknown

Note:

$$\partial_t f + \left(\begin{pmatrix} v \\ -E \end{pmatrix} \cdot \nabla_{(x,v)} \right) f = 0 \quad \Rightarrow \quad f(t,x,v) = f_0(\Phi_t^0(x,v))$$

Where:

$$\frac{\mathrm{d}}{\mathrm{d}t} \Phi_t^s(x,v) = a(t,x,v), \quad \Phi_s^s(x,v) = (x,v)$$

Approximating the phase-flow

Iterative approximation of $\Phi_{t_k}^{t_{k-1}}$ via **symplectic Euler**:

$$\boxed{\begin{aligned} x_{k-1} &:= x_k - \Delta t \cdot v_k, \\ v_{k-1} &:= v_k + \Delta t \cdot E\big(t_{k-1}, x_{k-1}\big), \end{aligned}}$$

Electric field is known at **previous time-step**!

For **nonlinear case**, $t = t_k$:

1. Compute ρ_k using numerical quadrature.
2. Compute electric potential φ_k from ρ_k using FFT.
3. Save coefficients of φ_k in B-Spline basis.

Repeat for $t_k \rightarrow t_{k+1}$!

Memory usage

No need to save f!

- Saving φ_h.
- Shift workload from "memory access" to "computation on the fly":
  - ➢ High flop/Byte ratio.
  - ➢ Cache can store coefficients of φ_h for several time-steps.

$$\boxed{\begin{aligned} \text{Memory Requirement }\textbf{Grid} &= N_x^d \times N_v^d \times \text{size of one floating point value} \\ \text{Memory Requirement }\textbf{NuFI} &= n \times N_x^d \times \text{size of one floating point value} \end{aligned}}$$

Where n number of time-steps: $n \ll N_v^d$.

d	$N_x = N_v$	Memory of φ_h per step	Memory for storing f
2	128	0.125 MiB	$\approx 2.0 \cdot 10^3$ MiB
2	256	0.5 MiB	$\approx 3.3 \cdot 10^4$ MiB
2	512	2 MiB	$\approx 5.2 \cdot 10^5$ MiB
3	128	0.015 GiB	$\approx 3.3 \cdot 10^4$ GiB
3	256	0.125 GiB	$\approx 2.1 \cdot 10^6$ GiB
3	512	1 GiB	$\approx 1.3 \cdot 10^8$ GiB

Conservation properties of NuFI

$$\boxed{f_h(t,x,v) = f_0(\Phi^0_{t,h}(x,v))}$$

- Satisfies the **maximum principle** exactly

$$0 \le f_h(t,x,v) \le \|f_0\|_{L^\infty(\Omega\times\mathbb{R}^d)}$$

- **Exact** conservation of all L^p**-norms**

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\int f_h\,(t,x,v)^p\,\mathrm{d}x\mathrm{d}v\right) = 0$$

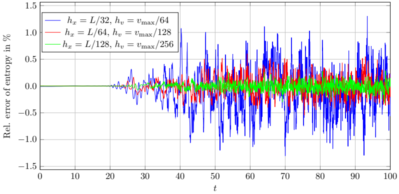
  and the **kinetic entropy**

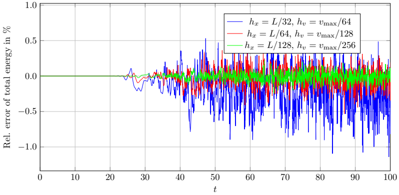
$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\int_{\mathbb{R}^d}\int_{\mathbb{R}^d} f_h\log(f_h)\mathrm{d}x\mathrm{d}v\right) = 0.$$

- **Numerical proof** for conservation of **total energy**

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\int_{\mathbb{R}^d}\int_{\mathbb{R}^d} v^2 f_h\mathrm{d}x\mathrm{d}v + \int_{\mathbb{R}^d} E_h^2\mathrm{d}x\right) = 0.$$

Relative errors - computed using **numerical quadrature**:

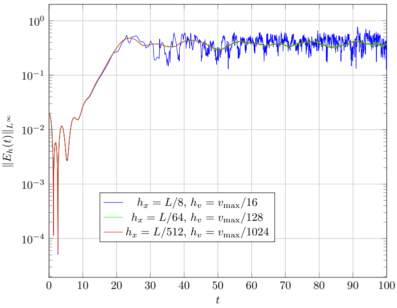




Numerical experiments

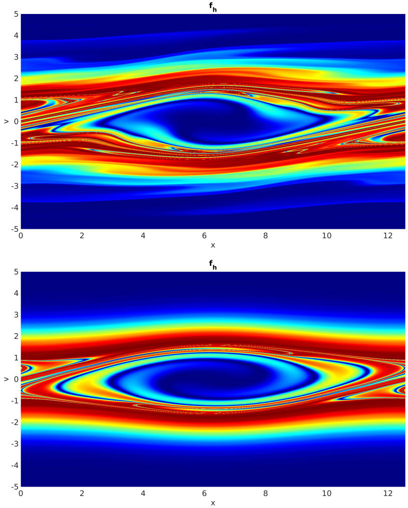
Two-Stream instability

$$f_0(x,v) = \frac{1}{\sqrt{2\pi}} e^{-\frac{v^2}{2}} v^2 \left(1 + 0.01\cos(0.5x)\right)$$



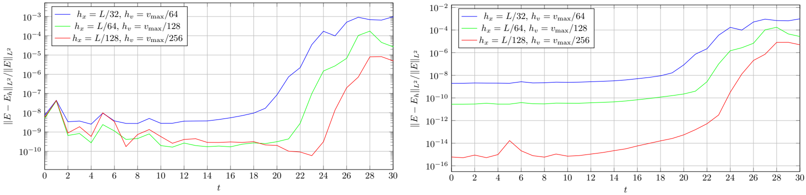
- ➢ Dynamics captured correctly.
- ➢ Low resolution leads to distortion.

t = 50

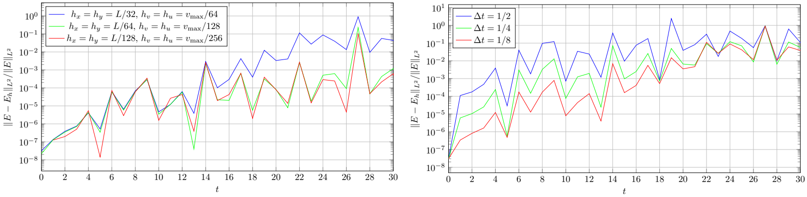


Errors

Two Stream 1d ($h \to 0$, Δt fixed): Single- (left) & double precision (right).

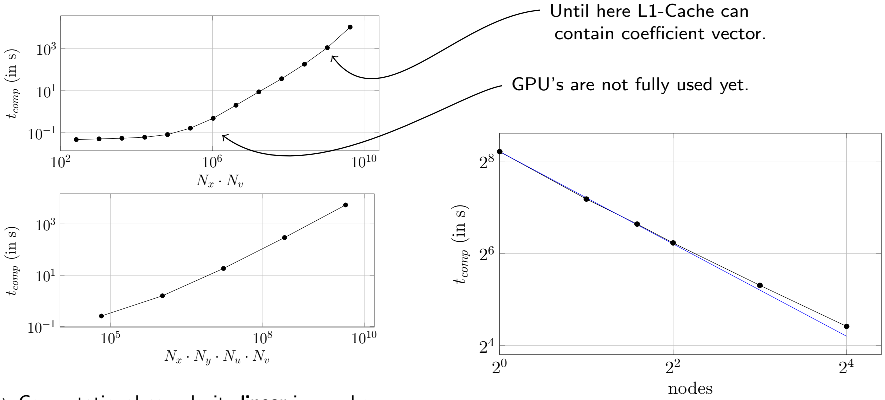


Two Stream 2d (single precision): $h \to 0$ left & $\Delta t \to 0$ right.



Computational complexity and scaling

Two-stream benchmark: $d = 1$ (left) & $d = 2$ (right). $N_T = 481$ time-steps.



Until here L1-Cache can contain coefficient vector.

GPU's are not fully used yet.

⇒ Computational complexity **linear** in number of quadrature points!

⇒ **Strong scaling.**

Conclusion

Summary

Conclusion:

- All DOF in phase-space are independent of each other!
  - ➢ Embarrassingly parallel algorithm.
- Very low memory usage.
  - ➢ **Never save f!**
- Comes at cost of increasing complexity with number of time-steps.
- Exact conservation of properties.

Future research:

- Enabling restart of simulation & starting from experimental data.
- Collision-handling. $\longrightarrow$ Evaluation of $Q(f,f)$ still expensive.

Wilhelm, R.-P., ACoM RWTH Aachen, wilhelm@acom.rwth-aachen.de
Kirchhart, M., ACoM RWTH Aachen and Intel Co., matthias.kirchhart@intel.com