Adaptive physics refinement in a Vlasov-Maxwell code on GPUs

Martin Rieke, Ruhr-Universität Bochum

Programming for GPUs with CUDA

+ Feels like C/C++ or FORTRAN
+ Many libraries available
+ Enormous speedups
+ Easy to code

- Hard to get it right
- Strict rules for memory access
- Not suitable for all algorithms
- Limits on size / speed of memory

The Vlasov-Maxwell system of equations:

Vlasov:

$$\partial_t f_s + \mathbf{v} \cdot \nabla_{\mathbf{x}} f_s + \frac{q_s}{m_s} \left(\mathbf{E} + \frac{\mathbf{v} \times \mathbf{B}}{c} \right) \cdot \nabla_{\mathbf{v}} f_s = 0$$

Problems for the computer:

- 6D → memory consumption
- Physical properties of f → complicated numerics

The Vlasov-Maxwell system of equations:

Maxwell:

$$\partial_t \mathbf{B} = -\nabla \times \mathbf{E} \qquad \nabla \cdot \mathbf{E} = \frac{\rho}{\epsilon_o}$$

$$\partial_t \mathbf{E} = c^2 \left(\nabla \times \mathbf{B} - \mu_0 \mathbf{j} \right) \qquad \nabla \cdot \mathbf{B} = 0$$

Speed of light → Implicit scheme or small timestep

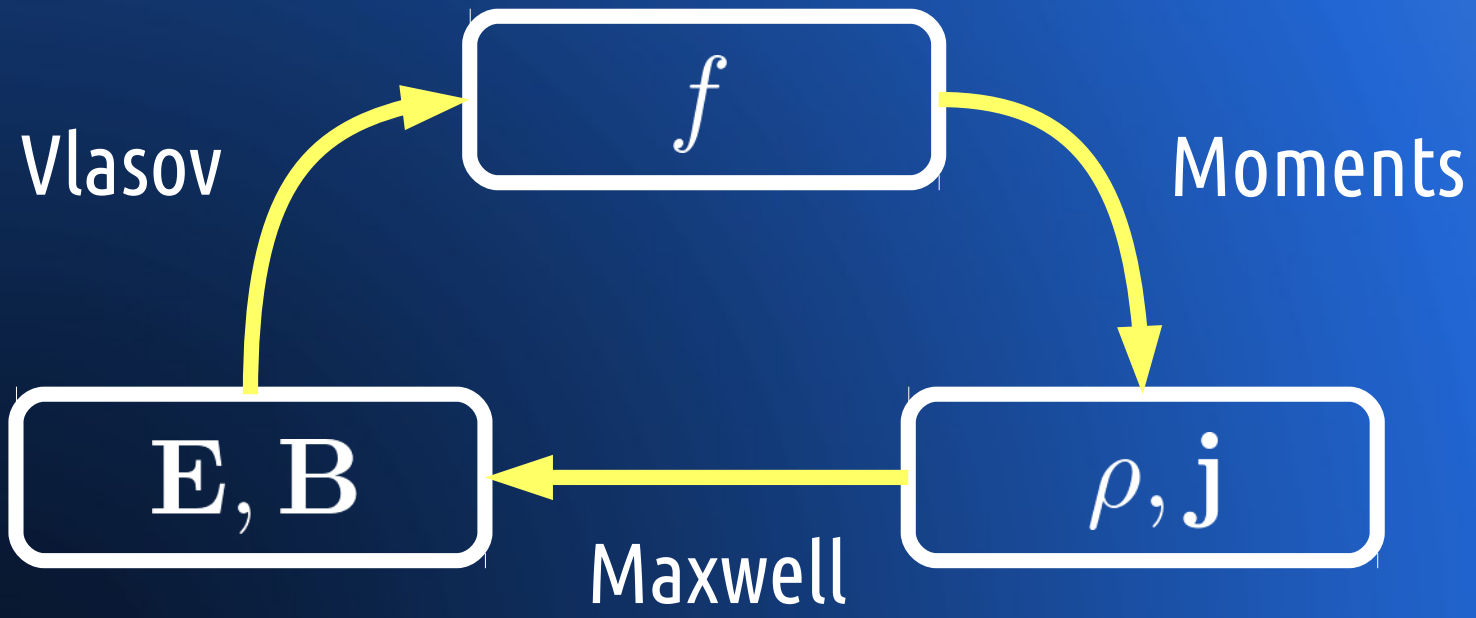
The Vlasov-Maxwell system of equations:

Moments of f :

$$\rho = \sum_s q_s \int f_s \mathrm{d}^3 v \qquad \mathbf{j} = \sum_s q_s \int \mathbf{v} f_s \mathrm{d}^3 v$$

Sources for Maxwell → Satisfy continuity eqn.

The Vlasov-Maxwell system of equations:



Requirements for the algorithm

- f is a probability density
  → should be conserved
  → should be positive
- Code will run on GPUs
  → should be local in memory access
- f is very large
  → scheme should not need temp fields

Semi-Lagrangian schemes

Hyperbolic PDE $\partial_t f + \partial_x u f = 0$

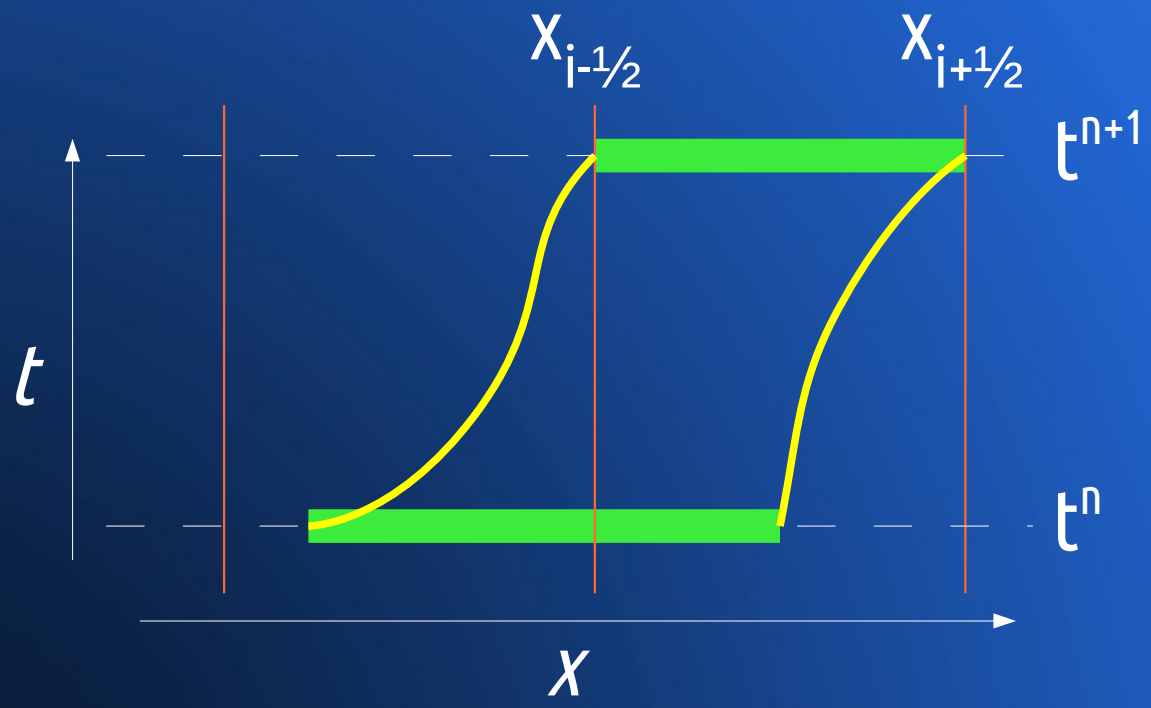
Characteristics $\frac{\mathrm{d}X}{\mathrm{d}s}(s) = u(s, X(s))$

$X(t) = x$

Solution: $X(s, t, x)$

$\Rightarrow f$ is solution to ODE along characteristics

Semi-Lagrangian schemes



$$\int_{x_{i-}}^{x_{i+}} f(t^{n+1},x)\mathrm{d}x = \int_{X(t^n,t^{n+1},x_{i-})}^{X(t^n,t^{n+1},x_{i+})} f(t^n,x)\mathrm{d}x$$

Semi-Lagrangian schemes

Reconstruction using *PFC* scheme:

I. Build polynomial approximation to cummulative f_s

II. Take derivative

III. Limit the slope to avoid negative f_s

Local stencil, conserves density,
no negative values, third-order in space

F. Filbet et al. 2001, JCP

Splitting

Consider $\partial_t f = \mathcal{A} f + \mathcal{B} f$

With $\mathcal{A}, \mathcal{B}$ linear, time-independent

Solution: $f(t) = \exp\left((\mathcal{A} + \mathcal{B})t\right) f(0)$

If $[\mathcal{A}, \mathcal{B}] = 0$

$f(t) = \exp(\mathcal{B}t) \exp(\mathcal{A}t) f(0)$

Splitting

Consider $\partial_t f = \mathcal{A} f + \mathcal{B} f$

What if $[\mathcal{A}, \mathcal{B}] \neq 0$?

$$f(t^{n+1}) = \exp(\mathcal{B}\tau) \exp(\mathcal{A}\tau) f(t^n) + \mathcal{O}(\tau^2)$$

better: Strang-Splitting

$$f(t^{n+1}) = \exp(\mathcal{A}\tau/2) \exp(\mathcal{B}\tau) \exp(\mathcal{A}\tau/2) f(t^n) + \mathcal{O}(\tau^3)$$

G. Strang 1968, SIAM JNA

Splitting

Construction of arbitrary order splitting using *Baker-Campbell-Hausdorff*

Sheng-Suzuki-Theorem:
"No splitting better than Strang's is possible without negative time-steps."

H. Yoshida 1990, Physics Letters A -- Q. Sheng 1989, IMA J.Num.Ana. -- M. Suzuki 1991, J.Math.Phys.

Splitting Vlasov

Keep $\mathbf{E}, \mathbf{B}$ constant over time τ

$$\partial_t f_s + \underbrace{\mathbf{v} \cdot \nabla_{\mathbf{x}}}_{-\mathcal{A}} f_s + \underbrace{\frac{q_s}{m_s}\left(\mathbf{E} + \frac{\mathbf{v} \times \mathbf{B}}{c}\right) \cdot \nabla_{\mathbf{v}}}_{-\mathcal{B}} f_s = 0$$

$$f(t^{n+1}) = \exp(\mathcal{A}\tau/2)\exp(\mathcal{B}\tau)\exp(\mathcal{A}\tau/2) f(t^n) + \mathcal{O}(\tau^3)$$

This is equivalent to the *leapfrog* scheme in PIC codes

C.Z. Cheng, G. Knorr 1976, JCP

Splitting Vlasov

$$\partial_t \tilde{f}_s + \mathbf{v} \cdot \nabla_{\mathbf{x}} \tilde{f}_s = 0$$

$$\Leftrightarrow \partial_t \tilde{f}_s + \partial_x v_x \tilde{f}_s + \partial_y v_y \tilde{f}_s = 0$$

Operators commute

$\Rightarrow$ Exact splitting!

$$\tilde{f}(t^{n+1}) = \exp(\mathcal{X}\tau) \exp(\mathcal{Y}\tau) \tilde{f}(t^n)$$

Splitting Vlasov

$$\partial_t \tilde{f}_s + \frac{q_s}{m_s}\left(\mathbf{E} + \frac{\mathbf{v}\times\mathbf{B}}{c}\right)\cdot\nabla_{\mathbf{v}}\tilde{f}_s = 0$$

Operators do not commute!
Strang-Splitting:

$$\begin{aligned}\tilde{f}(t^{n+1}) = &\exp(\mathcal{V}_x\tau/4)\exp(\mathcal{V}_y\tau/2)\exp(\mathcal{V}_x\tau/4)\\ &\times\exp(\mathcal{V}_z\tau)\\ &\times\exp(\mathcal{V}_x\tau/4)\exp(\mathcal{V}_y\tau/2)\exp(\mathcal{V}_x\tau/4)\tilde{f}(t^n)+\mathcal{O}(\tau^3)\end{aligned}$$

Not satisfying

Cascade interpolation / Backsubstitution

Idea: Modify

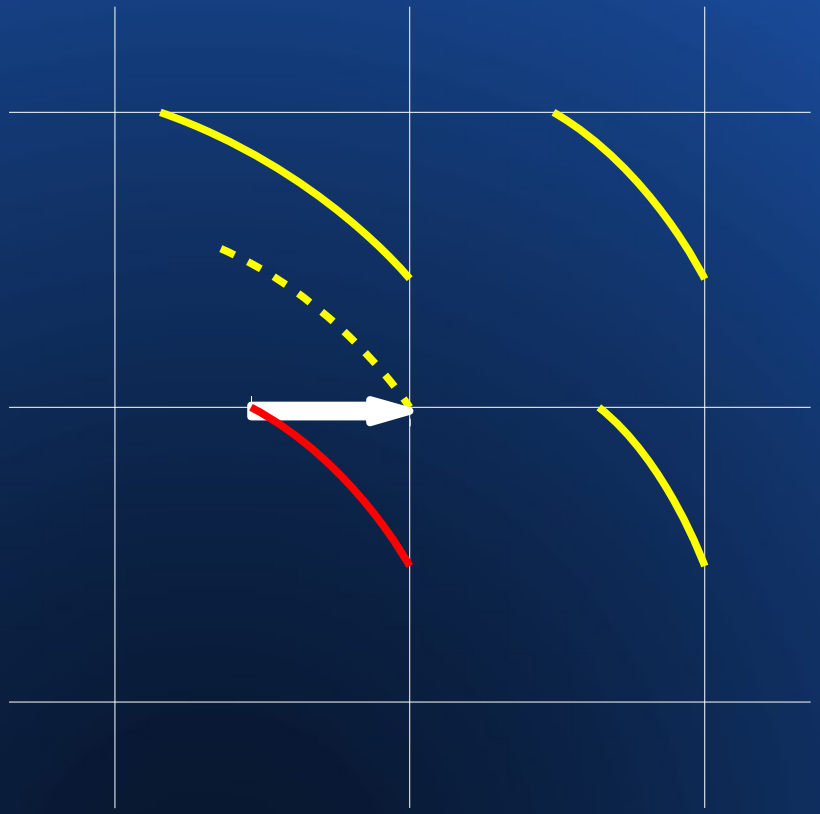
$$\partial_t \tilde{f}_s + \frac{q_s}{m_s}\left(\mathbf{E} + \frac{\mathbf{v}\times\mathbf{B}}{c}\right)\cdot\nabla_{\mathbf{v}}\tilde{f}_s = 0$$

to cancel error from first-order splitting

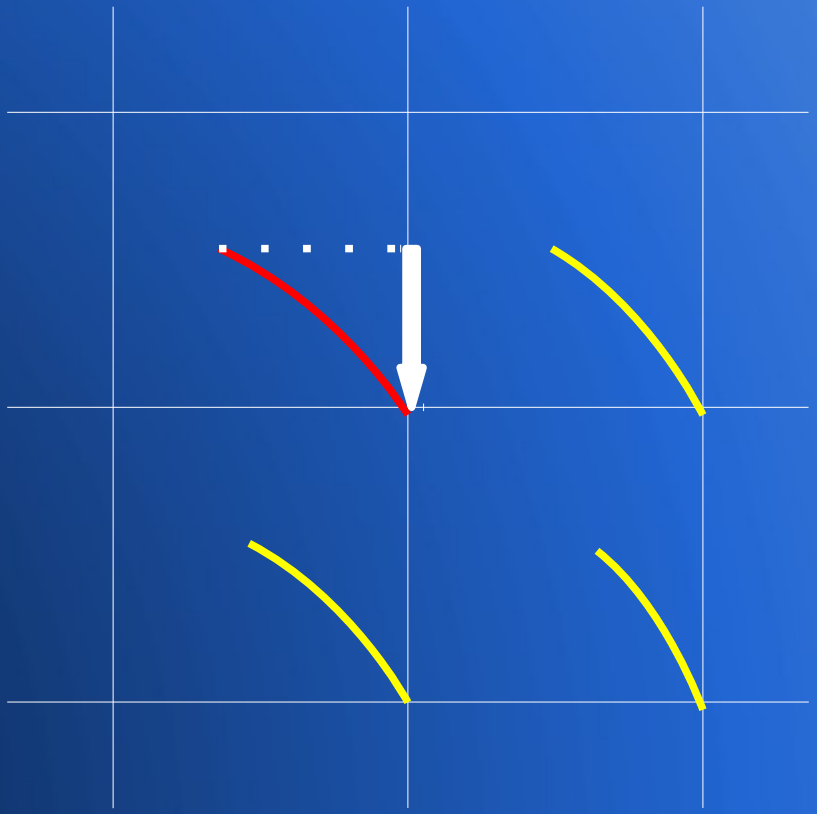
$$\tilde{f}(t^{n+1}) = \exp(\mathcal{B}\tau)\exp(\mathcal{A}\tau)\tilde{f}(t^n) + \mathcal{O}(\tau^2)$$

Nair et al. 1999, Quart. Journal Royal Met. Soc. -- Leslie et al. 1995, Month. Weather Rev.
H. Schmitz et al. 2006, Comp. Phys. Com.

Backsubstitutiton in 2D



Step 1



Step 2

Backsubstitution

In general:

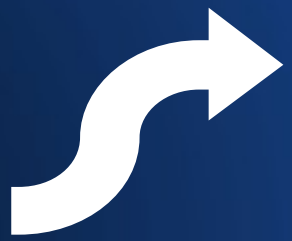
Characteristic going from $(A(x,y,z), B(x,y,z), C(x,y,z))$ to (x,y,z)

Solve for

$A = A(x, B, C)$

$B = B(x, y, C)$

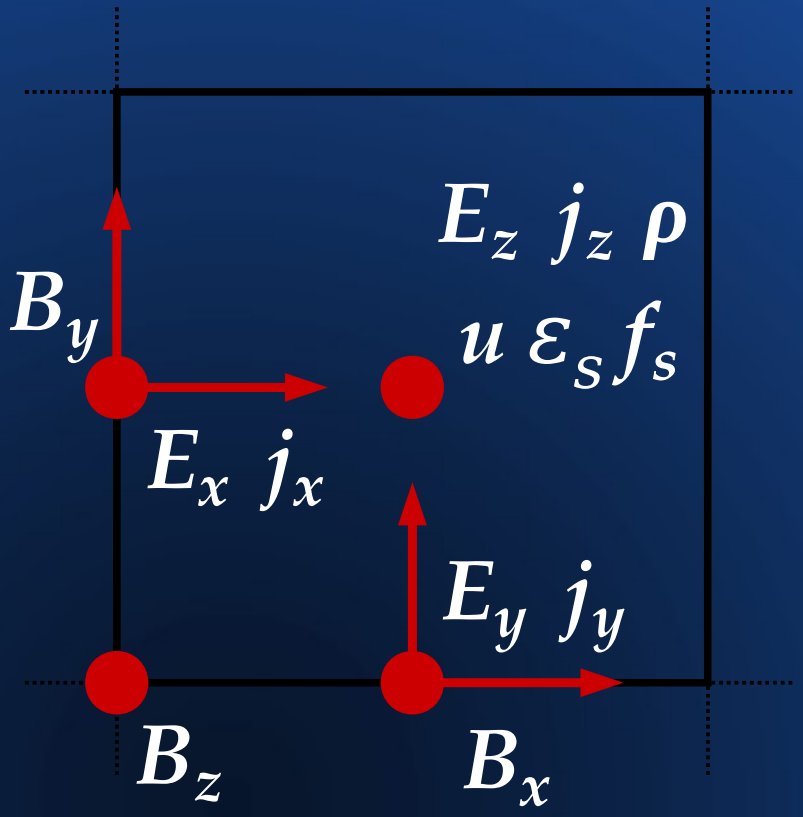
$C = C(x, y, z)$



Boris' scheme for PIC can be brought into this form!

Maxwell solver

Yee-Grid



$$\partial_t \mathbf{B} = -\nabla \times \mathbf{E}$$

$$\partial_t \mathbf{E} = c^2 \left(\nabla \times \mathbf{B} - \mu_0 \mathbf{j} \right)$$

Two fluid equations

Take moments:

$$\rho_s = m_s \int f_s \mathrm{d}^3 v \qquad \text{mass density}$$

$$\mathbf{u}_s = m_s \int \mathbf{v} f_s \mathrm{d}^3 v \qquad \text{momentum density}$$

$$\mathcal{E}_s = \frac{1}{2} m_s \int \mathbf{v}^2 f_s \mathrm{d}^3 v \qquad \text{energy density}$$

Two fluid equations

Time evolution:

$$\partial_t \rho_s = -\nabla \cdot \mathbf{u}_s$$

$$\partial_t \mathbf{u}_s = -\nabla \cdot \left(\frac{\mathbf{u}_s \otimes \mathbf{u}_s}{\rho_s} \right) - \nabla p_s + \frac{q_s}{m_s} \left(\rho_s \mathbf{E} + \mathbf{u}_s \times \mathbf{B} \right)$$

$$\partial_t \mathcal{E}_s = -\nabla \cdot \left(\frac{\mathcal{E}_s + p_s}{3\rho_s} \mathbf{u}_s \right) - \nabla \cdot \mathbf{Q}_s + \frac{q_s}{m_s} \mathbf{u}_s \cdot \mathbf{E}$$

Isotropic, monoatomic gas: $p_s = \frac{2}{3}\mathcal{E}_s - \frac{\mathbf{u}_s^2}{3\rho_s}, \quad \mathbf{Q}_s = 0$

The daVinci cluster in Bochum

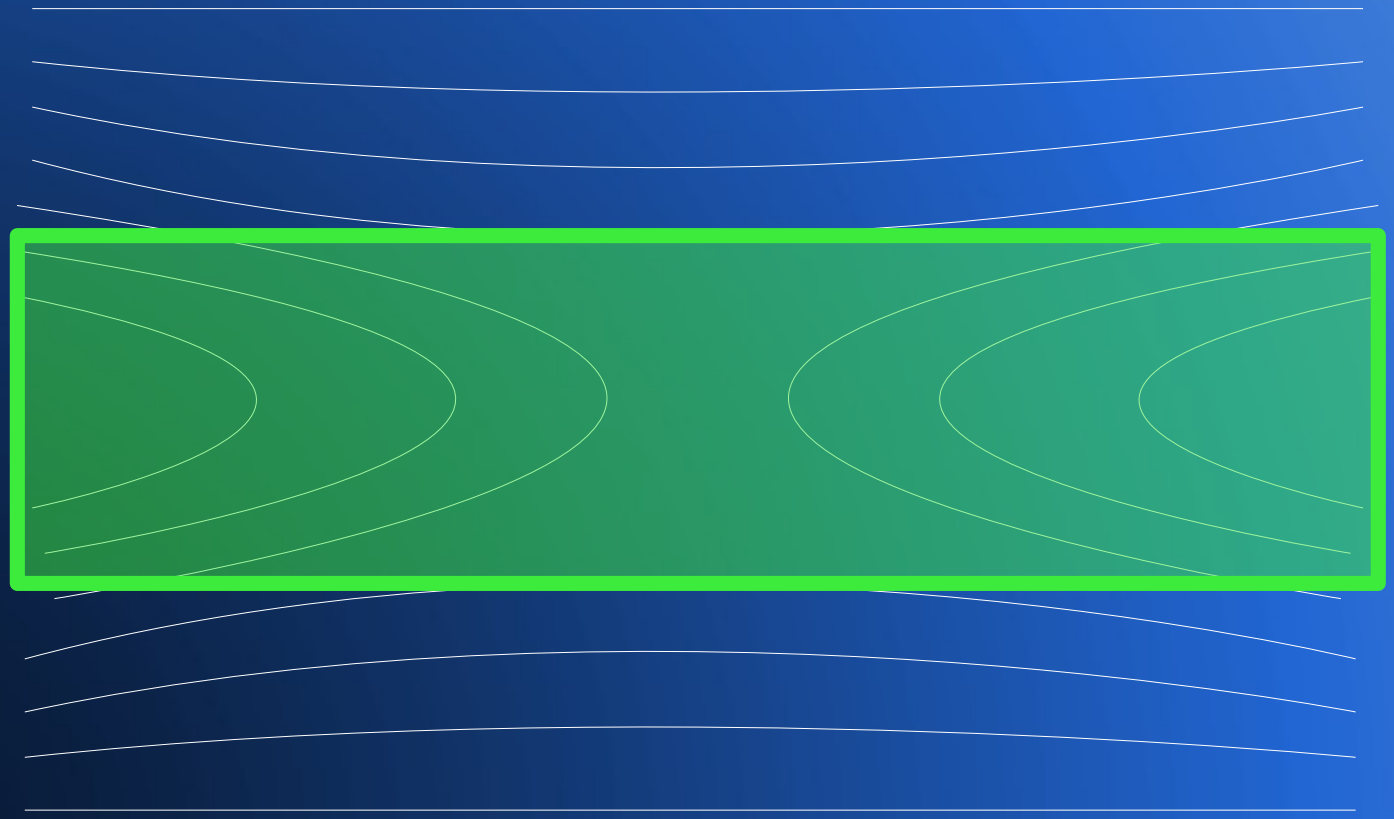
- 68 Tesla S1070 cards with 16320 cores and 272 GB RAM
- 34 Xeon E5530 Quadcore CPUs with 408 GB RAM
- Infiniband network

Coupling

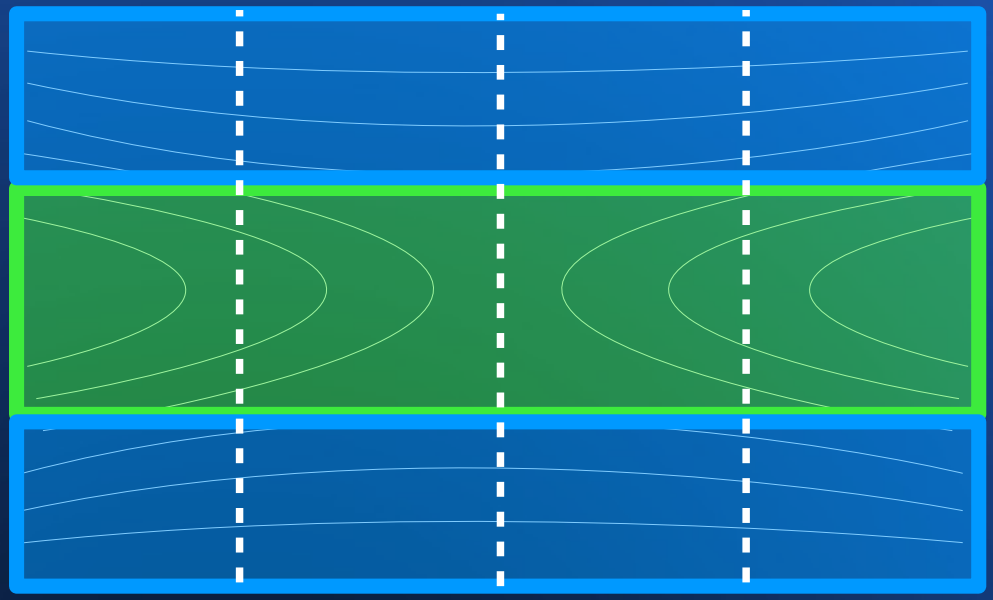
Two fluid

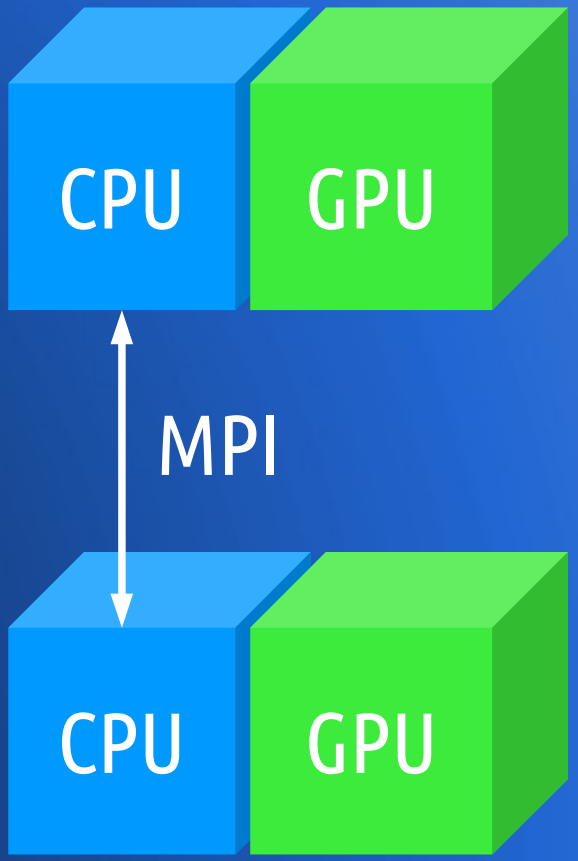
Vlasov

Two fluid



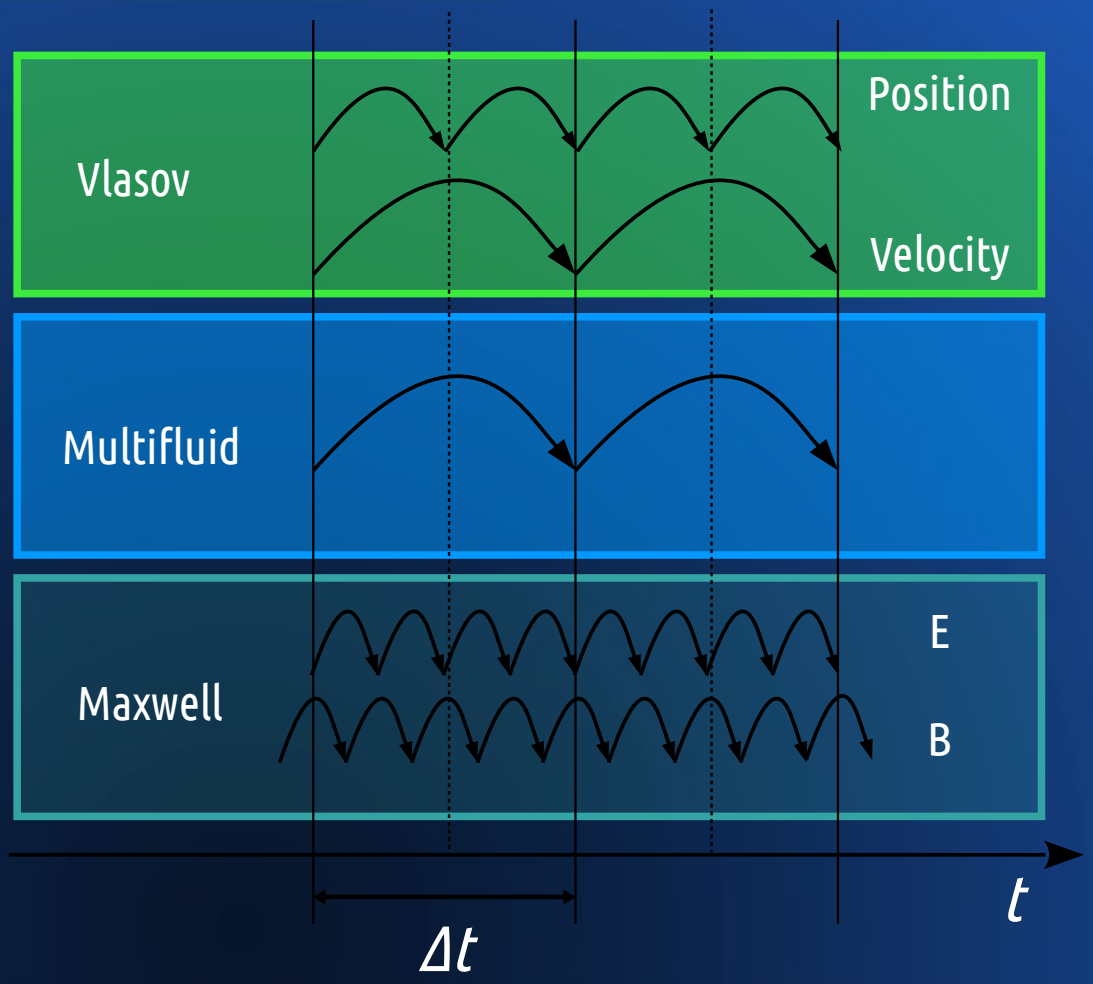
Parallelization





Domain decomposed into blocks
Vlasov on GPU, Fluid on CPU
Maxwell solved globally

Time stepping



- Match time levels of different schemes
- Keep second order accuracy in time

Coupling vlasov to fluid

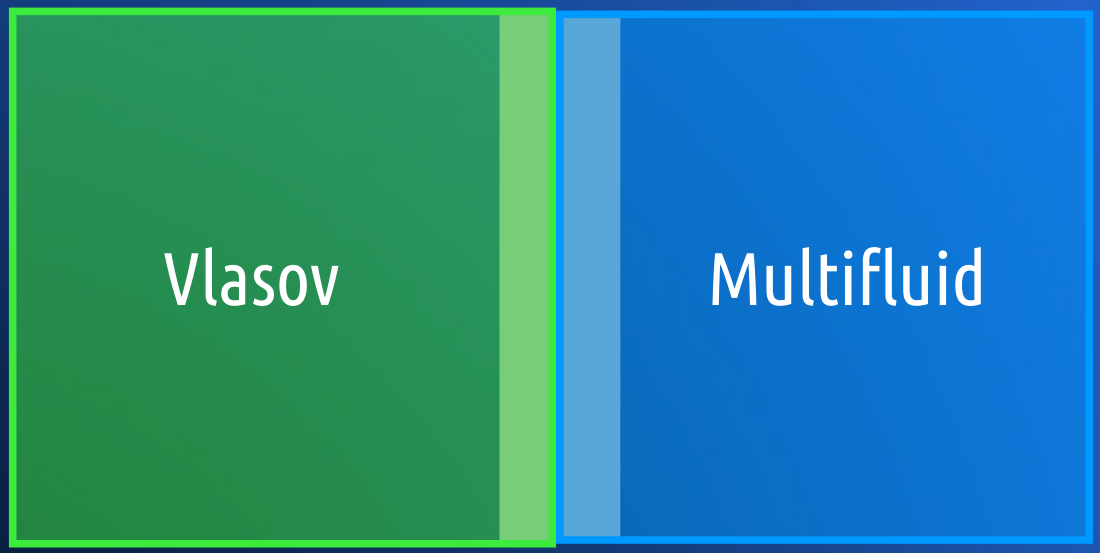
$$\rho_s = m_s \int f_s \mathrm{d}^3 v$$

$$\mathbf{u}_s = m_s \int \mathbf{v} f_s \mathrm{d}^3 v$$

$$\mathcal{E}_s = \frac{1}{2} m_s \int \mathbf{v}^2 f_s \mathrm{d}^3 v$$

Fluid quantities are
given by the moments of f

Coupling fluid to vlasov



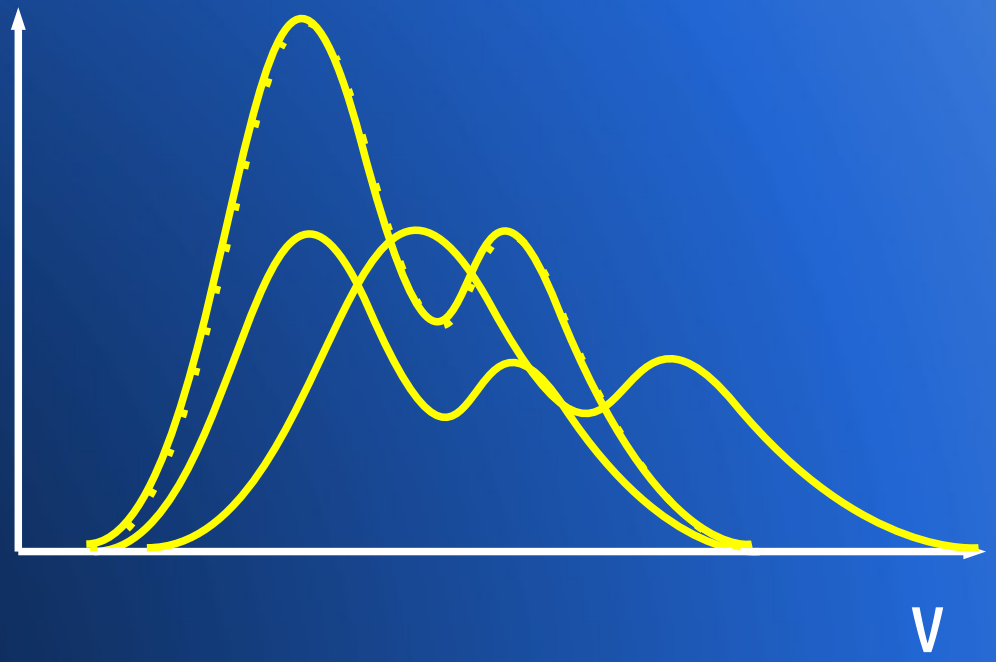
1. Extrapolate f into ghost cells
2. Modify f to fulfill fluid moments

Coupling fluid to vlasov

$$\rho_s = m_s \int f_s \mathrm{d}^3 v$$

$$\mathbf{u}_s = m_s \int \mathbf{v} f_s \mathrm{d}^3 v$$

$$\mathcal{E}_s = \frac{1}{2} m_s \int \mathbf{v}^2 f_s \mathrm{d}^3 v$$



Solve $\partial_t f_s + \partial_v((av + b) f_s) = 0$

Ion acoustic wave

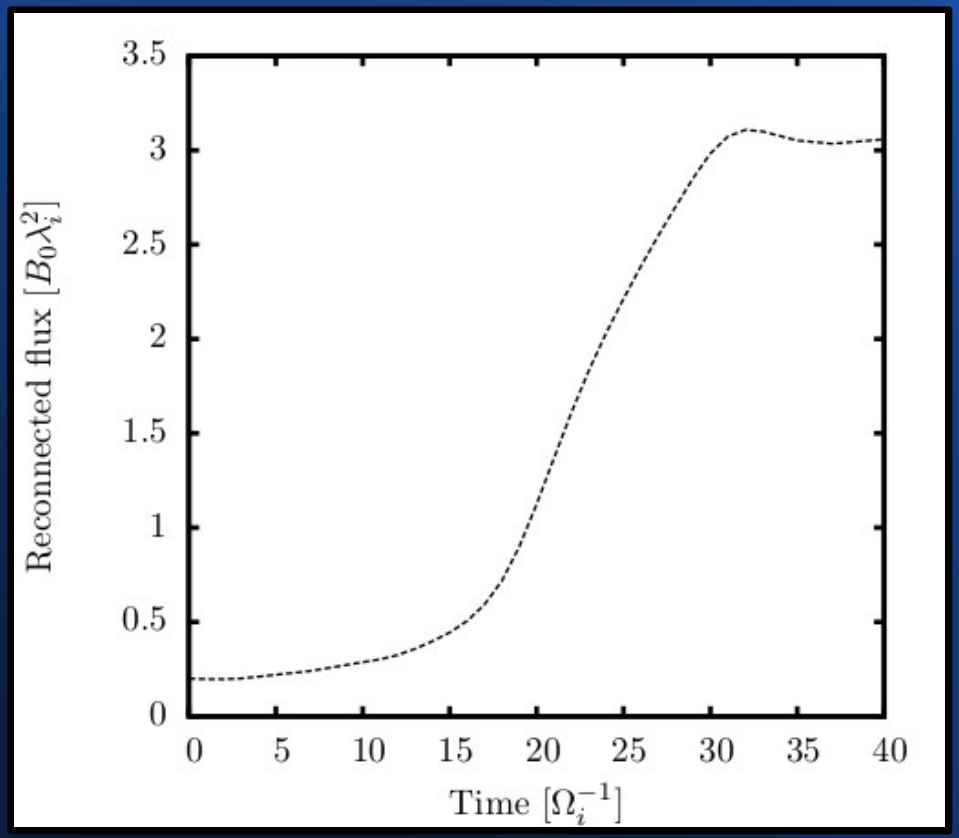
Multiple program / multiple data

- Each code can run independently or in coupled mode (multiple program / multiple data)
- GEM simulations of $512 \times 256 \times 64^3$ until $t = 40\ \Omega_i^{-1}$
- Comparison to full Vlasov on CPU:

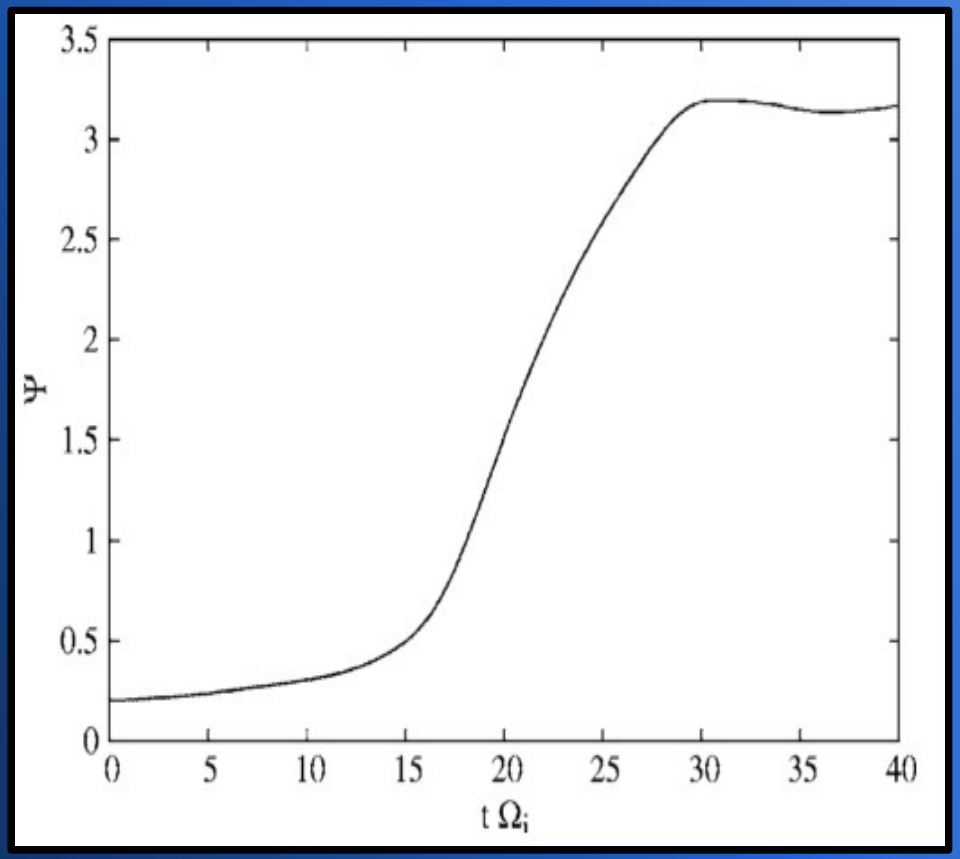
system	resolution	duration
CPU	$256 \times 128 \times 30^3$	~150h
GPU	$256 \times 128 \times 32^3$	~8h

Reconnection

Reconnection



Coupled



Full vlasov

H. Schmitz et al 2006, PoP

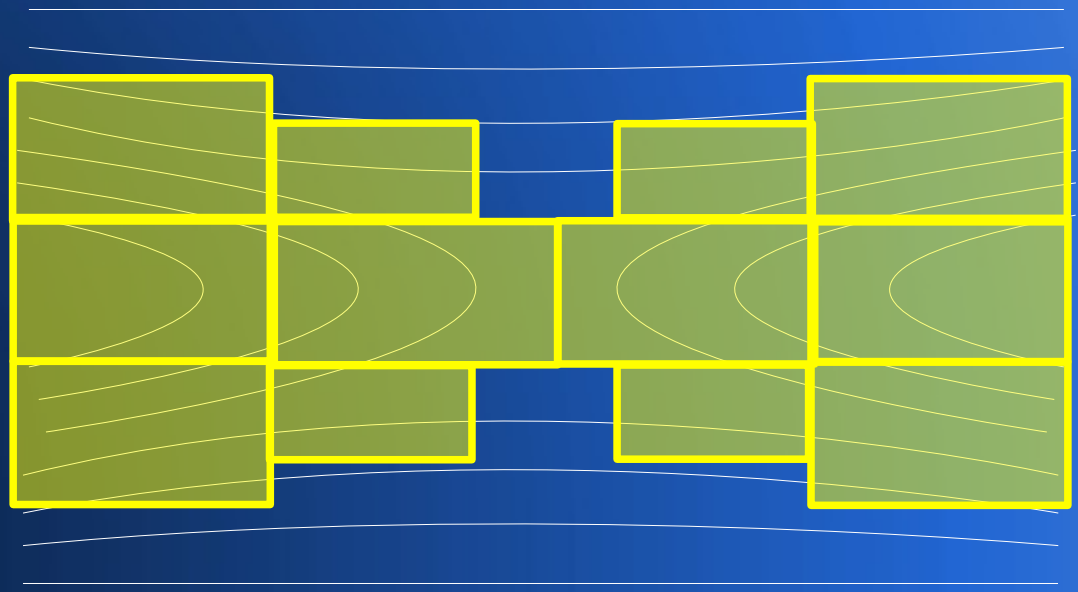
Reconnection

Outlook

- Find criterion for automatic refinement

  Difference between 5- and 10-moment fluid?
  Strong gradients?
  Large velocities?

- Coupling to different fluid models



Thank you!

“Software is slowing faster than hardware is accelerating.”

Martin Reiser