Numerical methods for the Gyrokinetic (GK) Vlasov equation

Eric Sonnendrücker

Max-Planck Institute for Plasma Physics

and

TU Munich

ITER Summer School, August, 25-29, 2014

Overview: Discretisation of gyrokinetic (GK) Vlasov equation.

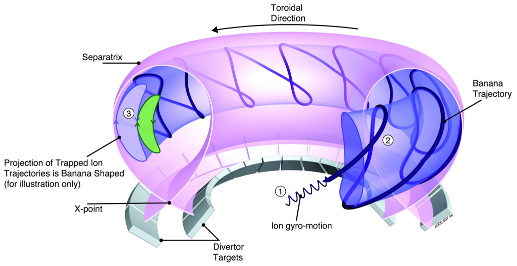
- Starting point:
  - Start from field theoretic Lagrangian $\rightarrow$ exact conservation properties.
  - Start from Euler-Lagrange eqns. (Vlasov + field eqns)
- Different classes of numerical methods
  - Lagrangian: Solve for characteristics (particle motion): PIC method
  - Semi-Lagrangian: Lagrangian + projection on phase space grid
  - Eulerian: solve Vlasov eq on phase space grid using FV or DG typically.
- Content of this talk
  - PIC method:
    - Bridge the gap between PIC and Monte Carlo method. Derive PIC from known concepts of MC methods in statistics.
    - Put PIC in a variational framework starting from gyrokinetic Lagrangian (following Lewis 1970).
  - Introduction to different kinds of semi-Lagrangian methods

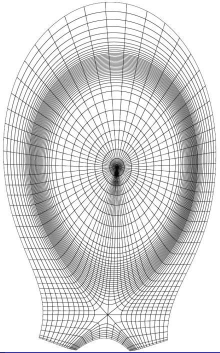
Outline

Outline

Tokamak simulations

- Large magnetic field.
- Particle trajectories mostly confined along magnetic field lines.
- Toroidal geometry
- Tokamak is axisymmetric in the toroidal direction.
- Large anisotropy of physics along and across magnetic field lines puts strong constraints on numerical method and mesh that is used.
- Need good alignment with magnetic flux surfaces and or high-order methods.





Approximation of Vlasov-Poisson equations in tokamak

- Specificities of tokamak, in particular large magnetic field and quasi-neutrality used to derive approximate models where smallest space and time scales are removed.
- In magnetic fusion plasmas at the time scale of micro turbulence two small scales need to be removed for efficient simulation:
  1. Rotation around magnetic field: gyrokinetic model.
  2. Debye length: quasi-neutral model.

The electrostatic gyrokinetic model

- The Vlasov-Poisson gyrokinetic model reads

$$\frac{\partial f}{\partial t} + \frac{d\mathbf{X}}{dt} \cdot \nabla_x f + \frac{dV_\parallel}{dt}\frac{\partial f}{\partial v_\parallel} = 0,$$

with

$$\begin{aligned} B_\parallel^* \frac{d\mathbf{X}}{dt} &= \mathbf{b} \times \nabla J(\phi) + \frac{1}{q}(mV_\parallel^2 \nabla \times \mathbf{b} + \mu \mathbf{b} \times \nabla B) + V_\parallel \mathbf{B} \\ B_\parallel^* \frac{dV_\parallel}{dt} &= -(\mathbf{B} + \frac{m}{q} V_\parallel \nabla \times \mathbf{b}) \cdot (\frac{\mu}{m} \nabla B + \frac{q}{m} \nabla J(\phi)) \end{aligned}$$

and $B_\parallel^* = B + \frac{m}{q} V_\parallel \nabla \times \mathbf{b} \cdot \mathbf{b}$.

- The gyroaverage operator J transforms the guiding center distribution to the distribution at the particle position which allows to take into account the finite Larmor radius effects.

Hamiltonian formalism

- Starting with Littlejohn 1983, and followed by Brizard, Hahm, Qin, Sugama, Scott-Smirnov and many others, derivation of the gyrokinetic theory starts from a particle phase space Lagrangian in canonical coordinates and finds suitable change of coordinates to decouple the gyrophase up to some given order and average it out.
- Following Sugama we start from the phase space Lagrangian, splitting between particle and field Lagrangian:

$$L = \sum_{s} \int f_s(\mathbf{Z}_0, t_0) L_p(\mathbf{Z}(\mathbf{Z}_0, t_0; t), \dot{\mathbf{Z}}(\mathbf{Z}_0, t_0; t), t) \, \mathrm{d}\mathbf{Z}_0 + \int \frac{E^2 - B_\perp^2}{8\pi} \, \mathrm{d}\mathbf{X}$$

- Distribution function f expressed at initial time. Particle Lagrangian L_p is of the form

$$L_p(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{p}, t) = \mathbf{p} \cdot \dot{q} - H$$

where H is the particle Hamiltonian.

Gyrokinetic Lagrangian

- Gyrokinetic equations can be derived from a Lagrangian with the same structure for the gyrocenters.
- Two changes of variables in phase space: guiding center transformation and gyrocenter transformation yield a Lagrangian in the new phase space coordinates $\mathbf{z} \in \mathbb{R}^6$ obtained as a coordinate transformation of $\mathbf{q} \in \mathbb{R}^3$ and $\mathbf{p} \in \mathbb{R}^3$:

$$L_p(\mathbf{z}, \dot{\mathbf{z}}, t) = \mathbf{p} \cdot \dot{q} - H$$

  where now H, $\mathbf{p}$ and $\mathbf{q}$ are functions of $\mathbf{z}, t$.
- Canonical hamiltonian formulation

$$\frac{d}{dt}\begin{pmatrix} \mathbf{q} \\ \mathbf{p} \end{pmatrix} = J\nabla_{q,p}H \text{ with } J = \begin{pmatrix} 0 & \mathbb{I}_3 \\ -\mathbb{I}_3 & 0 \end{pmatrix}.$$

- This becomes in new coordinate system $\frac{d\mathbf{z}}{dt} = \Omega\nabla_z H$,
  Ω is new cosymplectic matrix whose components are Poisson brackets of coordinates, with Poisson structure.

Advective form of GK Vlasov equation

- The GK Vlasov equation is a scalar hyperbolic equation in phase space
- Advective form

$$\frac{\partial f}{\partial t} + \mathbf{A} \cdot \nabla f = 0$$

- The solution of the Vlasov equation can be expressed with help of characteristics which are the solutions of the differential system

$$\frac{d\mathbf{Z}}{dt} = \mathbf{A}(t)$$

- Characteristics are denoted by $\mathbf{Z}(t; \mathbf{z}, s)$
- Solution of Vlasov equation then writes $f(\mathbf{z}, t) = f_0(\mathbf{Z}(0; \mathbf{z}, t))$.

Conservative form of GK Vlasov equation

- Arbitrary coordinates in configuration space $\sqrt{g}$ Jacobian of coordinate transformation.
- Jacobian of 6D coordinate transformation to gyrocenter coordinates then becomes $B_\parallel^* \sqrt{g}$
- Using $\nabla \cdot (B_\parallel^* \sqrt{g} \mathbf{A}) = 0$, advective form can be transformed to conservative form

$$\frac{\partial f}{\partial t} + \frac{1}{B_\parallel^* \sqrt{g}} \nabla \cdot (B_\parallel^* \sqrt{g} \mathbf{A} f) = 0$$

- Conservative form can be extended to include collisions $\rightarrow$ Fokker Planck or Kolmogorov forward equation.

Conservation properties

Maximum principle

$$0 \leq f(\mathbf{x}, \mathbf{v}, t) \leq \max_{(\mathbf{x},\mathbf{v})}(f_0(\mathbf{x}, \mathbf{v})). \tag{1}$$

Follows from advective form

$$f(\mathbf{x}, \mathbf{v}, t) = f_0(\mathbf{X}(0; \mathbf{x}, \mathbf{v}, t), \mathbf{V}(0; \mathbf{x}, \mathbf{v})).$$

Conservation of phase space volume

For any volume V of phase space

$$\int_V f(\mathbf{z}, t) B_\parallel^* \sqrt{g}\, \mathrm{d}\mathbf{z} = \int_{F^{-1}(V)} f_0(\tilde{\mathbf{z}}) B_\parallel^* \sqrt{g}\, \mathrm{d}\tilde{\mathbf{z}}. \tag{2}$$

Numerical simulation of gyrokinetic equations

- Difficulties:
  - defined in 5D phase space
  - Appearance of small scales. Filamentation.
  - Necessity to use a mesh of configuration space adapted to magnetic field lines to handle efficiently anisotropy of transport: curvilinear coordinates.
- Particle methods: more efficient in high dimensions.
  - Good qualitative results at relatively low cost
  - Numerical noise and slow convergence: need efficient variance reduction techniques.
- Methods using a grid of phase space:
  - Large grid of 5D phase space. Size reduced by field alignement.
  - No numerical noise, but diffusion.
  - Small scales at some point will not be resolved by the grid.
    Need to dissipate them for stability. How?
    Entropy cannot be conserved for long time simulations
  - Hamiltonian structure harder to preserve.

Outline

Discretization of the Vlasov equation by a particle method

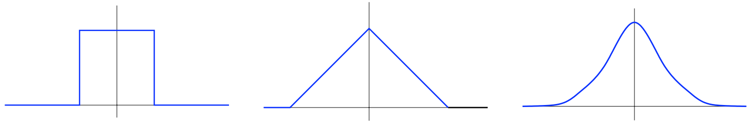
- Usually seen with a purely deterministic point of view as an approximation of a function by a sum of Dirac masses.
- Particle approximation of the Vlasov equation. Distribution function is approximated by

$$f_h(x, v, t) = \sum_k w_k \delta(x - x_k(t))\delta(v - v_k(t)).$$

- In standard PIC all weights are equal.
- Variance reduction methods (from Monte Carlo methods in statistics):
    - Weighted PIC corresponds to importance sampling.
    - δf corresponds to control variates.
- Deterministic, pseudo-random or Monte-Carlo approximation of f_0.
- Once particles have been initialized, they are advanced using deterministic equations of motion.

Coupling particles with fields through shape functions

- Particle method defines point particles (Dirac masses).
- Regularization by convolution with a finite width smoothing kernel generally called weighting function.
  - Splines of different orders on are generally used on structured grids



  - P^1 Finite Element shape functions on unstructured grids. Generally fields obtained by Poisson or Maxwell field solver first interpolated at vertices of mesh
  - Truncated Gaussians have also been used by some authors (Jacobs-Hesthaven).

Classical Poisson solvers

We need to approximate $-\Delta\phi = \rho$ with periodic boundary conditions.

- Finite Difference method. The standard centred second order finite difference method on a uniform grid yields the circulant system
$$-\phi_{j+1} + 2\phi_j - \phi_{j-1} = \Delta x^2 \rho_j, \quad 0 \le j \le N_x - 1$$
to set constant assume $\sum \phi_j = 0$. Electric field computed also with centred finite differences $E_j = (\phi_{j-1} - \phi_{j+1})/(2\Delta x)$.
- The FD solver verifies: $\sum E_j \rho_j = 0$.
- Finite Element method with B-splines as test functions: $\phi_h(x) = \sum \phi_j S_j(x)$ verifies
$$\int \nabla\phi_h(x) \nabla S_i(x)\,\mathrm{d}x = \int \rho_N S_i(x)\,\mathrm{d}x, \quad 0 \le i \le N_x - 1.$$
- Particle smoothing provided by Finite Element test function.
- Using ϕ_h itself as a test function, the Finite Element solver verifies
$$\int |\nabla\phi_h|^2\,\mathrm{d}x = \int \rho_N \phi_h(x)\,\mathrm{d}x = -\sum_{k=1}^{N} w_k \phi_h(x_k(t)).$$

Momentum conserving FD-PIC

- In classical PIC literature distinction between momentum conserving and energy conserving PIC is made.
- FD-PIC can be made momentum conserving by using symmetric procedure for computing ρ at grid points from particles and computing E at particles from grid values:

$$\rho_j = \sum_{k=1}^{N} w_k S(x_k - x_j), \qquad E(x_k) = \sum_{j=0}^{N_x-1} E_j S(x_k - x_j).$$

- Evolution of total momentum given by velocity advance:

$$\sum_{k=1}^{N} w_k \frac{v_k^{n+1/2} - v_k^{n-1/2}}{\Delta t} = \sum_{k=1}^{N} w_k E(x_k) = \sum_{k=1}^{N} w_k \sum_{j=0}^{N_x-1} E_j S(x_k - x_j)$$
$$= \sum_{j=0}^{N_x-1} E_j \sum_{k=1}^{N} w_k S(x_k - x_j) = \sum_{j=0}^{N_x-1} E_j \rho_j = 0.$$

Energy conserving FE-PIC without time discretisation

- Energy conserving PIC was derived by Lewis (1970) from a variational principle. Equivalent to our FE-PIC formulation.
- Automatically energy conserving without time discretisation *e.g.* for Vlasov-Poisson

$$\frac{d}{dt}\left(\sum_{k=1}^{N}\frac{1}{2}w_k\mathbf{v}_k^2(t)+\int|\nabla\phi_h|^2\,\mathrm{d}\mathbf{x}\right)=\sum_{k=1}^{N}w_k(\mathbf{v}_k(t)\cdot\frac{d\mathbf{v}_k}{dt}-\nabla\phi_h(\mathbf{x}_k(t))\cdot\frac{d\mathbf{x}_k}{dt}))$$
$$=\sum_{k=1}^{N}w_k(\mathbf{v}_k(t)\cdot\nabla\phi_h(\mathbf{x}_k(t))-\nabla\phi_h(\mathbf{x}_k(t))\cdot\mathbf{v}_k(t)))=0.$$

Monte Carlo interpretation of the particle in cell method

- Up to now the PIC method has be introduced as a deterministic particle method.
- When collisions are introduced, the natural way within the PIC framework is to use stochastic differential equations (SDE) and the Langevin formalism.
- With collisions deterministic interpretation makes is not adequate.
- Even without collisions for long enough times phase mixing occurs and numerical fluctuations are introduced, which makes Monte Carlo interpretation necessary.
- Consequences: only expected values are well defined. Point values of the distribution function at particle positions should not be used. This introduces also for PIC a resolution issue.

Monte Carlo simulation. Some background notions.

- A random variable X is a mapping from a probability space to $\mathbb{R}$. It is characterised by its probability density function (PDF) f.
- The realisations of X are randomly drawn according to f.
- The transfer theorem links the expected value of a real function of a random variable to a classical integral involving the PDF of X

$$\mathbb{E}(g(X)) = \int_{-\infty}^{+\infty} g(x) f(x) \, \mathrm{d}x.$$

- A Monte Carlo simulation consists in approximating a number that can be expressed as an expected value using the law of large numbers:

$$\mathbb{E}(g(X)) \approx \frac{1}{N} \sum_{i=1}^{N} g(X_i)$$

where the X_i are independent identically distributed (iid) random variables (distributed like X, *i.e.* having the same PDF as X).

Monte Carlo PIC

- Initialisation: Particle positions in phase space are drawn randomly according to the initial PDF f_0 that is the initial condition of the Vlasov equation (normalised to one).
- There is a one to one correspondance between a random variable and its PDF. The PDF is characterised by the particle positions in phase space
- The Kolmogorov forward equation describes the evolution of the PDF and the associated SDE, which is exactly the equation of characteristics (an ODE) when no diffusion term is present, describes the evolution of the random variable associated to the PDF.
- In a PIC method the time evolution is performed by solving (numerically) the SDE (or the ODE) and the PDF $f(t)$ can be recovered from the particle positions using a kernel density estimate (KDE).

Expected values in a PIC code

- In the Monte Carlo point of view, all the quantities of interest are expressed as expected values and approximated by the law of large numbers.
- Examples:
  - Kinetic energy:

$$\int f v^2 \,\mathrm{d}x\,\mathrm{d}v = \mathbb{E}(V^2).$$

  - Charge density at a point defined using smooth kernel S (cloud):

$$\rho(x_j) = \int S(x - x_j) f(x, v)\,\mathrm{d}x\,\mathrm{d}v = \mathbb{E}(S(X - x_j)).$$

  - Current density at a point:

$$J(x_j) = \int S(x - x_j) v f(x, v)\,\mathrm{d}x\,\mathrm{d}v = \mathbb{E}(S(X - x_j)V).$$

Monte Carlo error

- Common measure for the error in Monte Carlo simulation is mean squared error (MSE)

$$MSE(\hat{\theta}) = \mathbb{E}((\hat{\theta} - \theta)^2) = \mathbb{V}(\hat{\theta}) + Bias(\hat{\theta})^2.$$

- For an unbiased simulation the error is given by the variance of the estimator.
- The variance of the sample mean for a random variable X is $\mathbb{V}(X)/N$.
- So the root mean square error for an approximation using the law of large number is proportional to

$$\sqrt{\mathbb{V}(X)}/\sqrt{N}.$$

- This can be made precise using the central limit theorem.
- From this we get the statistical convergence in $1/\sqrt{N}$. We note also that the error is smaller if the variance of the random variable being simulated is smaller.

Kernel density estimate

- To reconstruct a probability density function f from particle positions we use in PIC codes a smooth kernel S that we assume to be even and a tensor product of 1D functions for simplicity.
- Even if it is in general not necessary to reconstruct f except for diagnostic, we will always need it to reconstruct ρ
- This kernel density estimate is defined by:

$$f_{h,N}(x_1,\dots,x_d) = \frac{1}{Nh^d}\sum_{i=1}^{N} S\left(\frac{Y_{1,i}-x_1}{h}\right)\dots S\left(\frac{Y_{d,i}-x_d}{h}\right).$$

- Its mean squared error can be computed as sum of variance and bias squared. Note that here the bias is unavoidable

$$MSE(f_{h,N}) = \frac{R(S)^d}{Nh^d} f + \frac{h^4}{4}\kappa_2^2(S)(\Delta f)^2 + O(\frac{1}{N}) + O(h^6).$$

- We can compute N so that both terms balance which yields a MSE of the order $N^{-\frac{4}{4+d}}$, which is larger than the standard statistical MSE N^{-1} and depends on d. Curse of dimensionality

Derivation of Monte Carlo PIC code from a Lagrangian

- Derivation of PIC code from Lagrangian introduce by Lewis 1970. Recent work by Evstatiev 2013 and Shadwick 2014.
- Consider Lagrangian of form

$$L[Z,\phi](t) = \sum_{\mathrm{s}} \int f_s(\mathbf{Z}_0, t_0) L_p(\mathbf{Z}(\mathbf{Z}_0, t_0; t), \dot{\mathbf{Z}}(\mathbf{Z}_0, t_0; t), t)\, \mathrm{d}\mathbf{Z}_0 + \int |\nabla\phi|^2\, \mathrm{d}\mathbf{X}.$$

- First part directly translates into an expected with respect to initial distribution of particles for which we perform a MC approximation

$$L[Z,\phi](t) = \sum_{\mathrm{s}} \sum_{k=1}^{N} L_p(\mathbf{Z}_s(\mathbf{Z}_{0,k}, t_0; t), \dot{\mathbf{Z}}_s(\mathbf{Z}_{0,k}, t_0; t), t) + \int |\nabla\phi_h|^2\, \mathrm{d}\mathbf{X}.$$

- When ϕ_h is constrained to live in finite dimensional subspace, variations with respect to ϕ_h directly yield finite element formulation of field equation.
- This formulation is implemented in ORB5 family of codes.

Variance reduction techniques

- The Monte Carlo error for a simulation based on a random variable X is given by $\sqrt{\mathbb{V}(X)/N}$.
- The idea of variance reduction techniques that are essential for efficient Monte Carlo simulations is to find a random variable $\tilde{X}$ so that

$$\mathbb{E}(\tilde{X}) = \mathbb{E}(X) \qquad \text{and } \mathbb{V}(\tilde{X}) \ll \mathbb{V}(X).$$

- Two such techniques are efficiently used in PIC simulations
  1. Importance sampling: weighted PIC
  2. Control variates: δf PIC
- Both have been historically developed for other purposes. First MC interpretation by Aydemir 1994.
- Both techniques are still not mainstream in PIC simulations because of weight mixing and weight spreading issues. We will see how those can be mitigated.

Importance sampling (1)

- We want to compute via Monte Carlo simulation

$$\int \psi(\mathbf{z}) f(\mathbf{z})\,\mathrm{d}\mathbf{z} = \mathbb{E}(\psi(\mathbf{Z})).$$

- Depending on ψ it might not be the best approach to use directly the density f for drawing the random variable used in the simulation.
- If g is any other probability density that does not vanish in the support of f one can express our integral as an expectation using a random variable $\tilde{\mathbf{Z}}$ of density g:

$$\int \psi(\mathbf{z}) f(\mathbf{z})\,\mathrm{d}\mathbf{z} = \int \psi(\mathbf{z}) \frac{f(\mathbf{z})}{g(\mathbf{z})} g(\mathbf{z})\,\mathrm{d}\mathbf{z} = \mathbb{E}(W(\tilde{\mathbf{Z}})\psi(\tilde{\mathbf{Z}})),$$

where the random variable $W(\tilde{\mathbf{Z}}) = f(\tilde{\mathbf{Z}})/g(\tilde{\mathbf{Z}})$ is called weight.

Importance sampling (2)

- The Monte Carlo approximation using independent random variables distributed identically with density g can be expressed as

$$\tilde{M}_N = \frac{1}{N}\sum_{i=1}^{N} W(\tilde{\mathbf{Z}}_i)\psi(\tilde{\mathbf{Z}}_i),$$

from which we get

$$\mathbb{E}(\tilde{M}_N) = \mathbb{E}(W(\tilde{\mathbf{Z}})\psi(\tilde{\mathbf{Z}})) = \int \psi(\mathbf{z})f(\mathbf{z})\,\mathrm{d}\mathbf{z}.$$

- $\tilde{M}_N$ is another unbiased estimator of the integral we wish to compute and the approximation error for a given number of samples N is determined by its variance.

Application of importance sampling to the PIC method

- Instead of initialising the particle positions according to initial particle distribution f_0, use adequately chosen marker distribution g_0.
- For each marker z_k weight is defined by $w_k = f_0(z_k)/g_0(z_k)$.
- Let marker density evolve like particle density: g is solution of the same Fokker-Planck (or Vlasov) equation as f, only with different initial condition.
- As f and g are conserved along the same characteristics w_k is constant in time:

$$w_k = \frac{f(t, \mathbf{z}_k(t))}{g(t, \mathbf{z}_k(t))} = \frac{f_0(\mathbf{z}_k(0))}{g_0(\mathbf{z}_k(0))}.$$

- Good way to initialise marker dependent on physics problem.
- Each expected value computed from g yields different variance. Low noise for computation of ρ essential.

Control variate method

- We want to define a Monte Carlo approximation of $a = \mathbb{E}(X)$ for some given random variable X.
- Assume there exists a random variable Y with known expected value correlated to X.
- For $\alpha \in \mathbb{R}$ define a new random variable

$$Z_\alpha = X - \alpha(Y - \mathbb{E}(Y)).$$

- For any α, $\mathbb{E}(Z_\alpha) = \mathbb{E}(X) = a$.
- The sample mean of Z_α

$$M_{N,\alpha} = \frac{1}{N}\sum_{i=1}^{N}(X_i - \alpha(Y_i - \mathbb{E}(Y))) = \alpha\mathbb{E}(Y) + \frac{1}{N}\sum_{i=1}^{N}(X_i - \alpha Y_i)$$

  can be used instead of the sample mean of X to approximate a.
- The random variable αY is called a control variate for X.
- If X and Y are not independent, there exists a value of α for which the variance of Z_α is smaller than the variance of X.

Application of control variate in PIC

- We couple control variate with importance sampling by assuming an arbitrary marker distribution g.
- Related to δf method originally introduced for linear tokamak simulations.
- Idea: f always stays close to analytically known distribution function $\tilde{f}(t, \mathbf{z})$. Typically initial condition or equilibrium distribution.
- Let $Y(t)$ be random variable associated to f, build control variate $\tilde{Y}_t$ associated to $\tilde{f}(t, \mathbf{z})$ such that

$$Y_t = \psi(\mathbf{Z})\frac{f(t,\mathbf{Z})}{g(t,\mathbf{Z})}, \qquad \tilde{Y}_t = \psi(\mathbf{Z})\frac{\tilde{f}(t,\mathbf{Z})}{g(t,\mathbf{Z})}.$$

Indeed we have

$$\mathbb{E}(\tilde{Y}_t) = \int \psi(\mathbf{z})\frac{\tilde{f}(t,\mathbf{z})}{g(t,\mathbf{z})}g(t,\mathbf{z})\,\mathrm{d}\mathbf{z} = \int \psi(\mathbf{z})\tilde{f}(t,\mathbf{z})\,\mathrm{d}\mathbf{z}$$

can be computed analytically for simple enough functions ψ and $\tilde{f}$.

- If $\tilde{f}$ is close enough to f then $\tilde{Y}_t$ will be close to Y_t

Implementation of control variates in PIC (1)

- Small modification in existing PIC code.
- Add weights that will evolve in time and will be used for computation of expected values: ρ, J, kinetic energy, ...
- Algorithm as follows:
- Initialisation:
    - Sample initial marker positions according to the random variable Z^0 of density g_0.
    - Importance weights for f defined by random variable $W = f_0(\mathbf{Z}^0)/g_0(\mathbf{Z}^0)$
    - Importance weights for $\delta f = f - \alpha\tilde{f}$ defined by the random variable

$$W_\alpha^0 = \frac{f_0(\mathbf{Z}^0) - \alpha\tilde{f}(t_n, \mathbf{Z}^n)}{g_0(\mathbf{Z}^0)} = W - \alpha\frac{\tilde{f}(0, \mathbf{Z}^0)}{g_0(\mathbf{Z}^0)} = W - \alpha\tilde{W}(0).$$

Implementation of control variates in PIC (2)

- Time stepping
  - Markers advanced using characteristics of Vlasov
  - Because f and g satisfy the same Vlasov-Poisson equation, they are conserved along the same characteristics so that

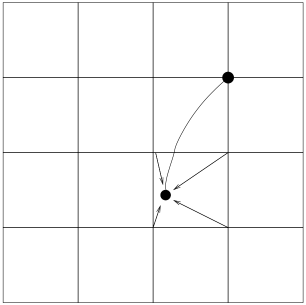
    $$W = \frac{f(t_n, \mathbf{Z}^n)}{g(t_n, \mathbf{Z}^n)} = \frac{f_0(\mathbf{Z}^0)}{g_0(\mathbf{Z}^0)}$$

    W does not evolve in time.
  - We know $\tilde{f}$ analytically and know that f and g are conserved along the characteristics, so that we can compute the importance weight for δf at time t_n from the phase space positions of the markers at the same time:

    $$W_\alpha^n = \frac{f(t_n, \mathbf{Z}^n) - \alpha \tilde{f}(t_n, \mathbf{Z}^n)}{g(t_n, \mathbf{Z}^n)} = \frac{f_0(\mathbf{Z}^0) - \alpha \tilde{f}(t_n, \mathbf{Z}^n)}{g_0(\mathbf{Z}^0)} = W - \alpha \frac{\tilde{f}(t_n, \mathbf{Z}^n)}{g_0(\mathbf{Z}^0)}.$$
  - W_α^n is a time dependent random variable which can be computed explicitly using the analytical functions $\tilde{f}$, f_0 and g_0.
  - These values can be used to express the sample mean for the new simulated random variable $\tilde{Y}_\alpha = Y - \alpha(\tilde{Y} - \mathbb{E}(\tilde{Y}))$.

Outline

The backward semi-Lagrangian Method



- *f* conserved along characteristics
- Find the origin of the characteristics ending at the grid points
- Interpolate old value at origin of characteristics from known grid values $\rightarrow$ High order interpolation needed

- Typical interpolation schemes.
    - Cubic spline (Cheng-Knorr)
    - Cubic Hermite with derivative transport (Nakamura-Yabe)

Interpolation

- Cubic spline interpolation originally proposed by Gagné and Shoucri 1977 is still our method of choice.
- Other methods have been tried: different variants of Lagrange, Hermite, higher order splines. None has proved superior to cubic splines for our applications.
- Features needed by interpolation: accuracy and robustness. Needs to degrade well when distribution is not resolved on the mesh.
- New implementations. Local splines
  - Series approximation of derivative on boundary: Crouseilles, Latu, ES: JCP 2007.
  - Fast algorithm by Unser IEEE Trans on Pattern Analysis and Machine Intelligence, vol 13 (3), 1991 for signal processing. Cholesky decomposition with constant coefficients on diagonal and off-diagonal. Iterations started with series using directly f.

Computation of the origin of the characteristics

- Transport equation
$$\frac{\partial f}{\partial t} + \mathbf{a} \cdot \nabla f = 0,$$
- Characteristics
$$\frac{dX}{dt} = \mathbf{a}$$
- Computation of the origin of the characteristics :
    - Explicit solution if **a** does not depend on x and t
    - Else, numerical algorithm needed.

Splitting for exact computation of characteristics

- In many cases splitting can enable to solve a constant coefficient advection at each split step. Ideal case.
- E.g. separable Hamiltonian $H(\mathbf{q},\mathbf{p}) = U(\mathbf{q}) + V(\mathbf{p})$.
  - Vlasov equation in canonical coordinates reads

$$\frac{\partial f}{\partial t} + \nabla_p H \cdot \nabla_q f - \nabla_q H \cdot \nabla_p f = 0.$$

  - Split equations then become

$$\frac{\partial f}{\partial t} + \nabla_p V \cdot \nabla_q f = 0, \qquad \frac{\partial f}{\partial t} - \nabla_q U \cdot \nabla_p f = 0,$$

    where U does not depend on $\mathbf{p}$ and V does not depend on $\mathbf{q}$
    $\Rightarrow$ characteristics can be solved explicitly.
  - Vlasov-Poisson falls into this category with $\mathbf{q} = \mathbf{x}$, $\mathbf{p} = \mathbf{v}$, $H(\mathbf{x},\mathbf{v}) = \frac{1}{2}m\mathbf{v}^2 + q\phi(\mathbf{x},t)$.

Backward computation of characteristics in general case.

- Consider general case. Characteristics defined by

$$\frac{d\mathbf{X}}{dt} = \mathbf{a}(\mathbf{X}, t).$$

- Backward solution: $\mathbf{X}^{n+1}$ is known and $\mathbf{a}^n$ known on the grid. Does not allow standard ODE integrator.
- Numerical method for EDO can be derived using a quadrature formula on RHS of system by integrating on one time step, e.g. left or right rectangle rule for 1st order:

$$\mathbf{X}^{n+1} - \mathbf{X}^n = \Delta t\, \mathbf{a}^n(\mathbf{X}^n) \quad \text{or} \quad \mathbf{X}^{n+1} - \mathbf{X}^n = \Delta t\, \mathbf{a}^{n+1}(\mathbf{X}^{n+1}).$$

- No explicit solution:
  - Fixed point procedure needed in first case (e.g. Newton).
  - Predictor-corrector method on **a** needed in second case.

A two step second order method

- Solve characteristics defined by $\frac{d\mathbf{X}}{dt} = \mathbf{a}(\mathbf{X}, t)$.
- Centered quadrature on two time steps:

$$\mathbf{X}^{n+1} - \mathbf{X}^{n-1} = 2\Delta t\, \mathbf{a}^n(\mathbf{X}^n), \quad \mathbf{X}^{n+1} + \mathbf{X}^{n-1} = 2\mathbf{X}^n + O(\Delta t^2).$$

- Use fixed point procedure to compute $\mathbf{X}^{n-1}$ such that

$$\mathbf{X}^{n+1} - \mathbf{X}^{n-1} = \Delta t\, \mathbf{a}^n\left(\frac{\mathbf{X}^{n+1} + \mathbf{X}^{n-1}}{2}\right).$$

- Problem: compute f^{n+1} from f^{n-1}. Even and odd order time approximations become decoupled after some time. Artificial coupling needs to be introduced.

A one step predictor-corrector second order method

- Solve characteristics defined by $\frac{d\mathbf{X}}{dt} = \mathbf{a}(\mathbf{X}, t)$.
- Centered quadrature on one time step:

$$\mathbf{X}^{n+1} - \mathbf{X}^n = \Delta t\, \mathbf{a}^{n+\frac{1}{2}}(\mathbf{X}^{n+\frac{1}{2}}), \quad \mathbf{X}^{n+1} + \mathbf{X}^n = 2\mathbf{X}^{n+\frac{1}{2}} + O(\Delta t^2).$$

- Now $\mathbf{a}^{n+\frac{1}{2}}$ is unknown. But can be computed with first order scheme (like in Runge-Kutta methods) for global second order accuracy. Requires two updates of distribution function per time step.
- Use fixed point procedure to compute $\mathbf{X}^n$ such that

$$\mathbf{X}^{n+1} - \mathbf{X}^n = \Delta t\, \mathbf{a}^{n+\frac{1}{2}}\left(\frac{\mathbf{X}^{n+1} + \mathbf{X}^n}{2}\right).$$

- In practice use linear interpolation for evaluation of $\mathbf{a}^{n+\frac{1}{2}}(X)$ to get explicit solution for $\mathbf{X}^n$.

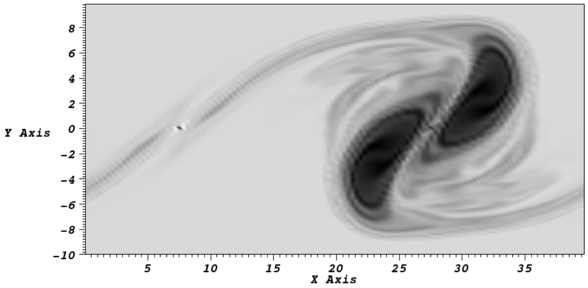
Conservativity

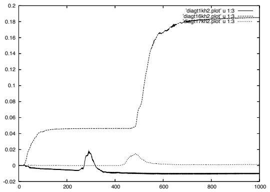
- Consider abstract Vlasov equation where $\mathbf{z}$ are all the phase space variables
$$\frac{\partial f}{\partial t} + \mathbf{a}(\mathbf{z}, t) \cdot \nabla_z f = 0 \quad \text{with } \nabla \cdot \mathbf{a} = 0.$$
- The equation is conservative: $\frac{d}{dt} \int f \, d\mathbf{z} = 0$.
- Consider splitting the equations by decomposing the variables into $\mathbf{z}_1$ and $\mathbf{z}_2$. Then the split equations read
$$\frac{\partial f}{\partial t} + \mathbf{a}_1(\mathbf{z}, t) \cdot \nabla_{z_1} f = 0, \quad \text{and} \quad \frac{\partial f}{\partial t} + \mathbf{a}_2(\mathbf{z}, t) \cdot \nabla_{z_2} f = 0.$$
- We have $\nabla \cdot \mathbf{a} = \nabla_{z_1} \cdot \mathbf{a}_1 + \nabla_{z_2} \cdot \mathbf{a}_2 = 0$, but in general $\nabla_{z_1} \cdot \mathbf{a}_1$ and $\nabla_{z_2} \cdot \mathbf{a}_2$ do not vanish separately.
- One or more of the split equations may not be conservative.

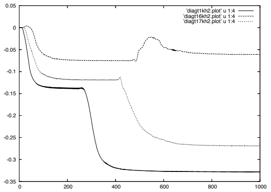
Problem with non conservative Vlasov solver

- When non conservative splitting is used for the numerical solver, the solver is not exactly conservative.
- Does generally not matter when solution is smooth and well resolved by the grid. The solver is still second order and yields good results.
- However: Fine structures develop in non linear simulations and are at some point locally not well resolved by the phase space grid.
- In this case a non conservative solvers can exhibit a large numerical gain or loss of particles which is totally unphysical.
- Lack of robustness.
- Classical SL solver can be made conservative at first order by appropriate coupling with field solver.

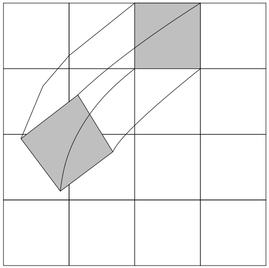
Vortex in Kelvin-Helmholtz instability







Conservative semi-Lagrangian method



- Start from conservative form of Vlasov equation

$$\frac{\partial f}{\partial t} + \nabla \cdot (f \mathbf{a}) = 0.$$

- $\int_V f \, dx \, dv$ conserved along characteristics
- Three steps:
  - High order polynomial reconstruction.
  - Compute origin of cells
  - Project (integrate) on transported cell.
- Efficient with splitting in 1D conservative equations as cells are then defined by their 2 endpoints. A lot more complex for 2D (or more) transport.
- Splitting on conservative form: always conservative.

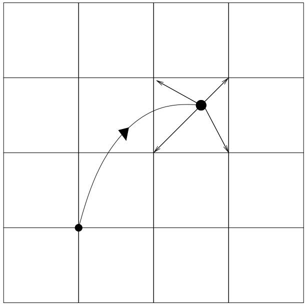
Link between classical and conservative semi-Lagrangian methods

- For constant coefficient advections it can be shown that

$$\text{C-Lag}(2d) \Longleftrightarrow \text{SL-Lag}(2d+1)$$
$$\text{PSM} \Longleftrightarrow \text{SPL}$$

- Consequences :
  1. Classical and conservative semi-Lagrangian methods equivalent for constant coefficients split equations.
  2. The PFC method (Filbet-ES-Bertrand, JCP 2001) corresponds for the Vlasov-Poisson (or Vlasov-Maxwell) systems to a classical semi-Lagrangian method with cubic Lagrange interpolation.

The forward semi-Lagrangian method



- f conserved along characteristics
- Characteristics advanced with same time schemes as in PIC method.
- Leap-Frog Vlasov-Poisson
- Runge-Kutta for guiding-center or gyrokinetic

- Values of f deposited on grid of phase space using convolution kernel.
- Identical to PIC deposition scheme but in whole phase space instead of configuration space only.
- Similar to PIC method with reconstruction introduced by Denavit (JCP 1972).

The δf semi-Lagragian method for Vlasov-Poisson

- To mitigate round-off and boundary errors, it can be useful to explicitely remove a known background distribution function from the actually computed function: $f = f^0 + \delta f$. E.g. $f^0(\mathbf{v}) = \frac{1}{\sqrt{2\pi}} e^{-v^2/2}$.
- Semi-Lagrangian with splitting.
- **x**-advection unchanged as f^0 does not depend on **x** and t.
- **v**-advection: $f = f^0 + \delta f$ conserved along characteristics:

$$\delta f^{n+1}(\mathbf{v}) = \delta f^n(\mathbf{V}(t_n; \mathbf{v}, t_{n+1})) + f^0(\mathbf{V}(t_n; \mathbf{v}, t_{n+1})) - f^0(\mathbf{v}).$$

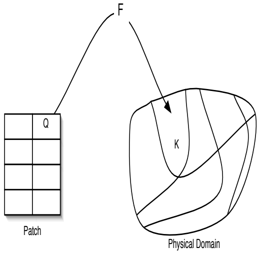
with $\mathbf{V}(t_n; \mathbf{v}, t_{n+1}) = \mathbf{v} - \mathbf{E}\Delta t$.

As f^0 explicitly known, only δf is interpolated.

- Poisson: $-\Delta \delta\phi = \int \delta f(t, \mathbf{x}, \mathbf{v})\, d\mathbf{v}$.

Mapped mesh

- A mapped mesh is defined by a mapping e.g. $F : \mathbb{R}^2 \to \mathbb{R}^2$ from a structured uniform logical mesh parametrized by (ξ_1, ξ_2) to the physical domain in cartesian coordinates
$$x_1 = F_1(\xi_1, \xi_2), \qquad x_2 = F_2(\xi_1, \xi_2).$$
- Mapping can be analytical coordinate transformation or defined by spline or NURBS curves.



- Using a given non cartesian coordinate system is one form of a mapped mesh.
- Can be more general. Full mapping not needed, only grid points and Jacobian matrix.
- Block structured mesh with one mapping in each block.

Vlasov equation in curvilinear coordinates

- Denote by A^i the contravariant (curvilinear) coordinates $A^i = A \cdot \nabla_x \xi_i$, with A velocity vector in cartesian coordinates.
- Advective form of Vlasov equation

$$\frac{\partial f}{\partial t} + \sum_k A^k \frac{\partial f}{\partial \xi_k} = 0.$$

- Conservative form of Vlasov equation

$$\sqrt{g}\frac{\partial f}{\partial t} + \sum_k \frac{\partial}{\partial \xi_k}\left(\sqrt{g}\ f\ A^k\right) = 0,$$

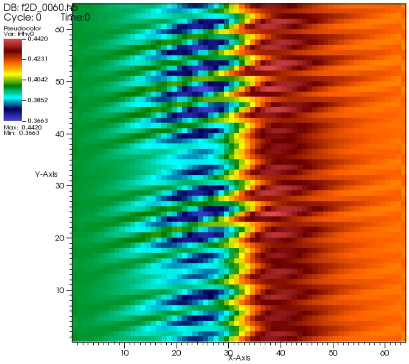
with the jacobian matrix $\sqrt{g} = det\left(\frac{D(x_1, x_2)}{D(\xi_1, \xi_2)}\right)$.

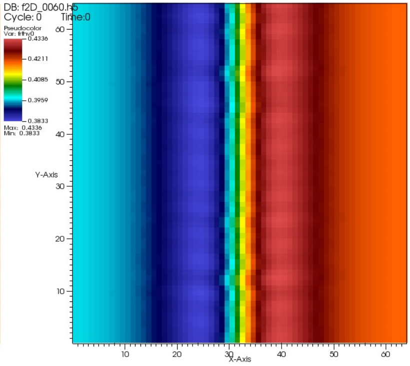
Specific problems on non cartesian meshes

- Two major issues when working on curvilinear meshes:
  1. Conservativity
  2. Free stream preservation (preservation of constant states).
- Classical SL scheme preserves constant states but is not conservative.
- Conservative SL scheme is conservative but does not preserve constant states.
- We like to do dimensional splitting to reduce the problem to a set of 1D problems for numerical efficiency. Not to be compatible with both exact conservation properties
- In practice both BSL and CSL work well, with conservation properties not enforced exactly but to lowest order in Δt.
- This is obtained by making sure that the advection field is exactly divergence free at the discrete level.

Problem with non free stream preservation

Drift-kinetic simulation





Advection field computed with cubic splines (left) and with one that satisfies the discrete divergence free condition (right).
4D simulation of 128x128x64x32 cells, result at time $t = 60$.

Selalib library

- Project started in December 2010 in France at Inria and university of Strasbourg.
- Main developers: E. Chacon-Golcher, P. Navaro, A. Back, M. Mehrenberger, ES and many others mainly in Strasbourg for now.
- Object oriented Fortran 2003.
- Aim: Develop a library to be used in physics codes for solving kinetic (including gyrokinetic) equations.
- Make parallel implementation transparent for the user.
- Curvilinear mesh by patches.
- Large panel of interpolation methods: Lagrange arbitrary order, spline arbitrary order, trigonometric, WENO
- Finite Element solvers on mapped meshes.
- Specific methods optimized, including accelerators (Cuda, OpenCL)

Conclusions

- Lagrangian (PIC), semi-Lagrangian (Lagrange + projection) and eulerian approaches have all been implemented to simulate GK Vlasov equations.
- Pros and cons for each method.
- PIC methods can be formulated in pure Monte Carlo formalism with the advantage of giving clear access to huge literature in this domain.
- PIC methods can be based on exact finite dimensional Lagrangian. Exact conservation properties in continuous time.
- Semi-Lagrangian method still uses characteristics and can be based on same Lagrangian, however projection step impedes conservation properties.