Gyrokinetic simulations of magnetic fusion plasmas

Tutorial 4

Virginie Grandgirard

CEA/DSM/IRFM, Association Euratom-CEA, Cadarache, 13108 St Paul-lez-Durance, France.
email: virginie.grandgirard@cea.fr

Acknowledgements: Yanick Sarazin

Time-splitting scheme

➠ Common to
Eulerian and Semi-Lagrangian
approaches

Time-splitting scheme concept (1/2)

- The Vlasov equation in its conservative form:

$$\frac{\partial}{\partial t} f(\mathbf{Z},t) + \mathrm{d}iv_{(x,v)}\left(\mathbf{U}(\mathbf{Z},t)\, f(\mathbf{Z},t)\right) = 0 \qquad (1)$$

- Then decomposing the components of **Z** into two sets **x** and **v**, the previous equation can then be written in the form

$$\frac{\partial}{\partial t} f(\mathbf{x},\mathbf{v},t) + \mathrm{d}iv_x\left(\mathbf{U}_x(\mathbf{x},\mathbf{v},t)\, f(\mathbf{x},\mathbf{v},t)\right) + \mathrm{d}iv_v\left(\mathbf{U}_v(\mathbf{x},\mathbf{v},t)\, f(\mathbf{x},\mathbf{v},t)\right) = 0$$

where $\mathbf{U}_x$ and $\mathbf{U}_v$ are the components of the advection field.

Time-splitting scheme concept (2/2)

- A method currently used is to split the both parts by what is called a splitting method (or an operator decomposition)

$$\frac{\partial}{\partial t} f(\mathbf{x},\mathbf{v},t) + \mathrm{d}iv_x\left(\mathbf{U}_x(\mathbf{x},\mathbf{v},t)\, f(\mathbf{x},\mathbf{v},t)\right) = 0 \quad \text{with fixed } \mathbf{v} \quad (2)$$

and

$$\frac{\partial}{\partial t} f(\mathbf{x},\mathbf{v},t) + \mathrm{d}iv_v\left(\mathbf{U}_v(\mathbf{x},\mathbf{v},t)\, f(\mathbf{x},\mathbf{v},t)\right) = 0 \quad \text{with fixed } \mathbf{x} \quad (3)$$

Constraint on the advective form (1/3)

- Rk: SL scheme does not solve Vlasov equation in the conservative form, but in the advective form

- Eqs. (2) and (3) can be put in the advective form

$$\frac{\partial f}{\partial t} + \mathbf{U}_x \cdot \nabla_x f = 0 \tag{4}$$

$$\frac{\partial f}{\partial t} + \mathbf{U}_v \cdot \nabla_v f = 0 \tag{5}$$

If and only if both conditions hold

$$\mathrm{d}iv_x(\mathbf{U}_x(\mathbf{x}, \mathbf{v}, t) = 0 \tag{6}$$

$$\mathrm{d}iv_v(\mathbf{U}_v(\mathbf{x}, \mathbf{v}, t) = 0 \tag{7}$$

Constraint on the advective form (2/3)

- On the contrary, if equations (6) and (7) are not satisfied, then splitting Eq. (1) is equivalent to solve advective equations with a source term

$$\frac{\partial f}{\partial t} + \mathbf{U}_x \cdot \nabla_x f = -f \mathrm{d}iv_x(\mathbf{U}_x) \tag{8}$$
$$\frac{\partial f}{\partial t} + \mathbf{U}_v \cdot \nabla_v f = -f \mathrm{d}iv_v(\mathbf{U}_v) \tag{9}$$

- Although from the divergence-free property of the full advection field $\mathbf{U}$, we have

$$\mathrm{d}iv_x(\mathbf{U}_x) = -\mathrm{d}iv_v(\mathbf{U}_v)$$

  The source term in Eqs. (8) and (9) do not vanish exactly since Eqs. (8) and (9) are not solved at the same time.

Constraint on the advective form (3/3)

- This will therefore introduce a cumulative systematic error at each time step, resulting in poor density conservation.

➠ A necessary condition for the time-splitting to preserve the conservative character is that the advections fields $\mathbf{U}_x$ and $\mathbf{U}_v$ are both divergence free.

- For instance in GYSELA , this constraint limit the decomposition in the (r, θ) direction
$\Rightarrow$ need to treat a 2D problem and not a 1D problem in r + a 1D problem in θ.

Formal writing of Vlasov equation (1/2)

- To estimate the splitting error, let consider that Vlasov equation can be formally written as

  $$\frac{\mathrm{d}f}{\mathrm{d}t} = (A+B)f \tag{10}$$

  where A and B are any differential operators in space (In our case, it would be $A = \mathbf{v}\cdot\nabla_{\mathbf{x}}$ and $B = \mathbf{E}\cdot\nabla_{\mathbf{v}}$ because $\mathbf{U}_x = \mathbf{v}$ and $\mathbf{U}_v = \mathbf{E}$), assumed constant between t_n and t_{n+1}.

- The formal solution of this equation (10) on a Δt time step reads

  $$f(t+\Delta t) = \exp\left[\Delta t(A+B)\right] f(t)$$

Formal writing of Vlasov equation (2/2)

- Then, the equation (10) is split in two formal equations

$$\frac{\mathrm{d}f}{\mathrm{d}t}(t) = Af(t) \tag{11}$$

$$\frac{\mathrm{d}f}{\mathrm{d}t}(t) = Bf(t) \tag{12}$$

- and the solutions are respectively

$$f(t+\Delta t) = \exp(\Delta t A) f(t) \quad \text{and}$$
$$f(t+\Delta t) = \exp(\Delta t B) f(t)$$

Standard splitting method

- The standard splitting method consists in solving successively
  1. Eq. (11) on a time step Δt followed by
  2. Eq. (12) also on a time step Δt

- which gives on one time step

$$\tilde{f}(t+\Delta t) = \exp(\Delta t B)\exp(\Delta t A) f(t)$$

- If the operators A and B commute the splitting is exact because

$$\exp(\Delta t B)\exp(\Delta t A) = \exp\left[\Delta t (A+B)\right]$$

- This is true for the particular case where advections are with constant coefficients.

Strang splitting (1/2)

- However, in most of the cases A and B do not commute,
- But Strang has proved *[Strang, SIAM (1968)]* that the splitting error can be reduce by solving symmetrically
  1. Eq. (11) on half a time step $\Delta t/2$, then
  2. Eq. (12) on a time step Δt and finally again
  3. Eq. (11) on half a time step $\Delta t/2$.
- This method is called Strang splitting.
- It corresponds to the formal solution

$$\tilde{f}(t+\Delta t) = \exp(\frac{\Delta t}{2}A)\exp(\Delta t B)\exp(\frac{\Delta t}{2}A)f(t)$$

Strang splitting (2/2)

- The splitting error on one time step, when the operators do not commute,
  - is of order 1 in time for the standard splitting
  - is of order 2 in time for the Strang splitting
- The proof can be simply find by using a Taylor expansion of the formal solutions
- The Strang splitting can be generalized to more than two operators. If $A = A_1 + \cdots + A_n$, the following decomposition gives a global order of 2 in time

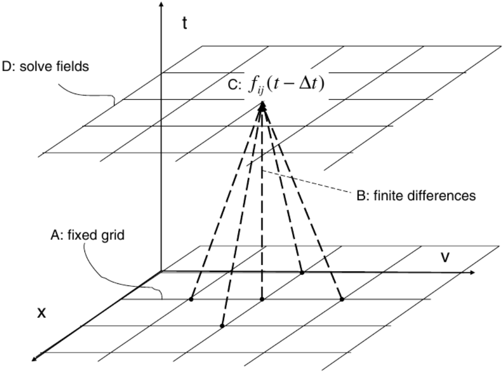
$$\exp(\frac{\Delta t}{2}A_1)\cdots\exp(\frac{\Delta t}{2}A_{n-1})\exp(\Delta t A_n)\exp(\frac{\Delta t}{2}A_{n-1})\cdots\exp(\frac{\Delta t}{2}A_1)$$

Strang splitting commonly used

- The advantage of such a method is that it change a 6D problem into a succession of 1D advection equations or 1D conservative equations.
- This technique has been used more than thirty years ago to solve a 2D Vlasov equation *[Shoucri, JOCP (1978)]*
- It is now currently used in Eulerian and semi-Lagrangian approaches
- Strang splitting has made possible the development of Eulerian and semi-Lagrangian codes of high dimensionality.

Eulerian approach

Eulerian method



In Eulerian methods, a fixed grid is defined in phase space (A). Finite difference expressions are used (B) in order to obtain the value of f at grid points at the next time step (C). Field equations are then solved (D) after integration over velocity space. (figure from *[Idomura, CR (2006)]*)

Eulerian schemes ➠ CFL conditions

- Due to the increase of HPC capabilities, Eulerian schemes have been more and more preferred to PIC approach, because they are not subject to the issue of marker sampling noise
- On the other hand, when explicit time integration is applied they are subject to the CFL stability condition (the maximum time step depends on grid space resolution)
- A way to improve this CFL condition as been find by using implicit or semi-implicit time integration schemes *[Idomura, NF (2009)]*
- This chronological list of Eulerian GK codes (not exhaustive): GS2 *[Kotschenreuther, CPC (1995)] and [Dorland, PRL (2000)*, GYRO *[Candy, JOCP (2003)]*, GENE *[Jenko, CPC (2000)]*, GKV *[Watanabe, NF (2006)]*, GKW *[Peeters, PoP (2004)]* and GT5D *[Idomura, CPC(2008)]*.

Advective and convective forms in 1D

- Therefore we will restrict ourselves, without loss of generality, to a 1D problem which leads to solving the following advection equation

$$\partial_t f + u(x,t)\partial_x f = 0, \qquad \forall (x,t) \in [x_{min}, x_{\max}] \times \mathbb{R}+ \tag{13}$$

  where we will assume that $u(x,t)$ is smooth enough; for instance u is Lipschitz continuous.

- Besides, taking into account the hypothesis $\partial u(x,t) = 0$, we will work with the 1D conservative form

$$\partial_t f + \partial_x(u(x,t)f) = 0, \qquad \forall (x,t) \in [x_{min}, x_{\max}] \times \mathbb{R}+ \tag{14}$$

Characteristic curves in 1D

- Then we can define the characteristic curves solution of the differential system corresponding to the transport equation

$$\begin{cases} \frac{\mathrm{d}X}{\mathrm{d}t}(t) = u(x(t),t) \\ X(s) = x \end{cases} \tag{15}$$

- Let us denote the solution of Eq. (15) by

$$X(t; x, s)$$

i.e. the position of a point at time t knowing that its position is x at time s.

Definitions of cells, cell centers and cell sizes

- Let us also introduce a finite set of mesh points $\{x_{i+1/2}\}_{i\in I}$ of the computational domain $[x_{\mathrm{m}in}, x_{\mathrm{m}ax}]$ as

$$x_{\mathrm{m}in} = x_{\frac{1}{2}} < x_{\frac{3}{2}} < \cdots < x_{N-\frac{1}{2}} < x_{N+\frac{1}{2}} = x_{\mathrm{m}ax}$$

- and denote cells, cell centers and cell sizes as, for $i = 1, 2, \cdots, N$, by

$$I_i \equiv \left[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}\right]; x_i \equiv \frac{1}{2}\left(x_{i-\frac{1}{2}} + x_{i+\frac{1}{2}}\right); \Delta x_i \equiv x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}$$

- and the maximum cell size by

$$\Delta x \equiv \max_{1\le i\le N} \Delta x_i$$

Properties of positivity and TVD (1/2)

- At first time, let suppose that u is not a function of f or x,

➠ The evolution of the advection equation for a time Δt

$$\partial_t f + u(x,t)\partial_x f = 0$$

is simply a uniform shift of the distribution function by a displacement $u\Delta t$.

- Let us also define λ as the CFL number

$$\lambda = u\frac{\Delta t}{\Delta x}$$

Properties of positivity and TVD (1/2)

- There are two particular properties of great importance in an advection solver:
  (a) The method should not introduce false extrema, i.e.:

$$\text{if} \quad 0 < \lambda < 1 \quad \text{and} \quad f_{i-1}^n < f_i^n < f_{i+1}^n \quad \text{then} \quad f_{i-1}^{n+1} < f_i^{n+1} < f_{i+1}^{n+1}$$

(and similarly for the monotone deacreasing),

  (b) The method should not accentuate already existing extrema, i.e.

$$\text{for} \quad 0 < \lambda < 1, \quad \text{if} \quad f_{i-1}^n < f_i^n > f_{i+1}^n \quad \text{then} \quad f_i^n \geq \max\{f_i^{n+1}, f_{i+1}^{n+1}\}$$

- Properties (a) and (b) together imply the method is positivity-preserving and total variation diminushing (TVD).

Properties of positivity and TVD (2/2)

- The problem is that f is only known at a set of discrete grid points $x_j = j\Delta x$.
- One solution would be to interpolate between the grid points by some function $\tilde{f}(y)$ and write

$$f(y_i, t + \Delta t) = \tilde{f}(y_i - \lambda, t), \quad \text{where } y = x/\Delta x \quad (\text{i.e. } y_i = i)$$

- However, by a corollary of Godunov's theorem, any interpolation scheme that is higher than first order breaks properties (a) and (b).
- Satisfying these properties has been one of the leitmotiv for the development of new schemes since more than 10 years.

Replace $f(x,t)$ by its smoothed approximation

- Let us come back to the general case of a non constant $u(x,t)$ coefficient.
- Our goal is to solve equation of type Eq. (14) on a grid, assuming the function f to be smooth in each elementary cell I_i.
- The main idea is to replace $f(x,t)$ by its smoothed approximation

$$\bar{f}_i(t) = \frac{1}{\Delta x_i} \int_{x_{i-1/2}}^{x_{i+1/2}} f(x,t) dx, \quad i = 1, 2, \cdots, N \tag{16}$$

Property of particle conservation

- Assuming the values of the approximation of the distribution function $\bar{f}_i$ known at time $t^n = n\Delta t$ for all $i = 1, \cdots, N$,

➠ we find the new values of $\bar{f}_i$ at time t^{n+1} by using the property of particle conservation and recalling that the Jacobian function $J(t^{n+1}; x, t^n)$ is positive, as

$$\bar{f}_i^{n+1} = \bar{f}_i(t^{n+1}) = \frac{1}{\Delta x_i} \int_{X(t^{n+1}; x_{i-1/2}, t^n)}^{X(t^{n+1}; x_{i+1/2}, t^n)} f(x,t)dx \qquad (17)$$

- Let us also define,

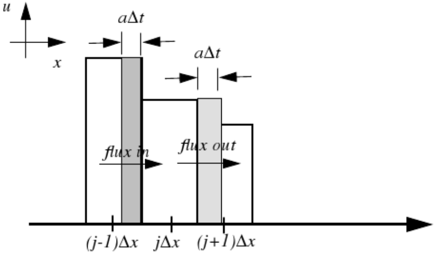
$$\Phi_{i+1/2}(t^n) = \int_{X(t^{n+1}; x_{i+1/2}, t^n)}^{x_{i+1/2}} f(x,t)dx \qquad (18)$$

- Then we finally obtain the conservative scheme

$$\bar{f}_i^{n+1} = \bar{f}_i^n - \frac{1}{\Delta x_i} \left[\Phi_{i+1/2}(t^n) - \Phi_{i-1/2}(t^n)\right] \qquad (19)$$

Interpretation of Φ

- Let us note α_i by
$$\alpha_i = x_{i+1/2} - X(t^{n+1}; x_{i+1/2}, t^n)$$



- Then $\Phi_{i+1/2}$ can be rewritten as
$$\Phi_{i+1/2}(t^n) = \int_{x_{i+1/2}-\alpha_i}^{x_{i+1/2}} f(x,t)dx$$

- Then, for the case $\alpha_i \geq 0$,
    - $\Phi_{i+1/2}$ represents the decrease of $\bar{f}_i(t)$ due to loss of fluid to the $(i+1)$-th cell
    - $\Phi_{i-1/2}$ is the gain from the $(i-1)$-th cell and is equal to the fluid the $(i-1)$-th cell looses.

Reconstruction of the distribution function

- An essential step is now to choose an efficient method to reconstruct f from the values on each cell I_i.
- In the Flux Balance Method (FBM), Fijalkow *[CPC (1999)]* proposed to use a linear interpolation for this reconstruction, i.e.:

$$D_i = \frac{f_{i+1} - f_{i-1}}{2\Delta x_i}$$

$$f_h(x) = f_i + D_i(x - x_i), \qquad \forall x \in [x_{i-1/2}, x_{i+1/2}]$$

then

$$\Phi_{i+1/2}(t^n) = \int_{x_{i+1/2}-\alpha_i}^{x_{i+1/2}} f_h(x,t)dx = \alpha_i f(x_i) + \frac{D_i}{2}\alpha_i(\Delta x_i - \alpha_i)$$

FBM improvements

- FBM scheme is only second-order accurate in space and there is no guarantee that it either preserves monotonicity or does not introduce false extrema.
- However this method has been shown to be successful for a variety of Vlasov problems *[Fijalkow, CPC (1999)]* when it is coupled with smoothing and averaging techniques to dissipate fine-scale structure.
- It is also the start point of lot of eulerian methods.
- Several improvements have been for instance proposed by Arber and Vann *[Arber, JOCP (2002)]* in two directions
  - (i) increasing the order of the gradient D_i
  - (ii) applying for instance Van-Leer gradient limiter to the gradient D_i before calculating the cell boundary fluxes

Reconstruction by the primitive

- Another solution to reconstruct f on each cell is to use a reconstruction via a primitive function.
- Let $F(x, t^n)$ be a primitive of the distribution function $f(x, t^n)$.
- Then using $\bar{f}_i$, the averaged value of f on a cell I_i, defined as previously by:

$$\bar{f}_i(t) = \frac{1}{\Delta x_i} \int_{x_{i-1/2}}^{x_{i+1/2}} f(x, t) dx, \quad i = 1, 2, \cdots, N$$

- We obtain

$$F(x_{i+\frac{1}{2}}, t_n) - F(x_{i-\frac{1}{2}}, t_n) = \Delta x_i \bar{f}_i^n$$

and

$$F(x_{i+\frac{1}{2}}, t_n) = \sum_{k=0}^{i} \Delta x_k \bar{f}_k^n$$

Positive Flux Conservative scheme (PFC)

- This method has been introduced in 2001 by Filbet *[JOCP (2001)]*.
- It is based on a reconstruction via primitive function using a Lagrange polynomial.
- For instance for a polynomial of degree two on the interval $[x_{i-1/2}, x_{i+1/2}]$, it leads to the following approximation:

$$f_h(x) = f_i + \epsilon_i (x - x_i) \frac{f_{i+1} - f_i}{\Delta x}, \quad \forall x \in [x_{i-1/2}, x_{i+1/2}] \qquad (20)$$

where ϵ_i the slope corrector is defined as

$$\epsilon_i = \begin{cases} \min(2(f_i - f_{min}); (f_{i+1} - f_i)) & \text{if } f_{i+1} - f_i \geq 0 \\ \min(-2(f_{max} - f_i); (f_{i+1} - f_i)) & \text{if } f_{i+1} - f_i < 0 \end{cases} \qquad (21)$$

where $f_{min} = 0$ and $f_{max} = f_\infty$.

PFC properties

- This approximation satisfies the conservation of the average

$$\text{for all } i \in I, \quad \int_{x_{1-1/2}}^{x_{1+1/2}} f_h(\xi)\,\mathrm{d}\xi = \Delta x\, f_i$$

- Its advantage compared for instance to the ENO method is that it preserves the positivity

$$\text{for all } x \in [x_{min}, x_{\max}], \quad 0 \leq f_h(x) \leq f_\infty$$

- The drawback of such a method, as seen in *[Grandgirard, JOCP (2006)]* applied to 4D drift-kinetic ITG simulations, is that it is dissipative and can lead to a loss of conservation of the total energy.

Modified versions of PFC (1/2)

- Several modified versions of the PFC method have been proposed.
- Schmitz and Grauer *[Schmitz, JOCP (2006)]* have modified the limiter, using $f_{max} = \infty$,
  - ➠ to avoid that the maximum value of the profile decreases and to allow the profile to rise uncontrollably.
- However as the origin one, in this method the profile has a local maximum and/or minimum and is not necessarily non-oscillatory.

Modified versions of PFC (2/2)

- Umeda *[EPS (2008)]*, has defined a non-oscillatory scheme (Positive Interpolation for Conservation), in the sense that the already-existing extrema are kept, by changing the extrema of Filbet's slope corrector as

$$f_{\mathrm{m}ax} = \max[f_{\mathrm{m}ax1}; f_{\mathrm{m}ax2}]$$
$$f_{\mathrm{m}in} = \max[f_{\mathrm{m}in1}; f_{\mathrm{m}in2}]$$

where
$$\begin{aligned} f_{\mathrm{m}ax1} &= \max[\max[f_{i-1}, f_i]; \min[2f_{i-1} - f_{i-2}; 2f_i - f_{i+1}]], \\ f_{\mathrm{m}ax2} &= \max[\max[f_{i+1}, f_i]; \min[2f_{i+1} - f_{i+2}; 2f_i - f_{i-1}]], \\ f_{\mathrm{m}in1} &= \min[\min[f_{i-1}, f_i]; \max[2f_{i-1} - f_{i-2}; 2f_i - f_{i+1}]], \\ f_{\mathrm{m}in2} &= \min[\min[f_{i+1}, f_i]; \max[2f_{i+1} - f_{i+2}; 2f_i - f_{i-1}]] \end{aligned}$$

- This new PFC scheme has been applied successfully to nonlinear beam-plasma interactions *[Ryu, PoP (2007)]*.

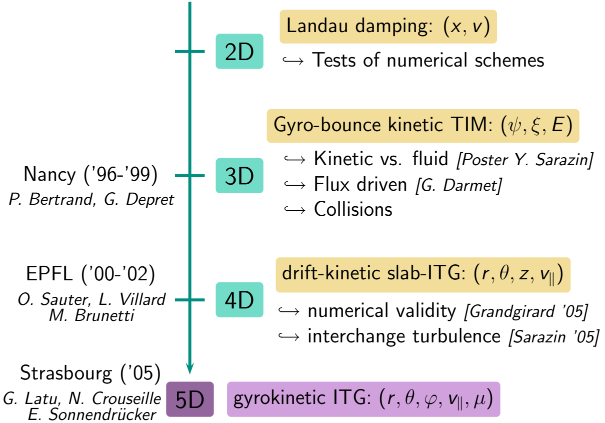
Morinishi scheme

- Morinishi et al *[Morinishi, JOCP (1998)]* proposed a new finite difference operator, which can be extended to higher order accuracy
- The advantage of this scheme is that it conserves both the momentum and kinetic energy by construction
- The drawback is that you need to store and evolve f but also ∇f
- This scheme has been successfully implemented in GT5D

Semi-Lagrangian approach

A family of Semi-Lagrangian codes at IRFM

Motivations: show that semi-lagrangian scheme offers a credible alternative for gyrokinetic turbulence simulations.



2D

Landau damping: (x, v)

↪ Tests of numerical schemes

Nancy ('96-'99)
P. Bertrand, G. Depret

3D

Gyro-bounce kinetic TIM: (ψ, ξ, E)

↪ Kinetic vs. fluid *[Poster Y. Sarazin]*

↪ Flux driven *[G. Darmet]*

↪ Collisions

EPFL ('00-'02)
O. Sauter, L. Villard
M. Brunetti

4D

drift-kinetic slab-ITG: $(r, \theta, z, v_{\|})$

↪ numerical validity *[Grandgirard '05]*

↪ interchange turbulence *[Sarazin '05]*

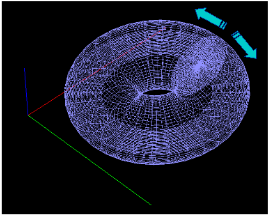
Strasbourg ('05)
G. Latu, N. Crouseille
E. Sonnendrücker

5D

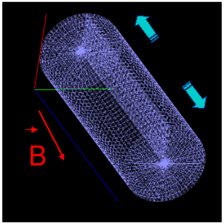
gyrokinetic ITG: $(r, \theta, \varphi, v_{\|}, \mu)$

Simplified model : 4D drift-kinetic ion turbulence (1/2)

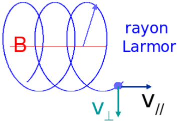
toric geometry



cylindrical geometry



- adiabatic electrons
- guiding center assumption
- uniform field $\vec{B}$ along z
- adiabatic invariant $\mu = \frac{mv_{\perp}^2}{2B} = cte$



$$\Downarrow$$

ion 4D distribution function : $f(r, \theta, z, v_{\parallel})$

Simplified model : 4D drift-kinetic ion turbulence (2/2)

A. Drift-kinetic equation (4*D*)

$$\boxed{\partial_t f + \mathbf{v}_E \cdot \boldsymbol{\nabla}_\perp f + v_\| \nabla_\| f + \dot{v}_\| \, \partial_{v_\|} f = \underbrace{0}_{\textit{no collisions}}}$$

$$\mathbf{v}_E = \frac{\mathbf{B} \times \boldsymbol{\nabla}\phi}{B^2} \; , \; \dot{v}_\| = -\frac{e}{m} \nabla_\| \phi \quad \text{with } \nabla_\perp = \left(\frac{\partial .}{\partial r}, \frac{1}{r}\frac{\partial .}{\partial \theta}\right) \text{ and } \nabla_\| = \partial_z$$

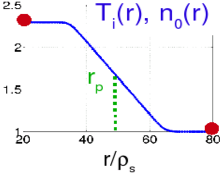
B. Quasi-neutrality equation (3*D*) $\delta n_e = \delta n_i \equiv n - n_{init}$

$$\boxed{-\frac{1}{n_0(r)} \boldsymbol{\nabla}_\perp \cdot \left[\frac{n_0}{B_0 \, \omega_c} \boldsymbol{\nabla}_\perp \phi \right] + \frac{e}{T_e(r)} \left(\phi - \langle \phi \rangle \right) = \frac{1}{n_0} \int dv_\| \left(f - f_{init} \right)}$$

$\langle \cdot \rangle \equiv$ flux surface average

Boundary and initial conditions

- Boundary conditions:
  - Periodic in θ and φ
  - Vanishing perturbations in non-periodic directions r and $v_{\|}$
- Initial conditions:
  - no source
    $\hookrightarrow$ profile relaxation

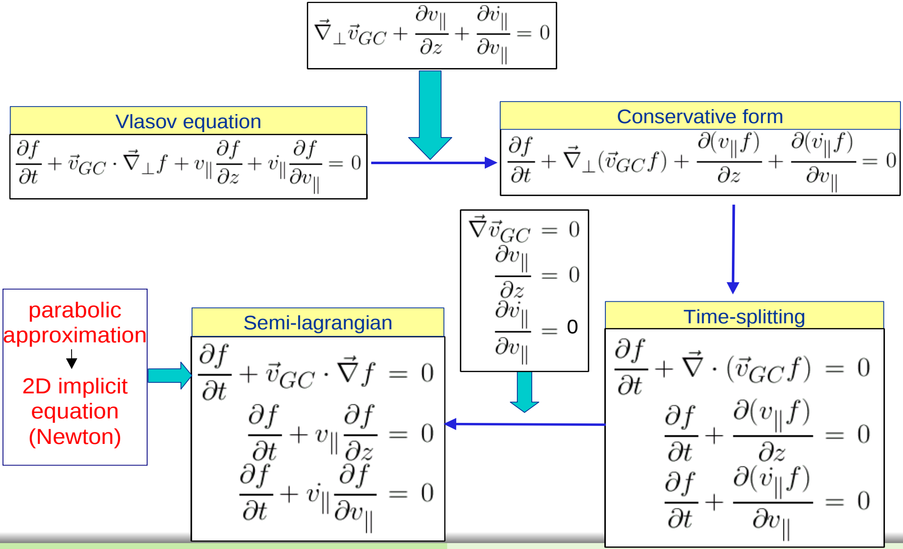


  - Initialisation of the global distribution function

$$\boxed{f = f_{eq}\left(1 + \delta f\right)}$$

$$f_{eq}(r, v_{\|}) = \frac{n_0(r)}{\left(2\pi T_i(r)/m_i\right)^{\frac{1}{2}}} \exp\left(-\frac{m_i v_{\|}^2}{2 T_i(r)}\right), \delta f = \sum_{m,n} \epsilon_{mn} \cos\left(\frac{2\pi n}{L_z} z + m\theta + \phi_{mn}\right)$$

How to apply a SL scheme ?



$$\vec{\nabla}_\perp \vec{v}_{GC} + \frac{\partial v_\parallel}{\partial z} + \frac{\partial \dot{v}_\parallel}{\partial v_\parallel} = 0$$

Vlasov equation

$$\frac{\partial f}{\partial t} + \vec{v}_{GC} \cdot \vec{\nabla}_\perp f + v_\parallel \frac{\partial f}{\partial z} + \dot{v}_\parallel \frac{\partial f}{\partial v_\parallel} = 0$$

Conservative form

$$\frac{\partial f}{\partial t} + \vec{\nabla}_\perp (\vec{v}_{GC} f) + \frac{\partial (v_\parallel f)}{\partial z} + \frac{\partial (\dot{v}_\parallel f)}{\partial v_\parallel} = 0$$

$$\begin{aligned} \vec{\nabla} \vec{v}_{GC} &= 0 \\ \frac{\partial v_\parallel}{\partial z} &= 0 \\ \frac{\partial \dot{v}_\parallel}{\partial v_\parallel} &= 0 \end{aligned}$$

Time-splitting

$$\begin{aligned} \frac{\partial f}{\partial t} + \vec{\nabla} \cdot (\vec{v}_{GC} f) &= 0 \\ \frac{\partial f}{\partial t} + \frac{\partial (v_\parallel f)}{\partial z} &= 0 \\ \frac{\partial f}{\partial t} + \frac{\partial (\dot{v}_\parallel f)}{\partial v_\parallel} &= 0 \end{aligned}$$

Semi-lagrangian

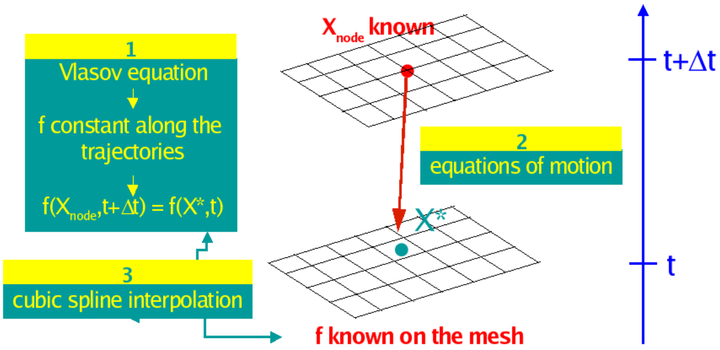
$$\begin{aligned} \frac{\partial f}{\partial t} + \vec{v}_{GC} \cdot \vec{\nabla} f &= 0 \\ \frac{\partial f}{\partial t} + v_\parallel \frac{\partial f}{\partial z} &= 0 \\ \frac{\partial f}{\partial t} + \dot{v}_\parallel \frac{\partial f}{\partial v_\parallel} &= 0 \end{aligned}$$

parabolic approximation

↓

2D implicit equation (Newton)

The backward semi-lagrangian Method



1

Vlasov equation

↓

f constant along the trajectories

↓

$f(X_{node},t+\Delta t) = f(X^*,t)$

2

equations of motion

3

cubic spline interpolation

Two important steps:

1. Computation of characteristic feet
2. High order interpolation needed ➔ Cubic splines

❶ Computation of characteristic feet

Problem: Solve backward in time, the characteristics defined by:

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

☛ A two step second order method:

- Centered quadrature on two time steps:

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

- Use fixed point procedure to compute X^{n-1} such that:

$$\mathbf{X}^{n+1} - \mathbf{X}^{n-1} = 2\Delta t\,\mathbf{u}^n(\frac{\mathbf{X}^{n+1} + \mathbf{X}^{n-1}}{2})$$

- 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 in the global leap-frog algorithm.

❷ Interpolation by cubic splines

High order interpolation needed to interpolate old value at origin of characteristics from known grid values.

☛ Cubic splines: Represent $g(x)$ in terms of piecewise cubic polynomials Λ_α, twice continuously differentiable, i.e: $g(x) \simeq s(x) = \sum_{\alpha=-1}^{N_x+1} c_\alpha \Lambda_\alpha(x)$

$$\text{where} \qquad \Lambda_\alpha(x) = \frac{1}{6h^3} \begin{cases} (x - x_{\alpha-2})^3 & \text{if } x_{\alpha-2} \le x \le x_{\alpha-1} \\ h^3 + 3h^2(x - x_{\alpha-1}) + 3h(x - x_{\alpha-1})^2 & \\ \quad -3(x - x_{\alpha-1})^3 & \text{if } x_{\alpha-1} \le x \le x_\alpha \\ h^3 + 3h^2(x_{\alpha+1} - x) + 3h(x_{\alpha+1} - x)^2 & \\ \quad -3(x_{\alpha+1} - x)^3 & \text{if } x_\alpha \le x \le x_{\alpha+1} \\ (x_{\alpha+2} - x)^3 & \text{if } x_{\alpha+1} \le x \le x_{\alpha+2} \\ 0 & \text{otherwise} \end{cases}$$

- ☺ A good compromise between between accuracy and simplicity
- ☹ Value of f on one grid point requires f over the whole grid
  - ➥ Spline coefficients requires tridiagonal system solve on each line of mesh.

The GYSELA code - global algorithm

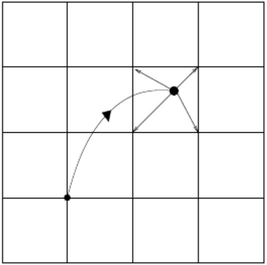
For each μ:

global algorithm of second order in time : leap-frog

- Solving of the non-linear Vlasov equation (4D):
  - Time-splitting of Strang :
    4D $(r,\theta,\varphi,v_{\|}) \Rightarrow$ 2D (r,θ) + 1D (φ) et 1D $(v_{\|})$
  - Semi-lagrangian
    - $\Rightarrow$ trajectories (Newton algorithm for (r,θ))
    - $\Rightarrow$ interpolation by cubic splines
- Solving of the quasi-neutrality equation (3D) :
  - Fourier in θ and φ + finite differences in r
- Gyroaverage ($J_0 \Leftrightarrow$ Padé approximation):
  - Fourier in θ + finite differences in r

end for

Forward Semi-Lagrangian method

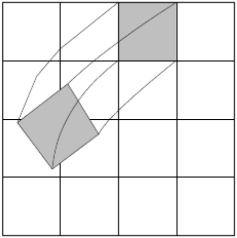


- *f* conserved along characteristics
- Characteristics advanced with same time schemes as in PIC method.
- Leap-Frog Vlasov-Poisson
- Runge-Kutta for guiding-center or gyrokinetic

- 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).

[Crouseilles, Respaud, Sonnendrücker (2008)]

Conservative semi-Lagrangian scheme (PSM)



- Start from conservative form of Vlasov equation

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

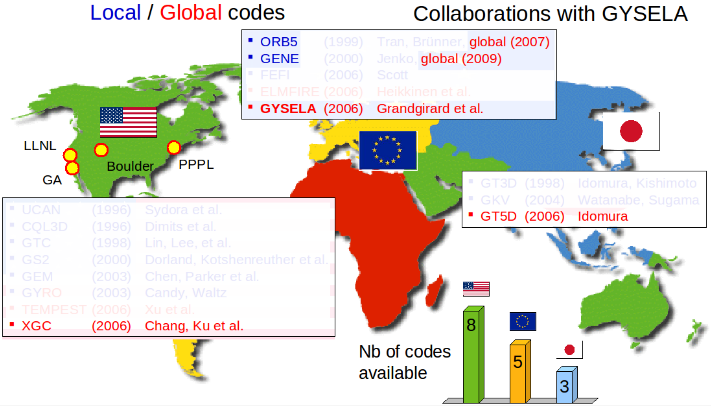
- $\int_V f \, dx \, dv$ conserved along characteristics
- Three steps:
  - High order polynomial reconstruction.
  - Compute origin of cells
  - Project (integrate) on transported cell.
- Efficient with splitting in 1D conservative equations as cells are then defined by their 2 endpoints. A lot more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

[Crouseilles, Mehrenberger, Sonnendrücker (2009)]

➠ Implementation in GYSELA in progress *by J.P Braeunig*

GK codes: a highly competitive activity

- No miraculous scheme ➠ all have their advantages and drawbacks
- Its important to continue to progress in the three directions
- More and more collaborations ➠ ↗ benchmark activity



Local / Global codes

Collaborations with GYSELA

▪ ORB5	(1999)	Tran, Brünner, global (2007)
▪ GENE	(2000)	Jenko, global (2009)
▪ FEFI	(2006)	Scott
▪ ELMFIRE	(2006)	Heikkinen et al.
▪ GYSELA	(2006)	Grandgirard et al.

▪ UCAN	(1996)	Sydora et al.
▪ CQL3D	(1996)	Dimits et al.
▪ GTC	(1998)	Lin, Lee, et al.
▪ GS2	(2000)	Dorland, Kotshenreuther et al.
▪ GEM	(2003)	Chen, Parker et al.
▪ GYRO	(2003)	Candy, Waltz
▪ TEMPEST	(2006)	Xu et al.
▪ XGC	(2006)	Chang, Ku et al.

▪ GT3D	(1998)	Idomura, Kishimoto
▪ GKV	(2004)	Watanabe, Sugama
▪ GT5D	(2006)	Idomura

Where are we in physics with GYSELA ?

1. Full-f
2. Collisions
3. Flux-driven simulations

Main issues for full-f gyrokinetic codes

- Full-F code ($F = F_{eq} + \delta F$) ➠ Time evolution of F_{eq} is retained

$$\partial_t F - \{H, F\} = \mathcal{C}(F) + S$$

[Sarazin-Grandgirard (GYSELA) 2009;
Angelino-Bottino (ORB5) 2008;
Idomura (GT5D) 2008;
Chang (XGC) 2008]

$$\Updownarrow$$

$$\begin{cases} \partial_t \delta F - \{H, \delta F\} = \{\delta H, F_{eq}\} + \mathcal{C}(\delta F) \\ \partial_t F_{eq} - \langle\{\delta H, \delta F\}\rangle = \mathcal{C}(F_{eq}) + S \end{cases}$$

- Main critical issues:
  - Collision operator: ensures equilibrium Maxwellian & recovers neoclassical theory
  - Source to sustain main profile (decaying turbulence otherwise)

Why collisions play an important role ?

➠ Core plasmas are widely believed to be collisionless, but recent evidence from experimental side and from GK modelisation are contributing to modify this classical idea.

➠ Great interest to the adjunction of NC theory to GK models:

❶ Regularisation of filamentation in the velocity phase space

$\hookrightarrow$ GK equation continuously produces fine scale structures due to linear and nonlinear mixing effects.

❷ Improvement of confinement regimes = regimes where turbulence can be locally suppressed

$\hookrightarrow$ NC transport becomes dominant

❸ Fondamental role of plasma rotation

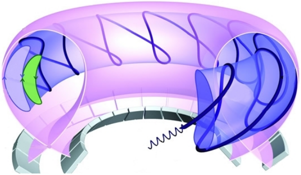
$\hookrightarrow$ poloidal rotation believed to be set by NC theory

❹ Impact on the turbulence

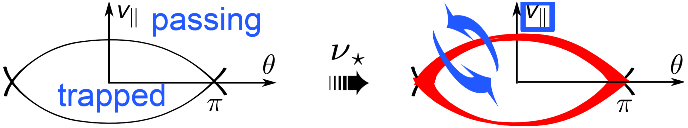
$\hookrightarrow$ ion-ion collisions enhance the damping of zonal flows.

Hand-waving physics for neoclassical theory

Neoclassical theory $\equiv$ describes the effects of binary Coulomb collisions in an inhomogeneous **B** field in the presence of trapping



$$\mathcal{H} = \frac{mv_\parallel^2}{2} + \mu B_0 - \underbrace{\epsilon \mu B_0 \cos\theta}_{\text{perturb.}}$$ ➠ island



- Inhomogeneities of **B** field ➠ local trapping
- without collisions: non-interacting populations trapped & passing
- with collisions: friction force along $v_\parallel$

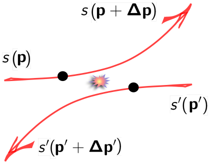
Reduced ion-ion collision operator valid for NC theory

- Particles weakly coupled → only two particle interactions considered

➠ Fokker-Planck collision operator:

$$\boxed{\mathcal{C}_{ss'} = \frac{\partial}{\partial \mathbf{v}}\left(\mathcal{D}(f_{s'})\,\frac{\partial f}{\partial \mathbf{v}} - \mathcal{V}(f_{s'})\,f\right)}$$

Collision of species s with field particles s'


$s\,(\mathbf{p}+\mathbf{\Delta p})$
$s\,(\mathbf{p})$
$s'(\mathbf{p}')$
$s'(\mathbf{p}'+\mathbf{\Delta p}')$

➠ preferred collisional friction –along $v_{\|}$

↪ efficient parallelisation (μ remains motion invariant)

- Reduced Fokker-Planck operator: $\boxed{\mathcal{C}_{ss'} = \partial_{v_{\|}}\left(\mathcal{D}\,\partial_{v_{\|}} f - \mathcal{V} f\right)}$

➠ $\dfrac{\mathcal{V}(r,v)}{\mathcal{D}(r,v)} = \dfrac{v_{\|}}{v_{th}^2} \quad \Rightarrow \quad f \longrightarrow f_{Maxwell}$

- Recovers correct NC regimes ⇒ *analytical proof [Garbet '09]*

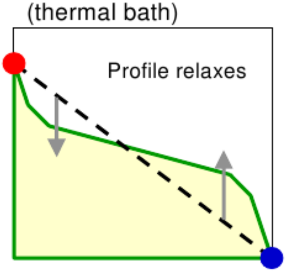
Flux-driven simulations

➠ Open question: What are the mechanisms that govern the averaged pressure gradient in flux-driven simulations.

- Nonlinear fluid simulations have long demonstrated the existence of a complex response *[Carreras '96; Garbet-Waltz '98; Sarazin-Ghendrih '98; Beyer '00;...]* as predicted theorically *[Diamond '95]*
- Such a complex interplay between turbulent transport, profile relaxation and zonal flows ⇒ analysed for the first time in a reduced GK model for trapped ion mode (TIM) *[Darmet '08]*
- Emergency of flux-driven GK codes with improvement of HPC ressources *[Chang '08; Idomura '08; Jolliet '08; Sarazin '09]*

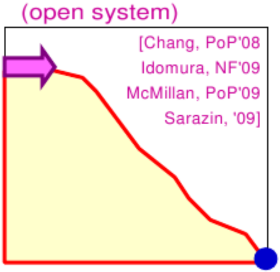
Importance of boundary conditions

Most kinetic simulations rely on transport between 2 thermal baths characterised by prescribed T_i



- Relaxation of T_i profile that tends to remain flat in the core
- ∇T_i being localized at the edge

More realistic: System is driven by some prescribed heat source



☺ Heat source achieve a statistical steady state

- Mean T_i profile does not systematically relax towards stability threshold.

Heat source in GYSELA

- Constant in time and radially localized heat source

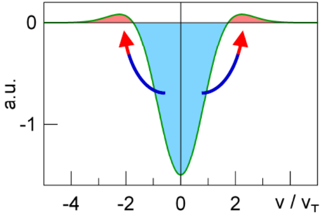
$$\frac{\mathrm{d}f}{\mathrm{d}t} = S_r(r)\left(\frac{\varepsilon}{T} - \frac{3}{2}\right)\exp\left(-\frac{\varepsilon}{T}\right) \quad \text{with} \quad \varepsilon = \frac{1}{2}m_i v_{\parallel}^2 + \mu B$$

$\swarrow$ $\downarrow$ $\searrow$ $\downarrow$

Radial localisation

- No particle source
- Transfers sub-thermal particles to supra-thermal region

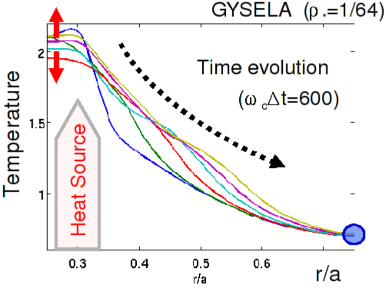
- Isotropic in $v_{\parallel}$ & $v_{\perp}$
- No injected torque



- Source already tested on a simplified model *[Darmet '08]*

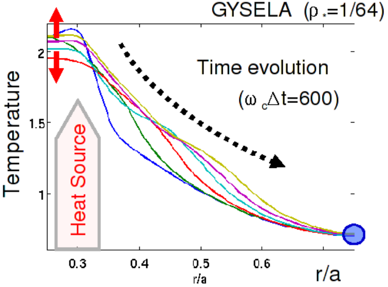
Physics of flux-driven regimes

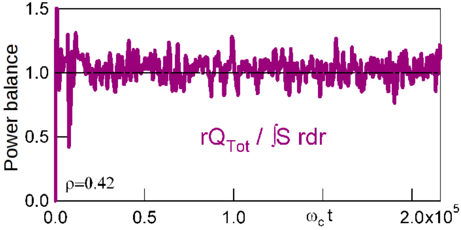
- Requires proper boundary conditions
- Profile relaxation
  - possible front propagation
  - requires global geometry to capture large scale events



Physics of flux-driven regimes

- Requires proper boundary conditions
- Profile relaxation
  - possible front propagation
  - requires global geometry to capture large scale events



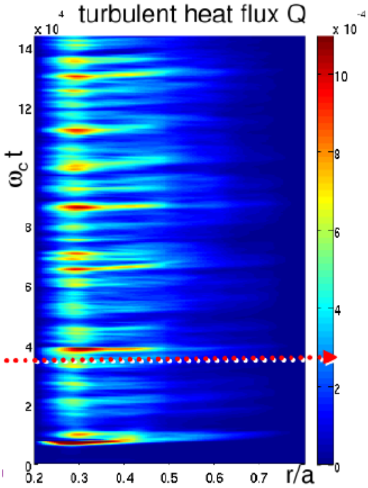


- Statistical steady-state regime when:

$$r\,Q_{\text{Total}}\;/\;\int S\,d^3x = 1$$

Avalanche-like dynamics

- Intermittent dynamics of turbulent heat flux Q
- Long range ballistic transport events at $v_{\text{burst}} \sim \rho^* v_T$



- PDF of flux positively skewed ($S_k > 0.5$)
- 1/f Fourier spectrum far from source

[Bak-Tang-Wiesenfeld '87; Hwa-Kardar '92]

Already reported in fluid simulations
[Carreras '96; Garbet-Waltz '98; Sarazin-Ghendrih '98; Beyer '00;...]

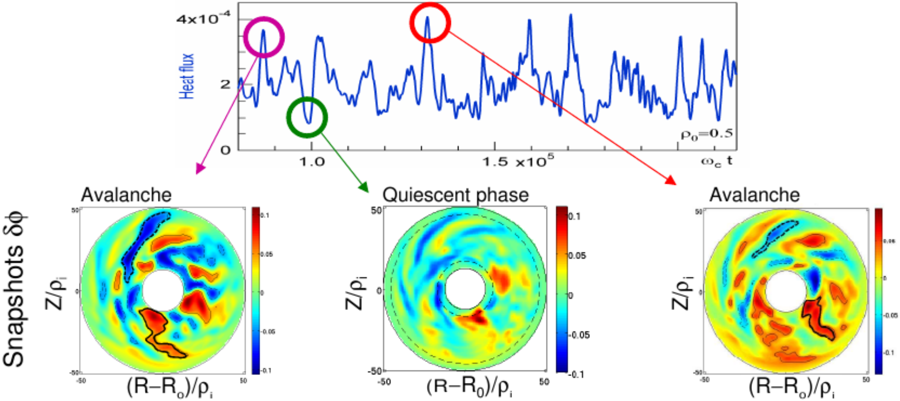
& gyrokinetic simulations
Idomura '08; Jolliet '08

& in experiments
Hidalgo-Sanchez '96; Antar '00; Politzer '00; Boedov-Rudakov '01]

➠ Pending issue: origin of these avalanche-like events ?

Streamers during avalanche events

- Radially elongated vortices (streamers) appear during bursts:
  - Ballooned character
  - Field aligned structures : $qR_0\langle k_{||}\rangle \sim 1$



High Performance Computing

1. Brief idea on the parallelisation
2. An idea of CPU time and memory size consumming

Parallelisation of GYSELA

- $\mu =$ adiabatic invariant

  $\hookrightarrow$ one μ per processor ➠ very performing in $//$

  - Each processor is solving separately the vlasov equation
  - Only communications for $\int d\mu$ (RHS of the Poisson equation)

- (r, θ) cross-section with the most important discretisation

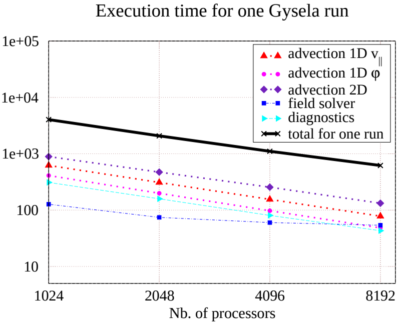
  $\hookrightarrow$ 2D domain decomposition in (r, θ) (2D local splines)

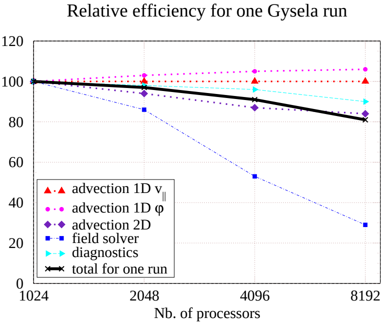
- sequential distribution in φ and $v_{\|}$ directions

➠ $\{$**MPI + OpenMP**$\}$ parallelisation

Results - Strong scaling

$N_r = 512, \quad N_\theta = 512, \quad N_\varphi = 256, \quad N_{v_\parallel} = 47, \quad N_\mu = 32$ (main data = 750 GB)





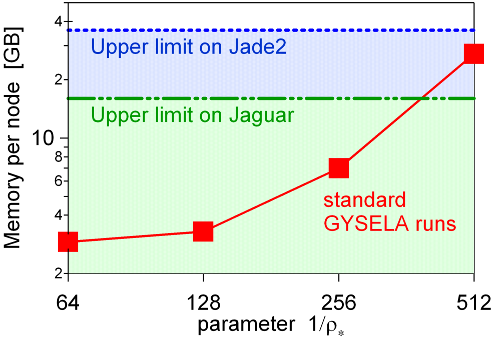
- Very good result: 82% relative efficiency

[Guillaume Latu, Chantal Passeron (2010)]

Parallel constraint: large memory / node

- Strong constraint on memory/node for ITER-like simulations

➠ (1/4 torus) : 272 billion points - 27.2 GBytes per node



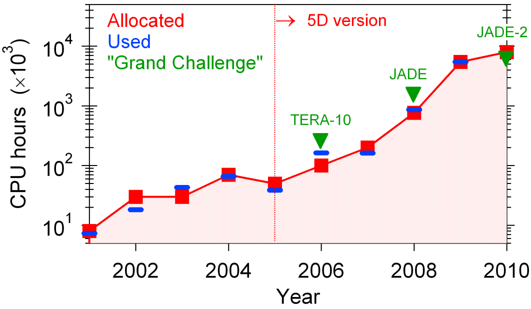
Jade
(CINES - France)
267 Tflop/s, 23 040 cores
36 GBytes per node

Jaguar
(Oak Ridge - US)
2.3 Pflop/s, 224 162 cores
16 GBytes per node

➠ BlueGene architecture not adapted for the code

An exponential increase of CPU consumption

- 2010: 8 million CPU hours allocated on european HPC:
  - 5 million: GENCI national resources (titane-CCRT+jade-CINES)
  - 3 million: HPC-FF (Jülich/Germany) dedicated to Fusion

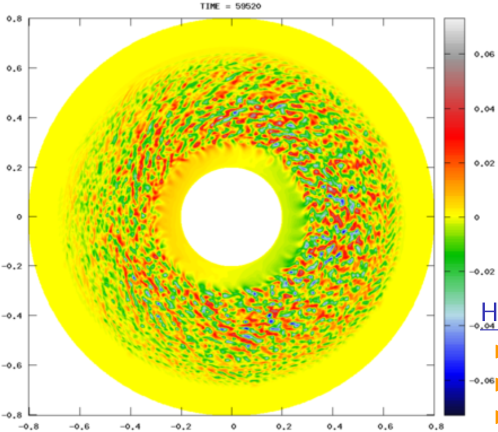


+ ∼ 5 million on Jaguar (Oak-Ridge/US):

→ collaboration G. Dif-Pradalier / P.H. Diamond at UC San Diego

GYSELA Grand Challenge GENCI/CINES

- 8192 processors during 30 days ➠ $>$ 6.7 million hours monoproc.



Mesh ($\rho^* = \rho^*_{\text{ITER}}$):

($1024 \times 1024 \times 128 \times 128 \times 16$)

272 billion points

Memory size:

4.7 TBytes for restart files

1.3 TBytes of 2D and 3D

$>$ 6600 files

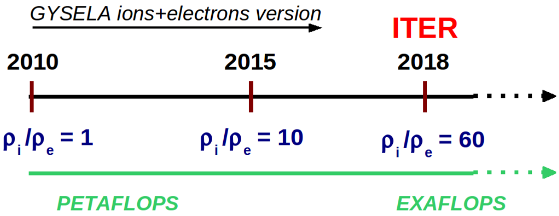
How to treat such quantities of datas ?

- storage ?
- analysis ?
- 3D visualization and more ?

Needs of Petaflop computers and more ...

➠ Next challenge: Global code with kinetic ions + electrons

- Now ITER-like ion simulation: 272 billion points
- With electrons: $\rho_{\text{ions}}/\rho_{\text{electrons}} = 60$
- mesh size $\times 60^3$ and time step/60 !!!



- GYSELA still requires huge efforts of parallelisation to scale to more than 10 000 processors

➠ Key question: How to overcome the constraint of large memory/node ?