Numerical and parallel computing challenges for the global full-f semi-lagrangian code GYSELA

V.Grandgirard, P. Angelino, N. Crouseilles†, G. Dif-Pradalier, X. Garbet, Ph. Ghendrih, M. Haefele†, G. Latu†, Ch. Passeron, Y. Sarazin, E. Sonnendrücker†

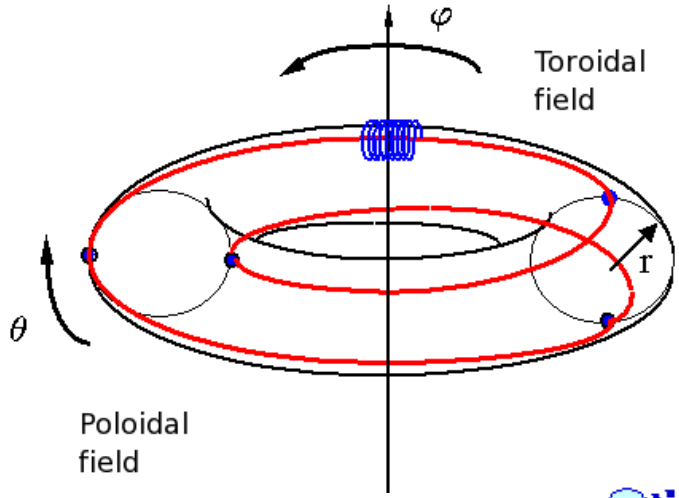
CEA, IRFM , F-13108 Saint-Paul-Lez-Durance.

†(INRIA-Calvi), IRMA-Univ. Louis Pasteur, F-67084 Strasbourg.

Gyrokinetic theory

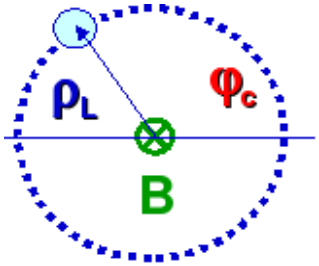
Phase space in 6D

- 3D in space :
  $\hookrightarrow$ toric geometry (r, θ, φ)
- 3D in velocity : $(v_\parallel, v_\perp, \alpha)$



$$\omega_{turb} \approx 10^5 s^{-1} \ll \dot{\varphi_c} \equiv \omega_c \approx 5.10^8 s^{-1}$$

➠ gyroaverage essential



adiabatic invariant $\mu = \frac{mv_\perp^2}{2B} \longrightarrow \mu$ as a parameter

4D+1D gyrokinetic problem $f(r, \theta, \varphi, v_\parallel, \mu)$

$\Downarrow$

Numerical complexity: Conservation properties of the scheme
Memory(5D) + CPU time $\longrightarrow$ massively parallel

Why can't we avoid parallelisation ?

- Simple question for developers but the theoricians have not always an idea of the memory size and the CPU time required

Memory size :

- Example in 4D: "small" mesh $= (128 \times 128 \times 128 \times 128)$
  - $\sim$ 280 millions of mesh points

  $\hookrightarrow$ Already 2 Gbytes just for one distribution function in 4D
- So in 5D: with only 4 points in the 5^{th} direction $\rightarrow$ not possible on actual PC.

CPU time :

- Typical simulations $\rightarrow$ 5 days on 1000 processors
  $\equiv$ 120.000 hours / monoprocessor $\rightarrow$ more than 13 years on 1 processor

Purpose of the talk

An idea of what are the numerical and parallel computing challenges for the development of semi-lagrangian code

- There is a gap between "academic" cases you can run on your laptop in 5 minutes and 5D simulations which require more than 1000 processors during several weeks.
- It requires a strong collaboration between physicists, mathematicians
  - Numerical methods adapted to your physical problem,
- But also a strong collaboration with specialists of parallel computing.
  - Performant algorithms for improving the parallelisation,
  - Parallelisation can constrainst your physical problem

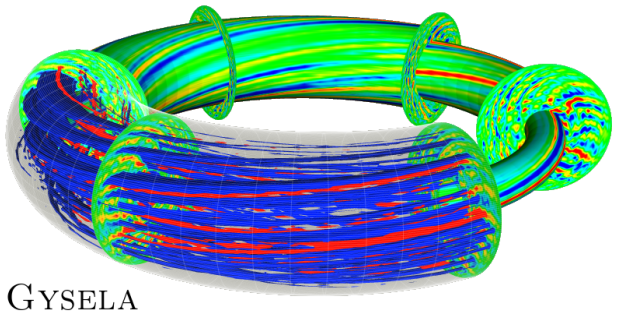
Challenges of physics for GYSELA code

Virginie GRANDGIRARD

Vienna - September 17, 2008

GYrokinetic SEmi-LAgrangian code

- **Full–f code** ($\neq \delta f$):
  - ☺ Equilibrium & fluctuations: no scale separation assumption
  - ☹ Non-Maxwellian background equilibrium
    ↪ *Careful calculation of E_r (canonical initial equilibrium)*
    [Idomura '03, Angelino '06, Dif-Pradalier '07]
  - ☹ Neoclassical theory
    ↪ *The full f must relax → $f_{Maxwell}$*


GYSELA

- **Global code** ($\neq$ flux-tube):
  - ☺ Large scale fast transport events
  - ☹ Profile relaxation ➠ steady state ➠ flux-driven conditions
  - ☹ Numerical challenge
- **Semi-lagrangian scheme** (mixed between PIC and eulerian)

The GYSELA code – hypotheses

i. Electrostatic approximation $\mathbf{E} = -\boldsymbol{\nabla}\phi$;

ii. Axisymmetric ($\partial_\varphi = 0$), concentric and circular nested magnetic flux surfaces ;

iii. Ion distribution function ;

iv. Adiabatic electrons ;

v. Low β approximation $\longrightarrow \frac{\boldsymbol{\nabla} N}{R} \approx \frac{\boldsymbol{\nabla} B}{B}$;

vi. $\mathbf{B}^\star$ at first order in ρ^*;

vii. Bessel function $J_0(k_\perp \rho_i) \approx \frac{1}{1+\frac{k_\perp^2 \rho_i^2}{4}}$ (Padé).

The equations –consistent up to the first order in $\rho_\star$

A. Gyrokinetic equation ($4D + 1D$)

$$\partial_t \bar{f} + \mathbf{v}_E \cdot \boldsymbol{\nabla}_\perp \bar{f} + \mathbf{v}_D \cdot \boldsymbol{\nabla}_\perp \bar{f} + v_\parallel \nabla_\parallel \bar{f} + \dot{v}_\parallel \, \partial_{v_\parallel} \bar{f} = \underbrace{0}_{\text{no collisions}} \quad or \quad \mathcal{C}(f)$$

$$\mathbf{v}_E = \frac{\mathbf{B} \times \boldsymbol{\nabla} \bar{\phi}}{B^2}$$

$$\mathbf{v}_D = \frac{m_i v_\parallel^2 + \mu B}{eB} \frac{\mathbf{B} \times \boldsymbol{\nabla} \mathbf{B}}{B^2}$$

$$\dot{v}_\parallel = \frac{dv_\parallel}{dt} = -\frac{e}{m} \nabla_\parallel \bar{\phi} - \frac{\mu}{m} \nabla_\parallel B + \frac{m v_\parallel}{B} \mathbf{v}_E \cdot \boldsymbol{\nabla} B$$

$$\nabla_\parallel = \frac{1}{R} \left(\partial_\varphi + \frac{1}{q(r)} \right)$$

B. Quasi-neutrality equation ($3D$)

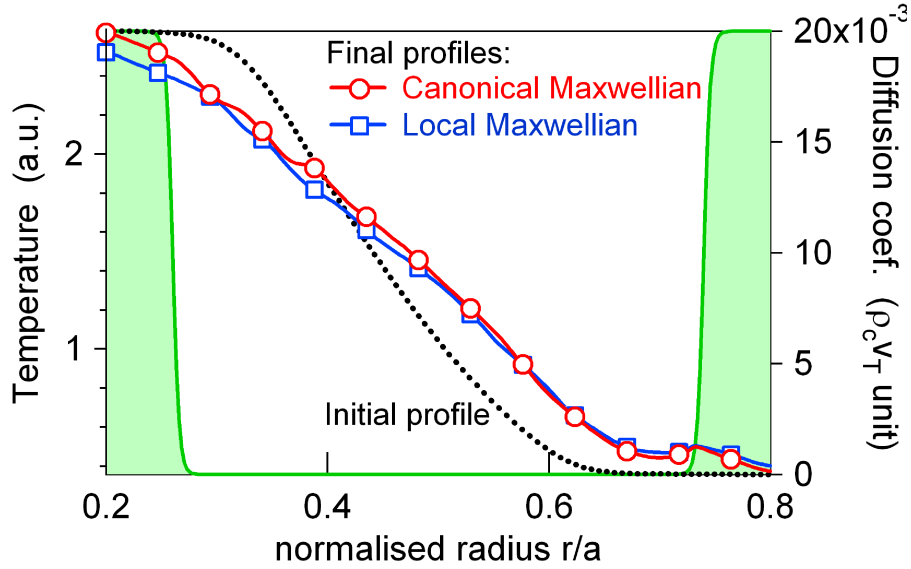
$\delta n_e = \delta n_i \equiv n - n_{init}$

$$-\frac{1}{n_0(r)} \boldsymbol{\nabla}_\perp \cdot \left[\frac{n_0}{B \, \omega_c} \boldsymbol{\nabla}_\perp \phi \right] + \frac{e}{T_e(r)} \left(\phi - \langle \phi \rangle \right) = \frac{2\pi B}{m n_0} \iint d\mu \, dv_\parallel \, J_0 \left(\bar{f} - f_{init} \right)$$

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

The GYSELA code – boundary conditions

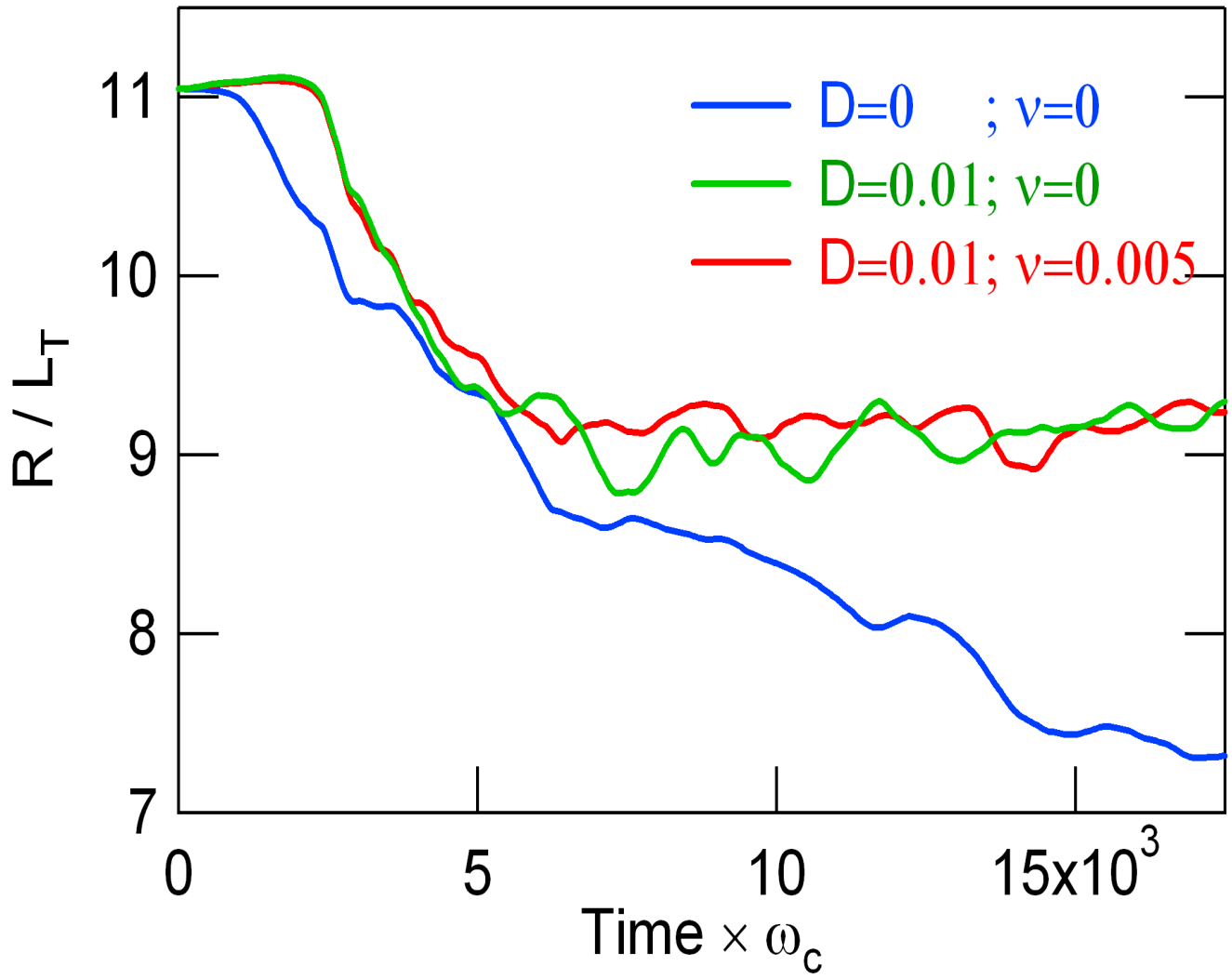
- Periodic in θ and φ
- Vanishing perturbations in non-periodic r and $v_{\|}$
- "Source": thermal baths
  ➠ profile relaxation



Alternatives to sustain the mean gradient:

- adaptative source at each radial position [GYRO ; GTC ; ...]
- diffusive buffer zones [ORB5, GYSELA]
- realistic flux-driven: controled input heat flux [GYSELA, *in progress*]

A thermal bath to sustain the mean gradient

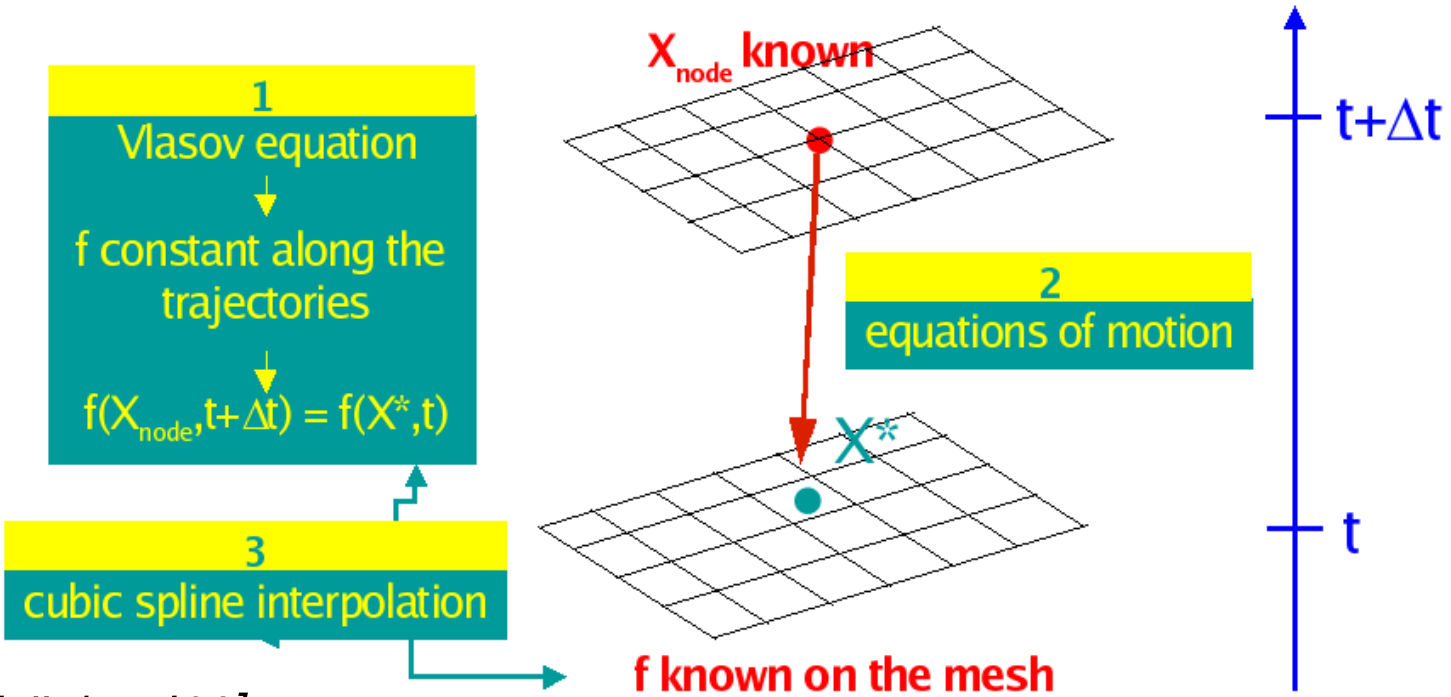


Challenges in numerics and parallel computing

for a Semi-Lagrangian scheme

Virginie GRANDGIRARD

Semi-Lagrangian basic concepts



[Sonnendrücker '98]

☺ Good property of energy conservation

[Grandgirard '05, for 4D-drift kinetic ITG]

☹ A supplementary step : INTERPOLATION

Parallelisation of a Semi-Lagrangian method

Advantage *(due to the eulerian aspect)* :

- fixed grid ➠ perfect load balancing

Drawback *(due to interpolation)* :

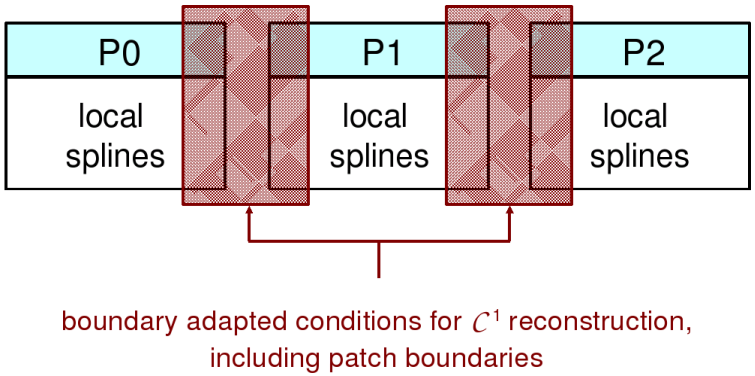
- Several choices for the interpolation
- But we use cubic splines interpolation :
  - ☺ Good compromise between accuracy and simplicity
  - ☹ Loss of locality
    (value of f on one grid point requires f over the whole grid)

➠ **Not possible to use a simple domain decomposition**

New approach : Local cubic splines

A new numerical tool has been developed

➠ Hermite Spline interpolation on patches *[Latu-Crouseilles '07]*

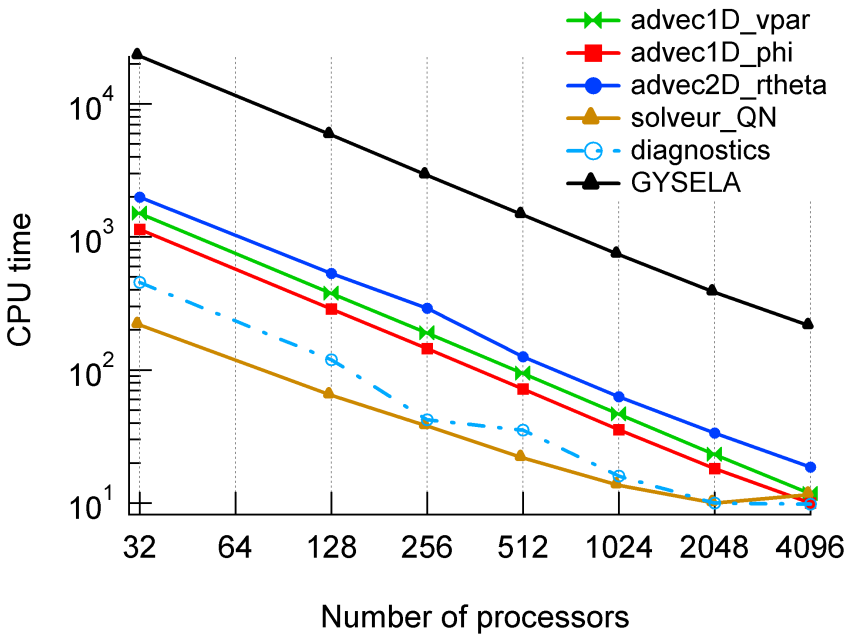


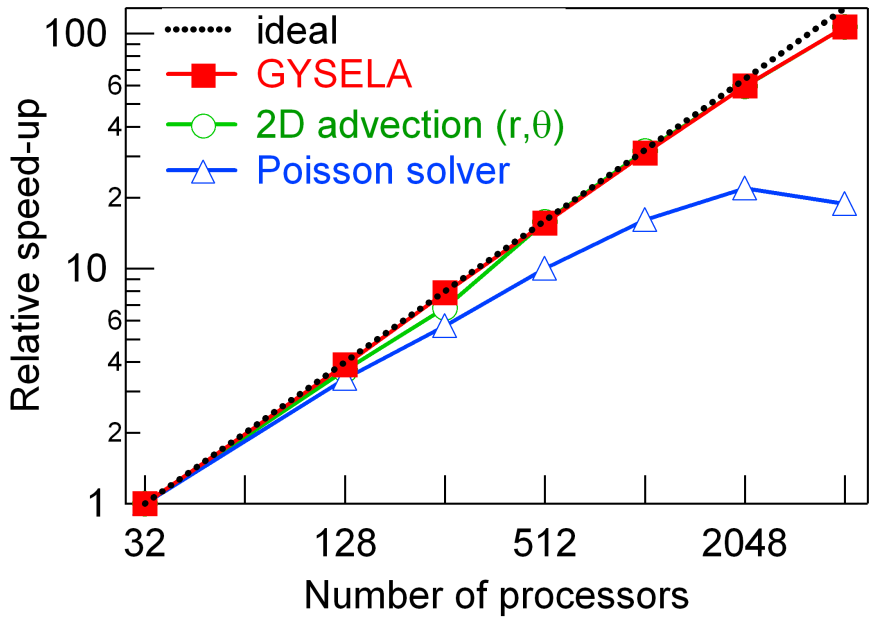
- Computational domain decomposed in subdomains
- Definition of local splines on each subdomains with Hermite boundary conditions
- Derivatives are defined so that they match as closely as possible those of global splines

Parallelisation in GYSELA – good performance

➠ {**MPI + OpenMP**} parallelisation

5D mesh $= (512 \times 128 \times 256 \times 16 \times 32) = 8.6$ billions of points





Efficiency of 93% sur 2048 processor

Efficiency of 83% on 4096 processors

How can the parallelisation constrainsts the physics ?

- Choice of the coordinate system
- Choice of the collision operator

Distribution of the 5D mesh

- μ = 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

Local splines not completely adapted to (r, θ, φ)

- ☺ Efficient parallelisation but only if you minimize the number of points in buffer regions
  $\hookrightarrow$ Minimize the size of the messages to transfer
- ☹ It imposes a CFL condition (displacement of no more than one cell)
  - As $\frac{d\theta}{dt} \gg \frac{dr}{dt} \rightarrow$ need to decrease drastically the time step
  - In practice, local spline just in radial direction $\rightarrow$ strong constrainst for memory size per node

Idea: Change the coordinates system to separate low and fast variables (field aligned coordinates).

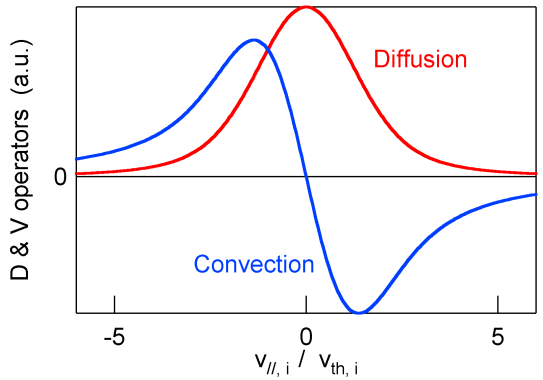
Collision operator on $v_{||}$ only

- If you want to keep an efficient parallelisation in μ
  - ➠ no differential operator in μ
- A strong constrainst in the choice of the collision operator

Try to find the best compromise: **mini. model ➠ max. physics ?**
exact NC transport at low computational cost [Garbet, Varenna '08]

Lorentz operator on $v_{||}$ only:

$$\frac{\mathrm{d}}{\mathrm{d}t} f = \partial_{v_{||}} \left(\mathcal{D}\, \partial_{v_{||}} f - \mathcal{V} f \right)$$



Goal: **interaction gyrokinetic turb. ↔ neoclassical theory**

[Dif-Pradalier, Varenna '08]

To conclude ➠ the road is long...

Example: CPU time and memory size for a $\rho^* = \rho_i/a$ scaling

Numerical problem: $\rho^*/2 \rightarrow$ mesh $\times 2^3 +$ time step $/2$

	ρ* = 1/128 (tore)	ρ* = 1/256 (1/2-tore)	ρ* = 1/512 (1/8-tore)	ρ* = 1/512 (1/4-tore)
Mesh (Nr x Ntheta x Nphi x 32 x 8)	128x256x128	256x512x128	512x1024x64	512x1024x128
Number of mesh points	1.15 billions	4.3 billions	8.6 billions	17.2 billions
Number of processors (8 threads at each time)	256	512	512	1024
Memory required per node	1.94 Go	4.5 Go	9.3 Go	13.5 Go
Total CPU time	14 h	62 h (2.5 days)	255 h (10.5 days)	302 h (12.5 days)
Number of monoprocessor hours	3.600 h	32.000 h	130.500 h	310.000 h
Memory for saving	400 GBytes	3 TBytes	12 TBytes	24.3 Tbytes

☹ Difficult to simulate a complet ITER tore with the actual HPCs.