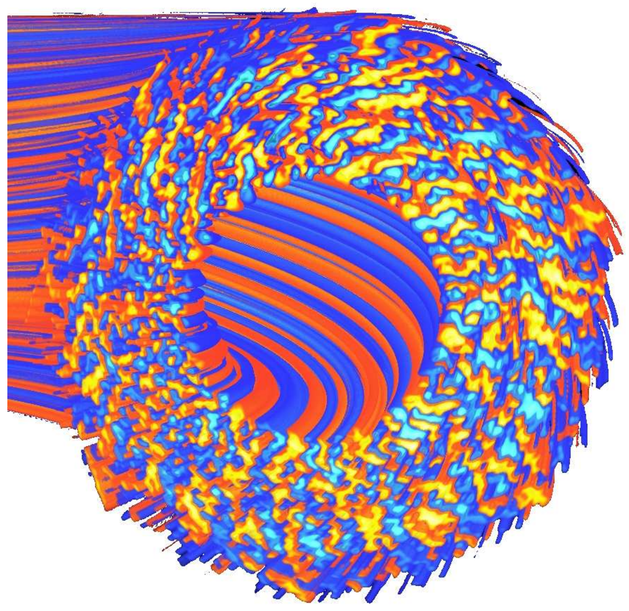
Solving the Vlasov equation using the Semi-Lagrangian method on multiple patches for the GYSELA code

Laura Mendoza*, Éric Sonnendrücker, Virginie Grandgirard (CEA - Cadarache)

Max-Planck-Institut für Plasmaphysik, D-85748 Garching, Germany
(*laura.mendoza@ipp.mpg.de)



Motivation

We are interested in the core of a tokamak, where the magnetic field lines are wrapped around closed surfaces. These surfaces, called magnetic surfaces can be described by a constant section around the torus. Therefore, to have an accurate simulation of the transport process in a tokamak, the description of the geometry of the magnetic surfaces is fundamental. When we work with non linear phenomena such as turbulence, perturbations play an important role. Gyrokinetics codes, that are specialized in these phenomena, need there fore to have a precise geometry's definition.

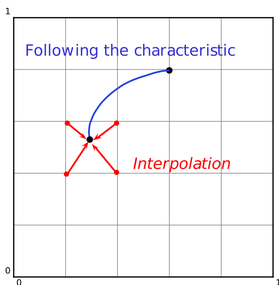
GYSELA

The GYrokinetic SEmi-LAgrangian code has the following properties:

- Used for studying plasma's ion turbulence in tokamaks
- Non linear 5D global gyrokinetic
- Uses circular toroidal geometry (r, θ, φ)
- Uses the Semi-Lagrangian method
- Strang splitting (advections 1D in φ and $v_{||}$ and 2D in (r, θ)

The BSL method

The **B**ackwards **S**emi-**L**agrangian method is now one of the most used numerical schemes to solve the Vlasov equation, in 1D and 2D. It uses a fixed grid in phase-space. The BSL scheme has two major components: it follows the characteristics back in time and it interpolates on the characteristic's foot.



To illustrate the scheme, we consider the constant advection equation:

$$\frac{\partial f}{\partial t} + a \cdot \nabla f = 0 \qquad (1)$$

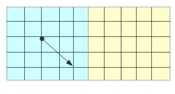
The characteristic curves are the solution of the following ODE:

$$\frac{\mathrm{d}X}{\mathrm{d}t} = a \quad \text{knowing} \quad X(s) = x$$

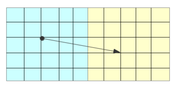
We note X^n the result and we use the property that f is conserved along the characteristics to obtain: $f^{n+1}(x) = f^n(X^n)$. As the computed foot of the characteristic is most certainly not a point on the mesh, we need to use an interpolation method to find $f^n(X^n)$.

Interpolation

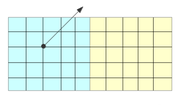
Often and in particular in the Gysela code the BSL method is based on cubic spline interpolation, which consists of an interpolation using piecewise third-degree polynomials. Let us see the three different possible BSL results that will impact on the interpolation:



(a) BSL - classical case



(b) BSL - particle to neighbour patch



(c) BSL - particle out of domain

IGA based Mesh

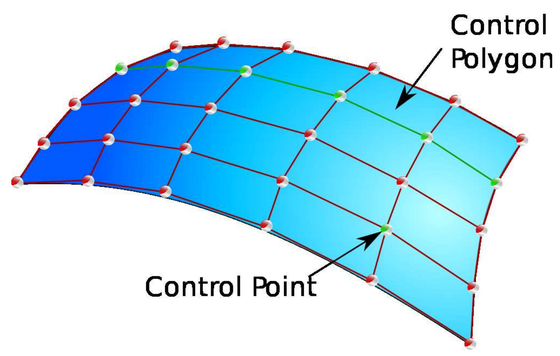
Our aim is to increase the flexibility of the discretization in the GYSELA code. Instead of hard coding one specific curvilinear coordinate system as is done in most gyrokinetic codes, we prefer to rely on an efficient mesh generating tool based on Isogeometric Analysis (IGA) which is based on Non-Uniform Rational B-Splines (NURBS) functions. A NURBS curve's general form is:

$$C(u) = \sum_{i=0}^{n} R_{i,p}(u) P_i \quad \text{with} \quad R_{i,p}(u) = \frac{N_{i,p}(u)\omega_i}{\sum_{j=0}^{n} N_{j,p}(u)\omega_j} \qquad (2)$$

where P is a vector of control points, w a vector of weights and $N_{i,p}(u)$ is the p-th degree B-Spline basis functions on the vector $U = \{0, ..., 0, u_{p+1}, ..., u_{m-p-1}, 1, ..., 1\}$. To go on higher dimension, a NURBS surface is basically a tensor product between two curves.

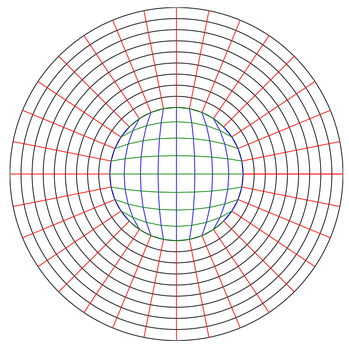
Isogeometry has the following advantages:

- Exact representation of any conic section
- Flexibility to define complex shapes
- Invariant to affine transformations
- Provides the mapping of the domain
- Provides the tools to pass from parametric to physical space



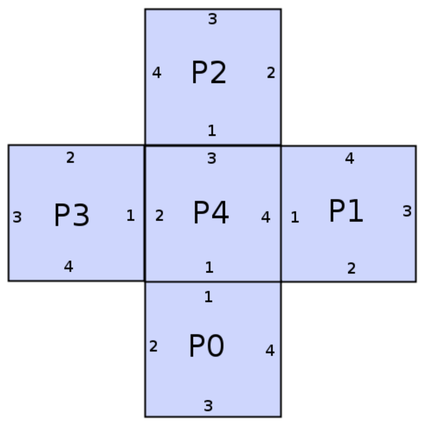
Poloidal section

To avoid the hole presented on the 2D poloidal section we have studied the reattachment of a second mapping.



The whole domain is then represented with NURBS instead of the usual analytical mapping.

Moreover we have taken advantage of the symmetry of the domain to divide it into 4 symmetrically identical patches forming the external crown and an internal patch.



The "connexion" between patches is made through the interpolation. Instead of using periodic or dirichlet boundary conditions, the boundary condition of each patch will depend on the patches at its neighbourhood. To make this process easy numerically we have introduced a notation system for the patches and each of their edges.

Results

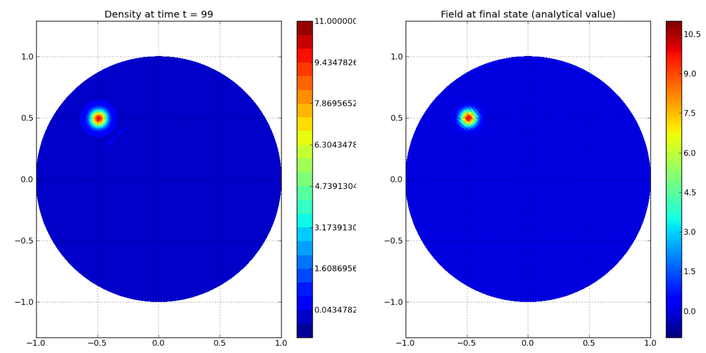
The first simulations were made to solve the constant advection equation (1). For the initial condition we have chosen to use a Gaussian function described as follows:

$$f_0(x, y) = \mathcal{A} \exp\left(\frac{x - x_c}{2\sigma_x} + \frac{y - y_c}{2\sigma_y}\right) \qquad (3)$$

We took an amplitude $\mathcal{A}$ of 10 and a width $\sigma = 0.05$. Our domain is the poloidal section, a disk, centred at the origin and of radius 1. We remind the analytical solution of the scalar advection equation:

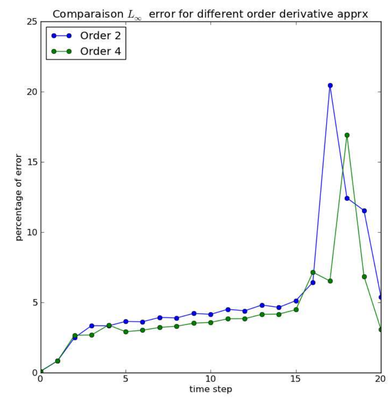
$$f_0(x - a_1 t, \quad y - a_2 t) \qquad (4)$$

where $A = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}$ is the advection coefficient. We ran our first simulation such that the gaussian density went through the whole patch, passing 3 patches total.



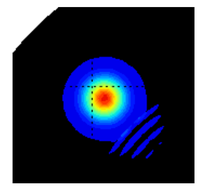
(d) Numeric result at $t = 100$ (e) Analytical result at $t = 100$

There is some numerical perturbation by the fact the strong density gradient goes through one of the (numerically) singular points of the interior mesh. We understand then the importance of using a good interpolation method and having a good approximation of the slopes. We have tried two different order (2 and 4) of finite differences methods for the computation of the derivatives (slopes):



When the boundary condition has some high gradient density, the error pikes and goes up to 10 times higher. Therefore the method used to compute the derivatives needs to be robust in the present of high gradients.

Perspectives



Before adapting our work to higher dimensions we wish to implement a new method for handling the boundary conditions. Our future work will include an LWENO schemes which should be more appropriated.

The next step will be solving the Vlasov-Poisson equation. And finally implementing the method in the GYSELA code.

References

J. Abiteboul, G. Latu, V. Grandgirard, A. Ratnani, E. Sonnendrücker, and A. Strugarek. Solving the vlasov equation in complex geometries. *ESAIM: Proceedings*, 32:103–117, 2011.

E. Sonnendrücker, J. Roche, P. Bertrand, and A. Ghizzo. The semi-lagrangian method for the numerical resolution of the vlasov equation. *JCP*, 149(2):201 – 220, 1999.

International Helmholtz Graduate School for Plasma Physics

IPP