Global nonlinear gyrofluid computation of interchange blobs in tokamak edge plasmas

M. Wiesenberger, A. Kendl

Institute for Ion Physics and Applied Physics, University of Innsbruck, Austria

Introduction

Field aligned filaments, generally referred to as blobs, are believed [1] to be one of the main causes of the large fluctuation levels in the scrape-off layer (SOL) of magnetically confined plasmas. It has been shown [2] that in order to model the behaviour of these plasma blobs finite Larmor radius (FLR) effects are important. However, the polarisation equation was linearized to save computation time although the fluctuation level in the SOL is of order unity. We present simulations of a global, energy conserving gyrofluid model [3] that keeps first order FLR effects and the nonlinear term in the quasineutrality constraint.

We employ discontinuous Galerkin methods [4] to discretise the system in space on a rectangular grid. These schemes have the advantage of being high-order accurate and highly parallelizable, and can be efficiently implemented on GPUs.

2D global gyrofluid equations

We setup our model in a slab geometry in a plane perpendicular to the magnetic field and use the standard Gyro-Bohm scaling to make our equations dimensionless. The fully nonlinear gyrofluid equations for electron particle and ion gyrocenter densities read [3]:

$$\partial_t n_e + \frac{1}{B}\{\phi, n_e\} + n_e \kappa \partial_y \phi - \kappa \partial_y n_e = \nu \Delta n_e \tag{1a}$$

$$\partial_t n_i + \frac{1}{B}\{\psi, n_i\} + n_i \kappa \partial_y \psi + \tau \kappa \partial_y n_i = \nu \Delta n_i \tag{1b}$$

$$\nabla\left(\frac{n_i}{B^2}\nabla_\perp \phi\right) = n_e - \Gamma_i n_i \tag{1c}$$

where $\psi := \Gamma_i \phi - \frac{1}{2}\left(\frac{\nabla\phi}{B}\right)^2$ is the generalized potential and $\frac{1}{B} = 1 + \kappa x$ is the magnetic field amplitude at the outboard midplane for slab geometry. $\Gamma_i := \left(1 - \frac{1}{2}\tau\Delta\right)^{-1}$ is the gyroaveraging operator and $\tau := \frac{T_i}{T_e}$. The viscosity ν is added in an ad-hoc manner.

Note that finite Larmor-radius (FLR) effects appear in the ion $E \times B$ - velocity through the generalized potential and on the right-hand side of the polarisation equation.

We simulate equations (1) in a rectangular box $[0, l_x] \times [0, l_y]$ with initial condition

$$n_e(t=0) = \Gamma_i n_i(t=0) = 1 + A_0 \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{2}$$

σ is the initial blob width and A_0 the initial amplitude. We choose periodic boundary conditions in y-direction and Dirichlet in x:

$$n_e(0,y) = n_e(l_x,y) = n_i(0,y) = n_i(l_x,y) = 1 \text{ and } \phi(0,y) = \phi(l_x,y) = 0 \tag{3}$$

Mass and energy conservation

We define the total blob mass $M_{\text{blob}} := \int d\Omega[n_i - 1]$. Mass conservation then reads

$$\frac{d}{dt} M_{\text{blob}} = \nu \int_{\partial\Omega} \mathbf{dA} \cdot \nabla n_i \tag{4}$$

The global energy theorem of our model system states:

$$\frac{d}{dt} E = \frac{d}{dt} \int_\Omega d\Omega \left[n_e \ln(n_e) + \tau n_i \ln(n_i) + \frac{1}{2} n_i \left(\frac{\nabla\phi}{B} \right)^2 \right] = \nu\Gamma \tag{5}$$

where the first two terms are the thermal energies and the last is the kinetic energy of the $E \times B$ - velocity. Γ represents the losses due to particle diffusion.

Our code numerically conserves both of these quantities.

Discontinuous Galerkin methods

We employ a discontinuous Galerkin [4] method to discretise the system, i.e. we expand each function by Legendre polynomials up to order $P-1$. In 1d this reads:

$$f(x) = \sum_{n=1}^{N} \sum_{k=0}^{P-1} f^{nk} p_{nk}(x) \tag{6}$$

where $p_{nk}(x)$ is the $k-th$ polynomial in cell n. Discretisations of derivatives then result in sparse block matrices. Discretisation errors are $\varepsilon \propto h^P$ in the L^2-norm. We typically use $P = 3$ or $P = 4$ in our simulations. The generalized poisson equation translates into a symmetric matrix equation which can be efficiently solved using a conjugate gradient method.

For time discretisation we use a standard explicit Adams-Bashforth multistep method.

Implementation

Our implementation follows modern C++ design principles (container free numerical algorithms), which enable the separation of numerics and optimisation, through template metaprogramming. At the time of this writing both a single-core CPU and a GPU backend is written using CUDA thrust and the cusp library. The code is in principle extendable to shared and distributed memory systems without having to change the front-end interfaces.

We observe good parallel efficiency of the code due to the high degrees of parllelism in the discontinuous Galerkin and the conjugate gradient methods.

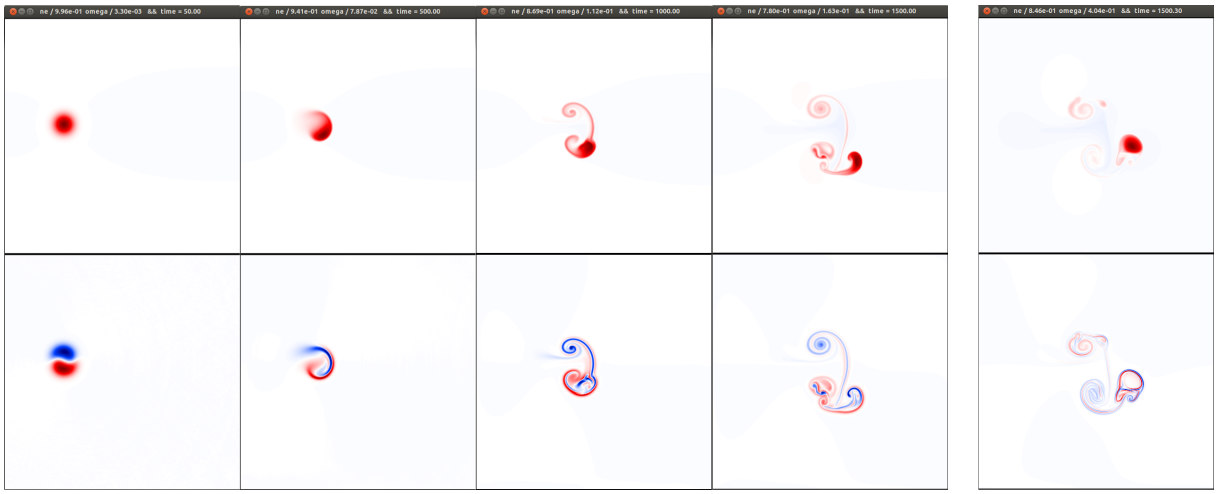

Figure 1: *Electron density* $n_e - 1$ *(top) and vorticity* $\nabla \times \mathbf{u}_E \simeq \frac{\Delta\phi}{B}$ *(bottom) for* $\tau = 1$, $\kappa = 0.0005$, $\nu = 10^{-3}$, *initial blob width* $\sigma = 5$ *and blob amplitude* $A_0 = 1$. *The box size is* $(192\rho_s)^2$. *Discontinuous Galerkin method with* 192^2 *grid cells and* $P = 4$. *The headers show maximum amplitude and time in units of* ω_0^{-1}. *The last picture shows the result of a spectral code applied to the linearized gyrofluid model.*

Simulation results and discussion

We present results of two initial simulations in (Fig. 1). Both runs share the same set of physical parameters and initial conditions. However, the first timeseries on the left-hand side presents results of the nonlinear gyrofluid model Eq. (1), while the second presents the final result of the local (linear) case. In both cases the blobs accelerates poloidally due to the generation of a tilted vorticity dipole that rolls up at later times. By comparing various timesteps we estimate the center of mass trajectory to be qualitativley the same. Differences are clearly seen in the vorticity field. In the local case strong gradients develop at the blob edge indicating a sheared flow around the blob. The nonlinear quasineutrality constraint seems to smoothen these gradients and shows less small scale structures.

3-d local computations of impurity convection

We further apply a 3-d multi-species electromagnetic isothermal gyrofluid flux-tube model in toroidal geometry with edge/SOL boundary conditions on (trace) impurity convection by interchange blobs in the SOL. This extended version of our code uses an Arakawa method for discretization of the Poisson brackets and a local (linear) gyrofluid polarization solver. We use similar parameters like above, here on a $(96\rho_s)^2$ perpendicular grid with resolution 256x256x8. The parallel dynamics is determined by the ratio $(qR/L_\perp) = 4280$, normalized beta $\hat{\beta} = 2$, and collisionality $\hat{C} = 3.5$. The resulting (particle) densities of the main plasma blob and of the (initially radially localized) impurity species are shown in the (Fig. 2) for 4 of the 8 parallel

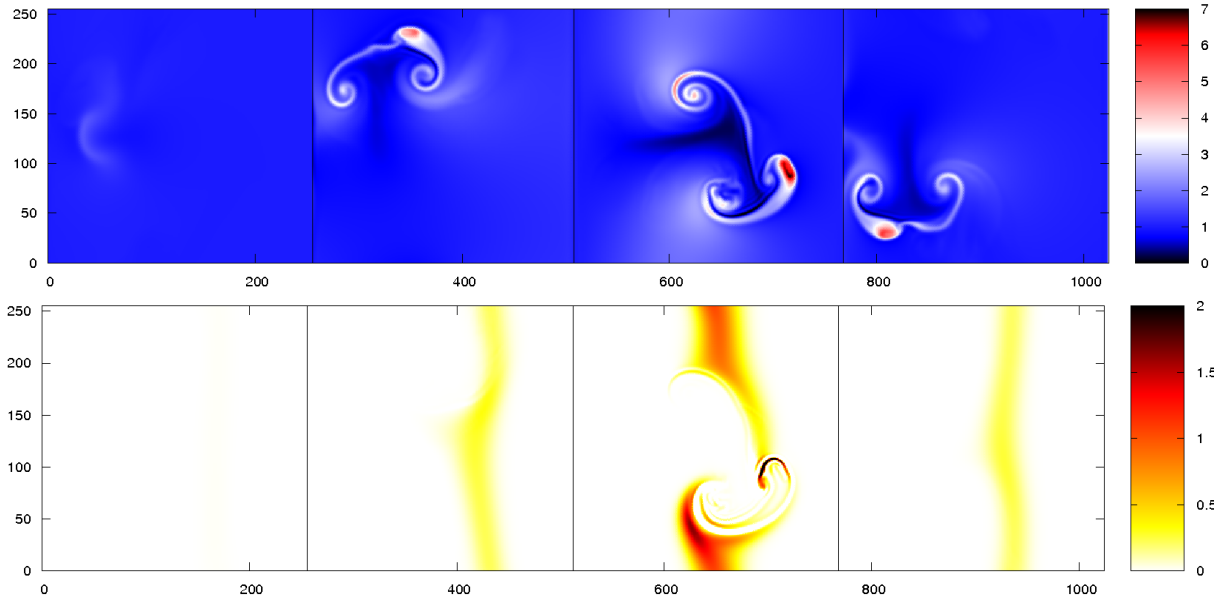

Figure 2: *Electron (top) and impurity (bottom) density for various poloidal positions in the local 3-d SOL blob computation.*

sections, corresponding to different poloidal locations (inside, bottom, outside, top). Note that the x-axis represents a radial coordinate, thus the overall convection is in each section always directed towards the lower field (outboard) side. As an outlook, the nonlinear polarization solver will be combined with the 3-d code version.

Acknowledgements

The authors would like to thank J. Madsen and A. H. Nielsen for helpful discussions and suggestions. This work was supported by the Austrian Science Fund (FWF) W1227-N16, and by the European Commission under the Contract of Association between EURATOM and ÖAW, carried out within the framework of the European Fusion Development Agreement (EFDA). The views and opinions expressed herein do not necessarily reflect those of the European Commission.

References

[1] O. E. Garcia, N. H. Bian and W. Fundamenski, Phys. Plasmas **13**, 082309 (2006)

[2] J. Madsen, O.E. Garcia, et al., Phys. Plasmas **18**, 112504 (2011)

[3] J. Madsen, Phys. Plasmas (2013) *(accepted for publication)*

[4] B. Cockburn and C. W. Shu, Journal of scientific computing, **16**, 173-261, (2001)