The semi-Lagrangian method on curvilinear grids

Adnane Hamiaz, Michel Mehrenberger, Hocine Sellama, Eric Sonnendrücker

The semi-Lagrangian method on curvilinear grids

Adnane Hamiaz, Michel Mehrenberger, Hocine Sellama, Eric Sonnendrücker

October 8, 2015

Abstract

We study the semi-Lagrangian method on curvilinear grids. The classical backward semi-Lagrangian method [1] preserves constant states but is not mass conservative. Natural reconstruction of the field permits nevertheless to have at least first order in time conservation of mass, even if the spatial error is large. Interpolation is performed with classical cubic splines and also cubic Hermite interpolation with arbitrary reconstruction order of the derivatives. High odd order reconstruction of the derivatives is shown to be a good ersatz of cubic splines which do not behave very well as time step tends to zero. A conservative semi-Lagrangian scheme along the lines of [2] is then described; here conservation of mass is automatically satisfied and constant states are shown to be preserved up to first order in time.[1]

1 Introduction

Plasmas, which are a collection of charged particles, can be described quite accurately by kinetic models like the Vlasov-Maxwell equations, or in some circumstances reduced models like the Vlasov-Poisson equation if low frequency phenomena are of interest or the gyrokinetic model when a strong background magnetic field is present. These models nonlinearly couple the Vlasov equation, which is a transport equation in phase space, with the Maxwell equations, which describe the evolution of the self-consistent electromagnetic field generated by the charged particles. The coupling is performed by solving alternatively the Vlasov and the Maxwell equations. In this paper, we are specifically interested by the Vlasov part and will not discuss the coupling, or the field equations. All these Vlasov or Vlasov type equations in standard cartesian coordinates in the d-dimensional phase space $\mathbb{R}^d$ can be written equivalently in the two following abstract forms

$$\partial_t f + \nabla_x \cdot (\mathbf{a} f) = 0, \tag{1.1}$$

[1]This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014-2018 under grant agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

or

$$\partial_t f + \mathbf{a} \cdot \nabla_x f = 0, \tag{1.2}$$

where $\mathbf{a} : \Omega \subset \mathbb{R}^d \to \mathbb{R}$ is a divergence free vector field, which means that it satisfies $\nabla_x \cdot \mathbf{a} = 0$, and the solution $f = f(t, x),\ t \in \mathbb{R}^+,\ x \in \mathbb{R}^d$. Equation (1.1) is called the conservative form and is equivalent, as the vector field $\mathbf{a}$ is divergence free, to (1.2), which is called the advective form.

Because of the high dimension of phase space, up to six for the full physical problem, the Monte Carlo type particle in cell method (PIC) is one of the most used numerical method for its solution, see the book [3] or the review article [4] for more information on this method. Another method, which has been very successful for plasma simulations is the semi-Lagrangian method, which couples particle tracking with a projection on a phase space grid at each time step. The idea in the simple setting of the Vlasov-Poisson equations dates back to Cheng and Knorr [5] and to [1] in the general setting we are interested in here, based on the semi-Lagrangian method, which was already quite successful for climate applications [6]. The big advantage of the semi-Lagrangian method, with respect to other grid based methods is that there is no strong CFL condition on the time step, which for many problems in plasma physics introduces a time step restriction, which is too severe due to the large velocity grid points compared to what is needed for physical accuracy. Since then, many variants of the semi-Lagrangian method have been introduced for plasma physics applications. A convergence proof of this method appears in [7] for linear interpolation and in [8] for high order interpolation. The estimates have been improved recently in [9]. A semi-Lagrangian method on unstructured meshes has been proposed in [10]. These methods, which solve the advective form (1.2) of the Vlasov equation, go in the standard class of semi-Lagrangian methods where characteristics are advected backward in time. We call this classical semi-Lagrangian method BSL for backward semi-Lagrangian method. This is opposed in particular to CSL, conservative semi-Lagrangian methods, which solve the conservative form (1.1) of the Vlasov equation still with backward characteristics as in [11, 12, 13], and FSL, forward semi-Lagrangian methods introduced in [14] for the Vlasov equation, which solves the characteristics forward in time. There have also been works on positivity preserving conservative Discontinuous Galerkin methods [15, 16] and also on semi-Lagrangian methods on adaptive grids based on wavelet interpolation [17, 18], see also [19] for a convergence proof. Note also a hybrid method using only a semi-Lagrangian method in the velocity space [20], and a conservative semi-Lagrangian method based on WENO reconstruction [21, 2]. Now somewhat outdated comparisons of different types of methods for the Vlasov equation can be found in [22, 23].

The present paper is dedicated to a difficulty appearing in applications of the semi-Lagrangian method to the gyrokinetic model used in turbulence simulations of magnetic fusion plasmas [24, 25, 26]. Due to the geometry of magnetic fusion devices and the strong magnetic field which creates a strong anisotropy on the solution and therefore imposes, if a reasonable cost is to be kept, that the mesh be aligned on magnetic flux surfaces, it is standard to use specific curvilin-

ear grids, aligned on magnetic field lines or flux surfaces, for these simulations. It is well known that such grids introduce additional numerical difficulties, in particular the free stream preservation issue for conservative methods [27, 28]. The free stream preservation enforces that constant states are exactly preserved by the numerical method. This is enforced trivially by the classical BSL method independently of the accuracy of the advection as constants are preserved by interpolation. For this reason it appears that they are more robust in curvilinear coordinates. On the other hand, a first attempt of using a split conservative semi-Lagrangian method has been reported in [29], pinpointing some specific difficulties.

An important property of the advection field for the method to be stable in curvilinear coordinates is that it is divergence free at the discrete level in some sense. This is most easily realised, when using a potential formulation of the advection field. This is also most easily understood in two-dimensions, but can be generalised to arbitrary dimensions as shown in [27]. So in order to make our point it is enough to study the problem in a two-dimensional space. We suppose that we can write the advection field $\mathbf{a} = (\partial_{x_2}\Phi, -\partial_{x_1}\Phi)^\top = \nabla^\perp\Phi$, where $\top$ stands for the transposition, and $\nabla^\perp = (\partial_{x_2}, -(\partial_{x_1}))^\top$. In applications $\mathbf{a}$ may depend on t and $f(t,\cdot)$, but we will omit this point for the moment.

The classical BSL methods automatically preserve constant states, but are not conservative. On the other hand the conservative CSL method is conservative, but does not preserve constant states. We will show in this paper that the BSL method can be made conservative up to at least first order in time and on the other some CSL methods can be made to preserve constant states up to at least first order in time. Moreover different interpolating methods will be compared for BSL to show its robustness. Some numerical investigations of the splitting issue raised in [30] will also be performed.

The outline of the paper is the following: First we are going to express the two abstract forms of the Vlasov equations in curvilinear coordinates. In the second part we recall the principles of the classical and conservative semi-Lagrangian methods. Then we are going to propose different versions of the BSL scheme and a new CSL scheme and prove their approximate conservation properties. Finally detailed numerical investigations will be performed to assess the strengths and weaknesses of the different schemes.

2 Obtention of the curvilinear equations

We consider here the curvilinear framework, that is we have a transformation (the mapping) $\mathscr{F} : \widehat{\Omega} \to \Omega; \boldsymbol{\eta} \mapsto \mathbf{x}(\boldsymbol{\eta})$ from a logical domain $\widehat{\Omega}$ to the physical domain $\Omega \subset \mathbb{R}^2$. Polar coordinates are a simple example of such a transformation, but more general, numerically defined transformations, are needed to align the grid with the magnetic flux surfaces in actual tokamak simulations. We want to define the equation for $\tilde{f}(\boldsymbol{\eta}) = f(\mathbf{x}(\boldsymbol{\eta}))$. In the sequel we will omit the *tilde* on f and other quantities, as on each identity it should be clear form the context if we deal with functions of $\mathbf{x}$ (that is, from the physical domain)

or functions of $\boldsymbol{\eta}$ (from the logical domain). We define the jacobian matrix $[J]$ and the jacobian J by

$$[J] = \begin{pmatrix} \partial_{\eta^1} x^1 & \partial_{\eta^2} x^1 \\ \partial_{\eta^1} x^2 & \partial_{\eta^2} x^2 \end{pmatrix}, \; J = \det([J]).$$

Using the chain rule we can relate the differential operators with respect to $\mathbf{x}$ to the differential operators with respect to $\boldsymbol{\eta}$. Denoting by $[J]^{-\top} = ([J]^{-1})^{\top}$, we get

$$\nabla_x f = [J]^{-\top} \nabla_\eta f, \qquad \nabla_x \cdot (\mathbf{a} f) = \frac{1}{J} \nabla_\eta \cdot ([J]^{-1} \mathbf{a} J f), \tag{2.1}$$

so that the advective form reads

$$\partial_t f + ([J]^{-1}\mathbf{a}) \cdot \nabla_\eta f = 0,$$

and the conservative form becomes

$$\partial_t (Jf) + \nabla_\eta \cdot ([J]^{-1} \mathbf{a}\, Jf) = 0$$

and the divergence free condition is

$$\nabla_\eta \cdot ([J]^{-1} \mathbf{a}\, J) = 0.$$

Again from the chain rule, we find that

$$\nabla_\eta^\perp \Phi = J[J]^{-1} \nabla_x^\perp \Phi.$$

So that if $\mathbf{a} = \nabla_x^\perp \Phi$, we get $J[J]^{-1}\mathbf{a} = \nabla_\eta^\perp \Phi$. The advective form then rewrites

$$\partial_t f + \frac{1}{J} \nabla_\eta^\perp \Phi \cdot \nabla_\eta f = 0, \tag{2.2}$$

or expanding the coordinates

$$\partial_t f + \frac{1}{J} \frac{\partial \Phi}{\partial \eta^2} \frac{\partial f}{\partial \eta^1} - \frac{1}{J} \frac{\partial \Phi}{\partial \eta^1} \frac{\partial f}{\partial \eta^2} = 0,$$

and the conservative form rewrites

$$\partial_t (Jf) + \nabla_\eta \cdot ((\nabla_\eta^\perp \Phi)\, f) = 0, \tag{2.3}$$

which becomes, expanding the coordinates

$$\partial_t (Jf) + \frac{\partial}{\partial \eta^1} (\frac{1}{J} \frac{\partial \Phi}{\partial \eta^2} Jf) - \frac{\partial}{\partial \eta^2} (\frac{1}{J} \frac{\partial \Phi}{\partial \eta^1} Jf) = 0,$$

and the divergence free condition is automatically satisfied, as

$$\frac{1}{J} \nabla_\eta \cdot ([J]^{-1} \mathbf{a}\, J) = \frac{1}{J} \nabla_\eta \cdot \nabla_\eta^\perp \Phi = \frac{1}{J} \frac{\partial}{\partial \eta^1} (\frac{\partial \Phi}{\partial \eta^2}) - \frac{1}{J} \frac{\partial}{\partial \eta^2} (\frac{\partial \Phi}{\partial \eta^1}) = 0.$$

3 The characteristics

The semi-Lagrangian method is based on conservation properties along the characteristics. Let us recall our Vlasov type equations in curvilinear coordinates that we are interested in, denoting by $\mathbf{b} = \nabla_\eta^\perp \phi$ the divergence free advection field and J the jacobian of our curvilinear grid. The advective form (2.2) then writes

$$\partial_t f + \frac{1}{J}\mathbf{b} \cdot \nabla_\eta f = 0, \tag{3.1}$$

and introducing $\bar{f} = Jf$ the conserved variable, the conservative form (2.3) becomes

$$\partial_t \bar{f} + \nabla_\eta \cdot (\frac{1}{J}\mathbf{b}\, \bar{f}) = 0. \tag{3.2}$$

To both of these equations we associate the same characteristics, which are the integral curves of the differential equation

$$\frac{\mathrm{d}\mathbf{H}}{\mathrm{d}t} = \frac{1}{J}\mathbf{b}, \qquad H(s) = \boldsymbol{\eta}. \tag{3.3}$$

We shall denote classically by $\mathbf{H}(t; \boldsymbol{\eta}, s)$, the solution at time t of the characteristic curve taking the value $\boldsymbol{\eta}$ at time s.

The solutions f of the advective form (3.1) and $\bar{f} = fJ$ of the conservative form (3.2) obey the following conservation properties along the characteristics:

Lemma 3.1 *For given smooth vector field* $\mathbf{b}$ *and function* J*, the solution* f *of* (3.1) *is conserved along the characteristics. More precisely*

$$\frac{\mathrm{d}}{\mathrm{d}t} f(t, \mathbf{H}(t; \boldsymbol{\eta}, s)) = 0. \tag{3.4}$$

And the solution $\bar{f} = fJ$ *of* (3.2) *satisfies*

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[\bar{f}(t, \mathbf{H}(t; \boldsymbol{\eta}, s)) \det(\nabla_\eta \mathbf{H}(t; \boldsymbol{\eta}, s))\right] = 0. \tag{3.5}$$

where $\nabla_\eta \mathbf{H}(t; \boldsymbol{\eta}, s)$ *denotes the Jacobian matrix of the transformation* $\boldsymbol{\eta} \mapsto \mathbf{H}(t; \boldsymbol{\eta}, s)$.

Proof. Using the chain rule we have

$$\frac{\mathrm{d}}{\mathrm{d}t} f(t, \mathbf{H}(t; \boldsymbol{\eta}, s)) = (\partial_t f + \frac{\mathrm{d}H}{\mathrm{d}t} \cdot \nabla_\eta f)(t, \mathbf{H}(t; \boldsymbol{\eta}, s)) = 0,$$

using the definition of the characteristics and the fact that f is a solution of (3.1).
For the second conservation property, let us first observe that

$$\frac{\mathrm{d}}{\mathrm{d}t} \det(\nabla_\eta \mathbf{H}(t; \boldsymbol{\eta}, s)) = (\nabla_\eta \cdot \frac{\mathbf{b}}{J})(t, \mathbf{H}(t; \boldsymbol{\eta}, s)) \det(\nabla_\eta \mathbf{H}(t; \boldsymbol{\eta}, s)).$$

This follows from the definition of the characteristics and the fact that the determinant is an n-linear alternating form. Then, a direct computation in (3.5) yields

$$\begin{aligned}
&\left(\frac{\partial \bar{f}}{\partial t}(t,\mathbf{H}(t;\boldsymbol{\eta},s)) + \frac{\mathrm{d}\mathbf{H}}{\mathrm{d}t}(t;\boldsymbol{\eta},s)\cdot\nabla_\eta \bar{f}(t,\mathbf{H}(t;\boldsymbol{\eta},s))\right)\det(\nabla_\eta \mathbf{H}(t;\boldsymbol{\eta},s)) \\
&\qquad + \bar{f}(t,\mathbf{H}(t;\boldsymbol{\eta},s))\frac{\mathrm{d}}{\mathrm{d}t}\det(\nabla_\eta \mathbf{H}(t;\boldsymbol{\eta},s)) = \\
&\left(\frac{\partial \bar{f}}{\partial t}(t,\mathbf{H}(t;\boldsymbol{\eta},s)) + \frac{\mathbf{b}(t,\mathbf{H}(t;\boldsymbol{\eta},s))}{J(t,\mathbf{H}(t;\boldsymbol{\eta},s))}\cdot\nabla_\eta \bar{f}(t,\mathbf{H}(t;\boldsymbol{\eta},s))\right)\det(\nabla_\eta \mathbf{H}(t;\boldsymbol{\eta},s)) \\
&\qquad + \bar{f}(t,\mathbf{H}(t;\boldsymbol{\eta},s))(\nabla_\eta\cdot\frac{\mathbf{b}}{J})(t,\mathbf{H}(t;\boldsymbol{\eta},s))\det(\nabla_\eta \mathbf{H}(t;\boldsymbol{\eta},s)) \\
&= \left(\frac{\partial \bar{f}}{\partial t}(t,\mathbf{H}(t;\boldsymbol{\eta},s)) + \left[\nabla_\eta\cdot(\frac{\mathbf{b}}{J}\bar{f})\right](t,\mathbf{H}(t;\boldsymbol{\eta},s))\right)\det(\nabla_\eta \mathbf{H}(t;\boldsymbol{\eta},s)) = 0,
\end{aligned}$$

as $\bar{f}$ is a solution of the conservation law at any point. ■

4 The Backward Semi-Lagrangian scheme (BSL2D)

The backward semi-Lagrangian scheme computes the time evolution of our equation in advective form (3.1) using the characteristics (3.3).

4.1 Formulation of the scheme

We consider a uniform cartesian grid of the logical space $(\eta_1,\eta_2)\in[0,1]^2$ defined by $N_1,\ N_2\in\mathbb{N}^*$ and $\eta_i^1=(i-1)\Delta\eta_1,\ \eta_i^2=(i-1)\Delta\eta_2,\ i\in\mathbb{R}$, using a *Fortran* like indexing, with $\Delta\eta_k=1/N_k,\ k=1,2$. We also define $t_n=n\Delta t,\ n\in\mathbb{R}$, with $\Delta t\in\mathbb{R}^{+*}$.

The unknowns at time t_n are

$$f_{i,j}^n\simeq f(t_n,\eta_i^1,\eta_j^2),\ i=1,\dots,N_1+1,\ j=1,\dots,N_2+1,\ n\in\mathbb{N}.$$

The classical backward semi-Lagrangian scheme uses the conservation property (3.4) at each grid point to update the solution from one time step to the next. This can be expressed as

$$f_{i,j}^{n+1}=f(t_n,\boldsymbol{\eta}_{i,j}^*,t_{n+1})),$$

where $\boldsymbol{\eta}_{i,j}=(\eta_i^1,\eta_j^2)$ is a grid point and $\boldsymbol{\eta}_{i,j}^*=\mathbf{H}(t_n;\boldsymbol{\eta}_{i,j},t_{n+1})$ the origin of the characteristic ending at that grid point. In order to get a numerical scheme from this equality, we need two steps:

1. Compute $\mathbf{H}(t_n;\boldsymbol{\eta}_{i,j},t_{n+1}))$ by numerically solving the characteristics equations (3.3) backward in time on one time step. If the advection field $\mathbf{b}/J$ is analytically known, any standard ODE solver can be used. When the

problem is non linear and the advection field depends on f, a simple solution is to use the backward Euler method. Note however that iterations are needed as the advection field is not known at time t_{n+1}. Two are generally sufficient. The scheme reads

$$\boldsymbol{\eta}^*_{i,j} = \boldsymbol{\eta}_{i,j} - \Delta t(\mathbf{b}/J)(t_{n+1}, \boldsymbol{\eta}_{i,j}), \qquad i = 1, \ldots, N_1 + 1,\ j = 1, \ldots, N_2 + 1.$$

2. Interpolate f^n the solution at t_n at the origin of the characteristics. From grid values $f_h = (f_{i,j})_{i=1,\ldots,N_1+1,\ j=1,\ldots,N_2+1}$, we need to define a reconstruction

$$\Pi[f_h]\ : \mathbb{R}^2 \to \mathbb{R}.$$

This reconstruction (the *interpolator*) should satisfy

$$\Pi[f_h](\eta^1_i, \eta^2_j) = f_{i,j},\ i = 1, \ldots, N_1 + 1,\ j = 1, \ldots, N_2 + 1.$$

The algorithm then becomes:

1. Initialisation:

$$f^0_{i,j} = f(t_0, \eta^1_i, \eta^2_j),\ i = 1, \ldots, N_1 + 1,\ j = 1, \ldots, N_2 + 1,$$

2. From t_n to t_{n+1}. Predict advection field $\mathbf{b}$ at time t_{n+1} for non linear problems and loop until convergence

   (a) Compute origin of characteristics

$$\boldsymbol{\eta}^*_{i,j} = \boldsymbol{\eta}_{i,j} - \Delta t(\mathbf{b}/J)(t_{n+1}, \boldsymbol{\eta}_{i,j}), \qquad i = 1, \ldots, N_1 + 1,\ j = 1, \ldots, N_2 + 1.$$

   (b) Interpolate

$$f^{n+1}_{i,j} = \Pi[f^n_h]\left(\eta^1_i - \Delta t a^1_{i,j}, \eta^2_j - \Delta t a^2_{i,j}\right), \qquad i = 1, \ldots, N_1 + 1,\ j = 1, \ldots, N_2 + 1$$

   with $f^n_h = (f^n_{i,j})_{i=1,\ldots,N_1+1,\ j=1,\ldots,N_2+1}$.

4.2 Preservation of constant states

As soon as the reconstruction Π reproduces the constants, we have preservation of constant states, which can be written in this form, as the operator is linear:

$$\left(f^n_{i,j} = 1,\ i = 1, \ldots, N_1 + 1,\ j = 1, \ldots, N_2 + 1\right) \Rightarrow \left(f^{n+1}_{i,j} = 1,\ i = 1, \ldots, N_1 + 1,\ j = 1, \ldots, N_2 + 1\right).$$

4.3 First order conservation of mass in BSL2D

A priori, conservation of mass is not satisfied for the BSL2D scheme. Using the jacobian at the grid points $J_{i,j} = J(\eta_i^1, \eta_j^2)$, this would read

$$\sum_{i,j} J_{i,j} f_{i,j}^{n+1} = \sum_{i,j} J_{i,j} f_{i,j}^{n}.$$

However, assuming our reconstruction is smooth enough, we can write a Taylor expansion of first order in Δt, from our scheme:

$$\begin{aligned} f_{i,j}^{n+1} = \Pi[f_h^n]\left(\eta_i^1 - \Delta t b_{i,j}^1/J_{i,j}, \eta_j^2 - \Delta t b_{i,j}^2/J_{i,j}\right) = \Pi[f_h^n](\eta_i^1, \eta_j^2) \\ - \frac{\Delta t}{J_{i,j}}\left(b_{i,j}^1 \frac{\partial \Pi[f_h]}{\partial \eta^1}(\eta_i^1, \eta_j^2) + b_{i,j}^2 \frac{\partial \Pi[f_h]}{\partial \eta^2}(\eta_i^1, \eta_j^2)\right) + O(\Delta t^2). \end{aligned}$$

So that

$$\begin{aligned} \sum_{i,j} J_{i,j} f_{i,j}^{n+1} = \sum_{i,j} J_{i,j} f_{i,j}^{n} - \Delta t \left(b_{i,j}^1 \frac{\partial \Pi[f_h]}{\partial \eta^1}(\eta_i^1, \eta_j^2) + b_{i,j}^2 \frac{\partial \Pi[f_h]}{\partial \eta^2}(\eta_i^1, \eta_j^2)\right) \\ + O(\Delta t^2). \end{aligned}$$

So a first order condition is the following

$$A = \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} b_{i,j}^1 \frac{\partial \Pi[f_h]}{\partial \eta^1}(\eta_i^1, \eta_j^2) + b_{i,j}^2 \frac{\partial \Pi[f_h]}{\partial \eta^2}(\eta_i^1, \eta_j^2) = 0,$$

which should be valid for all periodic sequences f_h. Now, considering that

$$b_{i,j}^1 = \frac{\partial \Phi_h}{\partial \eta^2}(\eta_i^1, \eta_j^2),\ b_{i,j}^2 = -\frac{\partial \Phi_h}{\partial \eta^1}(\eta_i^1, \eta_j^2),$$

with a reconstruction $\Phi_h \simeq \Phi$, the first order condition rewrites

$$A = \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} \left(\frac{\partial \Phi_h}{\partial \eta^2}\frac{\partial \Pi[f_h]}{\partial \eta^1} - \frac{\partial \Phi_h}{\partial \eta^1}\frac{\partial \Pi[f_h]}{\partial \eta^2}\right)(\eta_i^1, \eta_j^2) = 0. \tag{4.1}$$

Using a reconstruction of the derivatives of the form

$$\frac{\partial F}{\partial \eta^1}(\eta_i^1, \eta_j^2) = \sum_{\ell=r_1}^{s_1} \alpha_\ell^1 F(\eta_{i+\ell}^1, \eta_j^2),\ \frac{\partial F}{\partial \eta^2}(\eta_i^1, \eta_j^2) = \sum_{\ell=r_2}^{s_2} \alpha_\ell^2 F(\eta_i^1, \eta_{j+\ell}^2), \tag{4.2}$$

for both $F = \Pi[f_h]$ and $F = \Phi_h$ leads to

$$\begin{aligned} A &= \sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\sum_{k=r_1}^{s_1}\sum_{\ell=r_2}^{s_2} \alpha_k^1\alpha_\ell^2(\Pi[f_h](\eta_{i+k}^1,\eta_j^2)\Phi_h(\eta_i^1,\eta_{j+\ell}^2) \\ &\qquad\qquad -\Phi_h(\eta_{i+k}^1,\eta_j^2)\Pi[f_h](\eta_i^1,\eta_{j+\ell}^2)) \\ &= \sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\Pi[f_h](\eta_i^1,\eta_j^2)\sum_{k=r_1}^{s_1}\sum_{\ell=r_2}^{s_2}\alpha_k^1\alpha_\ell^2(\Phi_h(\eta_{i-k}^1,\eta_{j+\ell}^2)-\Phi_h(\eta_{i+k}^1,\eta_{j-\ell}^2)) \\ &= \sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\Pi[f_h](\eta_i^1,\eta_j^2)\sum_{k}\sum_{\ell}(\alpha_{i-k}^1\alpha_{\ell-j}^2-\alpha_{k-i}^1\alpha_{j-\ell}^2)\Phi_h(\eta_k^1,\eta_\ell^2), \end{aligned}$$

prolongating $\alpha_i^1 = 0,\ i \notin \{r_1,\dots,s_1\}$ and $\alpha_j^2 = 0,\ j \notin \{r_2,\dots,s_2\}$. The condition $A = 0$ is thus true as soon as $\alpha_{-i}^k = -\alpha_i^k,\ i \in \mathbb{Z},\ k = 1,2$, which is a reasonable assumption for the derivative formula (4.2).
We emphasize that such condition holds when the reconstructions $\Pi[f_h]$ and Φ_h use the *same* formula for the derivative.
We also remark that the condition (4.1) is the same for cartesian and curvilinear case, as the jacobian is simplified and does no more appear.
For splines, the derivative is generally not expressed like in (4.2): derivatives are obtained, by inverting a linear system. However, in the periodic setting, the matrix for computing the derivatives is circulant and its inverse also, which leads to a formula of type (4.2).

4.4 Examples of reconstructions: the $1D$ advective case

In this section, we review some choices for Π_h. We focus here on the $1D$ interpolation in the advective form, as the $2D$ case can be obtained through tensor product.

Spline interpolation: SPL(d) (see e.g. [1] (for $d = 3$), [8], [31] (for $d = 3$ and non-uniform grid))
We can consider here a non uniform mesh. Let $\eta_{\min} < \eta_{\max} \in \mathbb{R}$. We consider a discretization

$$\eta_1 = \eta_{\min} \le \eta_2 \le \cdots \le \eta_{N+1} = \eta_{\max}$$

of the interval $[\eta_{\min}, \eta_{\max}]$. Let $d \in \mathbb{N}^*$, an odd number. We define a knot sequence $\tau_i,\ i = 1,\dots,N+2d+1$. The knot sequence satisfies

$$\tau_{j+d} = \eta_j,\ j = 1,\dots,N+1.$$

In the case of open boundary conditions, we take

$$\begin{aligned} &\tau_j = \tau_{d+1},\ j = 1,\dots,d, \\ &\tau_{N+j} = \tau_{N+d+1},\ j = d+2,\dots,2d+1, \end{aligned}$$

and for periodic boundary conditions, we take

$$\tau_j = \tau_{j+N} - L,\ j = 1, \ldots, d,$$
$$\tau_{N+j} = \tau_j + L,\ j = d+2, \ldots, 2d+1,$$

where $L = \eta_{\max} - \eta_{\min}$. We can consider other possibilities for the $2d$ knots $\tau_1, \ldots, \tau_d$ and $\tau_{N+d+2}, \ldots, \tau_{N+2d+1}$; we consider that

$$\tau_1 \leq \cdots \leq \tau_d \leq \tau_{d+1},\ \tau_{N+d+1} \leq \tau_{N+d+2} \leq \cdots \leq \tau_{N+2d+1}.$$

We then can define, for $j = 1, \ldots, N+d$, the B-spline B_j^{d+1}, of degree d and order $d+1$, whose support is $[\tau_j, \tau_{j+d+1}]$.
For all $m \in \mathbb{N}^*$ such that $\tau_m < \tau_{m+1}$ and $m \geq d+1$ and $m \leq N+d$, the B-splines that are non-zero in the interval $[\tau_m, \tau_{m+1}[$ are

$$B_\ell^{d+1},\ \ell = m-d, \ldots, m,$$

with the exception that $B_m^{d+1}(\tau_m) = 0$. For $x \in [\tau_m, \tau_{m+1}[$, we can compute $b_j = b_j(x),\ j = 1, \ldots, d+1$, where $b_j(x) = B_{m-d-1+j}^{d+1}(x)$, by the following algorithm:

$b_1 \leftarrow 1$
for $\ell = 1, \ldots, d$ **do**
  $\alpha \leftarrow \frac{x - \tau_{m+1-\ell}}{\tau_{m+1} - \tau_{m+1-\ell}} b_1$
  $b_1 \leftarrow b_1 - \alpha$
  for $k = 2, \ldots, \ell$ **do**
    $\beta = \frac{x - \tau_{m+k-\ell}}{\tau_{m+k} - \tau_{m+k-\ell}} b_k$
    $b_k \leftarrow b_k + \alpha - \beta$
    $\alpha \leftarrow \beta$
  end for
  $b_{\ell+1} \leftarrow \alpha$
end for

Now, we look for a function f_h of the form

$$f_h(\eta) = \sum_{j=1}^{N+d} c_j B_j^{d+1}(\eta),\ \eta_{\min} \leq \eta < \eta_{\max}.$$

For $i = 1, \ldots, N$, we have

$$f_i = f_h(\eta_i) = \sum_{k=1}^{d} b_k(\tau_{i+d}) c_{i-1+k}.$$

This is enough for periodic boundary conditions. For open boundary conditions, we can add the d conditions

$$f_1^{(\ell)} = f_h^{(\ell)}(\eta_1) = \sum_{k=1}^{d} b_k^{(\ell)}(\tau_{d+1}) c_k,\ \ell = 1, \ldots, (d-1)/2,$$
$$f_{N+1}^{(\ell)} = f_h^{(\ell)}(\eta_{N+1}) = \sum_{k=2}^{d+1} b_k^{(\ell)}(\tau_{N+d+1}^-) c_{N-1+k},\ \ell = 0, \ldots, (d-1)/2.$$

Typically, when periodic boundary conditions are not used, we take 0 for the derivatives and extend the solution outside $[\eta_{\min}, \eta_{\max}[$ to be

$$f_h(\eta) = f_1,\ \eta \le \eta_{\min},\ f_h(\eta) = f_{N+1},\ \eta \ge \eta_{\min}.$$

Note that, from [32], we can then obtain explicitly the $d+1$ coefficients $c_k,\ c_{N-1+k},\ k = 1, \dots, (d+1)/2$, and the $N-1$ remaining coefficients are then solution of a d-diagonal system, with matrix

$$M = \begin{pmatrix} b_{(d+1)/2}(\tau_{2+d}) & \dots & b_d(\tau_{2+d}) & 0 & \dots & \dots & 0 \\ 0 \dots 0 & b_1(\tau_{i+d}) & \dots & b_{(d+1)/2}(\tau_{i+d}) & \dots & b_d(\tau_{i+d}) & 0 \dots 0 \\ 0 & \dots & \dots & 0 & b_1(\tau_{N+d}) & \dots & b_{(d+1)/2}(\tau_{N+d}) \end{pmatrix}.$$

Once the coefficients are computed, we obtain for $i = 1, \dots, N,\ 0 \le \alpha < 1$

$$f_h(\eta) = \sum_{k=1}^{d+1} c_{i-1+k} b_k(\tau_{i+d} + \alpha(\tau_{i+d+1} - \tau_{i+d})),\ \eta = \eta_i + \alpha(\eta_{i+1} - \eta_i).$$

Lagrange interpolation: LAG(2d+1) (see e.g. [23, 9, 33])
We consider here a uniform mesh. Lagrange interpolation of degree $2d+1$ reads

$$f_h(\eta) = \sum_{k=-d}^{d+1} w_k^d(\alpha) f_{i+k},\ \eta = \eta_i + \alpha(\eta_{i+1} - \eta_i),\ i = 1, \dots, N,\ 0 \le \alpha < 1,$$

with $w_k^d(\alpha) = \frac{\prod_{j=-d,\ j\neq k}^{d+1}(\alpha - j)}{\prod_{j=-d,\ j\neq k}^{d+1}(k-j)}$. As boundary conditions, apart from periodic, we often use

$$f_j = f_{N+1},\ j \in \mathbb{N},\ j \ge N+1;\ f_j = f_1,\ j\ \in \mathbb{Z},\ j \le 1. \tag{4.3}$$

Cubic Hermite interpolation: $H(p)$ (see e.g. [23] (for $p = 4$), [34]) Let $p \in \mathbb{N}$. We consider here again a uniform mesh, and write

$$f_h(\eta) = w_{0,0}(\alpha) f_i + w_{0,1}(\alpha) f'_{i^+} + w_{1,0}(\alpha) f_{i+1} + w_{1,1}(\alpha) f'_{(i+1)^-},$$
$$\eta = \eta_i + \alpha(\eta_{i+1} - \eta_i),\ i = 1, \dots, N,\ 0 \le \alpha < 1,$$

with $w_{1,0}(\alpha) = \alpha^2(3-2\alpha),\ w_{0,0}(\alpha) = 1 - w_{1,0}(\alpha), w_{0,1}(\alpha) = \alpha(1-\alpha)^2, w_{1,1}(\alpha) = \alpha^2(\alpha - 1)$.

The derivatives are reconstructed in the following way

$$f'_{i^\pm} = \sum_{\ell=r^\pm}^{s^\pm} b_\ell^\pm f_{i+\ell}.$$

For $p \in \mathbb{N}^*$, we take $r^+ = -\lfloor \frac{p}{2} \rfloor,\ s^+ = \lfloor \frac{p+1}{2} \rfloor,\ r^- = -s^+,\ s^- = -r^+$,

$$-b^-_{-\ell} = b^+_\ell = \frac{\prod_{j=r^+,\ j\notin\{0,\ell\}}^{s^+}(-j)}{\prod_{j=r^+,\ j\neq\ell}^{s^+}(\ell-j)},\ \ell = r^+,\dots,s^+,\ \ell\neq 0,$$

$$-b^-_0 = b^+_0 = -\sum_{j=r^+,\ j\neq 0}^{s^+} b^+_j.$$

As examples, we have (negative indices are before ; in the brackets)

$$\begin{array}{ll} p=1 & b^+ = (-1,1),\\ p=2 & b^+ = (-1/2;0,1/2),\\ p=3 & b^+ = (-1/3;-1/2,1,-1/6),\\ p=4 & b^+ = (1/12,-2/3;0,2/3,-1/12),\\ p=5 & b^+ = (1/20,-1/2;-1/3,1,-1/4,1/30),\\ p=6 & b^+ = (-1/60,3/20,-3/4;0,3/4,-3/20,1/60). \end{array} \tag{4.4}$$

Boundary conditions are treated like for the Lagrange case.

4.5 Extension to $2D$ for BSL2D

This is achieved using a tensor product formulation. For the splines, for each $j = 1,\dots,N_2+1$, we compute first coefficients $c^1_{ij},\ i = 1,\dots,N_1+d_1$, from the values $f_{ij},\ i = 1,\dots,N_1+1$. Then, for each $i = 1,\dots,N_1+d_1$, we compute coefficients $c_{ij},\ j = 1,\dots,N_2+d_2$, from the values $c^1_{ij},\ i = 1,\dots,N_1+d_1,\ j = 1,\dots,N_1+1$. The formula for interpolation is then

$$f_h(\eta^1,\eta^2) = \sum_{k=1}^{d_1+1}\sum_{\ell=1}^{d_2+1} c_{i-1+k,j-1+\ell} b_k(\tau_{i+d_1} + \alpha(\tau_{i+d_1+1}-\tau_{i+d_1}))$$
$$b_\ell(\tau_{j+d} + \beta(\tau_{j+d_2+1}-\tau_{j+d_2})),$$

for

$$\eta^1_i = \eta^1_i + \alpha(\eta^1_{i+1}-\eta^1_i),\ i = 1,\dots,N_1,\ 0\le\alpha<1,\ \eta^2_j = \eta^2_j + \beta(\eta^2_{j+1}-\eta^2_j),$$
$$j = 1,\dots,N_2,\ 0\le\beta<1.$$

Note that for open boundary conditions (in both directions), in addition to $f_{ij},\ i = 1,\dots,N_1+1,\ j = 1,\dots,N_2+1$, we have to specify

$$f^{(\ell,0)}_{ij} = \frac{\partial^\ell f_h}{\partial\eta^1}(\eta^1_i,\eta^2_j),\ j = 1,\dots,N_2+1,\ i\in\{1,N_1+1\},\ \ell = 1,\dots(d-1)/2,$$

and also

$$f^{(0,\ell)}_{ij} = \frac{\partial^\ell f_h}{\partial\eta^2}(\eta^1_i,\eta^2_j),\ i = 1,\dots,N_1+1,\ j\in\{1,N_2+1\},\ \ell = 1,\dots(d-1)/2,$$

together with

$$f_{ij}^{(\ell_1,\ell_2)} = \frac{\partial^{\ell_1}\partial^{\ell_2} f_h}{\partial\eta^1\partial\eta^2}(\eta^1,\eta^2), \eta^k \in \{\eta_1^k, \eta_{N_k+1}^k\},\ \ell_k = 1,\dots(d_k-1)/2,\ k = 1,2.$$

From $f_{ij}^{(\ell,0)}$ and f_{ij}, we compute $c_{ij}^1,\ i = 1,\dots,N_1+d,\ j = 1,\dots,N_2+1$. From $f_{ij}^{(\ell_1,\ell_2)}$ and $f_{ij}^{(0,\ell)}$, we compute missing coefficients called $c_{ij}^{1,(\ell)},\ i = 1,\dots,N_1+d,\ j \in \{1, N_2+1\},\ \ell = 1,\dots,(d_2-1)/2$, in order to compute finally, together with c_{ij}^1 the coefficients c_{ij}.
For Lagrange interpolation, the formula is

$$f_h(\eta^1,\eta^2) = \sum_{k=1}^{d_1+1}\sum_{\ell=1}^{d_2+1} w_k^{d_1}(\alpha) w_\ell^{d_2}(\beta) f_{i+k,j+\ell},$$

and for Hermite, we can write

$$f_h(\eta^1,\eta^2) = \sum_{k_1,k_2,\ell_1,\ell_2=0}^{1} w_{k_1,k_2}(\alpha) w_{\ell_1,\ell_2}(\beta) f_{i(k_2),j(\ell_2)}^{(k_1,\ell_1)},$$

defining $i(0) = i^+,\ i(1) = (i+1)^-$ and similarly, $j(0) = j^+,\ j(1) = (j+1)^-$. We take here $f_{i^\pm,\cdot}^{(0,\cdot)} = f_{i,\cdot}^{(0,\cdot)}$ and $f_{\cdot,j^\pm}^{(\cdot,0)} = f_{\cdot,j}^{(\cdot,0)}$. We have to compute the values

$$\begin{aligned}
&f_{i\pm,j}^{(1,0)},\ i = 1,\dots,N_1+1,\ j = 1,\dots,N_2+1,\\
&f_{i,j\pm}^{(0,1)},\ i = 1,\dots,N_1+1,\ j = 1,\dots,N_2+1,\\
&f_{i\pm,j\pm}^{(1,1)},\ i = 1,\dots,N_1+1,\ j = 1,\dots,N_2+1.
\end{aligned}$$

The values $f_{i\pm,j}^{(1,0)}$ are obtained from $f_{i,j}$ using the operator described in the $1D$ case in the first direction. The values $f_{i,j\pm}^{(0,1)}$ are obtained from $f_{i,j}$ using the same operator in the second direction. Finally, the values $f_{i\pm,j\pm}^{(1,1)}$ are obtained from $f_{i,j\pm}^{(1,0)}$ using the operator in the second direction (or equivalently from $f_{i\pm,j}^{(0,1)}$ using the operator in the first direction).

5 A split conservative semi-Lagrangian method

On curvilinear grids the free stream preservation property is essential for conservative finite difference, finite volume or finite element solvers. This means that for our problem a constant function must be preserved by a constant coefficient advection.

Although the natural form of the conservative semi-Lagrangian method, as proposed in [12], is based on a finite volume formulation where the unknown is a cell average, we find it simpler to use a point based conservative method with a point based unknown as in a Finite Difference scheme. This facilitates

a symmetric reconstruction between the stream function Φ of the advection field and the unknown, which for the conservative form of the equation (3.2) is $\bar{f} = fJ$ as we saw before. And, even more importantly this enables to define the Jacobian J only at the grid points, as well for the distribution function as for the computation of the characteristics. This avoids interpolation of the Jacobian, which is known to be bad for free stream preservation. In the classical Finite Volume type method of [12], the advected points are the cell edges and the Jacobian needs to be interpolated or defined in another manner there, which prevents a direct cancellation as we have for the point based method used in BSL and proposed here for the conservative scheme. A conservative semi-Lagrangian method of this type was introduced by Qiu and Shu [2]. We will consider here a specific example, that permits to have the first order free stream preservation; a further study may be pursued for designing a class of high order stable schemes, sharing this property, in the same spirit of the BSL method; one difficulty is to get the stability of the scheme, as already mentioned in [2]. A general discussion on the respective advantages and drawbacks of Finite Difference, Finite Volume and Discontinuous Galerkin discretisations for conservation laws is given in [35].

5.1 Formulation of the 1D split conservative semi-Lagrangian method (CSL)

Let us start from the model equation in conservative form (3.2) written in logical coordinates, with $\bar{f} = Jf$, which we recall

$$\frac{\partial \bar{f}}{\partial t} + \nabla_\eta \cdot (\frac{1}{J}(\nabla_\eta^\perp \phi)\bar{f}) = 0 \tag{5.1}$$

with $(\nabla_\eta^\perp \phi) = (\partial_{\eta_2}\phi, -\partial_{\eta_1}\phi)^\top$.

Using an operator splitting method, which can be of arbitrary order by alternating the 1D solves with adequate coefficients, we need to solve the following two 1D advections:

$$\frac{\partial \bar{f}}{\partial t} + \frac{\partial}{\partial \eta_1}\left(\frac{1}{J}\frac{\partial \phi}{\partial \eta_2}\bar{f}\right) = 0, \qquad \frac{\partial \bar{f}}{\partial t} - \frac{\partial}{\partial \eta_2}\left(\frac{1}{J}\frac{\partial \phi}{\partial \eta_1}\bar{f}\right) = 0. \tag{5.2}$$

This leads us to introducing the conservative algorithm for a 1D conservative problem of the form

$$\frac{\partial \bar{f}}{\partial t} + \frac{\partial}{\partial \eta}\left(\frac{a}{J}\bar{f}\right) = 0. \tag{5.3}$$

A key ingredient in our algorithm is the sliding average reconstruction, which is classical in conservative Finite Difference methods for conservations laws. This reconstruction aims at defining for a function G defined at grid points a high order approximation of its derivative $\partial_\eta G$ at the same grid points. For this, we introduce a function $\mathcal{G}$ related to G by the formula

$$G(\eta) = \frac{1}{\Delta\eta}\int_{\eta-\frac{\Delta\eta}{2}}^{\eta+\frac{\Delta\eta}{2}} \mathcal{G}(\tilde{\eta})\,\mathrm{d}\tilde{\eta},$$

so that

$$\frac{\partial G}{\partial \eta}(\eta_i) = \frac{\mathcal{G}(\eta_i + \frac{\Delta\eta}{2}) - \mathcal{G}(\eta_i - \frac{\Delta\eta}{2})}{\Delta\eta}. \tag{5.4}$$

When the averages of $\mathcal{G}$ on each cell, which are also the grid values $G(\eta_i)$, are known, the reconstruction by primitive, standard in Finite Volume methods can be used to approximate $\mathcal{G}$ at the cell extremities. Formulas for third and fifth order reconstructions are given in [2], we shall use here a fourth order reconstruction, for which it is straightforward to compute

$$\mathcal{G}(\eta_i + \frac{\Delta\eta}{2}) \simeq \frac{7}{12}(G(\eta_i) + G(\eta_{i+1}) - \frac{1}{12}(G(\eta_{i-1}) + G(\eta_{i+2})). \tag{5.5}$$

Now, a conservative semi-Lagrangian algorithm is based on the conservation property (3.5), which becomes for the 1D case

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\bar{f}(t, H(t;\eta,s))\frac{\partial H}{\partial \eta}(t;\eta,s)\right) = 0.$$

Introducing F a primitive in η of $\bar{f}$,

$$F(t,\eta) := \int_{\eta_1}^{\eta} \bar{f}(t,\tilde{\eta})\,\mathrm{d}\tilde{\eta},$$

where η_1 is chosen arbitrarily for fixing the constant, this can be written equivalently

$$\frac{\partial}{\partial t}\frac{\partial F^*}{\partial \eta}(t,\eta,s) = 0, \text{ with } F^*(t,\eta,s) := F(t, H(t;\eta,s)). \tag{5.6}$$

Let us introduce a uniform grid of our logical domain $[0,1]$ characterised by the number of cells N. The grid points are then defined by $\eta_i = (i-1)\Delta\eta$, $1 \le i \le N+1$, with $\Delta\eta = 1/N$. The unknowns are defined at the grid points $\bar{f}_i^n \simeq \bar{f}(t_n, \eta_i)$ and we denote by $\bar{f}_h^n = (\bar{f}_i^n)_{1\le i\le N+1}$ the collection of the grid values.
Integrating first in t between t_n and t_{n+1} in (5.6) and evaluating at $s = t_{n+1}$, it yields

$$\bar{f}(t_{n+1},\eta) = \frac{\partial F^*}{\partial \eta}(t_n,\eta,t_{n+1}),$$

as $H(t_{n+1};\eta,t_{n+1}) = \eta$ and $\frac{\partial H}{\partial \eta}(t_{n+1};\eta,t_{n+1}) = 1$. We suppose for the moment that $\eta_i^* = H(t_n;\eta_i,t_{n+1})$, the origin of the characteristic ending at η_i satisfies

$$\eta_i \le \eta_i^* < \eta_{i+1}. \tag{5.7}$$

Defining $G(\eta) = F^*(t_n,\eta,t_{n+1}) - F(t_n,\eta)$, we get

$$\bar{f}(t_{n+1},\eta_i) = \bar{f}(t_n,\eta_i) + \frac{\partial G}{\partial \eta}(\eta_i).$$

In order to get the scheme, we approximate $\frac{\partial G}{\partial \eta}(\eta_i)$ from values

$$G(\eta_i) = \int_{\eta_i}^{\eta_i^*} \bar{f}(t, \tilde{\eta})\, \mathrm{d}\tilde{\eta} \tag{5.8}$$

as explained before, that is using (5.4) together with (5.5).
To complete our algorithm we need to approximate $G(\eta_i)$. For this, we use Lagrange interpolation of order 3 on the interval $[\eta_i, \eta_{i+1}[$, using the values $f_{i+\ell}^n,\ \ell = -1, \ldots, 2$, in order to approximate $\bar{f}(t, \tilde{\eta}), \eta \in [\eta_i, \eta_{i+1}[$, in (5.8). Note that other choices may be possible, but we warn the reader that, as noted in [2], for stability reasons, such interpolation has to be combined carefully with the quadrature formula which is here (5.5).

To summarize, our conservative semi-Lagrangian algorithm, generalized to arbitrary displacement, as in [2], is based on the following steps:

1. The backward solution of the characteristics ending at the grid points η_i solution of
$$\frac{\mathrm{d}H}{\mathrm{d}t} = \frac{a}{J}, \qquad H(t_{n+1}) = \eta_i,$$
denoted by $\eta_i^* = H(t_n; \eta_i, t_{n+1})$. These are the same characteristics as for the split 1D BSL method and the algorithm to compute them is the same. We shall consider here the backward Euler method.

2. The computation of the flux $F_{i+1/2}$. We denote by $\eta_i^* = \eta_{i^*} + \alpha_i \Delta\eta,\ 0 \leq \alpha_i < 1$. Let P_i be the polynomial of degree ≤ 3 satisfying $P_i(\eta_{i^*+k}) = \bar{f}_{i^*+k}^n,\ k = -1, 0, 1, 2$. We compute
$$f_{i,k} = \frac{1}{\Delta\eta} \int_{\eta_{i^*+k}}^{\eta_{i+k}^*} P_i(\eta) d\eta,\ k = -1, 0, 1, 2,$$
by using Simpson rule for example. We then define
$$F_{i+1/2} = \frac{7}{12}(f_{i,0} + f_{i,1}) - \frac{1}{12}(f_{i,-1} + f_{i,2}) + \sum_{i+1 \leq k \leq i^*} \bar{f}_k^n - \sum_{i^*+1 \leq k \leq i} \bar{f}_k^n. \tag{5.9}$$

3. The update of the solution. The new value of $\bar{f}$ at the grid points is then given by
$$\bar{f}_i^{n+1} = \bar{f}_i^n + F_{i+1/2} - F_{i-1/2}. \tag{5.10}$$

Note that in the derivation of the scheme, the assumption (5.7) corresponds to the special case $i^* = i$. Formula (5.9) is chosen by considering that the local displacement is the decomposition of an exact displacement of a given number of cells plus a positive displacement less than one cell. We can check, as in [2] that the formula is continuous when α_i approachs 1. We clearly see from the flux form (5.10), that the scheme is mass conservative.
Note that in [2] explicit formulae are given for third and fifth order in the context of WENO. We consider here the example of fourth order and do not apply a WENO procedure. For constant displacement (with Jacobian equal to one), this scheme corresponds to use the BSL scheme with Lagrange interpolation of degree 4, where the stencil is chosen according to the sign of the displacement.

5.2 Free stream preservation

As for the BSL method, where the mass is only preserved up to first order in Δt, we shall prove here that $\bar{f}_{i,j}$ is preserved by the update formula up to first order in Δt if $f = 1$ or equivalently $\bar{f}_{i,j} = J_{i,j}$, which is our version of free stream preservation. For this, taking $\bar{f}_i^n = J_i$, we consider for simplifying the exposition of the proof that (5.7) occurs, but we could adapt the proof, removing the hypothesis. We use again the formulation of the update formula from the last section:

$$\bar{f}_i^{n+1} = \bar{f}_i^n + (F^* - F)'(\eta_i) = J_i + (F^* - F)'(\eta_i)$$

and compute an approximation of $(F^* - F)'(\eta_i)$ in the limit of small Δt.

As we saw in the previous subsection $F^* - F$ is obtained by reconstruction of the grid function with sliding average

$$\int_{\eta_i}^{\eta_i^*} f_h(\eta)\,\mathrm{d}\eta = -\Delta t \frac{a(\eta_i)}{J(\eta_i)} f_h(\eta_i) + O(\Delta t^2) = -\Delta t\, a(\eta_i) + O(\Delta t^2)$$

as we assume that $f_h(\eta_i) = J(\eta_i) = J_i$. Then denoting by $\mathcal{S}[a_h]$ the sliding average reconstruction of $a_h = (a(\eta_i))_{1\le\eta\le N+1}$, the update formula becomes for $\bar{f}_i^n = J_i$

$$\bar{f}_i^{n+1} = J_i - \Delta t\, \mathcal{S}[a_h](\eta_i) + O(\Delta t^2).$$

Let us now apply this formula to the two split steps of our method assuming a first order Lie splitting, but the same argument holds for higher order splittings. For the advection in the first direction we have for a given j, $a = \partial_{\eta_2}\Phi$. Numerically Φ is approximated by its grid values $\Phi_h = (\Phi_{i,j})_{1\le i\le N_1, 1\le j\le N_2}$, we denote also by $\Phi_{h,j} = (\Phi_{i,j})_{1\le i\le N_1}$ and $\Phi_{h,i} = (\Phi_{i,j})_{1\le j\le N_2}$ one grid line at constant j and i respectively. The derivative in the η_2 direction is approximated by the sliding average reconstruction in the η_2 direction $\mathcal{S}_2[\Phi_{h,i}]$. Then $a_{h,j} = (\mathcal{S}_2[\Phi_{h,i}](\eta_j))_{1\le i\le N_1}$. So that our first update reads

$$\bar{f}_{i,j}^* = J_{i,j} - \Delta t\, \mathcal{S}_1[\mathcal{S}_2[\Phi_{h,i}](\eta_j)](\eta_i) + O(\Delta t^2).$$

For the second split step, we first note that, as the reconstruction is linear the two terms above can be treated separately (and the higher order terms also). Then as there is a Δt in factor of the second term, this will be a $O(\Delta t^2)$ after reconstruction. So, as now $a = -\partial_{\eta_1}\Phi$, the formula is the symmetric of the other split step, which yields

$$\begin{aligned}\bar{f}_{i,j}^{n+1} = J_{i,j} - \Delta t \left(\mathcal{S}_1[\mathcal{S}_2[\Phi_{h,i}](\eta_j)](\eta_i) - \mathcal{S}_2[\mathcal{S}_1[\Phi_{h,j}](\eta_i)](\eta_j)\right) + O(\Delta t^2)\\ = J_{i,j} + O(\Delta t^2)\end{aligned}$$

because of the linearity and the symmetry of the sliding reconstruction in the two directions. This proves that the constant states are preserved up to first order, which is needed for a consistent conservative approximation on curvilinear grids.

6 Numerical results

6.1 Rotation

The domain is $\Omega = [-\pi, \pi]^2$. We take

$$\Phi(x^1, x^2) = -((x^1)^2 + (x^2)^2)/2$$

and solve (2.2). The mapping, referred to as deformed mesh (see e.g. [27]) is given by

$$\begin{array}{l} x^1(\eta^1, \eta^2) = \eta^1 + \alpha \sin(\eta^1 2\pi/L_{x^1}) \sin(\eta^2 2\pi/L_{x^2}), \\ x^2(\eta^1, \eta^2) = \eta^2 + \alpha \sin(\eta^1 2\pi/L_{x^1}) \sin(\eta^2 2\pi/L_{x^2}) \end{array} \tag{6.1}$$

for $(\eta^1, \eta^2) \in \hat{\Omega} = \Omega$ and for $0 \leq \alpha < 1$. Here, $L_{x^1} = L_{x^2} = 2\pi$.
We use the *cos-bell* initial function defined by

$$f_0(x^1, x^2) = \begin{cases} \cos(r)^6, \text{ for } r < \pi/2, \\ 0, \text{ else,} \end{cases}$$

with $r = \sqrt{(x^1 - x_c^1)^2 + (x^2 - x_c^2)^2}$ and $x_c^1 = 1,\ x_c^2 = -0.2$.
We use a Verlet scheme for computing the characteristics backward in BSL2D, which leads to a second order in time scheme; for the evaluation of the fields, we use cubic splines in the fixed point algorithm (tolerance is put to 10^{-12} an a maximum of 1000 iterations is allowed; when the convergence is not reached, a warning message is given). Periodic boundary conditions are used for this test, which can lead to small errors as the solution is not periodic but vanishes in the vicinity of the boundary. For the choice of the interpolation, we consider here only piecewise cubic polynomial reconstructions: SPL(3) and $H(p)$. Note that LAG3 corresponds to $H(3)$.

When not specified, we use the *natural* derivative for the potential, that is: the same derivative, as the derivative of the interpolation scheme. For $H(p)$, when p is odd, the derivative is discontinuous; we choose the derivative of order $p+1$ as natural derivative for the potential (we do not consider a discontinuous electric field here); this corresponds to taking the mean between the right and left derivative of order p.

We will refer to FD4 (resp. FD6) for computing the derivative using (4.4), with $p = 4$ (resp $p = 6$).
We consider different discretizations in space: $N = N_1 = N_2 \in \{32, 64, 128, 256\}$ and time: $\Delta t = 2^{-i},\ i \in \{0, 1, \ldots, 10\}$. The schemes under consideration are here H2, H3, H4, H5, H6, H9, H10, H17 and SPL3.
Numerical results are given on Figure 1 (solution and error at time $T = 20$), Figure 2 (error vs time step) and Figure 3 (evolution of mass).
On Figure 1, we see the solution and error of H17, using $\Delta t = 2^{-6}$, $N = 32$ (top) and $N = 64$ (bottom) at time $T = 20$ that is after $10/\pi$ turns. We see that the error is reduced, taking a finer mesh and that the error is not located on the boundary, that is periodic boundary conditions do not lead to problems, in this test. The mesh is superposed to the solution, and we see that it is quite

deformed. Note that the Jacobian becomes singular for $\alpha = 1$. The higher α is, the more difficult it is to get accurate solution (see [36]). Here we always take $\alpha = 0.9$ to see the robustness of the different methods.
On Figure 2, we give the error in function of i, with $\Delta t = 2^{-i}$ using the different reconstructions and $N = 32$ (top left), $N = 64$ (top right), $N = 128$ (bottom left) and $N = 256$ (bottom right). The error is computed in L^∞ norm taking the best constant that approaches in least square sense the error between $T = 10$ and $T = 20$ (we use the `fit` command of gnuplot). The error is multiplied by $(N/32)^3$ as we expect third order convergence in space for the different schemes (except $H2$).
So, we observe how the error behaves with respect to time for different discretizations in space. One can expect to have an error of the form

$$C_1 \Delta t^2 + \frac{C_2}{N^3} \tag{6.2}$$

(see [9] for example). We distinguish basically three zones. A first zone ($i = 0, 1, 2, 3$) when Δt is too big to observe convergence. Typically the time step is too big for solving properly the characteristics which are quite stiff to solve, as the mesh is very distorted. Then we observe a second zone, where the error decreases like Δt^2. This zone becomes bigger as N increases. Finally, there is a third zone, where the error saturates. Multiplying by $(N/32)^3$ permits to have the error for $N = 32$ and to see how the "constant" in the space error behaves (which is C_2 if the error is of the form (6.2)). For $N = 32$ (top left), we observe that increasing the value of p in $H(p)$ increases the accuracy and SPL3 is little better that H6. Moreover, when Δt becomes smaller, we see bad effect for the schemes that use a centered reconstruction for the derivatives: SPL3 and H10 (looking at the solution, we see that the error comes from the boundary; using other boundary conditions may lead to improvements); for H6, which is a little more diffusive, it is not observed here. When $i <= 3$ the error is big and does not depend on the reconstruction. H2 gets worser increasing N, as it is not a third order scheme; we also observe that taking smaller time steps leads to bigger error, when time error is not predominant. For H3, the error saturates or little increase with Δt. We remark that is of order 3, comparing $N = 128$ and $N = 256$. For the other schemes, increasing N, the difference between the schemes is diminished, as the reconstruction for the derivatives is higher than third order (for cubic splines it is fifth order, and the constant seems to be smaller than for $H5$).
On Figure 3, we represent the evolution of mass, for the different reconstructions, with $N = 32$ (left), $N = 64$ (right), and $\Delta t = 2^{-5}$ (top), $\Delta t = 2^{-6}$ (middle) and $\Delta t = 2^{-7}$ (bottom).

-We observe an increase of error of mass, when the bell crosses the region where the mesh is the most distorted.

-Increasing N improves the mass for the lower accuracy methods (essentially H2, H3, H4 and H5); it improves also the mass for all the methods, when the bell is not touching the most distorted region.

-For fixed value of N, by diminishing the time step, mass conservation is generally improved.

-For $N = 64$, we only see 3 types of curves: H2, H3 and the others which are almost the same. In order to see more differences on the different methods, we have to take a smaller value of N, like $N = 32$.

-We also can check that the centered reconstruction H2 (resp. H4) is worse for mass conservation than H3 (resp H5) when Δt is big, but they become better, when Δt is smaller, as expected from the analysis.

-We can also observe worse mass conservation using non natural reconstructions for the derivatives (not shown here), when $N = 32$, but the differences are slight, and we will see more difference in the guiding center case.

Kelvin-Helmholtz instability in a periodic box

We refer for example to [12] for this test case. This is a non linear test case, where the stream function Φ depends on f via the solution of the Poisson equation $-\Delta\Phi = f$. The initial distribution f_0 is given by the formula :

$$f_0(x^1, x^2) = \sin(x^2) + \beta \cos(\sigma x^1)$$

where $\beta = 0.015$ and $\sigma = 0.5$. Periodic conditions are considered both in x^1 and x^2 direction. The domain is $[0, L_{x^1}] \times [0, L_{x^2}]$, with $L_{x^1} = \frac{2\pi}{\sigma}$, $L_{x^2} = 2\pi$. We use the same deformed mesh as in (6.1), with $\alpha = 0.9$. We use mudpack [37] for the curvilinear Poisson solver.
For BSL2D, we use a predictor-corrector scheme for the time loop as in [36], together with Verlet for the characteristics (as in the case of the rotation).
For BSL1D, we replace the $2D$ advection, by $1D$ advection, using Strang splitting, as in [12]; a trapezoidal rule is used for the $1D$ characteristics, and again cubic splines for the evaluation of the fields, in the fixed point algorithm.
For CSL, we compute the characteristics as for BSL1D and use the same Strang splitting.
The schemes under consideration are here SPL3, SPL3-FD6 (SPL3 with FD6 instead of cubic splines for the derivatives of the potential), SPL3-1d (BSL1d with SPL3), H6, H6-SPL3 (H6 with cubic splines instead of FD6 for the derivatives of the potential), H17, H17-1d (BSL1d with H17) and CSL.

Numerical results for BSL are given on Figure 4 (solution at time $T = 50$) and Figure 5 (evolution of mass). Numerical results for CSL are given on Figure 6 (solution at time $T = 50$) and Figure 7 (evolution of free streaming error). On Figure 7, L^2 norm et energy evolution are also shown for CSL and BSL.

On Figure 4, we represent f at time $T = 50$ using H17 (left) and SPL3 (right), with $N = 256$, $\Delta t = 2^{-10}$ (top), $N = 128, \Delta t = 2^{-10}$ (middle) and $N = 128$, $\Delta t = 2^{-5}$ (bottom). Using $N = 128$, we have less details than with $N = 256$ for H17. This scheme leads to similar results, using different small enough time steps for a given N. Note that if we would take bigger time steps

(like $\Delta t = 2^{-3}$ which can be taken in the case of uniform mesh with $\alpha = 0$), the error in time would increase and we would get different (wrong) results. We see that this scheme is particularly robust with respect to Δt. For example, for the fixed value of $N = 128$ (the exercise could be repeated for $N = 256$), using $\Delta t = 2^{-5}$ or $\Delta t = 2^{-10}$ leads to similar result. On the other hand, SPL3, works very badly when the time step is small; but, using a bigger time step as $\Delta t = 2^{-5}$, for $N = 128$, gives an acceptable result almost as accurate as H17 for the same value of N. Indeed, we see some more details in H17, which are also present on the "reference solution" (H17, $N = 256$ top left).
On Figure 5, we represent the evolution of mass for $N = 256$, $\Delta t = 2^{-5}$ (top left), $\Delta t = 2^{-6}$ (top right), $\Delta t = 2^{-7}$ (middle left), $\Delta t = 2^{-8}$ (middle right), $\Delta t = 2^{-9}$ (bottom left) and $\Delta t = 2^{-10}$ (bottom right).

-We remark that BSL1d (here SPL3-1d and H17-1d) leads to huge mass error, when the time step is big (see Figure top left); this confirms previous analysis on uniform grid (see [12] for example); on the other hand, diminishing the time step ($i = 5$ to $i = 9$) BSL1d and BSL2d become similar: SPL3-1d converges to SPL3, and similarly H17-1d to H17.

-Diminishing the time step ($i = 5$ to $i = 9$), we get better conservation of mass, except for the schemes that do not use the natural reconstruction of the derivatives (SPL3-FD6 and H6-SPL3). This confirms the theoretical analysis of Section 4.3, which was more difficult to point out in the rotation case.

-For $i = 10$, we have first better conservation of mass for SPL3 than for H17 (which was the best for $i \leq 9$) this is coherent with the theoretical analysis of Section 4.3. On the other hand, the solution develops a lot of oscillations (as shown on Figure 4 top right) and a numerical instability appears. Note that instability appears before for SPL3-1d than for SPL3. For H6, we also get an instability (which was not in the rotation case).

On Figure 6, we represent f at time $T = 50$ using CSL, with $N = 128$ (left) and $N = 256$ (right), for $\Delta t = 2^{-5}$ (top), $\Delta t = 2^{-6}$ (middle) and $\Delta t = 2^{-7}$ (bottom). The solution seems to converge to the "reference solution" (H17, $N = 256$ top left). Sporadic out of bounds value of f appear and become bigger by taking a smaller time step and these values are attenuated by refining the space grid. We remark some slight grid effect, taking the biggest Δt (that is $\Delta t = 2^{-5}$); this may be due to the fact that the characteristics are not computed accurately enough. Some numerical oscillations are present and in the mean time some details are diffused; this may be explained from the order of the scheme and the behaviour as Δt tends to 0 (here fourth order).

On Figure 7, we represent the free stream error. To compute it, we consider a function initialized to 1 and let it evolve as f with the same advection field and the same scheme. As the field is divergence free, the function should remain 1, as it is the case for BSL. The free stream error is then the L^∞ error between the solution and 1. It is represented versus time. On top left, we multiply the error by $(2^{-5}/\Delta t)^2$ and otherwise not. It is represented on top for CSL with the natural reconstruction of the field, that is, FD4. On Figure 7 middle, we change

the reconstruction to FD6 (left) and SPL3 (right). On Figure 7, bottom, we represent the evolution of L^2-norm (left) and energy (right) for CSL, and BSL (H3,H6,H17 and SPL3), using $\Delta t = 2^{-5}$, or $\Delta t = 2^{-6}$ and $N = 256$. As seen on Figure 7 top left, we check the order 2 accuracy of the free stream preservation. We can notice, that is holds both in the linear phase, when everything is smooth and remains in the non linear phase, where we are far from convergence. On the other hand, the other reconstructions of the fields do not lead to this property. Taking $\Delta t = 2^{-7}$, $N = 128$, we even observe that the scheme can become unstable (Figure 7 middle left). So the natural reconstruction of the field seems here to prevent from the appearance of instability; the sporadic out of bounds values can become higher and higher; so that at a certain level, this destroys the solution. Other ingredients may be needed for garantying the stability of the scheme and to permit to go to higher order schemes. On Figure 7, bottom, it is confirmed that diffusion is not as strong as $H3$, but more than other usual BSL schemes ($H6, SPL3, H17$). We also see that energy is quite well preserved for CSL, certainly because the scheme is mass conservative; on the other hand, diminishing the time step also improves energy conservation for the BSL method (except for $H3$); this is coherent with the analysis about mass conservation.

On Figure 8, we represent the free streaming error, the mass, the L^2 norm and the energy, in the cartesian case (i.e. $\alpha = 0$). We observe that free streaming error is less important for the conservative method (Figure 8, top left). We remark that it is more important for $N = 256$ than $N = 128$, when $\Delta t = 2^{-5}$. This may be due to the fact that the solution is more complex when $N = 256$. On the other hand, diminishing the time step, we observe that the free streaming error diminishes, as expected, because we use FD4 for the reconstruction of the derivatives. Mass conservation is really improved, using $\alpha = 0$ (Figure 8 top right) for the BSL methods. The effect of diminishing the time step leads to small improvement of the mass conservation, in constrast to the curvilinear case ($\alpha = 0.9$), but the conservation is already a lot better. We remark no real difference for the L^2 norm (Figure 8, bottom left) and the energy is strangely not conserved as well; the curve is however less oscillating.
On Figure 9, we represent the mass evolution, for bigger time steps: $\Delta t = 2^{-3}$ or $\Delta t = 2^{-4}$, which corresponds to more standard time steps that are used for this test case in cartesian geometry (see [12] e.g.) and for $\alpha \in \{0, 0.1, 0.25, 0.5\}$. We remark that the mass is really better conserved on the cartesian mesh ($\alpha = 0$), and as α increases, mass conservation is degraded (for non conservative methods, of course). This can be seen even for $\alpha = 0.1$ which is almost uniform. As already shown, diminishing the time step has a beneficial effect, which is not so clear for $\alpha = 0$ or $\alpha = 0.1$ but which becomes clearer increasing the value of α. L^2 norm and energy evolutions are plotted on Figure 10 for $\alpha \in \{0, 0.5, 0.9\}$. On these quantities which are less accurately conserved in the cartesian case, we do not see differences between the cartesian case $\alpha = 0$ and the case $\alpha = 0.5$. This is good news, as it tells us that the time step restriction is not so severe; diminishing the time step mainly helps in the conservation of the mass. In the case of $\alpha = 0.9$, for which reasonable simulation are shown for $\Delta t \leq 2^{-5}$, exhibit

very bad mass conservation (here, there is no picture; the result is similar to $\alpha = 0.5$, but with values around 1 for $\Delta t = 2^{-4}$ and between 6 and 7 for $\Delta t = 2^{-4}$ at final time $T = 100$). We see on Figure 10 that for $\alpha = 0.9$, the quantities are changed, especially for $\Delta t = 2^{-3}$, and also for the CSL method. Note that for $\Delta t = 2^{-3}$, the simulation stops in the CSL method, as we have added a test that prevents from having intersection of characteristics. We see that the energy is less accurately conserved for $\Delta t = 2^{-4}$, for the CSL method, whereas it was not changing much from $\alpha = 0$ to $\alpha = 0.5$. For the BSL method, the energy conservation is really worse for $\Delta t = 2^{-3}$. On the other hand, we remark that H17 performs the best for $\Delta t = 2^{-4}$ in comparison to the other methods.
A picture of the meshes for $\alpha = 0.25$ and $\alpha = 0.5$ is depicted on Figure 11, using $N = 32$ instead of $N = 256$ in order to better see the mesh.

7 Conclusion

We have studied the semi-Lagrangian method on curvilinear grids and looked at mass and constant states preservation issues. A (mass) conservative method that preserves constant states up to first order in time has been exhibited and compared with the classical advective backward semi-lagrangian method, which in turn always preserves constant states and for which we have shown that it conserves mass up to first order in time. When the mesh becomes distorted, the time step has to be small enough; otherwise important errors appear, as in particular characteristics are badly solved. In this setting, Hermite interpolation with high order odd reconstruction of the derivatives exhibits more favorable behavior than cubic splines, but the Hermite method can be also more expensive; so we may stick to cubic splines when the mesh is not too distorted. The next step is to adapt such methods to gyrokinetic simulations. The classical BSL method may be preferred, as we can make it work for several reconstructions and it is more robust, even though there is a loss of mass conservation. The CSL method could also be tried, but we fear for stability and robustness issues, as the design of such a scheme was already not easy in this simplified context. We have tested the methods for a quite severely deformed mesh; we have seen that, when the mesh is less deformed, the time constraint is relaxed. The grid in a gyrokinetic simulation is imposed by the background magnetic field, which does not vary in time. Hence the methodology described in this paper can be directly applied as we can define a transformation, based on a B-spline representation, which maps a logical domain onto the desired physical domain. Typical gyrokinetic simulations have magnetic configurations not far from equilibrium, thus including the fluctuations of the magnetic field in the grid information is not necessary for most applications. Among the new difficulties that we can encounter in 5D gyrokinetic simulations, one is due to the dimension, which leads to heavy computations; the model can also lead to numerical instabilities, but collisions can be added to prevent from this fact; the whole system can also become quite complex with all the physical terms; boundary conditions and

change of geometry have also to be treated correctly. A possible further study is also to add slope limiters or other tools in order to prevent from sporadic out of bounds values which can destroy the solution when they become too high.

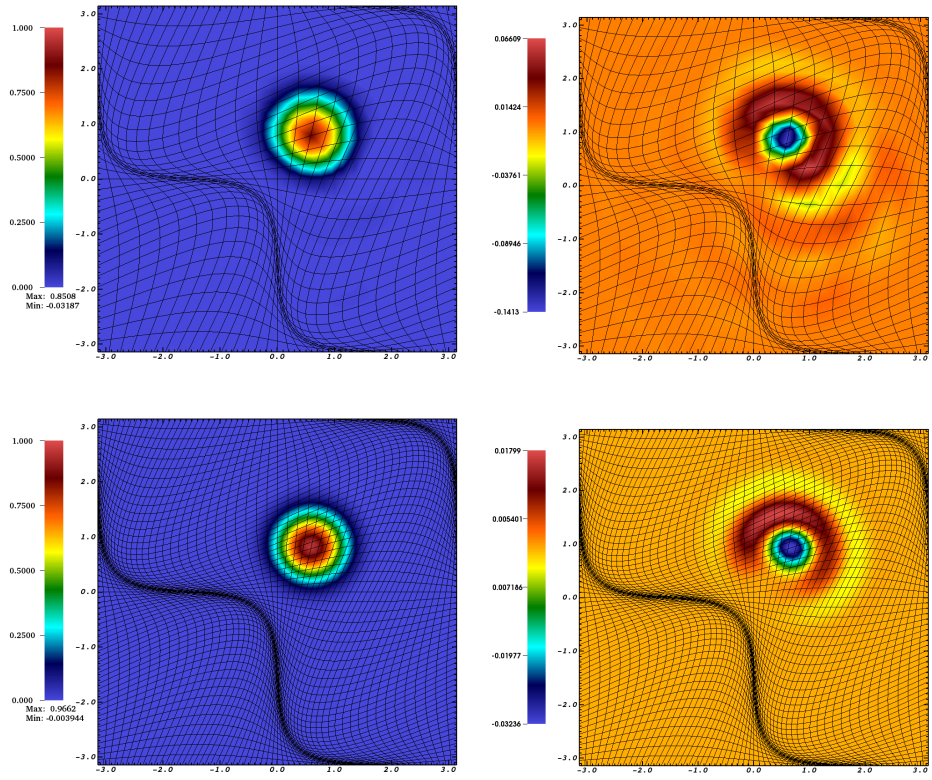

Figure 1: Rotation case. Solution at time $T = 20$, on deformed mesh: H17 $\alpha = 0.9, \Delta t = 1/2^6$ (left: solution, right: error), with $N = 32$ (top) and $N = 64$ (bottom).

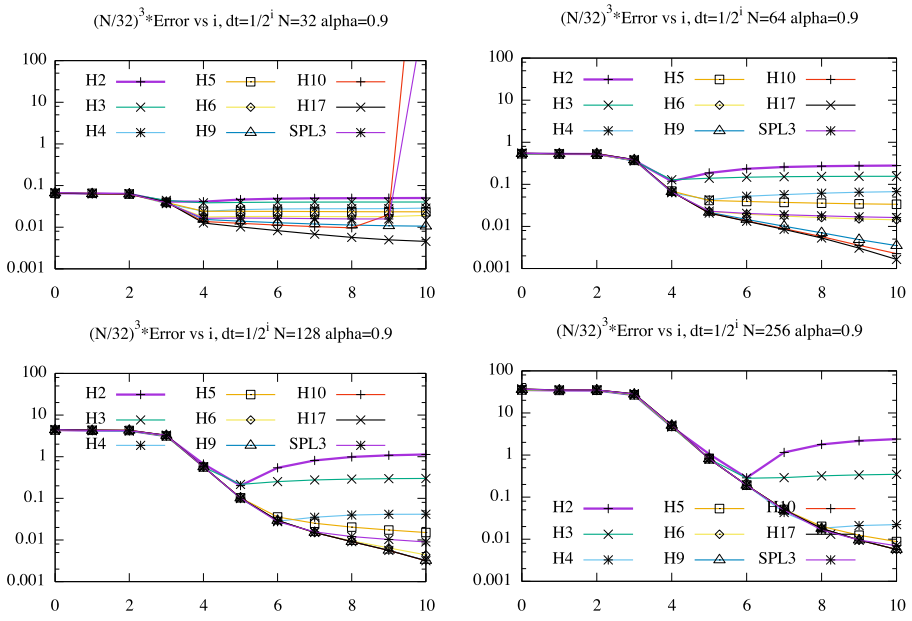

Figure 2: Rotation case. Average of the error in maximum norm between $t = 10$ and $t = 20$ divided by t versus i, where $\Delta t = 1/2^i$.

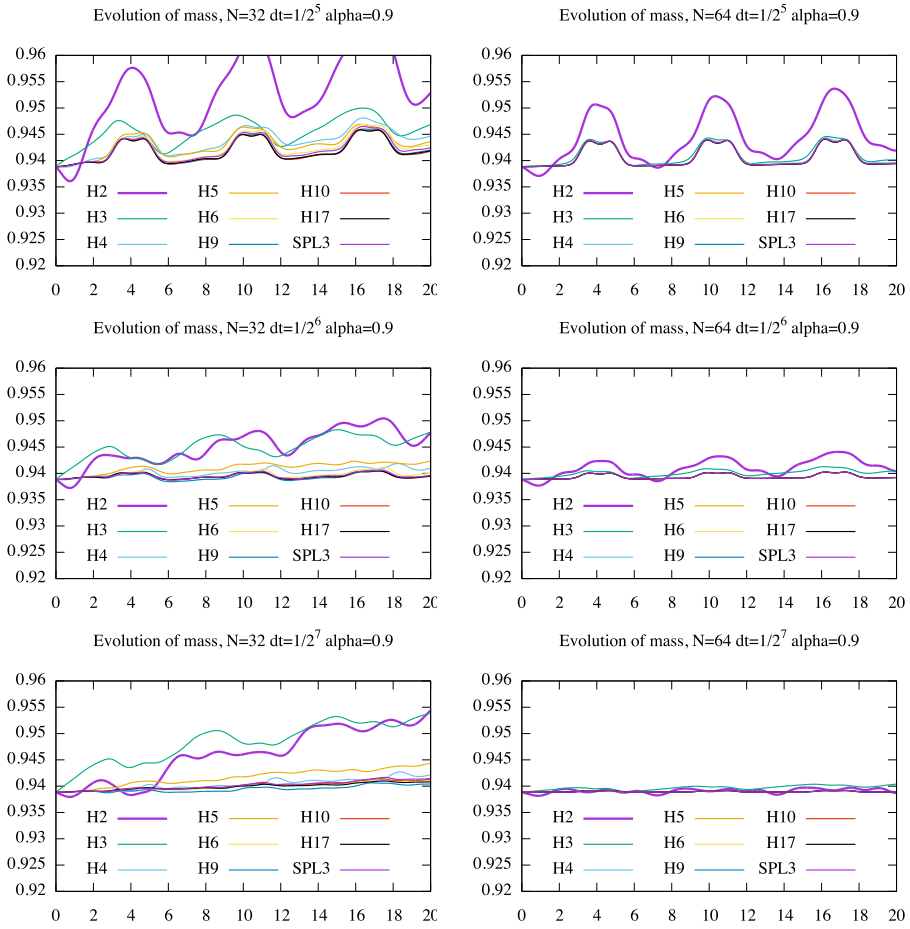

Figure 3: Rotation case. Evolution of mass.

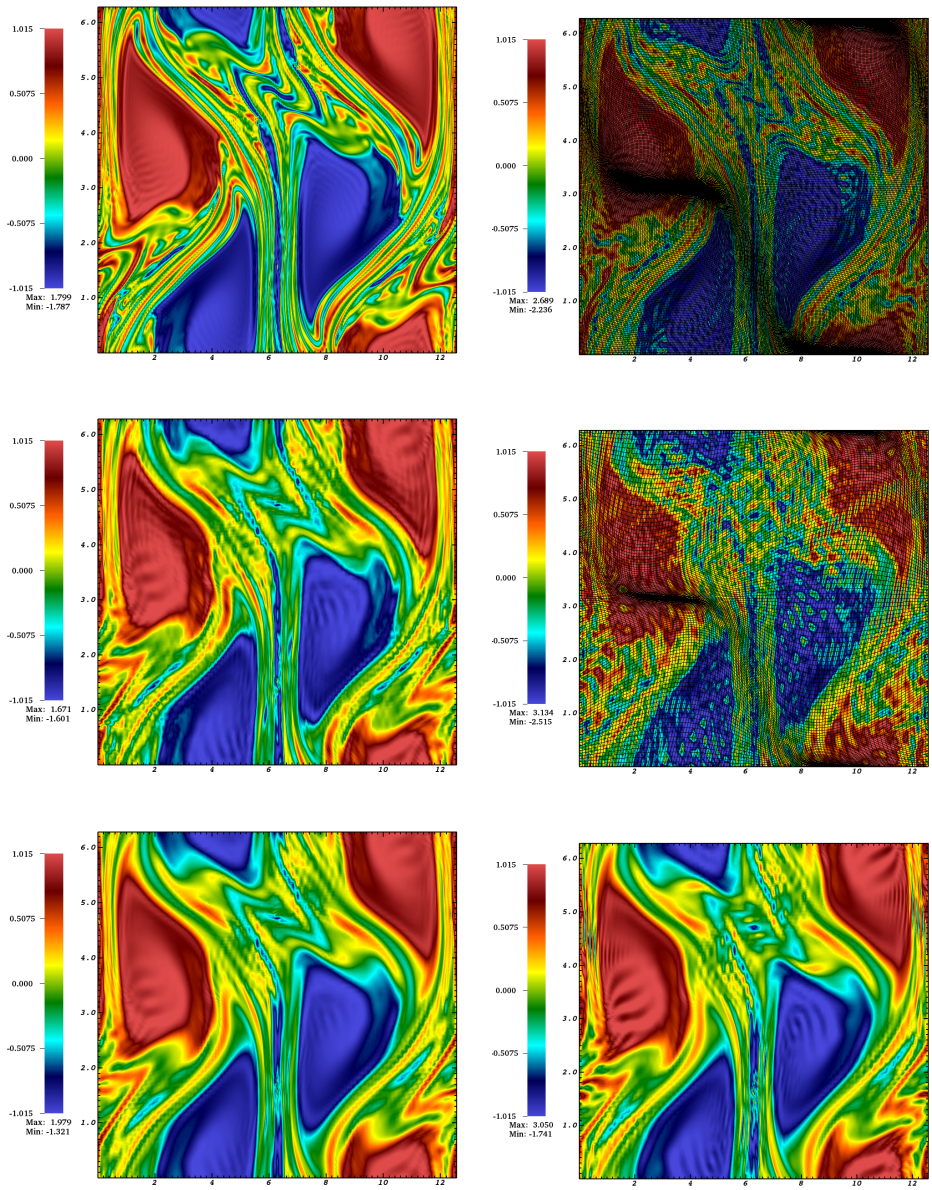

Figure 4: Guiding center case. Solution at time $T = 50$, on deformed mesh, with $\alpha = 0.9$ (left: H17, right: SPL3), $N = 256$, $\Delta t = 2^{-10}$ (top), $N = 128$, $\Delta t = 2^{-10}$ (middle) and $N = 128$, $\Delta t = 2^{-5}$ (bottom). Mesh for $N = 256$ (top right) and $N = 128$ (middle right) are also shown.

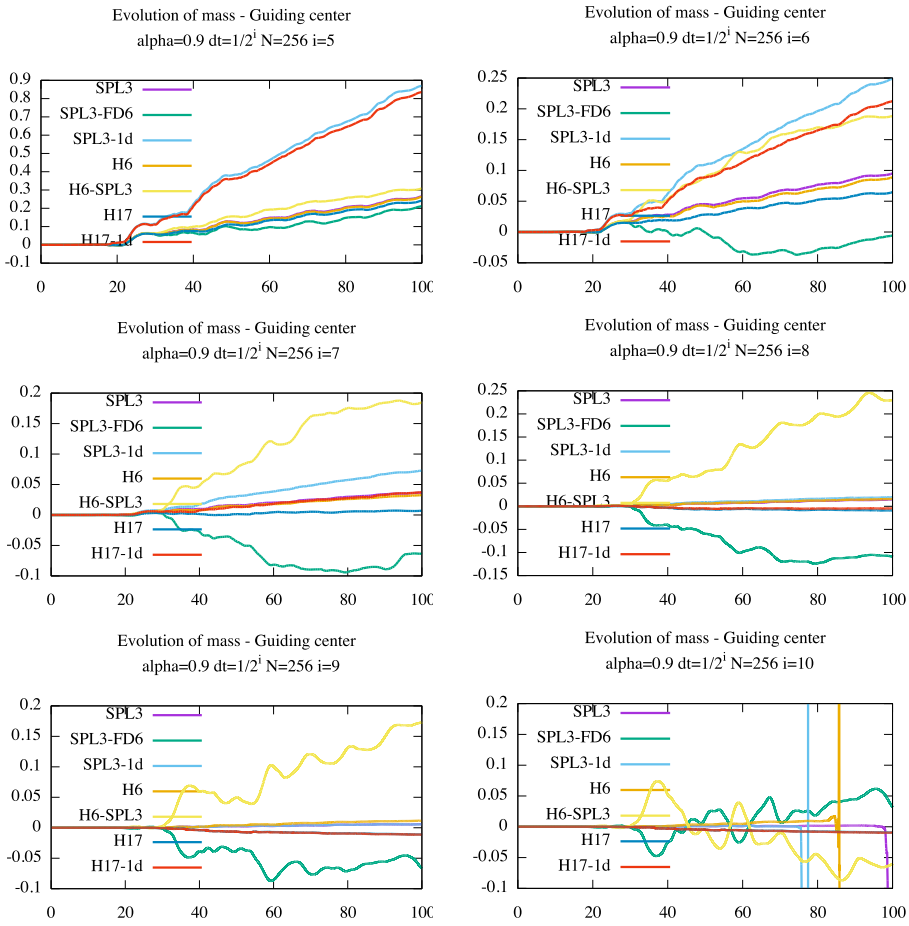

Figure 5: Guiding center case. Evolution of mass.

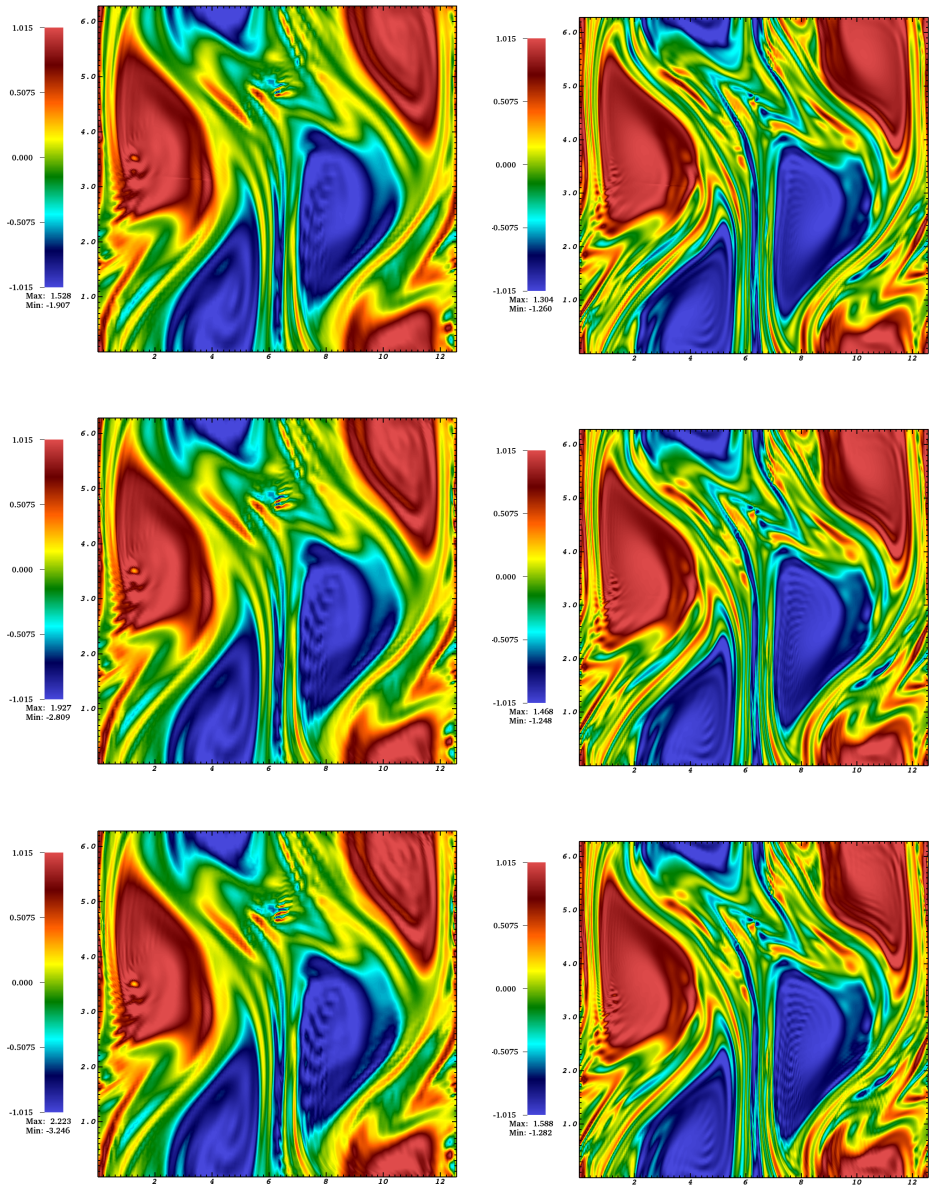

Figure 6: Guiding center case. Solution at time $T = 50$, on deformed mesh for CSL, with $\alpha = 0.9$ (left: $N = 128$, right: $N = 256$), $\Delta t = 2^{-5}$ (top), $\Delta t = 2^{-6}$ (middle) and $\Delta t = 2^{-7}$ (bottom).

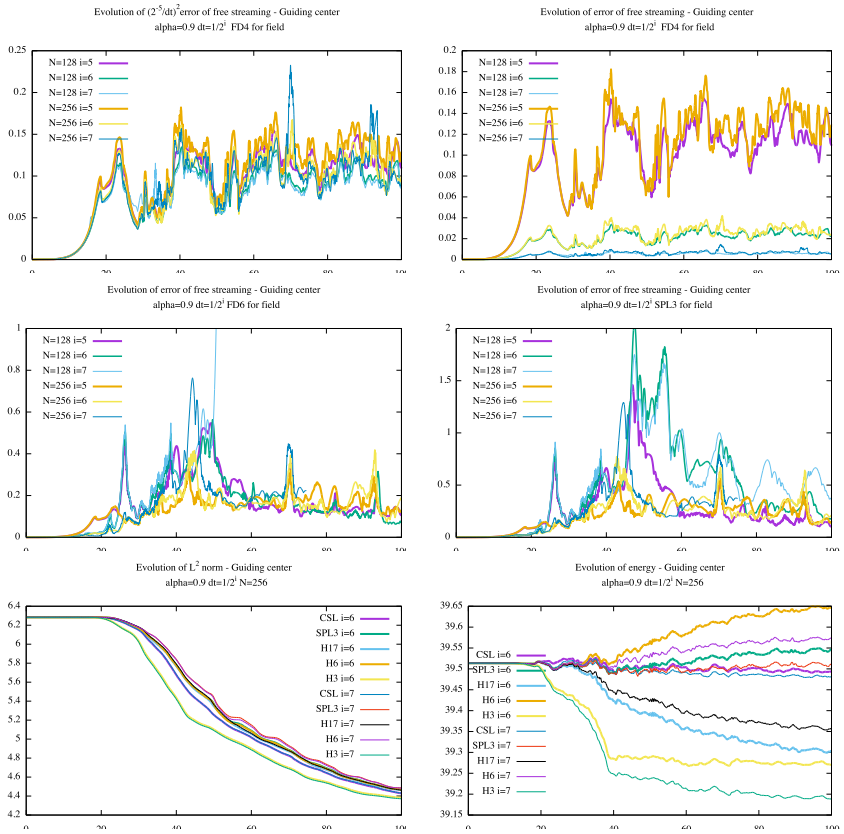

Figure 7: Guiding center case. Evolution of free stream error: FD4 for field (top), FD6 (middle left) SPL3 (middle right). Evolution of L^2 norm (bottom left) and energy (bottom right).

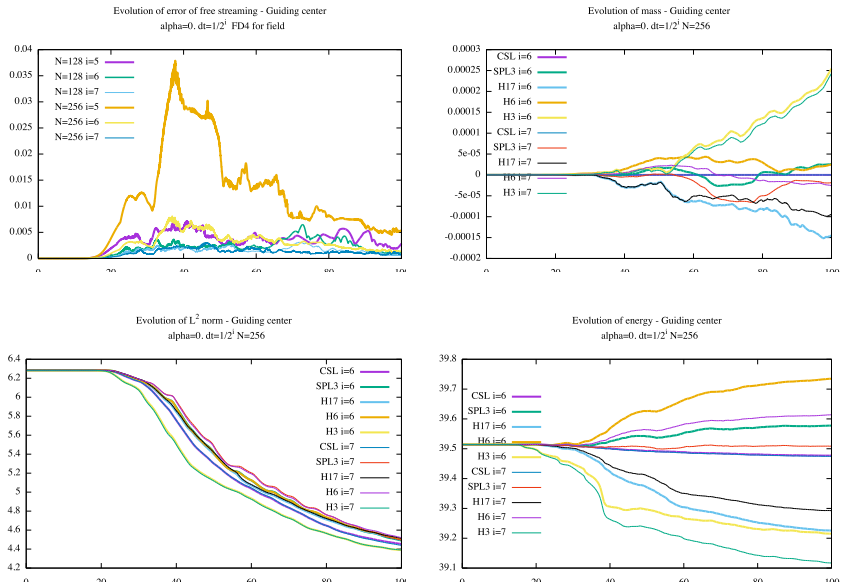

Figure 8: Guiding center case in the cartesian case ($\alpha = 0$). Evolution of free stream error with FD4 for field (top left). Evolution of mass (top right), L^2 norm (bottom left) and energy (bottom right).

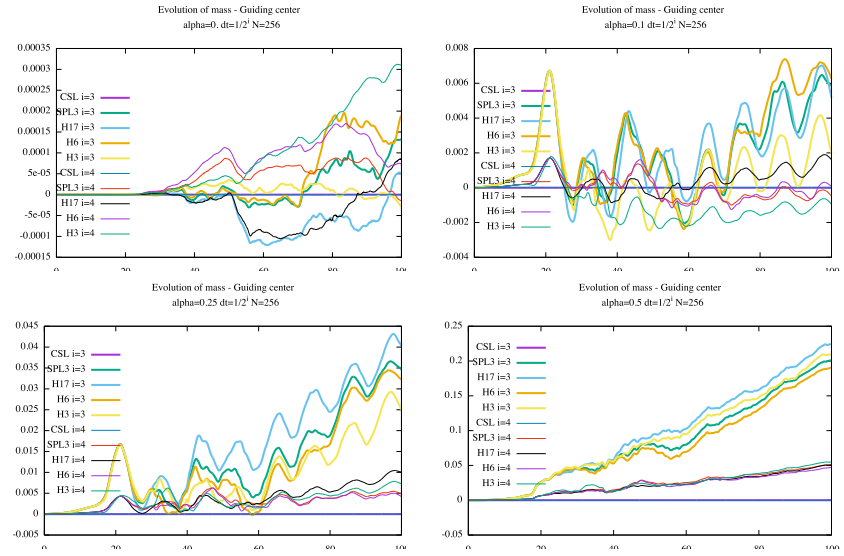

Figure 9: Guiding center case. Evolution of mass for $\alpha = 0,\ 0.1,\ 0.25,\ 0.5$; $N = 256$ (from left to right, top to bottom).

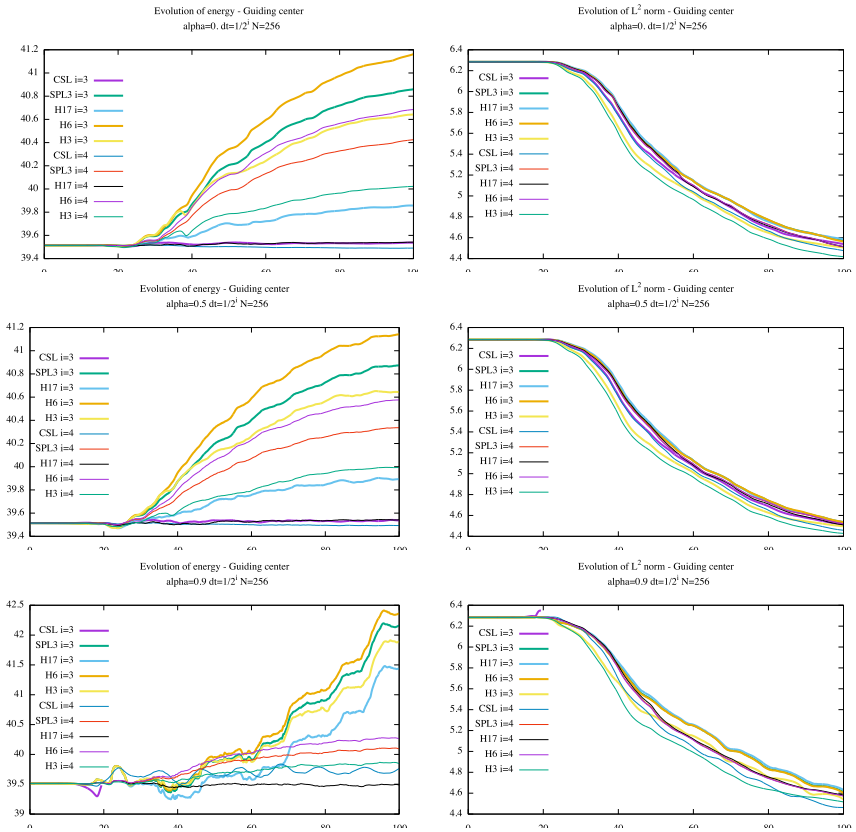

Figure 10: Guiding center case. Evolution of energy (left) and L^2 norm (right) for $\alpha = 0,\ 0.5,\ 0.9$ (from top to bottom).

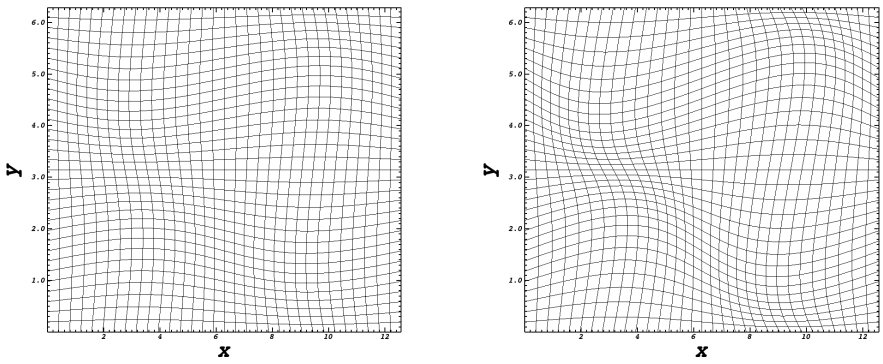

Figure 11: Guiding center case. Representation of mesh, on 32×32 grid, for $\alpha = 0.25$ and $\alpha = 0.5$.

References

[1] E. Sonnendrücker, J. Roche, P. Bertrand, and A. Ghizzo, The semi-lagrangian method for the numerical resolution of the vlasov equation, *Journal of computational physics*, vol. 149, no. 2, pp. 201–220, 1999.

[2] J.-M. Qiu and C.-W. Shu, Conservative high order semi-lagrangian finite difference weno methods for advection in incompressible flow, *Journal of Computational Physics*, vol. 230, no. 4, pp. 863–889, 2011.

[3] C. K. Birdsall and A. B. Langdon, *Plasma physics via computer simulation*. CRC Press, 2004.

[4] J. P. Verboncoeur, Particle simulation of plasmas: review and advances, *Plasma Physics and Controlled Fusion*, vol. 47, no. 5A, p. A231, 2005.

[5] C.-Z. Cheng and G. Knorr, The integration of the vlasov equation in configuration space, *Journal of Computational Physics*, vol. 22, no. 3, pp. 330–351, 1976.

[6] A. Staniforth and J. Côté, Semi-lagrangian integration schemes for atmospheric models-a review, *Monthly weather review*, vol. 119, no. 9, pp. 2206–2223, 1991.

[7] N. Besse, Convergence of a semi-lagrangian scheme for the one-dimensional vlasov–poisson system, *SIAM Journal on Numerical Analysis*, vol. 42, no. 1, pp. 350–382, 2004.

[8] N. Besse and M. Mehrenberger, Convergence of classes of high-order semi-lagrangian schemes for the vlasov–poisson system, *Mathematics of computation*, vol. 77, no. 261, pp. 93–123, 2008.

[9] F. Charles, B. Després, and M. Mehrenberger, Enhanced convergence estimates for semi-lagrangian schemes application to the vlasov–poisson equation, *SIAM Journal on Numerical Analysis*, vol. 51, no. 2, pp. 840–863, 2013.

[10] N. Besse and E. Sonnendrücker, Semi-lagrangian schemes for the vlasov equation on an unstructured mesh of phase space, *Journal of Computational Physics*, vol. 191, no. 2, pp. 341–376, 2003.

[11] T. Nakamura, R. Tanaka, T. Yabe, and K. Takizawa, Exactly conservative semi-lagrangian scheme for multi-dimensional hyperbolic equations with directional splitting technique, *Journal of Computational Physics*, vol. 174, no. 1, pp. 171–207, 2001.

[12] N. Crouseilles, M. Mehrenberger, and E. Sonnendrücker, Conservative semi-lagrangian schemes for vlasov equations, *Journal of Computational Physics*, vol. 229, no. 6, pp. 1927–1953, 2010.

[13] T. Umeda, Y. Nariyuki, and D. Kariya, A non-oscillatory and conservative semi-lagrangian scheme with fourth-degree polynomial interpolation for solving the vlasov equation, *Computer Physics Communications*, vol. 183, no. 5, pp. 1094–1100, 2012.

[14] N. Crouseilles, T. Respaud, and E. Sonnendrücker, A forward semi-lagrangian method for the numerical solution of the vlasov equation, *Computer Physics Communications*, vol. 180, no. 10, pp. 1730–1745, 2009.

[15] J.-M. Qiu and C.-W. Shu, Positivity preserving semi-lagrangian discontinuous galerkin formulation: theoretical analysis and application to the vlasov–poisson system, *Journal of Computational Physics*, vol. 230, no. 23, pp. 8386–8409, 2011.

[16] J. A. Rossmanith and D. C. Seal, A positivity-preserving high-order semi-lagrangian discontinuous galerkin scheme for the vlasov–poisson equations, *Journal of Computational Physics*, vol. 230, no. 16, pp. 6203–6232, 2011.

[17] M. Gutnic, M. Haefele, I. Paun, and E. Sonnendrücker, Vlasov simulations on an adaptive phase-space grid, *Computer Physics Communications*, vol. 164, no. 1, pp. 214–219, 2004.

[18] N. Besse, G. Latu, A. Ghizzo, E. Sonnendrücker, and P. Bertrand, A wavelet-mra-based adaptive semi-lagrangian method for the relativistic vlasov–maxwell system, *Journal of Computational Physics*, vol. 227, no. 16, pp. 7889–7916, 2008.

[19] M. C. Pinto and M. Mehrenberger, Convergence of an adaptive semi-lagrangian scheme for the vlasov-poisson system, *Numerische Mathematik*, vol. 108, no. 3, pp. 407–444, 2008.

[20] W. Guo and J.-M. Qiu, Hybrid semi-lagrangian finite element-finite difference methods for the vlasov equation, *Journal of Computational Physics*, vol. 234, pp. 108–132, 2013.

[21] J.-M. Qiu and C.-W. Shu, Conservative semi-lagrangian finite difference weno formulations with applications to the vlasov equation, *Communications in Computational Physics*, vol. 10, no. 4, p. 979, 2011.

[22] T. Arber and R. Vann, A critical comparison of eulerian-grid-based vlasov solvers, *Journal of computational physics*, vol. 180, no. 1, pp. 339–357, 2002.

[23] F. Filbet and E. Sonnendrücker, Comparison of eulerian vlasov solvers, *Computer Physics Communications*, vol. 150, no. 3, pp. 247–266, 2003.

[24] V. Grandgirard, M. Brunetti, P. Bertrand, N. Besse, X. Garbet, P. Ghendrih, G. Manfredi, Y. Sarazin, O. Sauter, E. Sonnendrücker, *et al.*, A drift-kinetic semi-lagrangian 4d code for ion turbulence simulation, *Journal of Computational Physics*, vol. 217, no. 2, pp. 395–423, 2006.

[25] V. Grandgirard, Y. Sarazin, X. Garbet, G. Dif-Pradalier, P. Ghendrih, N. Crouseilles, G. Latu, E. Sonnendrücker, N. Besse, and P. Bertrand, Computing itg turbulence with a full-f semi-lagrangian code, *Communications in Nonlinear Science and Numerical Simulation*, vol. 13, no. 1, pp. 81–87, 2008.

[26] G. Latu, N. Crouseilles, V. Grandgirard, and E. Sonnendrücker, Gyrokinetic semi-lagrangian parallel simulation using a hybrid openmp/mpi programming, in *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, pp. 356–364, Springer, 2007.

[27] P. Colella, M. R. Dorr, J. A. Hittinger, and D. F. Martin, High-order, finite-volume methods in mapped coordinates, *Journal of Computational Physics*, vol. 230, no. 8, pp. 2952–2976, 2011.

[28] P.-O. Persson, J. Bonet, and J. Peraire, Discontinuous galerkin solution of the navier–stokes equations on deformable domains, *Computer Methods in Applied Mechanics and Engineering*, vol. 198, no. 17, pp. 1585–1595, 2009.

[29] J.-P. Braeunig, N. Crouseilles, V. Grandgirard, G. Latu, M. Mehrenberger, and E. Sonnendrücker, Some numerical aspects of the conservative psm scheme in a 4d drift-kinetic code, *arXiv preprint arXiv:1303.2238*, 2013.

[30] F. Huot, A. Ghizzo, P. Bertrand, E. Sonnendrücker, and O. Coulaud, Instability of the time splitting scheme for the one-dimensional and relativistic vlasov–maxwell system, *Journal of Computational Physics*, vol. 185, no. 2, pp. 512–531, 2003.

[31] B. Afeyan, F. Casas, N. Crouseilles, D. A., E. Faou, M. Mehrenberger, and E. Sonnendrücker, Simulations of kinetic electrostatic electron nonlinear (keen) waves with two-grid, variable velocity resolution and high-order time-splitting, *The European Physical Journal D*, vol. 68:295, pp. 1–21, 2014.

[32] Y. S. Volkov, On complete interpolation spline finding via b-splines, *Sibirskie Èlektronnye Matematicheskie Izvestiya [Siberian Electronic Mathematical Reports]*, vol. 5, pp. 334–338, 2008.

[33] M. Mehrenberger, C. Steiner., L. Marradi, M. Mehrenberger, N. Crouseilles, E. Sonnendrücker, and B. Afeyan, Vlasov on gpu (vog project), *ESAIM: PROCEEDINGS*, vol. 43, pp. 37–58, 2013.

[34] C. Steiner., *Numerical computation of the gyroaverage operator and coupling with the Vlasov gyrokinetic equations*. PhD thesis, IRMA, University of Strasbourg, France, December 2014.

[35] C.-W. Shu, High-order finite difference and finite volume weno schemes and discontinuous galerkin methods for cfd, *International Journal of Computational Fluid Dynamics*, vol. 17, no. 2, pp. 107–118, 2003.

[36] A. Hamiaz, M. Mehrenberger, and A. Back, "Guiding center simulations on curvilinear grids." https://hal.archives-ouvertes.fr/hal-00908500, Dec. 2014.

[37] "Mudpack." https://www2.cisl.ucar.edu/resources/legacy/mudpack.