

The Semi-Lagrangian Method for the Numerical Resolution of the Vlasov Equation

Eric Sonnendrücker,* Jean Roche,* Pierre Bertrand,† and Alain Ghizzo‡

*IECN—Projet Numath, †LPMI, Université Henri Poincaré Nancy 1,
54506 Vandoeuvre-lès-Nancy Cedex, France

E-mail: Eric.Sonnendrucker@iecn.u-nancy.fr, Jean.Roche@iecn.u-nancy.fr,
Pierre.Bertrand@lpmi.u-nancy.fr, Alain.Ghizzo@lpmi.u-nancy.fr

Received May 18, 1998; revised October 23, 1998

The numerical resolution of kinetic equations and, in particular, of Vlasov-type equations is performed most of the time using particle in cell methods which consist in describing the time evolution of the equation through a finite number of particles which follow the characteristic curves of the equation, the interaction with the external and self-consistent fields being resolved using a grid. Another approach consists in computing directly the distribution function on a grid by following the characteristics backward in time for one time step and interpolating the value at the feet of the characteristics using the grid point values of the distribution function at the previous time step. In this report we introduce this last method, which couples the Lagrangian and Eulerian points of view and its use for the Vlasov equation and equations derived from it. © 1999 Academic Press

Key Words: computational plasma physics; Vlasov equations; semi-Lagrangian method; time splitting.

1. INTRODUCTION

The numerical resolution of kinetic equations, the solution of which depends, in addition to the time, on three space variables and three velocity variables, is performed most of the time using particle-in-cell (PIC) methods which enable us to get satisfying results with relatively few particles. However, for some applications, in particular when particles in the tail of the distribution play an important physical role or when the numerical noise due to the finite number of superparticles becomes too important, methods which compute the solution on a grid in phase space have proven to better describe the physics [1–3]. Such methods are all the more interesting when using parallel computers as they are very scalable [4]. On the other hand, although PIC methods can be parallelized fairly well, it is very hard to maintain good load balancing as particles move from domain to domain, whereas when

using a fixed grid it is straightforward to achieve a perfect load balancing throughout the run.

In order to compute the solution of the one-dimensional Vlasov equation

$$\frac{\partial f}{\partial t} + v \frac{\partial f}{\partial x} + E(x, t) \frac{\partial f}{\partial v} = 0, \quad (1)$$

coupled with Poisson's equation,

$$\frac{\partial E}{\partial x} = \int_{-\infty}^{+\infty} f(x, v, t) dv - 1 \quad (2)$$

for E , the following procedure, originally introduced by Cheng and Knorr [5] and making full use of time-splitting, can be used to go from time step t_n to t_{n+1} :

1. Perform a half time step shift along the x -axis $f^*(x, v) = f^n(x - v\Delta t/2, v)$
2. Compute the electric field at time $t_{n+1/2}$ by substituting f^* in the Poisson equation (2)
3. Perform a shift along the v -axis $f^{**}(x, v) = f^*(x, v - E(x, t_{n+1/2})\Delta t)$
4. Perform a second half time step shift along the x -axis $f^{n+1}(x, v) = f^{**}(x - v\Delta t/2, v)$.

This method, called Eulerian because it used a phase space grid instead of particles, proved to work very well in this case. Moreover, the shifts along x or v are nothing but Lagrangian advections since the method is equivalent to solving the characteristics of the Vlasov equation. The advantage is that the scheme can use larger time steps than explicit Eulerian ones, the price to pay is to reconstruct a regular grid using interpolation. However, it could not be applied as easily to other kinds of Vlasov problems like, for instance, the guiding-centre approximation,

$$\frac{\partial f}{\partial t} + \frac{\mathbf{E}(\mathbf{x}, t) \times \mathbf{B}}{B^2} \cdot \nabla_x f = 0, \quad (3)$$

coupled with Poisson's equation, where the advection term $\mathbf{E}(\mathbf{x}, t) \times \mathbf{B}/B^2$ depends on \mathbf{x} and the time-splitting method cannot be applied, or to a reduced relativistic Vlasov equation like

$$\frac{\partial f}{\partial t} + \frac{p}{\gamma} \frac{\partial f}{\partial x} + \left(E_x - \frac{1}{2\gamma} \frac{\partial}{\partial x} (A_{\perp}^2(x, t)) \right) \frac{\partial f}{\partial p} = 0, \quad (4)$$

where

$$\gamma = \sqrt{1 + p^2 + A_{\perp}^2(x, t)},$$

coupled with Maxwell's equations for the vector potential A_{\perp} , where a similar problem occurs. These problems pushed us to try to extend the scope of application of the method. This could be done, using the so-called semi-Lagrangian method or backward characteristics method which had already been investigated by the fluid dynamics community and especially in weather and climate simulation [6]. A mathematical analysis of the method has also been performed by Bermejo [7] and more recently by Falcone and Ferreti [8], and cost-effectiveness was investigated by Bartello and Thomas [10].

Let us mention that FCT which is purely eulerian has also been used for solving the Vlasov equation [11] and for stiff problems a semi-Lagrangian method using shape-preserving splines for the interpolation was introduced in [12].

This paper is organised as follows. In the next section we are going to present a number of Vlasov equations that we would like to be able to solve with our method, this will lead to an abstract formulation which will enclose all those equations. Then we shall introduce the semi-Lagrangian method for our abstract formulation and discuss some simplifications in special cases. After that we shall describe the implementation of the method in the two-dimensional case and finally numerical results will be presented for the guiding center model which is the easiest problem where both difficulties not addressed by the previously used methods are present, namely that the advection field depends on the advected variable and time splitting cannot be performed.

2. DIFFERENT TYPES OF VLASOV EQUATIONS

The ultimate model we would like to use in order to describe the behaviour of a plasma is the relativistic three-dimensional Vlasov model which reads, for instance for an electron gas with $e = m = 1$,

$$\frac{\partial f}{\partial t} + \frac{\mathbf{p}}{\gamma} \cdot \nabla_x f + \left(\mathbf{E} + \frac{\mathbf{p}}{\gamma} \times \mathbf{B} \right) \cdot \nabla_p f = 0, \quad (5)$$

where γ is the Lorentz factor $(1 + p_x^2 + p_y^2 + p_z^2)^{1/2}$, or for its nonrelativistic counterpart,

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_x f + (\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \nabla_v f = 0, \quad (6)$$

coupled to Maxwell's equations or some approximation of those. However, this would require a six-dimensional grid and, as a minimum of points are required in each direction to represent the physics correctly, this would be too large even for the largest multiprocessor computer available today. Therefore, we shall investigate reduced models, which are adequate for describing specific physical problems.

The simplest model we are interested in, beyond the 1D electrostatic model, is the 2D electrostatic one which reads

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla_x f + \mathbf{E}(\mathbf{x}, t) \cdot \nabla_v f = 0, \quad (7)$$

coupled with Poisson's equation

$$-\Delta\phi = 1 - \int f d\mathbf{v},$$

where $\mathbf{E} = -\nabla\phi$.

The next category of models is motivated by the important problems of the nonlinear interaction of high intensity ultrashort laser pulses with plasmas, with specific application to particle acceleration or inertial confinement fusion purpose. It consists of $1\frac{1}{2}D$, $1\frac{2}{2}D$, and $2\frac{1}{2}D$ electromagnetic models,

$$\frac{\partial f}{\partial t} + \frac{\mathbf{p}}{\gamma} \cdot \nabla_x f + \left(\mathbf{E} + \frac{\mathbf{p}}{\gamma} \times \mathbf{B} \right) \cdot \nabla_p f = 0, \quad (8)$$

where, compared to the full relativistic model (5) f , \mathbf{E} and \mathbf{B} depend only on one or two space variables and f depends on two or three momentum components. Moreover, some

components of the fields may vanish according to the wave polarisation, and the remaining are computed using some reduced set of Maxwell's equations.

For the case of a laser wave propagating in one direction (say x), f depends only on x , p_x , and \mathbf{p}_\perp . So one can further reduce the relativistic Vlasov (5) equation by looking for a specific class of exact solutions of the form, say in $1\frac{1}{2}D$:

$$f(x, p_x, p_y, p_z, t) = F(x, p_x, t)\delta(p_y - P_y(x, t))\delta(p_z - P_z(x, t)).$$

Then F is solution of

$$\frac{\partial F}{\partial t} + \frac{p_x}{\gamma} \frac{\partial F}{\partial x} + \left(E_x + \frac{P_y B_z - P_z B_y}{\gamma} \right) \frac{\partial F}{\partial p_x} = 0,$$

with $\gamma = (1 + p_x^2 + P_y(x, t)^2 + P_z(x, t)^2)^{1/2}$. This results from the conservation of the generalised canonical momentum along the wave plane, i.e. $P_y - A_y = \text{const}$, and $P_z - A_z = \text{const}$. Deriving with respect to time and using

$$\mathbf{E}_\perp = -\frac{\partial \mathbf{A}_\perp}{\partial t},$$

we get

$$\frac{\partial P_y}{\partial t} = E_y, \quad \frac{\partial P_z}{\partial t} = E_z.$$

This mono-kinetic perpendicular description in the longitudinal axis is consistent with the very low temperature of the targets (a few electron volts) while the longitudinal electron temperature can reach the mega electron volt range during the interaction. Finally, using $\mathbf{B}_\perp = \text{curl } \mathbf{A}_\perp$, we find that

$$P_y B_z - P_z B_y = \frac{1}{2} \frac{\partial A_\perp^2}{\partial x},$$

and we recover Eq. (4).

Instead of reducing the dimension or assuming a specific form for the solution one can also obtain another type of Vlasov equation by averaging over the magnetic orbits in a highly magnetised plasma like a tokamak plasma; this yields the drift-kinetic model which reads, in the case of a uniform external magnetic field,

$$\frac{\partial f}{\partial t} + \left(\mathbf{v}_\parallel + \frac{\mathbf{E} \times \mathbf{B}}{B^2} \right) \cdot \nabla_x f + \mathbf{E}_\parallel \cdot \nabla_v f = 0.$$

If the computational box is reduced to the plane perpendicular to \mathbf{B} , the drift-kinetic equation reduces to the guiding-centre equation (3).

Finally, some simplification of the Vlasov equation may be obtained in some cases by using some other coordinate system. One example of this case is the $1\frac{1}{2}D$ axisymmetric model, where $f = f(r, v_r, v_\theta, t)$, for which we have

$$\frac{\partial f}{\partial t} + v_r \frac{\partial f}{\partial r} + \left(E_r + \frac{v_\theta^2}{r} \right) \frac{\partial f}{\partial v_r} - \frac{v_r v_\theta}{r} \frac{\partial f}{\partial v_\theta} = 0.$$

All the types of Vlasov equations we have listed above can be written

$$\frac{\partial f}{\partial t} + U(X, t) \cdot \nabla_X f = 0, \quad (9)$$

where X stands for the phase space coordinates and U is a divergence free vector field having up to six components in the full three-dimensional case. For example, in the case of the 3D nonrelativistic Vlasov equation, $X = (x, y, z, v_x, v_y, v_z)$ and $U(X, t) = (v_x, v_y, v_z, E_x + v_y B_z - v_z B_y, E_y + v_z B_x - v_x B_z, E_z + v_x B_y - v_y B_x)$ with all components of the electric and magnetic fields depending on x, y, z , and t .

3. THE TIME-SPLITTING PROBLEM

For an advection field U which is divergence-free, as is the case for our Vlasov equations, Eq. (9) can also be written in conservative form,

$$\frac{\partial f}{\partial t} + \text{div}_X(U(X, t)f) = 0. \quad (10)$$

For numerical purposes, conservation laws like (10) can be further reduced by splitting. Indeed, splitting the components of X into two sets X_1 and X_2 (10) can be written

$$\frac{\partial f}{\partial t} + \text{div}_{X_1}(U_1(X_1, X_2, t)f) + \text{div}_{X_2}(U_2(X_1, X_2, t)f) = 0. \quad (11)$$

Moreover, it has been verified (see, for example, [13]) that one can use a second order in time numerical scheme for solving separately at each time step,

$$\frac{\partial f}{\partial t} + \text{div}_{X_1}(U_1(X_1, X_2, t)f) = 0$$

and

$$\frac{\partial f}{\partial t} + \text{div}_{X_2}(U_2(X_1, X_2, t)f) = 0,$$

and keep the second-order accuracy for the whole equation (11) by alternating the solves.

For our purposes, this will be useful in the case where both $\text{div}_{X_1}U_1(X_1, X_2, t) = 0$ and $\text{div}_{X_2}U_2(X_1, X_2, t) = 0$, as in this case U_1 and U_2 can be taken out of the div operator and yield the traditional advective form.

Applying this, first for the nonrelativistic Vlasov equation (6), we notice that $\partial_i v_i = 0$ and $\partial_{v_i}(\mathbf{E} + \mathbf{v} \times \mathbf{B})_i = 0$, with i standing for x, y , or z . Hence, it is possible to split (6) into six 1D advections. This method has already been applied in the 1D case, as we mentioned in the Introduction and also in the 2D case [14, 15].

In the case of the relativistic Vlasov equation (5), we still have obviously $\partial_i p_i / \gamma = 0$, but no such property for the force term which now depends on the Lorentz factor, $\gamma = \sqrt{1 + p_x^2 + p_y^2 + p_z^2}$.

However, a straightforward computation shows that we still have $\text{div}_p(\mathbf{E} + (\mathbf{p}/\gamma) \times \mathbf{B}) = 0$. Hence we could split the equation into one 3D advection for the momentum space and three 1D advection for the physical space. Whether the splitting of the physical space

could be a problem needs some further investigation, as it results in a loss of isotropy having three special directions. Finally, in the other models we introduced, splitting does not seem justified a priori and a full multidimensional advection scheme for Eq. (9) is needed.

4. THE SEMI-LAGRANGIAN METHOD

Let us now introduce the semi-Lagrangian method on the abstract model (9). For this, we need to introduce the characteristics of (9), which are the solutions of the dynamical system

$$\frac{d\mathbf{X}}{dt} = \mathbf{U}(\mathbf{X}(t), t). \quad (12)$$

Let us denote by $\mathbf{X}(t; \mathbf{x}, s)$ the solution at time t whose value is \mathbf{x} at time s . Taking $\mathbf{X}(t)$ a solution of (12), we have

$$\begin{aligned} \frac{d}{dt}(f(\mathbf{X}(t), t)) &= \frac{\partial f}{\partial t} + \frac{d\mathbf{X}}{dt} \cdot \nabla_{\mathbf{X}} f \\ &= \frac{\partial f}{\partial t} + \mathbf{U}(\mathbf{X}(t), t) \cdot \nabla_{\mathbf{X}} f = 0, \end{aligned}$$

which means that f is constant along the characteristics. This can also be written

$$f(\mathbf{X}(t; \mathbf{x}, s), t) = f(\mathbf{X}(s; \mathbf{x}, s), s) = f(\mathbf{x}, s)$$

for any times t and s and phase space coordinate \mathbf{x} . It is this property which will be used in the semi-Lagrangian method to solve a discrete problem, which is defined by introducing a finite set of mesh points $(\mathbf{x}_m)_{m=1, \dots, N}$ which may or may not be equally spaced. Then, given the value of the function f at the mesh points at any given time step, we obtain the new value at mesh point \mathbf{x}_m using that

$$f(\mathbf{x}_m, t_n + \Delta t) = f(\mathbf{X}(t_n - \Delta t; \mathbf{x}_m, t_n + \Delta t), t_n - \Delta t).$$

For each mesh point \mathbf{x}_m , f is computed in two steps:

1. Find the starting point of the characteristic ending at \mathbf{x}_m , i.e. $\mathbf{X}(t_n - \Delta t; \mathbf{x}_m, t_n + \Delta t)$.
2. Compute $f(\mathbf{X}(t_n - \Delta t; \mathbf{x}_m, t_n + \Delta t), t_n - \Delta t)$ by interpolation, as f is known only at mesh points at time $t_n - \Delta t$.

In the case of the 1D Vlasov–Poisson system (1)–(2), the numerical scheme described in the introduction actually involves these two steps. But, since the advection field for each half time step does not depend on the variable to be advected, step 1 is straightforward.

Now, for the general case, in order to deal with step 1, we need to introduce a time discretisation of (12). As in general no information on the advection function U is known at any given time, we need to use a two time step scheme in order to remain second order in time. The starting point of the characteristic is obtained, to second-order accuracy, by

$$\frac{\mathbf{x}_m - \mathbf{X}(t_n - \Delta t)}{2\Delta t} = \mathbf{U}(\mathbf{X}(t_n), t_n), \quad (13)$$

writing, still to second-order accuracy,

$$\mathbf{X}(t_n) = \frac{\mathbf{X}(t_n + \Delta t) + \mathbf{X}(t_n - \Delta t)}{2};$$

there exists \mathbf{d}_m such that $\mathbf{X}(t_n) = \mathbf{x}_m - \mathbf{d}_m$ and $\mathbf{X}(t_n - \Delta t) = \mathbf{x}_m - 2\mathbf{d}_m$. Then (13) becomes

$$\mathbf{d}_m = \Delta t \mathbf{U}(\mathbf{x}_m - \mathbf{d}_m, t_n) \quad (14)$$

which can be solved iteratively for the unknown \mathbf{d}_m .

Once \mathbf{d}_m is known $f(\mathbf{x}_m - 2\mathbf{d}_m, t_n - \Delta t)$ is interpolated using a tensor product of cubic B-splines.

Remark 4.1. Let us recall a few properties of the semi-Lagrangian methods that have been derived in previous investigations (see [6]):

- If U is known independently of f a one time step method can be used by introducing an intermediate time step.
- The time step is not restricted by the usual Courant condition, but by the deformational Courant number $\|(\partial U / \partial x) \Delta t\| < 1$ which is often less restrictive. In practice, accuracy conditions impose the time step which is usually a few Courant time steps.
- Cubic interpolation appears to be a good compromise between accuracy and cost. Linear interpolation is too dissipative to be used.
- When using cubic interpolation for f , linear interpolation is sufficient for U .
- Bermejo [7] interpreted this method as a finite-element PIC method and proved its convergence.

5. IMPLEMENTATION OF THE GUIDING-CENTRE MODEL

Let us now give a few more details on the implementation of the semi-Lagrangian method in the two-dimensional case of the guiding-centre model, a case where all the additional difficulties we are investigating are concentrated. Time-splitting cannot be performed so we need a full two-dimensional scheme, the advection field, which is, in this case, the drift velocity depends on \mathbf{x} . Moreover, passing to a higher dimension does not add any new difficulty. So the implementation of this specific case is a good benchmark for our method.

The model we consider is the guiding-centre Vlasov equation,

$$\frac{\partial \rho}{\partial t} + \mathbf{v}_D \cdot \nabla_x \rho = 0, \quad (15)$$

where

$$\mathbf{v}_D = \frac{\mathbf{E} \times \mathbf{B}}{B^2},$$

coupled to the Poisson equation

$$-\Delta \phi = \rho \quad (16)$$

with $\mathbf{E} = -\nabla \phi$ and \mathbf{B} is a given external magnetic field.

In this case the function \mathbf{U} of the previous section is the drift velocity $\mathbf{v}_D = \mathbf{E} \times \mathbf{B} / B^2$ whose coordinates we denote by v_{Dx} and v_{Dy} .

5.1. Computation of the Origin of the Characteristics

Applying the two-time step scheme as described in the previous section, the problem comes down to solving the fixed point equation (14). Denoting by $(\alpha_{ij}, \beta_{ij})$ the coordinates of the displacement \mathbf{d}_m at the mesh point \mathbf{x}_m whose coordinates are denoted by (x_i, y_j) , Eq. (14) becomes

$$\alpha_{ij} = \Delta t v_{Dx}(x_i - \alpha_{ij}, y_j - \beta_{ij}, t_n) \quad (17)$$

$$\beta_{ij} = \Delta t v_{Dy}(x_i - \alpha_{ij}, y_j - \beta_{ij}, t_n). \quad (18)$$

In order to solve this nonlinear system (17)–(18), as α_{ij} and β_{ij} are small, the first obvious approach is to take

$$v_{Dx}(x_i - \alpha_{ij}, y_j - \beta_{ij}, t_n) = v_{Dx}(x_i, y_j, t_n)$$

and

$$v_{Dy}(x_i - \alpha_{ij}, y_j - \beta_{ij}, t_n) = v_{Dy}(x_i, y_j, t_n).$$

However, this method is only first-order accurate in Δt and it will not be convenient, or at least, as we shall see later in the numerical test, it would require much smaller time steps. In order to remain second-order accurate, the system can be solved iteratively by

$$\alpha_{ij}^{k+1} = \Delta t v_{Dx}(x_i - \alpha_{ij}^k, y_j - \beta_{ij}^k, t_n),$$

$$\beta_{ij}^{k+1} = \Delta t v_{Dy}(x_i - \alpha_{ij}^k, y_j - \beta_{ij}^k, t_n),$$

starting from an initial guess $(\alpha_{ij}^0, \beta_{ij}^0)$.

This scheme can be proved to be convergent for Δt small enough. Notice that \mathbf{v}_D , known only on the mesh, needs to be interpolated at the point $(x_i - \alpha_{ij}^k, y_j - \beta_{ij}^k)$. Numerical experiments performed for climate equations [6] have proved that linear interpolation is sufficient, which was confirmed in our problems.

Using the ansatz that \mathbf{v}_D is linear in each grid cell, we also tried the more sophisticated Newton method in order to compute the solution of (17)–(18). The algorithm reads

$$\begin{pmatrix} \alpha_{ij}^{k+1} \\ \beta_{ij}^{k+1} \end{pmatrix} = \begin{pmatrix} \alpha_{ij}^k \\ \beta_{ij}^k \end{pmatrix} - (Df(\alpha_{ij}^k, \beta_{ij}^k))^{-1} f(\alpha_{ij}^k, \beta_{ij}^k),$$

where

$$f(\alpha_{ij}^k, \beta_{ij}^k) = \begin{pmatrix} \alpha_{ij}^k - \Delta t v_{Dx}(x_i - \alpha_{ij}^k, y_j - \beta_{ij}^k) \\ \beta_{ij}^k - \Delta t v_{Dy}(x_i - \alpha_{ij}^k, y_j - \beta_{ij}^k) \end{pmatrix},$$

and the Jacobian matrix of f is given by

$$Df(\alpha_{ij}^k, \beta_{ij}^k) = \begin{pmatrix} \frac{\partial f_x}{\partial \alpha_{ij}^k} & \frac{\partial f_x}{\partial \beta_{ij}^k} \\ \frac{\partial f_y}{\partial \alpha_{ij}^k} & \frac{\partial f_y}{\partial \beta_{ij}^k} \end{pmatrix} = \begin{pmatrix} 1 + \Delta t \partial_x v_{Dx} & \Delta t \partial_y v_{Dx} \\ \Delta t \partial_x v_{Dy} & 1 + \Delta t \partial_y v_{Dy} \end{pmatrix},$$

where we omitted indicating explicitly the dependence of \mathbf{v}_D on α_{ij}^k and β_{ij}^k for the sake of clarity. Then $(Df(\alpha_{ij}^k, \beta_{ij}^k))^{-1} f(\alpha_{ij}^k, \beta_{ij}^k)$ can be computed using Cramer's formulae for a 2×2 system, which yields the algorithm

$$\begin{aligned}\alpha_{ij}^{k+1} &= \alpha_{ij}^k - \frac{1}{\Delta} [(\alpha_{ij}^k - \Delta t v_{Dx}(x - \alpha_{ij}^k, y - \beta_{ij}^k))(1 + \Delta t \partial_y v_{Dy}(x - \alpha_{ij}^k, y - \beta_{ij}^k)) \\ &\quad - (\beta_{ij}^k - \Delta t v_{Dy}(x - \alpha_{ij}^k, y - \beta_{ij}^k)) \Delta t \partial_y v_{Dx}(x - \alpha_{ij}^k, y - \beta_{ij}^k)] \\ \beta_{ij}^{k+1} &= \beta_{ij}^k - \frac{1}{\Delta} [(\beta_{ij}^k - \Delta t v_{Dy}(x - \alpha_{ij}^k, y - \beta_{ij}^k))(1 + \Delta t \partial_x v_{Dx}(x - \alpha_{ij}^k, y - \beta_{ij}^k)) \\ &\quad - (\alpha_{ij}^k - \Delta t v_{Dx}(x - \alpha_{ij}^k, y - \beta_{ij}^k)) \Delta t \partial_x v_{Dy}(x - \alpha_{ij}^k, y - \beta_{ij}^k)],\end{aligned}$$

where

$$\begin{aligned}\Delta &= (1 + \Delta t \partial_x v_{Dx}(x - \alpha_{ij}^k, y - \beta_{ij}^k))(1 + \Delta t \partial_y v_{Dy}(x - \alpha_{ij}^k, y - \beta_{ij}^k)) \\ &\quad - \Delta t^2 \partial_x v_{Dy}(x - \alpha_{ij}^k, y - \beta_{ij}^k) \partial_y v_{Dx}(x - \alpha_{ij}^k, y - \beta_{ij}^k).\end{aligned}$$

The values of \mathbf{v}_D and its derivatives which are needed in this algorithm are computed using its mesh point values and the ansatz that it is linear in each cell.

5.2. The Two-Dimensional Spline Interpolation

Once α_{ij} and β_{ij} are known, $\rho(x_i - 2\alpha_{ij}, x_j - 2\beta_{ij})$ is interpolated using a tensor product of cubic B-splines. In order to do this we first need to compute the coefficients $\eta_{v\kappa}$ of the cubic spline interpolation function $s(x, y)$ given by

$$s(x, y) = \sum_{-2 \leq v \leq N_x - 1} \left(\sum_{-2 \leq \kappa \leq N_y - 1} \eta_{v\kappa} B_{3v}(x) B_{3\kappa}(y) \right).$$

The spline s must satisfy the interpolation conditions

$$s(x_i, y_j) = \rho(x_i, y_j, t_n - \Delta t)$$

for $i = 1, \dots, N_x$; $j = 1, \dots, N_y$, and the two boundary conditions of the function ρ in each direction, which in our case are periodic in the x -direction and natural in the y -direction.

In order to compute the $\eta_{v\kappa}$ coefficients, we first solve the N_y one-dimensional interpolation problems,

$$s(x, y_j) = \sum_{-2 \leq v \leq N_x - 1} \gamma_v^j B_{3v}(x) \quad \text{for } j = 1, \dots, N_y,$$

each verifying the N_x interpolation conditions $s(x_i, y_j) = \rho(x_i, y_j, t_n - \Delta t)$ and the periodic boundary conditions which we denote by

$$\gamma_v(y) = \sum_{-2 \leq \kappa \leq N_y - 1} \eta_{v\kappa} B_{3\kappa}(y)$$

and by

$$\gamma_v^j = \gamma_v(y_j).$$

Using these interpolation and boundary conditions, we need to solve N_y linear systems, one for each value of j , involving the same $(N_x + 2)$ -dimensional matrix which reads, denoting $h = \Delta x$,

$$\frac{1}{6} \begin{pmatrix} -\frac{3}{h} & -\frac{3}{h} & 0 & \frac{3}{h} & \dots & \dots & \frac{3}{h} & 0 \\ -\frac{6}{h^2} & \frac{6}{h^2} & -\frac{12}{h^2} & \frac{6}{h^2} & \dots & \dots & -\frac{6}{h^2} & \frac{12}{h^2} \\ 0 & 1 & 4 & 1 & & & & 0 \\ \vdots & 0 & 1 & 4 & \ddots & & & \vdots \\ \vdots & \vdots & & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \vdots & & & \ddots & \ddots & \ddots & 0 \\ 0 & \vdots & & & & 1 & 4 & 1 \\ 1 & 0 & \dots & \dots & \dots & 0 & 1 & 4 \end{pmatrix}. \tag{19}$$

Note that we have written this matrix such that the first two rows correspond to the boundary conditions, the remaining N_x to the interpolation conditions, and the $(N_x + 2)$ -dimensional unknown vector reads $(\gamma_{N_x-1}^j, \gamma_{-2}^j, \dots, \gamma_{N_x-2}^j)$.

Then we obtain $\eta_{v\kappa}$ by solving the $N_x + 2$ interpolation problems,

$$\gamma_v(y) = \sum_{-2 \leq \kappa \leq N_x-1} \eta_{v\kappa} B_{3\kappa}(y) \quad \text{for } v = -2, \dots, N_x - 1,$$

verifying the N_y interpolation conditions $\gamma_v(y_j) = \gamma_v^j$ and natural boundary conditions. Using these interpolation and boundary conditions, we need to solve $N_x + 2$ linear systems, one for each value of v , involving the same $(N_y + 2)$ -dimensional matrix which reads, denoting $k = \Delta y$,

$$\frac{1}{6} \begin{pmatrix} -\frac{6}{k^2} & 0 & 0 & \dots & \dots & \dots & \frac{6}{k^2} & -\frac{12}{k^2} \\ 0 & \frac{6}{k^2} & -\frac{12}{k^2} & \frac{6}{k^2} & \dots & \dots & 0 & 0 \\ 0 & 1 & 4 & 1 & & & & 0 \\ \vdots & 0 & 1 & 4 & \ddots & & & \vdots \\ \vdots & \vdots & & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \vdots & & & \ddots & \ddots & \ddots & 0 \\ 0 & \vdots & & & & 1 & 4 & 1 \\ 1 & 0 & \dots & \dots & \dots & 0 & 1 & 4 \end{pmatrix}. \tag{20}$$

This matrix is written for the $(N_y + 2)$ -dimensional unknown vector $(\eta_{v, N_y-1}, \eta_{v, -2}, \dots, \eta_{v, N_y-2})$ and the right-hand-side vector $\gamma_v^{N_y}, \gamma_v^1, \dots, \gamma_v^{N_y-1}$.

In both matrices (19) and (20), the terms not explicitly written are all zeros, except for fours on the diagonal and ones on the upper and lower diagonals.

Once the B-spline coefficients $\eta_{\nu\kappa}$ for all ν and κ have been computed, the value of ρ at the origin of the characteristics is taken to be the value of the B-spline $s(x_i - 2\alpha_{ij}, y_j - 2\beta_{ij})$. If $(x_i - 2\alpha_{ij}, y_j - 2\beta_{ij})$ belongs to $[x_i, x_{i+1}] \times [y_j, y_{j+1}]$ the approximation of the function $\rho(x_i - 2\alpha_{ij}, y_j - 2\beta_{ij}, t_n - \Delta t)$ is given by

$$s(x_i - 2\alpha_{ij}, y_j - 2\beta_{ij}) = \sum_{i-3 \leq \nu \leq i} \left(\sum_{j-3 \leq \kappa \leq j} \eta_{\nu\kappa} B_{3\nu}(x_i - 2\alpha_{ij}) B_{3\kappa}(y_j - 2\beta_{ij}) \right),$$

where

$$B_{3\nu}(x) = \frac{1}{6h^3} \begin{cases} (x - x_\nu)^3, & x_\nu \leq x < x_{\nu+1}, \\ h^3 + 3h^2(x - x_{\nu+1}) + 3h(x - x_{\nu+1})^2 - 3(x - x_{\nu+1})^3, & x_{\nu+1} \leq x < x_{\nu+2}, \\ h^3 + 3h^2(x_{\nu+3} - x) + 3h(x_{\nu+3} - x)^2 - 3(x_{\nu+3} - x)^3, & x_{\nu+2} \leq x < x_{\nu+3}, \\ (x_{\nu+4} - x)^3, & x_{\nu+3} \leq x < x_{\nu+4}, \end{cases}$$

and $B_{3\nu}(x) = 0$ for $x < x_\nu$ and $x \geq x_{\nu+4}$. To compute $s(x_i - 2\alpha_{ij}, y_j - 2\beta_{ij})$ for all $N_x N_y$ mesh points, requires $O(N_x N_y)$ floating point operations.

This algorithm can be fully parallelised. A description of the B-spline tensor product procedure of interpolation can be found in De Boor [16] and Hämmerlin and Hoffmann [17].

5.3. Numerical Resolution of the ‘‘Almost’’ Tridiagonal Systems

In order to compute the B-spline coefficients with the method described above, we are confronted with the resolution of linear systems which are tridiagonal except for two rows and columns. Instead of using full system solves, an ad hoc block decomposition of the matrices leads to a tridiagonal system solve coupled to a 2×2 system solve. All those matrices (19) and (20) have been brought by row and column exchanges to the block form

$$M = \left(\begin{array}{cc|c} \zeta_1 & \zeta_2 & \lambda \\ \zeta_3 & \zeta_4 & \\ \hline & & A \end{array} \right),$$

where γ is a matrix with two columns and A a square positive definite tridiagonal matrix.

Then the block linear system

$$M \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix} \quad (21)$$

can be solved using the following procedure: First, denoting by

$$\delta = \begin{pmatrix} \zeta_1 & \zeta_2 \\ \zeta_3 & \zeta_4 \end{pmatrix},$$

we can eliminate y ; this gives the system

$$(\delta - \lambda A^{-1} \gamma)x = b - \lambda A^{-1}c. \quad (22)$$

Then x being known, we compute y by resolving

$$Ay = c - \gamma x. \quad (23)$$

The system (22) is a 2×2 system which can be solved explicitly.

We are now ready to write down the numerical algorithm; let us first notice that only the right-hand side of system (21) is time dependant. Hence, we can factorise the matrix A and assemble the 2×2 matrix (22) once for all in an initialisation step.

1. *Initialisation*

- factorise A and store it
- compute and store $A^{-1}\gamma$ using the previously computed factorisation
- assemble the matrix $\delta - \lambda A^{-1}\gamma$

2. *Time loop*

- compute $A^{-1}c$ using the stored factorisation of A
- assemble $b - \lambda A^{-1}c$
- solve (22) using the explicit formulae
- compute y using the already compute values of $A^{-1}c$, $A^{-1}\gamma$ and of x .

The factorisation of A and the subsequent resolutions of the systems involving A , which involves a large part of the computations performed in the code, are performed using optimised library subroutines.

5.4. *The Time Marching Scheme*

After having detailed the implementation of the semi-Lagrangian method for the guiding-center Vlasov equations let us now write down the time marching algorithm for the full guiding-center Vlasov–Poisson problem:

Start with an initial charge density $\rho_0(x, y)$. In order to initialise our two time step advance, we first need to get the value of ρ at the first time step. For this purpose, we perform a Poisson solve which yields \mathbf{E}_0 , compute the drift velocity \mathbf{v}_{D0} and use Eq. (13) over half a time step using \mathbf{v}_{D0} as an approximation of $\mathbf{v}_D(t = \frac{1}{2}\Delta t)$ in order to get ρ_1 . This procedure provides the needed accuracy in our particular problem; however, it might be necessary to use a more sophisticated method for different problems.

Then ρ_{n-1} and ρ_n being given, the time loop reads

1. Compute \mathbf{E}_n with a Poisson solve from ρ_n .
2. Compute the drift velocity \mathbf{v}_{Dn} from \mathbf{E}_n .
3. Compute ρ_{n+1} , using the two time step semi-Lagrangian algorithm, from ρ_{n-1} and \mathbf{E}_n .

6. THE KELVIN–HELMHOLTZ INSTABILITY IN A PLASMA

In order to validate our code, we used two problems introduced by Ghizzo *et al.* [18].

6.1. *First Test Case*

First, we took a case where the growth rate of the instability can be computed analytically. This enabled us to check the accuracy of the code by comparing the analytical values with the computed ones.

Shoucri's analysis [19] shows how to start a Kelvin–Helmholtz instability for our model (15)–(16) by picking an initial condition of the form $\rho(x, y, t = 0) = \rho^0(y) + \epsilon\rho^1(y) \cos(kx)$ which yields through Poisson's equation a ϕ of the form $\phi = \phi^0(y) + \epsilon\phi^1(y) \cos(kx)$. The instability occurs by choosing properly the function ρ^1 which defines the perturbation around the equilibrium solution (ρ^0, ϕ^0) . Shoucri picks $\rho^0(y) = \sin y$. Then he shows that taking $\phi^1 = \psi_s = \sin(y/2)$, which he calls the neutrally stable solution and which amounts to taking $\rho^1 = (k^2 - \frac{1}{4}) \sin(y/2)$. The perturbation should remain constant for all time t if $k = k_s = \sqrt{3}/2$; there should be exponential growth for $k < k_s$ and no instability for $k > k_s$. Moreover, the growth rate of the instabilities should be such that

$$\frac{\text{Im}(\omega)}{k} = \frac{\sqrt{3}}{2}(k_s - k). \quad (24)$$

In order to compare our code with this theory, we ran two simulations with the initial condition

$$\rho(x, y, t = 0) = \sin y + 0.015 \sin \frac{y}{2} \cos(kx),$$

where $k = 2\pi/L_x$, L_x being the length of the domain in the x -direction. In both runs the domain in the y -direction went from 0 to 2π , the number of mesh points in each direction was 128 and the time step 0.5.

In the first case we took $L_x = 7$ which yields $k = 2\pi/7 = 0.89 > k_s = \sqrt{3}/2 = 0.86$. Hence, we should be in the stable case. The results of this run are displayed in Fig. 1, which clearly shows that there is no instability. In Fig. 2 we also see that the first mode is not growing.

In the second case we took $L_x = 10$ which yields $k = 2\pi/10 = 0.62 < k_s = \sqrt{3}/2 = 0.86$. This time we should be in the unstable case, which is confirmed by the results displayed in Fig. 3. Figure 2 also shows that the first mode is growing. Moreover, the growth rate of the instability $\text{Im}(\omega)$, according to Eq. (24) should have a value of 0.129, for $k_s = \sqrt{3}/2$ and $k = 2\pi/10$. In Fig. 4, where we zoom on the linear part of the temporal behaviour of the first mode, the slope for $\log \rho_k^2$ is 0.248. This should be twice the value we are looking for. The computed growth rate is 0.124, which is in very good agreement with the theoretical value.

In this case, we also displayed the evolution of the energy $\int E^2 dx$ and enstrophy $\int \rho^2 dx$, which are theoretically invariants of the system, in Fig. 5. As expected for a semi-Lagrangian Vlasov code (see [3]) the energy decreases during the smoothing phase, where

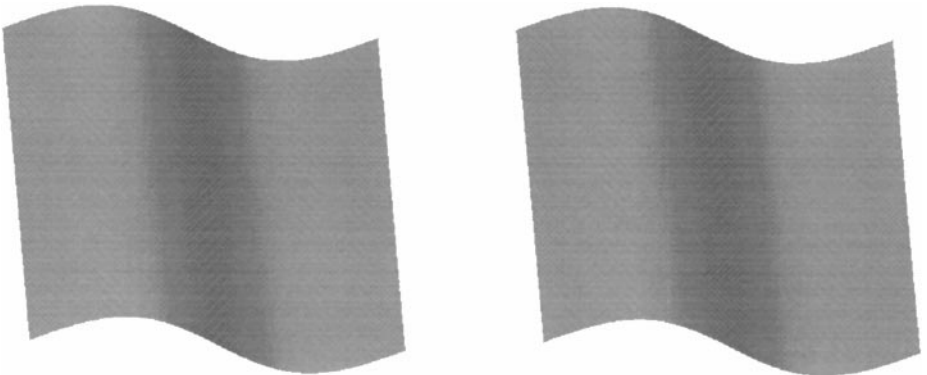


FIG. 1. The stable case: $t = 0$ (left) and $t = 1000$ (right).

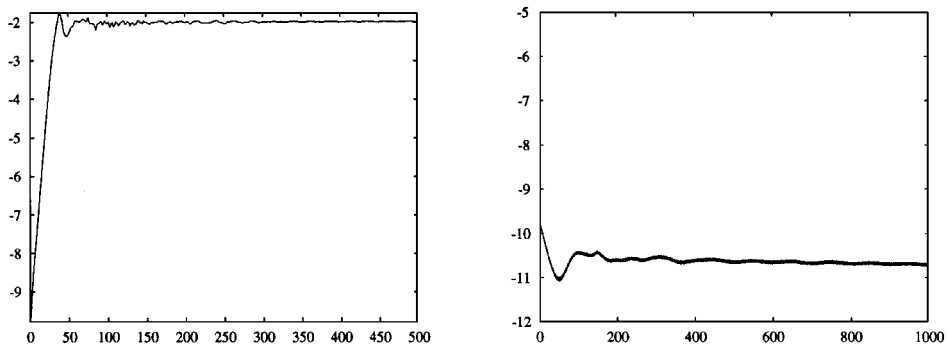


FIG. 2. The first Fourier mode: $\log \rho_k^2$ with respect to time, unstable case (left) and stable case (right).

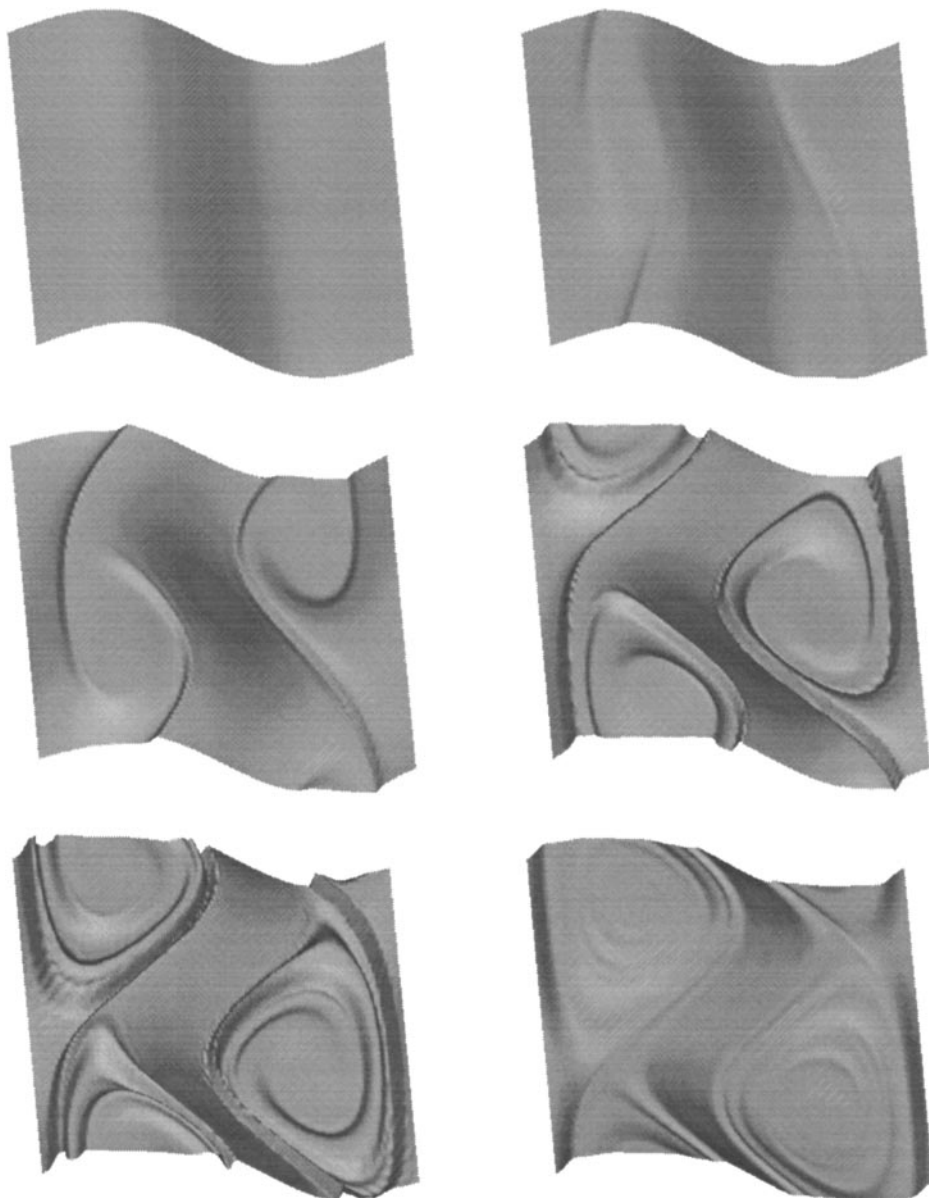


FIG. 3. The unstable case: from left to right and top to bottom $t = 0$, $t = 20$, $t = 30$, $t = 40$, $t = 50$, and $t = 1000$.

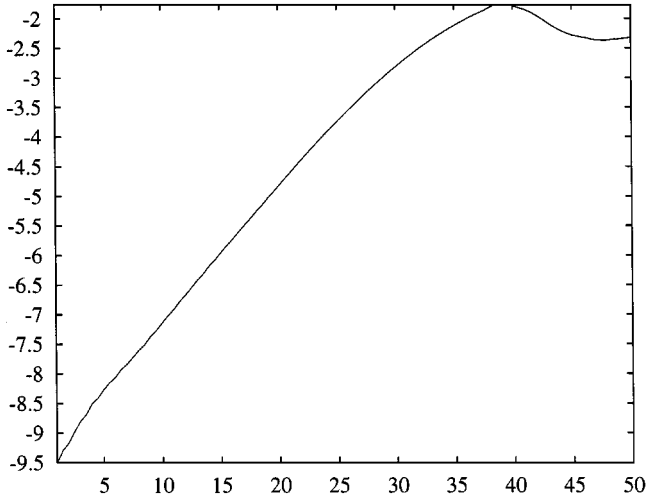


FIG. 4. Computation of the growth rate of the linear part of the first Fourier mode.

the microstructures cannot be resolved within a cell and, thus, get smeared out. Except for this phenomenon, which is not unphysical, although it does not appear in the Vlasov equation, as it mimics through a numerical artefact the effect collisions would have in a real system, the energy is very well conserved over many time steps.

6.2. Second Test Case

In the second test case, still following Ghizzo *et al.* [18], we started the computation with an initial charge density

$$\rho(x, y, t = 0) = 1.5 \operatorname{sech}(y/0.9)(1 + 0.08 \sin(2k_0x))$$

in order to start a Kelvin–Helmholtz instability. The computational mesh consisted of 128 points in each direction with $0 \leq x \leq 40$ and $-5 \leq y \leq 5$. We used a time step of 0.5 in the dimensionless units.

In order to solve the fixed point equation (14), we tried the three different procedures we described in Section 5.1:

1. Fixed point iterations which took around six iterations to converge.

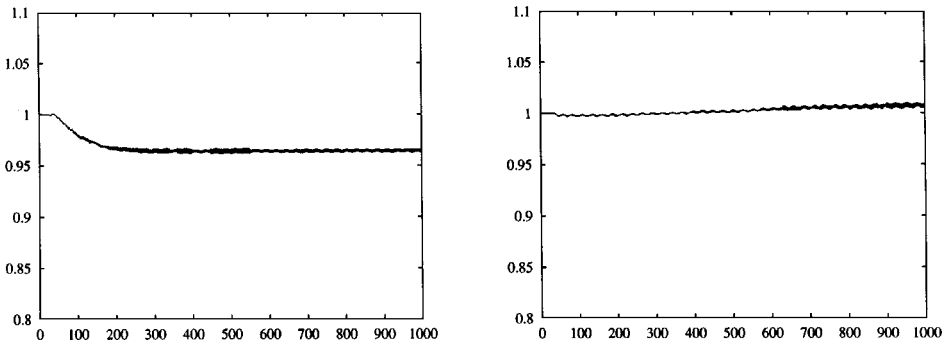


FIG. 5. The invariants: energy (left) and enstrophy (right).

2. The Newton method which took only two iterations to converge after the first time step. For both of these methods we kept the values of α and β from the previous time step to start the iterations.

3. Doing no iterations at all, assuming that $U(x_i - \alpha_{ij}, y_j - \beta_{ij})$ was close enough to $U(x_i, y_j)$. This worked fine, provided we divided the time step by 10.

Through the choice of the initial condition, we excited the second mode. Hence, the instability started right from the beginning of the run and saturated around time $t = 20$, as we can see on Fig. 6, yielding two circular roll-up vortex structures. When the microstructures become of the order of the mesh size, smoothing occurs; see Fig. 7. Then after some time a

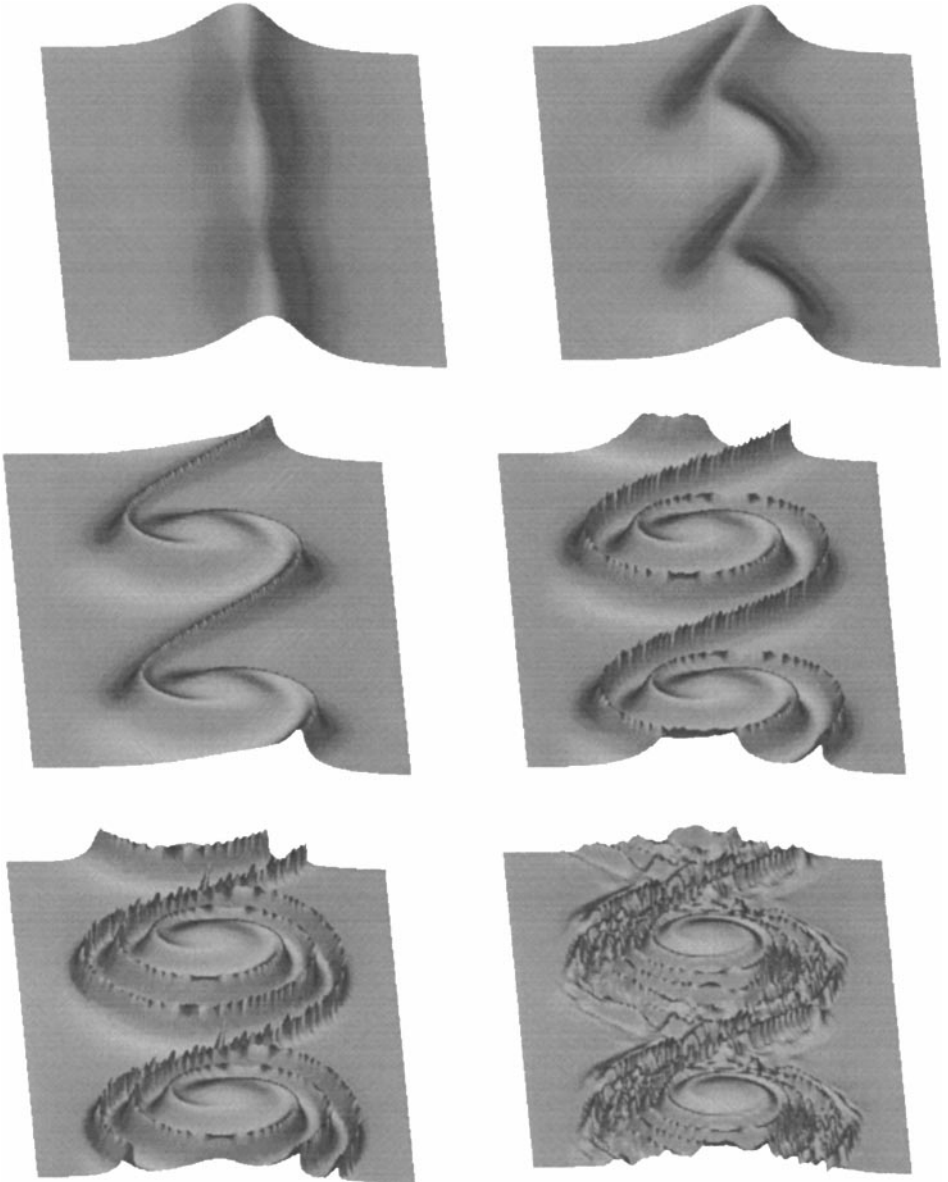


FIG. 6. Development of the first instability, from left to right and top to bottom $t = 0$, $t = 5$, $t = 10$, $t = 15$, $t = 20$, and $t = 50$.

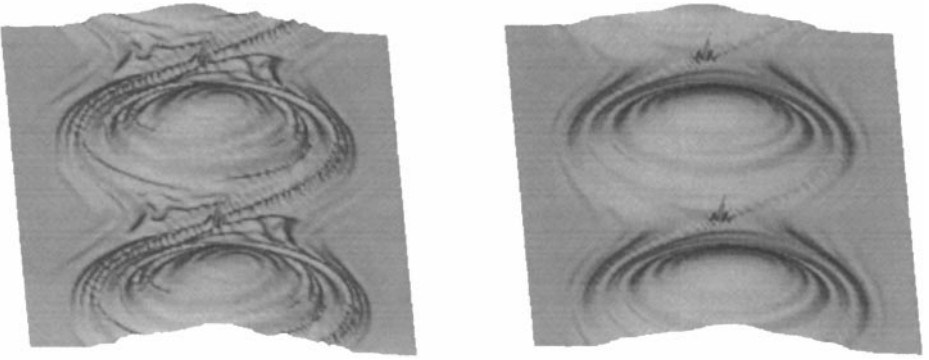


FIG. 7. Between the instabilities, left $t = 100$ and right $t = 200$.

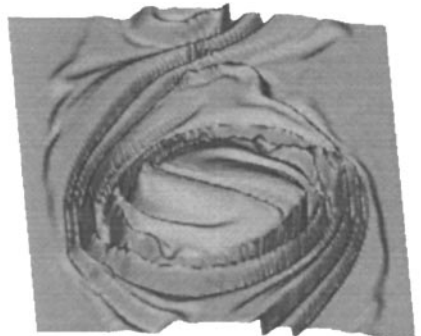
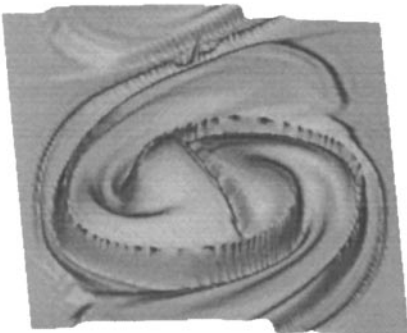
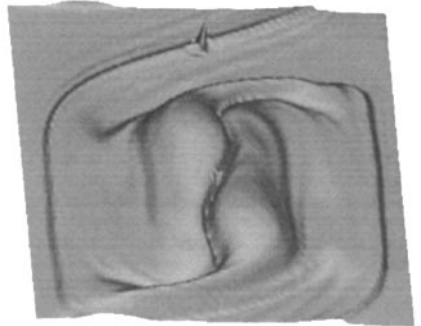
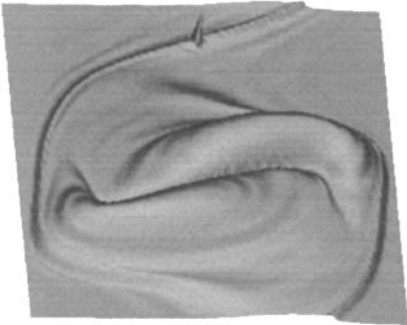
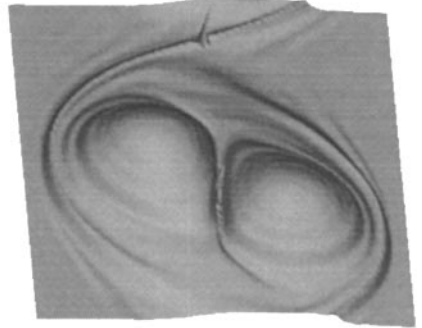
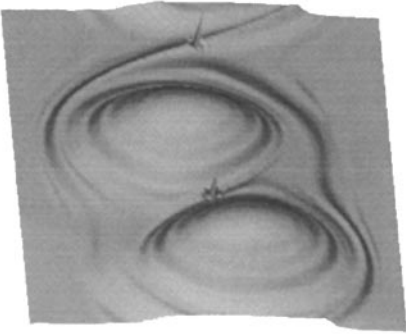


FIG. 8. Development of the second instability, from left to right and top to bottom $t = 235$, $t = 240$, $t = 245$, $t = 250$, $t = 260$, and $t = 275$.

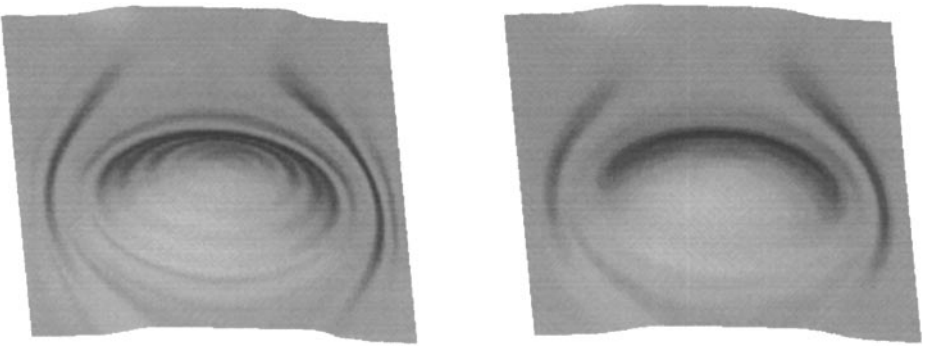


FIG. 9. Steady-state, left $t = 1000$ and right $t = 5000$.

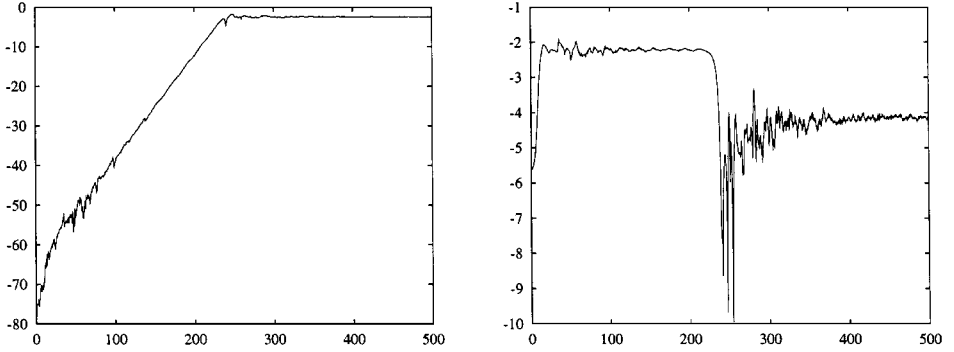


FIG. 10. Evolution of log of the first (left) and second (right) Fourier modes with respect to time between $t = 0$ and $t = 500$.

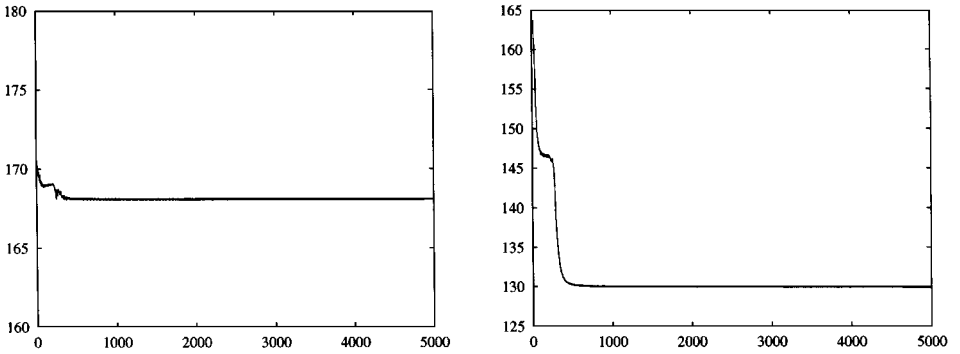


FIG. 11. Graphs of the L^1 norm (left) and L^2 norm (right) with respect to time.

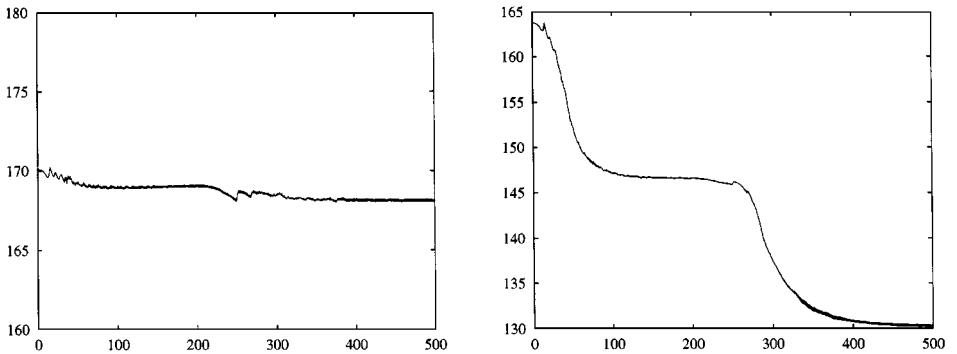


FIG. 12. Evolution of the L^1 norm (left) and L^2 norm (right) with respect to time between $t = 0$ and $t = 500$ for the Kelvin–Helmholtz instability.

second instability takes place (Fig. 8), where the two distinct structures merge and roll-up around each other. Here again, there are microstructures which smooth out when they reach the scale of the mesh size, as we can see on Fig. 9. This second instability can be explained by the fact that the two-vortex structure is an unstable equilibrium state. Mode 1 acts as a perturbation, and at some time, depending on the roundoff errors, the equilibrium is lost and the second instability takes place until the second mode vanishes, as we can see on Fig. 10.

As represented on Figs. 11 and 12, the L^1 norm, i.e. $\int f dx dy$, even though it decreases slightly during the two unstable phases, is conserved with an accuracy better than 1%. The L^2 norm, i.e. $\int f^2 dx dy$, decreases strongly on two occasions and is stable the rest of the time; this is the same phenomenon we discussed in the previous test case and which occurs when the microstructures get smoothed out as they become smaller than a cell size.

The first and second Fourier modes are also shown on Fig. 10. The first mode shows a linear growth, until the second instability, and becomes steady afterward. The second mode growth during the first instability remains steady until the second instability, and then decreases and stabilises at a smaller value.

7. CONCLUSION AND PERSPECTIVES

In this paper, we introduced the semi-Lagrangian method for several types of Vlasov equations and discussed the simplifications that can be used in the implementation for some specific models, where a time-splitting procedure can be applied. On the other hand, we described the full method which works for any type of Vlasov equation and implemented it for the case of the guiding-centre Vlasov–Poisson model which is an example that could not be solved accurately using the splitting procedure. The numerical results obtained on this example were very satisfying.

Building on the methods we introduced here we are now ready to develop a Fortran 90 module library, implementing the different kinds of advection types that are needed. Assembling these modules will then enable us to treat many problems occurring in plasma physics using the semi-Lagrangian methodology. Let us also mention that this methodology does not rely upon the use of regular grids. All we need is a set of lines in each direction for the spline interpolation, but these need not be equally spaced. On the other hand, as mostly through the use of parallel computers, more and more memory is available on today's supercomputers it becomes quite conceivable to solve 4D problems with this method with good resolutions and even 5D or 6D problems, where the resolution requirements are not too important, can be solved and in a few years medium sized 5D or 6D problems should become feasible.

ACKNOWLEDGMENTS

The authors acknowledge the Centre Charles Hermite in Nancy for computer time allocation on the Origin 2000 super computer. A.G. and P.B. acknowledge G. Knorr for fruitful discussions on the topic.

REFERENCES

1. A. Ghizzo, P. Bertrand, M. Shoucri, T. W. Johnston, E. Filjakow, and M. R. Feix, A Vlasov code for the numerical simulation of stimulated Raman scattering, *J. Comput. Phys.* **90**, 431 (1990).
2. A. Ghizzo, P. Bertrand, M. L. Begue, T. W. Johnston, and M. Shoucri, A Hilbert–Vlasov code for the study of high-frequency plasma beatwave accelerator, *IEEE Trans. Plasma Sci.* **24**, 370 (1996).

3. M. R. Feix, P. Bertrand, and A. Ghizzo, Eulerian codes for the Vlasov equation, in *Series on Advances in Mathematics for Applied Sciences*. Vol. 22. *Kinetic Theory and Computing*, edited by B. Perthame (1994), p. 45.
4. O. Coulaud, E. Sonnendrücker, E. Dillon, P. Bertrand, and A. Ghizzo, Parallelization of semi-Lagrangian Vlasov codes, *J. Plasma Phys.*, to appear.
5. C. Z. Cheng and G. Knorr, The integration of the Vlasov equation in configuration space, *J. Comput. Phys.* **22**, 330 (1976).
6. A. Staniforth and J. Côté, Semi-Lagrangian integration schemes for atmospheric models—A review, *Mon. Weather Rev.* **119** (1991).
7. R. Bermejo, Analysis of an algorithm for the Galerkin-characteristic method, *Numer. Math.* **60** (1991).
8. M. Falcone and R. Ferreti, Convergence analysis for a class of high-order semi-Lagrangian advection schemes, *SIAM J. Numer. Anal.* **35**(3), 909 (1998).
9. P. K. Smolarkiewicz and J. A. Pudykiewicz, A class of semi-Lagrangian approximations for fluids, *J. Atmos. Sci.* **49**(22), 2082 (1992).
10. P. Bartello and S. J. Thomas, The cost-effectiveness of semi-Lagrangian advection, *Mon. Weather Rev.* **124**, 2883 (1996).
11. J. P. Boris and D. L. Book, Solution of continuity equations by the method of flux-corrected transport, in *Methods of Comput. Phys.* Vol. 16, edited by J. Killeen (Academic Press, New York, 1976), p. 85.
12. M. Shoucri, P. Bertrand, M. Feix, and B. Izrar, The application of shape preserving splines for the modified Korteweg–de Vries equation, *Comput. Phys. Commun.* **46**(1), 1 (1987).
13. G. I. Marchuk, *Methods of Numerical Mathematics* (Springer-Verlag, New York, 1982).
14. E. Fijalkow, E. Jamin, M. Feix, P. Bertrand, A. Ghizzo, and M. Shoucri, A fully parallel 2D Vlasov code, in *14th Conference on the Numerical Simulations of Plasmas, Annapolis, 1991*, Pt. 2.
15. M. Shoucri, Numerical solution of the two-dimensional Vlasov equation, *IEEE Trans. Plasma Sci.* **PS-7**(2), 69 (1979).
16. C. De Boor, *A Practical Guide to Splines* (Springer-Verlag, New York, 1978).
17. M. Shoucri, A two-level implicit scheme for the numerical solution of the linearized vorticity equation, *Int. J. Numer. Meth. Eng.* **17**, 1525 (1981).