Journal of Computational Physics **179,** 495–538 (2002)
doi:10.1006/jcph.2002.7071

A Numerical Scheme for the Integration of the Vlasov–Maxwell System of Equations

A. Mangeney,* F. Califano,† C. Cavazzoni,‡ and P. Travnicek#

*DESPA, Observatoire de Paris, Meudon, France; †Istituto Nazionale Fisica della Materia, Sez. A, Dipartimento di Fisica, Pisa, Italy; ‡Cineca, Bologna, Italy; Mullard Space Science Laboratory, University College, London, United Kingdom; and #Institute of Atmospheric Physics, Prague, Czech Republic
E-mail: califano@df.unipi.it

Received January 16, 2002; revised April 8, 2002

We present a discussion of some numerical algorithms for the solution of the Vlasov–Maxwell system of equations in the magnetized, nonrelativistic case. We show that a splitting scheme combined with a Van Leer type of discretization provides an efficient and accurate scheme for integrating the motion of charged particles in their self-consistent electromagnetic field. The problem of open boundary conditions is also considered. We then discuss the parallelization strategy as used on large parallel computers. Finally, we present an example of the evolution of an electromagnetic beam plasma instability as a typical problem of interest in plasma physics research which can be studied with the Vlasov code.

1. INTRODUCTION

Space and laboratory plasmas can be considered collisionless in many situations and their dynamics, driven in part by kinetic effects, i.e., more or less transient deviations from thermodynamical equilibrium, can be described by the Vlasov–Maxwell (or Vlasov–Poisson) equations. The Vlasov equation plays a central role in classical and semiclassical time-dependent mean field theory and has been used to model a wide range of many-body processes from plasma physics (see, for example, Akhiezer *et al.* [1]) to the gravitational N-body problem (see, for example, Peebles [2]) and nuclear dynamics (see, for example, Bertsch and Das Gupta [3]). In plasma physics, the physical content of these equations is conceptually simple: Coulomb interactions between the charged particles are replaced by a common mean electromagnetic field determined by the Maxwell or Poisson equations, with electric charge and current densities determined by the particle distribution functions. However, for any realistic (even simplified) problem, this system of equations cannot be dealt with analytically, especially in the full nonlinear regime, and large-scale numerical

0021-9991/02 $35.00

simulations are presently the most important resource for investigating the dynamics of such systems.

The particle-in-cell (PIC) approach (see, for example, Birdsall and Langdon [4]) has been the favorite tool essentially because it allows the modeling of large-scale phenomena in one, two, and even three dimensions while solving the Vlasov equation, which is a partial differential equation for a distribution function depending on six coordinates (besides the time) and requires in general numerical resources which are far beyond the present day possibilities. However, a number of interesting problems can be studied in a phase space of lower dimensionality as, for example, two spatial and two or three velocity coordinates, which brings their numerical resolution via the Vlasov equation within the possibilities of existing computers.

The interest of such an approach resides mainly in the fact that one is not bothered by the statistical noise which is intrinsic to PIC simulation, so it is possible to study with high accuracy problems where small-scale structures in phase space play an important role, such as in the formation of electron or ion holes. The knowledge of the particle distribution function and its evolution with a good space and time resolution is then essential to the understanding of the physical phenomena under study.

However, the free-streaming evolution described by the advective term in the Vlasov equation usually leads to the formation of increasingly smaller and smaller scales in velocity space, resulting in a numerically unstable behavior when these scales become comparable to the velocity resolution. This is the "filamentation problem," which has been one of the reasons why Vlasov simulations have been poorly considered compared to particle simulations, which are insensitive to these small scales in velocity space. Several ways of dealing with this problem have been devised (see, for example, Klimas and Farrell [5]) and we shall discuss this problem in what follows (see the last section).

In this paper we describe a numerical algorithm for the solution of the Vlasov equation in the phase space self-consistenly coupled to the Maxwell equations. This algorithm is based on the so-called "splitting scheme" widely used in numerical fluid dynamics; this scheme was first applied by Cheng and Knorr [6] to the solution of the Vlasov–Poisson in the electrostatic limits and later used by several authors. We give here an extension to the fully magnetized case. In this algorithm the advection equation plays a key role and its solution is used repeatedly; therefore it is crucial to implement a performing and accurate scheme to solve this equation. A significant part of this paper is devoted to the description of a general framework where the different methods which have been used so far and their possible generalizations can be analyzed; we present a detailed numerical comparison between the most frequently used methods. We discuss also the problem of open boundary conditions which, to the authors knowledge, has been treated by ad hoc methods. We propose a specific scheme which we found to perform successfully in our simulations. We discuss in Appendix II the corresponding open boundary conditions for the Poisson equation in the electrostatic limit and for the Maxwell equations in the full electromagnetic regime. We also discuss the implementation of this algorithm on large parallel computers and finally an example of the use of this code is presented in the case of a problem of interest in plasma physics research.

The paper is organized as follows. In Section 2 we introduce the equations and the charactristic parameters. In Section 3 we discuss the numerical algorithm for the integration of the Vlasov equation and in Section 4 we present a scheme for implementing open boundary conditions on the Vlasov equation. A comparison between the most used algorithm for the solution of the Vlasov equation is discussed in Section 5. We then discuss the parallel

implementation of the algorithm in Section 6 and we present in Section 7 an example of the use of the code in the five-dimensional case (2d in space and 3d in velocity). Finally, we discuss in Section 8 some issues like coarse-grained evolution and the related problem of phase space filamentation and we present our conclusions.

2. THE EQUATIONS

The behavior of a multispecies collisionless magnetized plasma is described by the Vlasov–Maxwell system of equations, which, in physical units, reads

$$\frac{\partial f_\alpha}{\partial t} + \mathbf{v}\cdot\frac{\partial f_\alpha}{\partial \mathbf{x}} + \frac{Z_\alpha e}{m_\alpha}(\mathbf{E}+\mathbf{v}\times\mathbf{B})\cdot\frac{\partial f_\alpha}{\partial \mathbf{v}} = 0, \tag{1}$$

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla\times\mathbf{E}, \quad \epsilon_0\mu_0\frac{\partial \mathbf{E}}{\partial t} = \nabla\times\mathbf{B} - \mu_0\,\mathbf{j}, \tag{2}$$

$$\nabla\cdot\mathbf{E} = \frac{q}{\epsilon_0}, \quad \nabla\cdot\mathbf{B} = 0. \tag{3}$$

Here $f_\alpha(\mathbf{x}, \mathbf{v}, t)$ is the distribution function of the particles of species α, m_α and $Z_\alpha e$ are the corresponding mass and electric charge, and $\mathbf{E}(\mathbf{x}, t)$ and $\mathbf{B}(\mathbf{x}, t)$ are the electric and magnetic fields determined by external and internal charge and current densities given by

$$q = e\sum_\alpha Z_\alpha \int f_\alpha\, d\mathbf{v}; \quad \mathbf{j} = e\sum_\alpha Z_\alpha \int \mathbf{v} f_\alpha\, d\mathbf{v}.$$

The physical phenomena which are described by the Vlasov–Maxwell equations (1)–(3) cover an enormous range of spatial and temporal scales, so that various dimensionless forms and approximations are used according to the specific problem one has to deal with. These equations have five dimensionless parameters (see, for example, Gott and Yurchenko [7]) which can be introduced as follows.

By defining $\bar{T}$ as a characteristic time scale, $\bar{m}$ as a characteristic mass, $\bar{E}$ and $\bar{B}$ as a characteristic electric and magnetic field, $\bar{U} = \bar{E}/\bar{B}$ as the associated characteristic (drift) velocity, and $\bar{v}_\alpha$ as the characteristic thermal speeds, the Vlasov–Maxwell equations can be expressed in dimensionless form as (see the appendix for the details)

$$\frac{\partial f_\alpha}{\partial t} + \lambda_\alpha\mathbf{v}\cdot\frac{\partial f_\alpha}{\partial \mathbf{x}} + Z_\alpha\zeta_\alpha(\mathbf{E}+\lambda_\alpha\mathbf{v}\times\mathbf{B})\cdot\frac{\partial f_\alpha}{\partial \mathbf{v}} = 0, \tag{4}$$

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla\times\mathbf{E}, \quad \frac{\partial \mathbf{E}}{\partial t} = R_E(\nabla\times\mathbf{B} - \mathbf{j}), \tag{5}$$

$$\nabla\cdot\mathbf{E} = R_E q, \quad \nabla\cdot\mathbf{B} = 0, \tag{6}$$

where $\lambda_\alpha = \bar{v}_\alpha/\bar{U}$, $\zeta_\alpha = (e\bar{E}/\bar{m})/(\bar{v}_\alpha\bar{T})$, and $R_E = (c/\bar{U})^2$.

We note that the field equations contain only one dimensionless parameter, R_E; the limit $R_E \to \infty$ corresponds to the Darwin approximation, where the displacement current is neglected. The Vlasov equation for each particle species contains two dimensionless parameters: λ_α, which characterizes the width of the distribution function f_α in velocity space, and ζ_α, which characterizes (with λ_α) the intensity of the Lorentz force acting on the particles. The corresponding (normalized) moments, the density n_α, the bulk velocity

$\mathbf{u}_\alpha$, the kinetic energy density K_α, and the kinetic energy flux $\mathbf{\Phi}_\alpha$, for each component, are given by

$$n_\alpha = \int f_\alpha \, d\mathbf{v}, \quad n_\alpha \mathbf{u}_\alpha = \lambda_\alpha \int \mathbf{v} f_\alpha \, d\mathbf{v}, \quad K_\alpha = \frac{\lambda_\alpha}{\zeta_\alpha} \int \frac{v^2}{2} f_\alpha \, d\mathbf{v}, \quad \mathbf{\Phi}_\alpha = \frac{\lambda_\alpha^2}{\zeta_\alpha} \int \frac{v^2}{2} \mathbf{v} f_\alpha \, d\mathbf{v}. \tag{7}$$

3. THE ALGORITHM

Two main steps have to be considered for the numerical solution of the Vlasov equation: the first one relates to the time discretization and the second one to the space and velocity discretizations. We start by laying out the general framework which will be used in this paper; in order to simplify the notations, we assume in this section that there is only one particle species, with parameters $\lambda_\alpha = 1$, $Z_\alpha = -1$, and $\zeta_\alpha = 1$.

The Vlasov equation is basically a multidimensional advection equation: even in the simplest case of a one-dimensional plasma interacting with an electrostatic field, the distribution function still depends on two independent variables (x, v). An important property of this advection equation is that its characteristics, the particle trajectories,

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}, \quad \frac{d\mathbf{v}}{dt} = -(\mathbf{E} + \mathbf{v} \times \mathbf{B}), \tag{8}$$

describe a Hamiltonian flow $\mathcal{T}^t$ in phase space, usually in noncanonical variables,

$$z_0 \to z(z_0, t) \equiv \mathcal{T}^t z_0,$$

using the notation $z = (\mathbf{x}, \mathbf{v})$ to represent a point in phase space. The flow is reversible,

$$z_0 = \mathcal{T}^{-t} z, \quad \mathcal{T}^{-t} \cdot \mathcal{T}^t = \textit{identity},$$

and preserves the volume element in phase space,

$$dz = dz_0,$$

a property characterizing symplectic transformations.

In terms of this flow, the solution of the Vlasov equation can be written formally as

$$f(z, t) = f_0(\mathcal{T}^{-t} z), \tag{9}$$

with f_0 being the distribution function at time $t = 0$. This relation defines an evolution operator T^t acting on sufficiently smooth functions of z,

$$f(z, t) = T^t f_0(z) = f_0(\mathcal{T}^{-t} z).$$

Assume for a moment that the flow $\mathcal{T}^t$ (or T^t) is known; the problem is then to obtain a discrete formulation by using the fact that the space and velocity discretizations involve the projection of f onto some finite dimensional function space W. This is easily done if

one knows a complete basis $\phi_\alpha(z)$ spanning W as well as a basis ψ_α of its dual (the set of all linear functionals acting on the elements of W), such that

$$\psi_\alpha(\phi_\beta) = \delta_{\alpha,\beta}. \tag{10}$$

Then f is approximated by a function $\tilde{f}$ which belongs to W and can be characterized by its components

$$a_\alpha = \psi_\alpha(\tilde{f})$$

in the basis ϕ_α,

$$\tilde{f}(z) = \sum_\alpha a_\alpha \phi_\alpha(z).$$

Knowing the coefficients $\{a_\alpha(0)\}$ of the initial distribution function and using Eq. (9), $\tilde{f}$ may be advanced in time,

$$f(z,t) = \sum_\alpha a_\alpha(0)\phi_\alpha(T^{-t}z),$$

so that

$$a_\alpha(t) = \sum_\beta \psi_\alpha(\phi_\beta(T^{-t}z))a_\beta(0) = \sum_\beta A^\alpha_\beta a_\beta, \tag{11}$$

where the matrix elements A^α_β of the projection $\tilde{T}^t$ of the evolution operator T^t on the approximation space W are given by

$$A^\alpha_\beta = \psi_\alpha(\phi_\beta(T^{-t}z)). \tag{12}$$

The approximated evolution operator is in general no longer reversible, with

$$\tilde{T}^{-t} \cdot \tilde{T}^t \neq \textit{identity}$$

expressing the fact that the Hamiltonian character of the phase space flow has been lost and that the resulting scheme is dissipative.

However, the flow $\mathcal{T}^t$ is usually not known explicitly. One has therefore to restrict oneself to time steps Δt sufficiently small so that Eq. (8) can be solved approximatively to some order Δt^r. If r is small ($r \leq 2$) this can be done analytically; this is the method used by Cheng [11], Fijalkow [12], and Gazdag [13] to treat the advance in a constant magnetic field. For higher order, or more complicated, motions one can use an iterative solution (see, for example, Bermejo [14] or Sonnendrücker *et al.* [15]).

3.1. *The Splitting Scheme*

To obtain an approximation of the flow $\mathcal{T}$ and of the associated evolution operator T, we have chosen here another method, which is to use the splitting scheme (see, for example, Blanes and Moan [16]), as was first done by Cheng and Knorr [6] in the electrostatic case

where the space and velocity advection terms are advanced separately. The main reason for this choice can be understood if one puts this splitting method in the perspective of "symplectic" integrations. Consider, for example, the electrostatic case where the particle motion is described by the dimensionless Hamiltonian

$$H = \frac{v^2}{2} + \Phi(x),$$

with a potential $\Phi(x)$ depending only on the position; then the Vlasov equation can be cast in the form

$$\frac{\partial f}{\partial t} = -[H, f] \equiv \Lambda f$$

using the Poisson brakets $[H, f] = \{(\partial H/\partial x)(\partial f/\partial v) - (\partial H/\partial v)(\partial f/\partial x)\}$, and the evolution operator T^t becomes

$$T^t = e^{\Lambda t}.$$

The Hamiltonian can be split into two parts, $H = H_1 + H_2$, with $H_1 = v^2/2$ and $H_2 = -\Phi(x)$, which have the properties that the corresponding operators $\exp(\Lambda_1 \tau)$ and $\exp(\Lambda_2 \tau)$, with $\Lambda_i f = -[H_i, f]$, are explicitly known as translations in x or v,

$$\exp(\Lambda_1 \tau) f(x, v) = f(x - v\tau, v), \quad \exp(\Lambda_2 \tau) f(x, v) = f(x, v - E(x)\tau), \tag{13}$$

where $E = -\partial \Phi/\partial x$ is the electric field. It can be shown that

$$e^{\Lambda t} = \lim_{N \to \infty} \left[\exp\left(\frac{\Lambda_2 t}{2N}\right) \exp\left(\frac{\Lambda_1 t}{N}\right) \exp\left(\frac{\Lambda_2 t}{2N}\right) \right]^N.$$

For a finite time step $\tau t = t/N$ one gets the second-order approximation:

$$e^{\Lambda \tau} = \exp\left(\frac{\Lambda_2 \tau}{2}\right) \exp(\Lambda_1 \tau) \exp\left(\frac{\Lambda_2 \tau}{2}\right) + O(\tau^3). \tag{14}$$

These formulae are valid for any decomposition of the Hamiltonian but if the two operators Λ_1 and Λ_2 commute, Eq. (14) is exact and reduces to $e^{\Lambda \tau} = \exp([\Lambda_1 + \Lambda_2]\tau)$. Higher order approximations can be found in [16].

Equation (14) corresponds to an approximation of the phase space flow $\mathcal{T}^t$ given by

$$x_\tau = \mathcal{T}^\tau x = x + \tau v + \frac{\tau^2}{2} E\left(x + v\frac{\tau}{2}\right), \quad v_\tau = \mathcal{T}^\tau v = v + \tau E\left(x + v\frac{\tau}{2}\right), \tag{15}$$

which is the Verlet algorithm for a second-order particle advance (cf. Andersen [8]). A very important property of the map generated by Eq. (15) is the conservation of the volume element in phase space,

$$dx_\tau \, dv_\tau = dx \, dv,$$

a property characterizing symplectic transformations. This property ensures that the error of the map is bounded, i.e., there will be no secular growth in the energy conservation error.

One may even show (see Yoshida [9]) that the map (15) describes the motion of a particle with an approximate Hamiltonian,

$$\tilde{H}(x, v, \tau) = H + \frac{\tau v}{2} E + \frac{\tau^2}{12}\left(E^2 + v^2 \frac{\partial E}{\partial x}\right),$$

and that $\tilde{H}$ is exactly conserved.

The same arguments may also be applied when the phase space variables are not canonical, for exemple in the presence of a magnetic field. Consider first the simplest case of a spatially uniform plasma imbedded in a uniform magnetic field of unit magnitude and directed along the z-axis. The corresponding Vlasov equation reduces to

$$\frac{\partial f}{\partial t} = -\left[v_y \frac{\partial}{\partial v_x} - v_x \frac{\partial}{\partial v_y}\right] f, \tag{16}$$

whose solution to order $0(\tau^2)$ can be expressed in terms of the operators $\Theta_x = -v_y \partial/\partial v_x$ and $\Theta_y = v_x \partial/\partial v_y$ generating translations along the v_x and v_y axis,

$$f(v_x, v_y, \tau) = \Omega_z(\tau) f(v_x, v_y, 0), \tag{17}$$

with

$$\Omega_z(\tau) = \exp\left(\frac{\Theta_x \tau}{2}\right) \exp(\Theta_y \tau) \exp\left(\frac{\Theta_x \tau}{2}\right) + O(\tau^3).$$

The corresponding map

$$v_{x\tau} = v_x(1 - \tau^2/2) + v_y \tau, \quad v_{y\tau} = v_y(1 - \tau^2/2) - v_x \tau \tag{18}$$

can be shown to be symplectic, since it is a product of symplectic operators; furthermore it conserves exactly the quantity $(v_{x\tau}^2 + v_{y\tau}^2)/(1 + \tau^4/4)$, indicating that the motion is very close to a circular one.

We have studied numerically the properties of this scheme by integrating numerically Eq. (16) with equal resolution, $Nv_x = Nv_y = Nv$, in both velocity coordinates. We start with an initial distribution function which is a Maxwellian with a bulk velocity directed along the x-axis and we follow its evolution in time, as described by Eq. (17), forgetting both the spatial dependance and the z velocity. In this case, the distribution function, which is the exact solution of Eq. (16), preserves its shape with its maximum rotating around the magnetic field axis, with constant angular velocity $\Omega = 1$. the numerical solution shows the same rotation, but with a slightly different frequency $\Omega_{num} = 1 + \delta\Omega$, with $\delta\Omega$ depending on the numerical resolution in velocity space. We found after roughly 10^3 iterations that $\delta\Omega \simeq 10^{-2}$ for $N_v = 41$ points in each velocity direction, scaling as $\delta\Omega \sim N_v^{-1.8}$. When the velocity resolution is sufficient ($N_v \geq 20$) the total energy decreases at a rate which scales as $\delta E \sim N_v^{-3}$. Thus the scheme (17) discussed above performs satisfactorily and it has been used by Johnson [10], with minor modifications. In the case of a constant magnetic field it is essentially equivalent to the integration of Eq. (16) along characteristics such as (18), used, for example, by Cheng [11] and Fijalkow [12].

In a more general case, with a nonvanishing electric field **E** and a magnetic field **B** which has an arbitrary direction with respect to the reference frame, a symplectic advance of the

velocities exact to second order in time requires the product of seven translation operators in velocity space. Indeed, let us define

$$\Theta_x = F_x \frac{\partial}{\partial v_x}, \quad \Theta_y = F_y \frac{\partial}{\partial v_y}, \quad \Theta_z = F_z \frac{\partial}{\partial v_z} \tag{19}$$

and by analogy with Eq. (17)

$$\Omega_z(\tau) = \exp(\Theta_x \tau/2) \exp(\Theta_y \tau) \exp(\Theta_x \tau/2), \tag{20}$$

where F_x, F_y, and F_z are now the components of the Lorentz force, $\mathbf{F} = Z_\alpha \zeta_\alpha (\mathbf{E} + \lambda_\alpha \mathbf{v} \times \mathbf{B})$. Then it can be shown that

$$f(v_x, v_y, v_z, \tau) = \left[\Omega_z\left(\frac{\tau}{2}\right) \exp(\Theta_z \tau) \Omega_z\left(\frac{\tau}{2}\right)\right] f(v_x, v_y, v_z, 0) \tag{21}$$

is a solution, correct to second order in time of the equation

$$\frac{\partial f}{\partial t} = -\left[\mathbf{F} \cdot \frac{\partial}{\partial \mathbf{v}}\right] f.$$

In Eq. (21) we have in some sense priviliged the v_z-axis. This choice is arbitrary and one can select any other axis, obtaining the same results at least to second order in the time step.

In the general electromagnetic 5d case with two space variables (x, y), three velocity variables (v_x, v_y, v_z), and space- and time-varying electromagnetic fields $\mathbf{E}$ and $\mathbf{B}$, the scheme we used for the time advance over a time interval $\tau = \Delta t$ is a combination of spatial and velocity translations which are summarized by the equation

$$\begin{aligned} f(x, y, v_x, v_y, v_z, \tau) = \exp\left(\frac{\Lambda_{xy}\tau}{2}\right) \exp\left[\Omega_z\left(\frac{\tau}{2}\right) \exp(\Theta_z \tau) \Omega_z\left(\frac{\tau}{2}\right)\right] \\ \times \exp\left(\frac{\Lambda_{xy}\tau}{2}\right) f(x, y, v_x, v_y, v_z, 0), \end{aligned} \tag{22}$$

where Θ_z, Ω_z are as defined in Eqs. (19) and (20) and

$$\Lambda_{xy} = -\left(v_x \frac{\partial}{\partial x} + v_y \frac{\partial}{\partial y}\right)$$

is the infinitesimal generator of translations in the physical space.

Equation (22) must be coupled to the field equations, (2) and (3), which are solved once per time step. To maintain second-order accuracy in Δt, the electric and magnetic fields which are used to calculate the force $\mathbf{F}$ in the velocity advance (21) are determined self consistently by solving the Maxwell equations using currents and charges calculated with the translated distribution function $\exp(\Lambda_{xy}\tau/2) f$, i.e., just after the first half-time step.

To the authors knowledge, this advancing scheme for the fully electromagnetic case is original; in principle it could be easily extended to the full 6d case. However, even working with the most powerful supercomputers of the last generation, the 6d case still requires too large a memory requirement.

3.2. *Solution of the One-Dimensionnal Advection Equation*

The advancement of the distribution function summarized by Eq. (22) is obtained through a combination of "elementary" translations which differ only on the variable which is concerned. We now describe how such a translation may be efficiently discretized and for the sake of simplicity we only consider the case of the space translation operator and ignore the fact that f is actually a function of the velocity v. We are then left with the protype advection equation for a distribution function $f(x, t)$ depending only on space and time,

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} = 0, \tag{23}$$

where v is a constant velocity, considered as a parameter. What follows applies equally well to the velocity translation operators.

The corresponding evolution operator reduces to a translation:

$$T^t f(x) = e^{\Lambda t} f(x) = f(x - vt), \quad \Lambda = -v\frac{\partial}{\partial x}.$$

A considerable amount of work has been devoted to the problem of obtaining accurate numerical solutions of equations similar to Eq. (23) (see, for example, Godlewski and Raviart [17]). We shall limit ourselves here only to what is relevant for our purpose, which is to provide a common framework within which we may compare the different methods most commonly used in the numerical solution of the Vlasov–Maxwell or Vlasov–Poisson equations and evaluate their relative performances.

As mentionned above, the discretization process implies usually a projection on some finite-dimensional space of functions. We shall limit ourselves to spaces of piecewise polynomial functions, the properties of which depend on the degree of regularity which is required from the approximating functions. Three particular cases are considered here: first the spaces used in the "discontinuous Galerkin" methods when no regularity is required at the cell boundaries, then the "smooth Hermite" spaces when the continuity of the function and its m first derivatives is imposed at the cell boundaries, and finally the "spline" spaces when still more smoothness is required.

Let us now introduce some notations. The numerical domain $[0, L]$ is divided into N cells C_i, $1 \le i \le N$, of equal (for the sake of simplicity) width $\Delta x = L/N$ centered on the N grid point $\bar{x}_i = (i - 1/2)\Delta x$ and with boundaries $x_i = i\Delta x$, $0 \le i \le N$. We denote by f_i the value $f(x_i)$ of the function f at the cell boundaries, by $\bar{f}_i$ the "cell average,"

$$\bar{f}_i = \frac{1}{\Delta x}\int_{x_{i-1}}^{x_i} dx\, f(x) = \frac{1}{2}\int_{-1}^{+1} f\left(\bar{x}_i + \frac{\Delta x}{2}\xi\right) d\xi, \tag{24}$$

and by $H_i(x)$ the characteristic function of the cell C_i, where $H_i(x) = 1$ if x belongs to the cell C_i and $H_i(x) = 0$ otherwise. In this section, we limit ourselves to functions $f(x)$ which are periodic, with the open, nonperiodic case being considered in the next section.

3.3. *Discretization of the Displacement Operator*

3.3.1. Discontinuous Galerkin methods. The "discontinuous Galerkin" methods (see Victory and Ganguly [18], and more recently Cockburn [19]) uses the $(M + 1) * N$ dimensional space $V_{N,M}$ of functions which reduce, on each cell C_i, to polynomials of degree equal to or less than M. A convenient basis of $V_{N,M}$ is provided by the functions

$\phi_{i,l}(x) = H_i(x) P_l[2(x - x_i)/\Delta x]$, the P_l being orthogonal polynomials on the interval $[-1, +1]$, such as the Legendre polynomials, which we use in the following. Any function $\tilde{f}$ belonging to $V_{N,M}$ can be expanded on this basis,

$$\tilde{f} = \sum_{i=1,N} \sum_{l=0,M} a_{i,l} \phi_{i,l}(x), \tag{25}$$

and the L^2 scalar product and the associated norm reduce to

$$\langle \tilde{g}, \tilde{f} \rangle = \int \tilde{g}(x) \tilde{f}(x)\, dx = \frac{\Delta x}{2} \sum_{i=1,N} \sum_{i=0,M} c_l^2 b_{il} a_{il}, \quad \|\tilde{f}\|_{N,M}^2 = \frac{\Delta x}{2} \sum_{i=1,N} \sum_{l=0,M} c_l^2 a_{il}^2,$$

for any pair of functions $\tilde{f} = \sum \sum a_{il} \phi_{i,l}(x)$ and $\tilde{g} = \sum \sum b_{il} \phi_{i,l}(x)$ of $V_{N,M}$, with

$$c_l^2 = \int_{-1}^{1} d\xi\, P_l^2(\xi) = \frac{2}{(2l+1)}$$

being normalization factors.

Since the basis $\phi_{i,l}$ is orthogonal for this scalar product, the dual basis $\psi_{i,l}$ is simply the projection on $\phi_{i,l}$. Therefore a function $f(x)$ is approximated by the expansion (25) with coefficients $a_{i,l}$ which are given by

$$a_{i,l} = \frac{2}{\Delta x c_l^2} \int_{C_i} \phi_{i,l}(x) f(x)\, dx = \frac{1}{c_l^2} \int_{-1}^{+1} P_l(\xi) f\left(\bar{x}_i + \frac{\Delta x}{2} \xi\right) d\xi. \tag{26}$$

These coefficients can be given a physical meaning: $a_{i,0}$ is the "cell average" $\bar{f}_i$ and it can be seen by using a Taylor expansion of $f(\bar{x}_i + \frac{\Delta x}{2}\xi)$ that the coefficients of degree l are related to the spatial derivatives of f of order at least equal to l,

$$a_{il} = \sum_{k=0,\ldots} \alpha_{l,k} D^{l+2k} f(x_i), \tag{27}$$

since $\int_{-1}^{1} \xi^k P_l(\xi)\, d\xi = 0$ if either $k < l$ or $k + l$ is odd. We have used the notation $D = \Delta x \partial/\partial x$, with the $\alpha_{l,k}$ being numerical coefficients.

For each cell, there are therefore $M + 1$ independent data, including the cell average and local derivatives up to order M; the order M of the truncation characterizes the number of details retained in the function $\tilde{f}$, which is a piecewise constant function for $M = 0$, a piecewise linear function for $M = 1$, etc.... No requirements for the continuity of $\tilde{f}$ and its derivatives have been imposed up to now and the approximating function $\tilde{f}$ is in general discontinuous (hence the name "discontinuous Galerkin" methods) at the cell borders $\{x_i\}$, since $\tilde{f}(x_i) = \sum_{l=0,M} a_{i,l} \equiv f_i^-$ in the cell C_i and $\tilde{f}(x_i) = \sum_{l=0,M} a_{i+1,l}(-1)^l \equiv f_i^+ \neq f_i^-$ in the cell C_{i+1}.

Let us come back to the solution of the advection equation, Eq. (23), and suppose that the initial condition is the approximate function $\tilde{f}(x, 0) = \sum_{i,l} a_{il}(0) \phi_{i,l}(x) \in V_{N,M}$; then its solution at time t is $\exp\{t\Lambda\} \tilde{f}(x, 0)$, which, in general, does not belong to $V_{N,M}$ but may be projected (cf. Eq. (26)) onto this space to give the approximate solution

$\tilde{f}(x,t) \equiv \tilde{T}^t(v, \Delta x, t)\tilde{f}(x,0)$, with components

$$a_{i,l}(t) = \frac{2}{\Delta x c_l^2} \int \phi_{i,l}(x) \exp\{t\Lambda\} \tilde{f}(x,0)\, dx = \sum_{j=1}^{N} \sum_{k=0}^{M} A_{j,k}^{i,l}\, a_{j,k}(0), \tag{28}$$

which is the analogue of Eq. (12) in the simple case of the 1d advection equation. The matrix elements of the projection $\tilde{T}^t(v, \Delta x, t)$ of the displacement operator $\exp\{t\Lambda\}$ on the space $V_{N,M}$ are given by elementary spatial integrations:

$$A_{j,k}^{i,l} = \frac{2}{\Delta x c_l^2} \int_0^L \phi_{i,l}(x)\phi_{j,k}(x - vt)\, dx. \tag{29}$$

Note that $\tilde{T}^t(v, \Delta x, t)$ depends only on the "signed" Courant number, $vt/\Delta x$. The explicit expression of the matrix elements can be found in Van Leer [20] for the case $M = 1$ and in Appendix III of the present paper for the cases $M = 1, 2$. The scheme given in Eq. (28) advances in time the $N * (M + 1)$ data, including the cell averages $\{\bar{f}_i\}$ as well as the local derivatives $\{\partial^k f_i/\partial x^k\}$ for $k = 1, M$ and its time accuracy $\sim O(\Delta t^{M+1})$, and is entirely determined by the order M of the projection.

Some important properties of this scheme are worth mentionning.

1. It is "local" in the sense that at a given time, the data $\{a_{i,l}\}_{l=0,M}$ depend only on the properties of the function to be approximated within the cell C_i, a fact which contributes considerably to the simplicity of the calculations.

2. If $vt/\Delta x = n$, n being a positive or negative integer, then $A_{j,k}^{i,l} = \delta_{j-(i+n)}\delta_{k,l}$, meaning that the approximate solution is simply displaced by n cells without deformation. Therefore the operator $\tilde{T}^t(v\Delta t/\Delta x)$ can always be split into a simple translation by an integer number of cells followed by a more "diffusive" displacement corresponding to the noninteger part of the Courant number. This property allows, in principle, use of large time steps (see Fijalkow [12]) but actually this is limited by the time accuracy which is required when using the splitting method.

3. The "mass,"

$$Q(L,t) = \int_0^L f(x)\, dx = \sum_{i=1,N} a_{i,0},$$

is exactly conserved since for $v \geq 0$ (the same arguments apply for negative velocities)

$$\begin{aligned} Q(L,t) &= \frac{1}{\Delta x} \sum_{i=1,N} \sum_{k=0,M} a_{j,k}(0) \int_{C_i} \phi_{i,0}(x)\phi_{j,k}(x - vt)\, dx \\ &= \frac{1}{\Delta x} \sum_{k=0,M} a_{j,k}(0) \int_0^L \phi_{j,k}(x - vt)\, dx = \frac{1}{\Delta x} \sum_{k=0,M} a_{j,k}(0) \int_0^L \phi_{j,k}(x)\, dx \\ &= Q(L,0) \end{aligned}$$

by using the assumption of periodicity and the fact that the cells cover the whole interval $[0, L]$, so that $\sum_{i=1,N} \int_0^L H_i(x)g(x)\, dx = \int_0^L g(x)\, dx$.

4. The operator $\tilde{T}^t$ keeps some properties of the displacement operator $\exp\{t\Lambda\}$; for example a very useful symmetry can be obtained if one makes the change of variable $y = x - vt$ in the expression (29) of the matrix elements and uses the periodicity assumption

$$c_l^2 A_{j,k}^{i,l}(vt/\Delta x < 0) = c_k^2 A_{i,l}^{j,k}(vt/\Delta x \geq 0).$$

Therefore, to calculate the matrix elements $A_{j,k}^{i,l}$, it is necessary to calculate only those corresponding to positive velocities.

After elementary manipulations, using in particular the periodicity assumption, we obtain

$$\langle \tilde{T}^{-t}\tilde{g}, \tilde{f}\rangle = \langle \tilde{g}, \tilde{T}^t \tilde{f}\rangle$$

for any couple of functions $\tilde{f}$ and $\tilde{g}$ of $V_{N,M}$. This can be applied now to the case where $\tilde{g} = \tilde{T}^t \tilde{f}$, showing that

$$\|\tilde{T}^t \tilde{f}\|_{N,M}^2 = \langle \tilde{f}, \tilde{T}^{-t}\tilde{T}^t \tilde{f}\rangle$$

with respect to the $V_{N,M}$ norm. Since $\tilde{T}^{-t}$ is not the inverse of $\tilde{T}^t$, except when $|vt/\Delta x|$ is an integer, the norm of the approximate solution does not remain constant. Actually the operator $\tilde{T}^{-t}\tilde{T}^t$ is "contracting," as illustrated by the simple case when $M = 0$ and only cell averages $\bar{f}_i$ are considered; in that case, $\bar{f}_i(t) = (1-\delta)\bar{f}_i(0) + \delta \bar{f}_{i-1}(0)$ for $i = 1, \ldots, N$ and

$$\begin{aligned}
\|f(t)\|_{N,0}^2 &= \Delta x \sum_{i=1,N} [(1-\delta)\bar{f}_i(0) + \delta \bar{f}_{i-1}(0)]^2 \\
&= \Delta x \sum_{i=1,N} \left[(1-\delta)^2 \bar{f}_i^2 + 2\delta(1-\delta)\bar{f}_i \bar{f}_{i-1} + \delta^2 \bar{f}_{i-1}^2\right] \\
&= \|f\|_{N,0}^2 + 2\delta(1-\delta)\Delta x \sum_{i=1,N} \bar{f}_i(\bar{f}_{i-1} - \bar{f}_i) \\
&= \|f\|_{N,0}^2 - \delta(1-\delta)\Delta x \sum_{i=1,N} (\bar{f}_{i-1} - \bar{f}_i)^2.
\end{aligned}$$

Since $0 \leq \delta(1-\delta) \leq 1/4$ for $0 \leq \delta \leq 1$, $\|f(t)\|_{N,0}^2$ is smaller than $\|f\|_{N,0}^2$ and the difference is smaller than the expression

$$0.25\Delta x \sum_{i=1,N} (\bar{f}_{i-1} - \bar{f}_i)^2,$$

which is fully determined by the initial jumps of $\bar{f}$ at the cell borders, a result similar to that obtained by Cockburn [19], who used the differential form of the advection equation.

Up to now, the coefficients $a_{i,l}$ have been considered independent quantities. However, they can be expressed in terms of the $\{\bar{f}_j\}$ at any desired order of accuracy. This operation may be understood as a projection from the function space $V_{N,M}$ onto a subspace of smaller dimension N. Following an argument given by Harten [21], we may proceed as follows. The first step is to notice that the cell averages $\bar{f}_i$ are the values at the cell centers $\bar{x}_i$ of the function $\bar{f}(x)$ obtained by smoothing the function f on a sliding box of width Δx:

$$\bar{f}(x) = \frac{1}{\Delta x}\int_{-\Delta x/2}^{\Delta x/2} f(x+x')\,dx' = \frac{1}{2}\int_{-1}^{1} f\left(x + \xi\frac{\Delta x}{2}\right) d\xi.$$

The derivatives are also obtained by smoothing the corresponding derivative of f:

$$\frac{\partial^k \bar{f}}{\partial x^k} = \frac{1}{\Delta x} \int_{-\Delta x/2}^{\Delta x/2} \frac{\partial f(x+x')}{\partial x'}\, dx'.$$

A Taylor expansion shows that

$$D^r \bar{f}(x) = \sum_{k=0,\ldots} \beta_{l,k} D^{r+2k} f(x), \tag{30}$$

where the $\alpha_{l,k}$ are numerical coefficients. This expression is very similar to (27) of the coefficients $a_{i,l}$ in terms of the derivatives of the function f. The next step is to truncate these expansions at order M, i.e., to assume that $D^{M+1} f(x) \simeq 0$, and to eliminate the remaining derivatives $D^{2k+1} f(x)$ between Eqs. (27) and (30) to obtain a system relating the $\{a_l\}$ and $\{D^l \bar{f}\}$. For example at order $M = 2$,

$$a_0(x) = \bar{f}(x), \quad a_1(x) = \frac{1}{2} D\bar{f}(x), \quad a_2(x) = \frac{1}{4} D\bar{f}(x).$$

Using standard formulas for the discrete derivatives, one obtains to second order in Δx (Fromm [22], Van Leer [20], Fijalkow [12])

$$a_{i,0} = \bar{f}_i, \quad a_{i,1} = \frac{1}{4}[\bar{f}_{i+1} - \bar{f}_{i-1}] \tag{31}$$

and

$$\begin{aligned} a_{i0} = \bar{f}_i, \quad a_{i1} &= \frac{1}{24}(\bar{f}_{i-2} - 8\bar{f}_{i-1} + 8\bar{f}_{i+1} - \bar{f}_{i+2}), \\ a_{i2} &= \frac{1}{24}(\bar{f}_{i-2} - 8\bar{f}_{i-1} + 8\bar{f}_{i+1} - \bar{f}_{i+2}) \end{aligned} \tag{32}$$

to fourth order. Then Eq. (28) reduces to the simple form ($\delta = |v\Delta t/\Delta x|$)

$$\bar{f}_i(t + \Delta t) = \sum_{j=-(M-1)}^{j=M} A_j(\delta) \bar{f}_{i+j}(t) \tag{33}$$

for positive velocities, or

$$\bar{f}_i(t + \Delta t) = \sum_{j=-M}^{j=M+1} A_j(\delta) \bar{f}_{i-j}(t) \tag{34}$$

for negative velocities. To second order in $\Delta x (M = 1)$, Eqs. (33) and (34) are known under the name of "Van Leer's scheme" and widely used in fluid dynamics (see, for example, Godlewski and Raviart [17]) and plasma physics (see, for example, Fijalkow [12]). The expressions for the coefficients $A_j(\delta)$ are given in Appendix III for the cases $M = 1$ and $M = 2$, and we designate by "VL2" and "VL3" the corresponding numerical schemes.

3.3.2. Hermite interpolation. Consider now interpolants $\tilde{f}(x)$ that are smooth, i.e., continuous, as are their derivatives up to some order m. The natural function spaces to consider are then the "smooth Hermite" spaces H_{NM} (see Ciarlet *et al.* [23]) with $M = 2m + 1$. Now $m + 1$ data $f_i, \ldots, f_i^{(m)}$ (the values of the function f and its first m derivatives) are given at the cell boundaries x_i and we use the fact that for each cell C_i there is a unique interpolation polynomial $Q_i(x)$ of degree $M = 2m + 1$ such that

$$D^k Q_i(x_{i-1}) = f_{i-1}^{(k)}, \quad D^k Q_i(x_i) = f_i^{(k)}, \quad 0 \le k \le m,$$

where, as above, $D = \Delta x \partial/\partial x$. The interpolating function $\tilde{f}(x)$, defined on the whole interval $[0, L]$ and equal to $Q_i(x)$ on the cell C_i, is then continuous and differentiable up to order m. The function space H_{NM} generated by all possible sets of data $\{f_i, \ldots, f_i^{(m)}\}$ is a linear vector space of dimension $(M + 1) * (N + 1)/2$ (almost one half smaller than) V_{NM} and contains functions which are smoother than those of $V_{N,M}$. An important point is that there is also a convenient basis $\{h_{i,k}(x)\}$ of H_{NM} which is made of piecewise polynomial functions which vanish, as do all their derivatives, outside the interval $[x_{i-1}, x_{i+1}]$ for $1 \le i \le N - 1$ and satisfy

$$D^l h_{i,k}(x_j) = \delta_{k,l}\delta_{i,j}, \quad 0 \le k, l \le m. \tag{35}$$

The corresponding basis $h_{i,k}^*$ of the dual space is easily seen to be $h_{i,k}^* = D^k\delta(x - x_i)$. Therefore,

$$\tilde{f}(x) = \sum_{i=0}^{N}\sum_{l=0}^{m} f_j^{(l)} h_{i,1}(x), \quad D^k\tilde{f}(x) = \sum_{j=0}^{N}\sum_{l=0}^{m} f_i^{(l)} D^k h_{i,1}(x), \quad 1 \le k \le m. \tag{36}$$

The basis $h_{i,k}$ is not orthogonal since the support of $h_{i,k}(x)$ overlap with that of $h_{i-1,k}(x)$ and $h_{i+1,k}(x)$ but its dual $h_{i,k}^*$ is particularly simple so that the anologue of Eq. (26) is trivial, since it is the values $\{f_i, \ldots, f_i^{(m)}\}$ of the function and its first m derivatives which appear in the expansion of Eq. (36). This can be used to obtain the discretization $\tilde{T}^t$ of the displacement operator in the H_{NM} space, since

$$\exp\{-t\Lambda\}\tilde{f}(x) = \sum_{j=0}^{N}\sum_{l=0}^{m} f_j^{(l)}(0) D^k h_{j,1}(x - vt)$$

can be projected onto the space H_{NM},

$$f_i^{(l)}(t) = \sum_{j=0}^{N}\sum_{k=0}^{m} A_{j,k}^{i,l} f_j^{(k)}(0),$$

with the matrix elements of $\tilde{T}^t$ being given by

$$A_{j,k}^{i,l} \equiv D^l h_{j,k}(x_i - vt). \tag{37}$$

The "cubic interpolated propagation scheme" proposed by Nakamura and Yabe [24] corresponds to the case $M = 3(m = 1)$. These authors have shown also that the scheme conserves the total mass, an argument which applies in the more general formulation given

here. The details for the case $M = 3$ are given in Appendix III, corresponding to a scheme which is of third order in the time step; in what follows this scheme is designated by the abreviation "HMN."

3.3.3. Spline interpolation. In the Hermite interpolation, the $(N + 1)(m + 1)$ quantities $f_i, \ldots, f_i^{(m)}$ are considered as independent. One may increase the regularity requirements by asking that the derivatives of the interpolant be continuous up to order $2m$; this means that m linear relations have to be satisfied at the cell boundaries (we consider here the periodic case) so that the derivatives $f_i^{(1)}, \ldots, f_i^{(m)}$ can be expressed in terms of the values f_i of the function (see Ciarlet *et al.* [23], Rubin and Khosla [25]). This corresponds to a projection of the space H_{NM} onto a much smaller subspace, the spline subspace S_{NM} of dimension N of very regular piecewise polynomial functions (of degree at most $M = 2m + 1$). In this space the so-called B-spline functions $\{B_i(x)\}$ provide a basis; their $2m$th derivative is a continuous piecewise linear function (see Ciarlet *et al.* [23]) and they have a minimal support covering $M + 1$ cells. However this basis and the associated one in the dual space do not allow a simple and direct determination of the coefficients of the development of the interpolant $\tilde{f}$ in terms of B-splines; in all cases, a linear system of equations needs to be solved to determine these coefficients. The cubic spline interpolation ($m = 1$, $M = 3$) has been used by a number of authors (see, for example, [26]) providing a scheme which is third-order accurate in space but requires the solution of a linear, tridiagonal system of equations; the details are presented in Appendix III and the corresponding scheme is designated in what follows by the abreviation "SPL."

3.4. *Semi-Lagrangian Method*

Another possible approach for the solution of the advection equation is obtained by integrating Eq. (23) over the cell C_i,

$$\frac{\partial \bar{f}_i}{\partial t} + \frac{v}{\Delta x}[f(x_{i+1}, t) - f(x_i, t)] = 0,$$

leading to an equation which relates the time variation of the cell averages, $\{\bar{f}_i\}_{1 \le i \le N}$, to its local values at the cell boundaries, $\{f_i\}_{0 \le i \le N}$, i.e., at the grid points x_i. Then, integration over the time interval $[0, t]$ gives

$$\bar{f}_i(t) = \bar{f}_i(0) + \Pi_i^+ - \Pi_i^-, \tag{38}$$

where the time-averaged fluxes across the cell boundaries $\Pi_i^{\pm}$ are given by

$$\Pi_i^+ = -\frac{v}{\Delta x}\int_0^t f(x_{i+1}, \tau)\, d\tau, \quad \Pi_i^- = -\frac{v}{\Delta x}\int_0^t f(x_i, t + \tau)\, d\tau. \tag{39}$$

Equation (38) is still an exact one; however, it mixes the cell averages $\bar{f}_i$ and time integrals of the distribution function at the cell boundaries. Two steps are necessary to calculate approximate values of these integrals in term of the cell averages. First, Eq. (23) implies that f is constant along a characteristic curve $x - v\tau = const.$ (hence the often-used name "semi-Lagrangian"). For positive velocities, $v \ge 0$ (the same kind of arguments applies to negative velocities), and sufficiently small time intervals t, the characteristic arriving at the point x_{i+1} at time $0 \le \tau \le t$ comes from a point $x_+ = \bar{x}_i + (\Delta x/2 - v\tau)$

inside the cell C_i while the characteristic arriving at the point x_i at the same time comes from a point $x_- = \bar{x}_{i-1} + (\Delta x/2 - v\tau)$ of the cell C_{i-1}. Therefore, the first step is to evaluate $f(x_-, 0)$, $f(x_+, 0)$ by any of the interpolation schemes described above, for example the discontinuous Galerkin approximation,

$$\tilde{f}_M(x_+) = \sum_{l=0}^{M} a_{i,l}(0)\phi_{i,l}(x_+), \quad \tilde{f}_M(x_-) = \sum_{l=0}^{M} a_{i-1,l}(0)\phi_{i-1,l}(x_-).$$

The next step is to integrate over τ to get the time-averaged fluxes; using as above the notation $\delta = |vt/\Delta x|$, one obtains immediately a relation between the "numerical fluxes" and the matrix elements $A^{i,0}_{i,l}$ of the projection $T_V(t)$ of the displacement operator

$$\Pi_i^+ = \left(A^{i,0}_{i,0}(\delta) - 1\right) a_{i,0}(0) + \sum_{l=1}^{M} A^{i,0}_{i,l}(\delta) a_{i,l}(0) \tag{40}$$

and

$$\Pi_i^- = -\sum_{l=1}^{M} A^{i,0}_{i-1,l}(\delta) a_{i-1,l}(0) \tag{41}$$

for a positive velocity $v \geq 0$.

4. OPEN BOUNDARY CONDITIONS

In the previous section, we assumed that the system is spatially periodic. However, there are a number of interesting problems where open boundaries are necessary and we discuss in the following a possible strategy to insert open boundary conditions in the Vlasov equation when the system is finite in one direction and periodic in all other space directions. The corresponding conditions on the electrostatic or the electromagnetic field are discussed in Appendix II.

We consider the one-dimensional situation where the numerical domain is open at both ends $x = 0$ and $x = L_x$ and we limit our analysis to the case of a constant flux of "entering" particles at both ends, the extension to the time-dependent case being straightforward. We have then to impose the following boundary conditions:

$$\begin{aligned} vf^{>}(v) &= \textit{given for} \quad v \geq 0 \quad \textit{at } x = 0, \\ vf^{<}(v) &= \textit{given for} \quad v \geq 0 \quad \textit{at } x = L_x. \end{aligned} \tag{42}$$

The most natural way to do that is to use the "flux" formulation. Consider a specific boundary, say $x = 0$, which is also the left boundary of the cell C_1, and positive velocities $v \geq 0$. In this case, the time-averaged flux Π_1^+ across the right cell boundary is given by the dynamics in the interior of the numerical domain,

$$\Pi_1^+(v) = -\frac{v}{\Delta x} \int_0^{\Delta t} d\tau \{f(\Delta x, v, t + \tau)\},$$

while the flux Π_1^- "transported" across the boundary $x = 0$ by a characteristic coming from outside the numerical domain represents the influence of the "outside world" and must be set to

$$\Pi_1^-(v) = -\delta f^>(v), \quad v > 0, \tag{43}$$

with $\delta = |v\Delta t/\Delta x|$. At the other boundary, the cell C_N, and negative velocities $v < 0$, the flux $\Pi_N^-(v)$ is fixed by the interior while

$$\Pi_N^+(v) = \delta f^<(v), \quad v < 0. \tag{44}$$

Besides the modification of the boundary fluxes, nonperiodic boundary conditions will affect the truncatures (31) and (32) since the numerical expressions of the spatial derivatives must be modified to use only points in the interior of the domain (i.e., a noncentered formula),

$$D\bar{f}_1 \cong \frac{1}{2}(-\bar{f}_3 + 4\bar{f}_2 - 3\bar{f}_1)$$

in the cell C_1 and

$$D\bar{f}_N \cong \frac{1}{2}(-\bar{f}_N + 4\bar{f}_{N-1} - 3\bar{f}_{N-2})$$

in the cell C_N.

In summary, the advancing scheme, Eqs. (33) and (34), must be somewhat modified in the case of open boundary conditions and, using the notation $\{\bar{f}\} = \{\bar{f}_1, \bar{f}_2, \ldots, \bar{f}_i, \ldots, \bar{f}_N\}$, may be cast in the compact matrix form

$$\{\bar{f}\}(t + \Delta t) = \mathbf{A}^*(\delta)\{\bar{\mathbf{f}}\} + \mathbf{B}(\delta). \tag{45}$$

In this equation, the boundary flux $\mathbf{B}$ is given by

$$\mathbf{B} = (-\Pi_1^-, 0, 0, \ldots, 0), \quad v > 0, \quad \mathbf{B} = (0, 0, 0, \ldots, \Pi_N^+(v)), \quad v \le 0,$$

while the matrix $\mathbf{A}^*$, when $M = 1$, corresponds exactly to the formula (33)

$$A^*_{i,j} = A_{j-i}(\delta), \quad v > 0, \quad A^*_{i,j+1} = A_{i-(j+1)}(\delta), \quad v \le 0, \tag{46}$$

inside the numerical domain, i.e., for $2 < i < N - 1$, $j = i - 2, i + 1$; the matrix elements $A^*_{1,j}$, $A^*_{2,j}$, $A^*_{N,j}$ corresponding to the boundary cells differ and are given in Appendix III. Despite these modifications the matrix $\mathbf{A}_*$ keeps some symmetry,

$$A^*_{i,i-l}(v < 0) = A^*_{i,i+l}(v > 0), \qquad l = -2, 1, \quad i = 3, N - 2; \tag{47}$$

$$A^*_{1,1+l}(v < 0) = A^*_{N,N-l}(v > 0), \qquad l = 0, 2; \tag{48}$$

$$A^*_{N-k,N-k-l}(v < 0) = A^*_{1+k,1+k+l}(v > 0), \quad k = 0, 1, \quad l = 0, 2.$$

If one multiplies the advection equation (23) by f and integrates over space and time one obtains an equation describing the conservation of the quantity $I_2 = \|f\|^2 = \int_0^L f^2(x', t)\,dx'$,

$$I_2(\Delta t) = I_2(0) - v\int_0^{\Delta t} d\tau\{f^2(x = L, \tau) - f^2(x = 0, \tau)\},$$

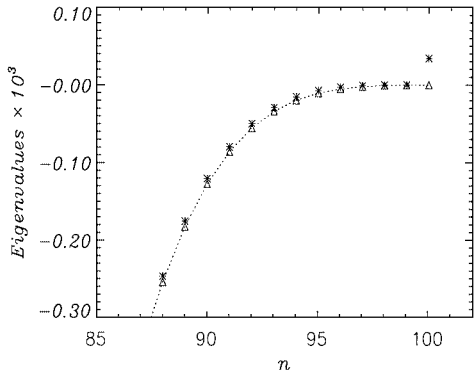

FIG. 1. The largest eigenvalues of the matrix $A^*(-v)A^*(v)$ (stars) and of $A^{**}(-v)A^{**}(v)$ (triangles). The parameters which have been used are $N = 100, \delta = 0.05$ and $\Delta t/t_{rel} = 0.1$.

whose discretized form is

$$\|f\|^2 = \langle f(0), A^*(-v)A^*(v)f(0)\rangle + \cdots \textit{boundary terms}.$$

The second term corresponds to the boundary terms and depends only on the values of f at $\bar{x}_1, \bar{x}_2$, and $\bar{x}_N$. The important point is that in the periodic case the first term on the RHS is always smaller than $\|f(0)\|^2$, expressing the stability and dissipation of the scheme. In the open case this is no longer true. Indeed, the eigenvalues of the matrix $A^*(-v)A^*(v)$ are displayed in Fig. 1 for the case when $\delta = 0.05$ and $N = 100$; they are real, positive and smaller than 1 except for the largest one, which lies slightly above 1, indicating the possibility of unstable behavior if the initial condition has a significant amplitude on the corresponding eigenvector, which is strongly localized close to the boundaries. Therefore, the scheme given by Eq. (45) is not unconditionally stable.

A way to improve the stability (see, for example, Abarbanel and Chertock [27]) consists of modifying the incoming fluxes Π_1^- and Π_N^+ (see Eqs. (43) and (44)) by

$$\Pi_1^- = \frac{v(\bar{f} - f^>)}{t_{rel}}, \quad \Pi_N^+ = \frac{-v(\bar{f}_N - f^<)}{t_{rel}},$$

expressing a relaxation of the border cell average fluxes toward the prescribed one. If t_{rel} is chosen adequately, then the scheme is stable. Indeed, to these expressions of the fluxes corresponds a new matrix A^{**}; the eigen values of $A^{**}(-v)A^{**}(v)$ are plotted in Fig. 1 for the same values as above and $\Delta t/t_{rel} = 0.1$. It may be seen that these eigenvalues are all smaller than 1 and differ very little from those of $A^*(-v)A^*(v)$ except for the unstable one.

5. COMPARISON BETWEEN VARIOUS SCHEMES

In this section we compare the performances of several of the schemes discussed above for solving the advection equation, Eq. (23): the standard second-order Van Leer scheme, VL2; the third-order Van Leer, VL3; the Hermite interpolation scheme, HMN; and the spline scheme, SPL.

It should be mentioned that a comparison, in the electrostatic limit, between the Van Leer scheme and cubic splines interpolation method was carried out by Sabatier *et al.* [28], who

found, for the same number of grid points, that cubic splines conserve better the energy of the system at the expense of a longer computational time (more or less a factor of three). This comparison was done for the full Vlasov–Poisson system; here we limit ourselves to the advective part in 1d with periodic boundary conditions where an exact solution of the advection equation (23) can be obtained in Fourier space,

$$f_k^{ex}(t) = e^{-ikvt} f_k(t=0), \tag{49}$$

where $f_k(t=0)$ is the Fourier transform of the initial distribution $f(x, t=0)$. Using the same initial condition and a large number of iterations ($t = N_{iter}\Delta t$, $N_{iter} \simeq 10^4$), the numerical solution at time t obtained by a particular scheme is Fourier transformed in space, giving the Fourier components $\tilde{f}_k(t)$; we calculate then the complex ratio $G_k(t)$ between the exact and approximate solution:

$$\tilde{f}_k(t) = G_k(t) f_k^{ex}(t). \tag{50}$$

The numerical damping rate and dispersion are given by $\gamma(k) = \log(\|G_k\|)/t$ and $\phi = d(\arg(G_k))/t$, respectively. In Figs. 2 and 3 we plot, for several values of the courant number $v_c = |v\Delta t/\Delta x|$, γ and ϕ as functions of $k/k_{\max}$, where the maximum wave number is given by $k_{\max} = \pi N_x/L_x$. In all the frames of these figures, the four curves correspond to the schemes SPL (solid line), HMN (dashed line), VL3 (dashed–dotted line), VL2 (dashed–three dotted line). For all cases, we used the same number of mesh points, $N = 128$, the time step $\Delta t = 0.05$, and the mesh size $\Delta x = 0.1$.

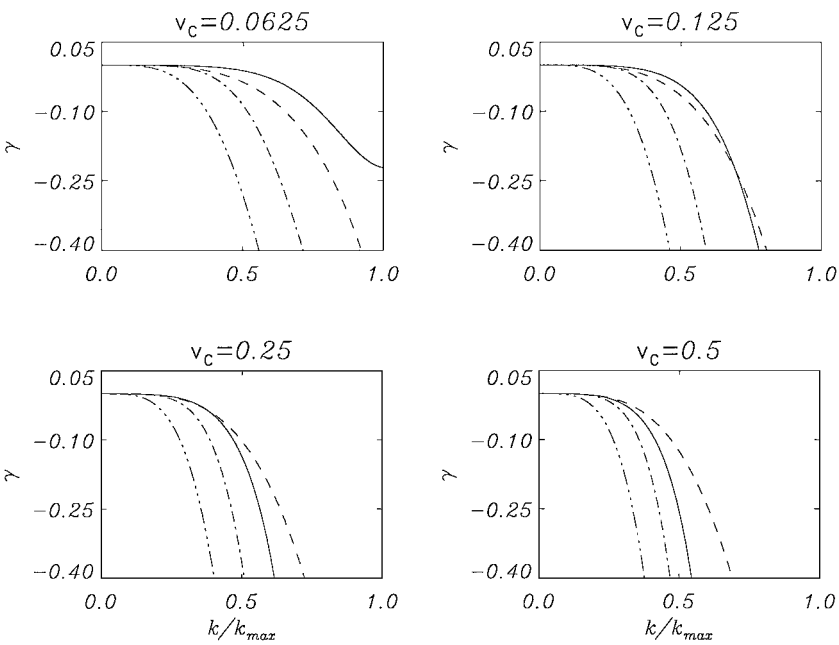

FIG. 2. The numerical damping rate vs $k/k_{\max}$ for the different schemes: SPL (solid line), HMN (dashed line), VL3 (dashed–dotted line), VL2 (dashed–three dotted line). The four frames correspond to different values of the Courant number, $v_c = |v\Delta t/\Delta x|$.

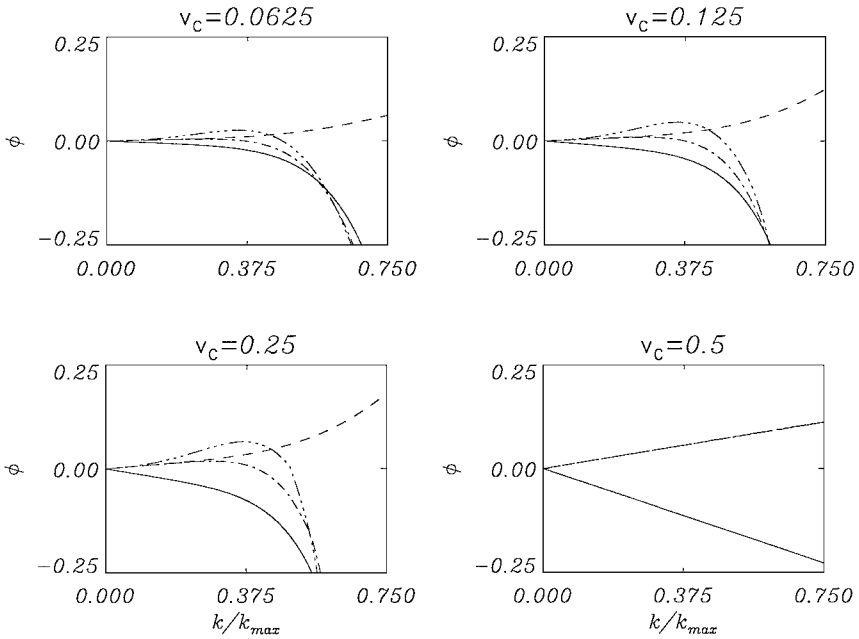

FIG. 3. The numerical dispersion vs $k/k_{\max}$ for the different schemes, SPL (solid line), HMN (dashed line), VL3 (dashed–dotted line), VL2 (dashed–three dotted line). The four frames correspond to different values of the Courant number, $v_c = |v\Delta t/\Delta x|$. No numerical dispersion corresponds to $\phi = 0$.

It is seen that in all cases $\gamma \simeq 0$ for $k/k_{\max} \leq 0.1$. The second-order Van Leer scheme, VL2, has the strongest damping rate, as expected for the lowest order scheme. For small Courant numbers the spline method, SPL, is the least damped, while for Courant numbers greater than 0.2 it is the Hermite interpolation, HMN, which performs the best.

On the other hand, Fig. 3 shows that the least dispersive scheme is always the HMN, probably because the first derivative is also advanced in time. We also observe a degradation of the phase accuracy of the SPL scheme when increasing the Courant number. Surprisingly, the second-order VL2 scheme performs as well as third-order schemes.

The results presented in these figures were obtained with the same number of grid points; therefore the computational times used by the different schemes differ strongly. It would be more meaningful to present this comparison for the same computational effort, even if in this case the memory requirement is different. This is not easy since the computational time depends on several parameters, such as the degree of optimization and the machine used for the calculations. We have chosen to use a different number of grid points for the different schemes corresponding roughly to the same CPU time. In particular, for the SPL method, which requires the inversion of a tridiagonal matrix, we estimated, for the same number of grid points, $T_{CPU}(\text{SPL}) \simeq 3T_{CPU}$ (VL2) (see Appendix V for the details), in agreement with the estimation given in Ref. [28]. Pohn *et al.* [29] find only a 10% increase in the computational time of SPL with respect to VL2; we underline the fact that our computational test is limited to the pure 1d advection equation with constant velocities, while Refs. [28] and [29] consider the full Vlasov equation in the 1d–1V and 1d–3V phase space, respectively.

The results of the comparison between the VL2, VL3, HMN, and SPL with 180, 120, 90, and 60 grid points, respectively, are shown in Figs. 4 and 5. All the other parameters have

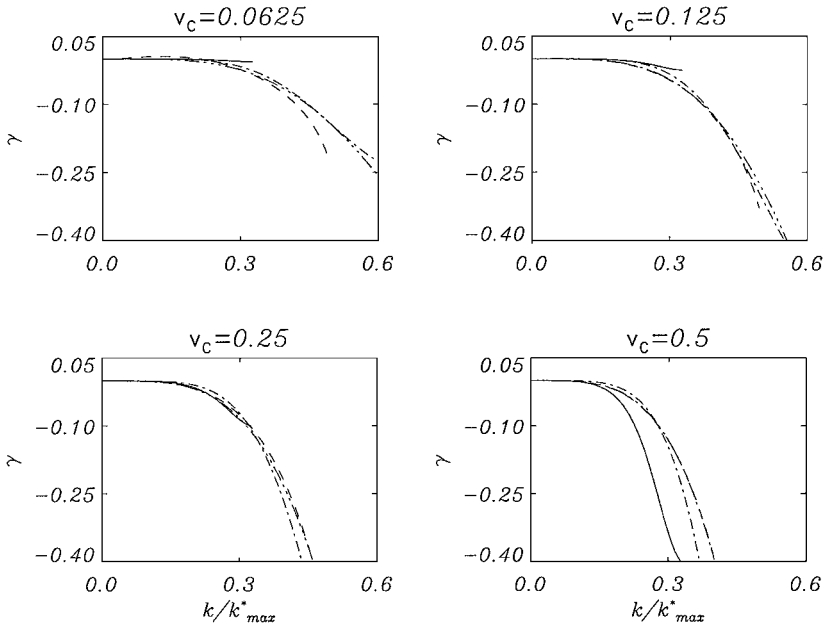

FIG. 4. The same as Fig. 2 with $N_x = 180$ 120, 90, and 60 for the VL2, VL3, HMN, and SPL schemes, respectively. Here $k^*_{\max}$ is the maximum wave vector of the VL2 case.

not been changed. In these figures, the wave vector is normalized to the maximum wave vector $k^*_{\max}$ of the VL2 case.

Now the damping rates are almost identical; on the other hand, we observe that in phase precision the VL2 scheme performs best while the SPL scheme is the most dispersive one.

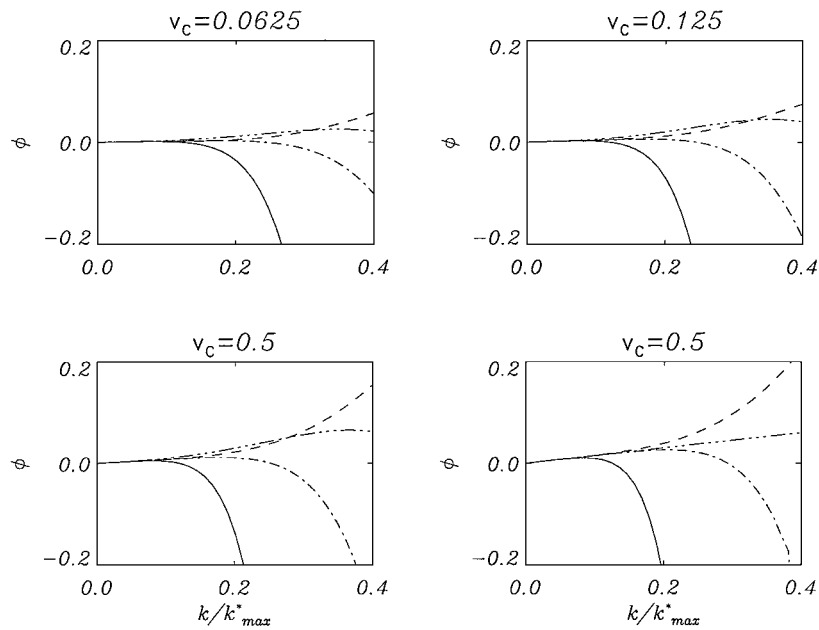

FIG. 5. The same as Fig. 3 with $N_x = 180$, 120, 90, and 60 for the VL2, VL3, HMN, and SPL schemes, respectively. Here $k^*_{\max}$ is the maximum wave vector of the VL2 case.

This is probably a direct consequence of the fact that the number of mesh points in the SPL method is significantly lower than in the other schemes.

It is worth mentioning that in all cases the total mass and energy are very well conserved. This is true also for the second- and third-order invariants $I^n = \int f^n \, dx \, dv$, $n = 2, 3$.

In conclusion, there is no compelling argument for the choice of a particular scheme; we just note that the SPL scheme requires less memory than the other schemes for the same total computational time, but at the expense of the phase precision. Furthermore, our test here is limited to the advection equation, while a full comparison between the different schemes should be made on the full Vlasov equation in the electrostatic and electromagnetic case, with particular attention on the behavior of the invariants of the system and on the small-scale features, as, for example, phase space filamentation.

6. PARALLELIZATION OF THE ALGORITHM

As discussed in the Introduction, the computational effort required to solve the Vlasov–Maxwell system of equations, Eqs. (4)–(6), is significant and today such simulations could only be afforded on large parallel supercomputers. To exploit the full computational power of these machines a scalable parallel algorithm for the resolution of the equations must be implemented. In this section we discuss the parallelization strategy we have chosen to follow and the structure of the resulting code, using in particular "metalanguage" schemes of the key parallel routines.

The most demanding equation in terms of computational resources is by far the Vlasov equation; therefore we limit our discussion to this equation for the case of a 5d phase space (x, y, v_x, v_y, v_z), where (x, y) are the spatial coordinates in a Cartesian reference frame and (v_x, v_y, v_z) the velocity coordinates. Indeed, at the moment the full 6d case, while being in principle a straightforward extension of the 5d one, could be solved only with a numerical resolution too small to be of interest to plasma physics research. Only one particle species is considered: periodic boundary conditions are used in the physical plane (x, y), while in the velocity space the distribution function is set to zero outside the numerical domain.

6.1. *Parallelization Strategy*

For the integration of the Vlasov equation, we use a uniform grid with $N_x \times N_y$ mesh points in physical space and $(2Nv_x + 1) \times (2Nv_y + 1) \times (2Nv_z + 1)$ points in velocity space covering the five-dimensional numerical domain $0 \le x \le L_x$, $0 \le y \le L_y$, $-v_x^{\max} \le v_x \le v_x^{\max}$, $-v_y^{\max} \le v_y \le v_y^{\max}$, $-v_z^{\max} \le v_z \le v_z^{\max}$. The corresponding distribution function $f(n_x, n_y, -Nv_x : Nv_x, -Nv_y : Nv_y, -Nv_z : Nv_z)$ is by far the largest data structure and its time advance uses repeatedly, both for the spatial and the velocity advance, the translation described by Eqs. (33) and (34) ($M = 1$), which corresponds in practice to an interpolation formula.

A simple profilling analysis done on a scalar implementation of the algorithm presented in Section 3 shows that the updating cycle of f takes (for all the translations involved in a time step) almost 90% of the total execution time.

Let us now look in some detail at the algorithm. First of all, we observe that the five-dimensional distribution and the updating algorithm as a whole have an high degree of symmetry with respect to the phase space coordinates; indeed, exactly the same computations

```
FOR jz = −Nv_z, Nv_z
    FOR jy = −Nv_y, Nv_y
        FOR jx = −Nv_x, Nv_x
            propagate along x
            FOR iy = 1, N_y
                FOR ix = 1, N_x
                    g(ix, iy) = Σ_c (A_x(c) * f^t(c))
                END
            END
            propagate along y
            FOR iy = 1, N_y
                FOR ix = 1, N_x
                    f^{t+Δt}(ix, iy, jx, jy, jz) = Σ_c (A_y(c) * g(c))
                END
            END
        END
    END
END
```

FIG. 6. The calculation of $\bar{f}_i$ in Eqs. (33) and (34) ($M = 1$) in the case of the spatial translation operator. Here g is a working array and $A_x(c)$ and $A_y(c)$ are the coefficients described in Section 11.2 (for example, Eq. (65) in the case of positive velocities). The sum index c runs on the neighbor cell.

are required to update the x and y coordinates, as shown explicitly in Fig. 6, and the same applies to to the velocity coordinates. Considering this symmetry property and the large size of the 5d distribution function array, the SPMD (single program multiple data) paradigm has been choosen for the parallelization of the code, so that each processor executes the same computations on its local block of data.

When analyzing the whole time-step cycle, it appears that the velocity space updating is local, at any given point (x_i, y_j) of physical space (i.e., does not require information from any other spatial point) and vice versa for the spatial updating. Furthermore, the spatial and velocity updating cycles are completely independent. On the other hand, the computational weight of the two updating algorithms are different since in the velocity 3d space seven successive translations are required, Eq. (21), while the updating in the 2d physical space requires only four translations, two translations before the updating with respect to the velocity coordinates and two after (see Eq. (22)). Therefore, since each translation along a given direction requires nearby data communication along that direction (see Eqs. (33) and (34) ($M = 1$)), it is more convenient to distribute the real space only.

6.2. *Data Distribution and Communication*

As anticipated the SPMD parallelization strategy requires the data to be equally distributed among processors; therefore the two-dimensional space grid points have been partitioned into rectangular blocks (the number of blocks is equal to the number of processors) and distributed blocklike across the processors logically arranged on a 2d mesh, with each block being assigned to a processor. This particular distribution minimizes the amount of data to be exchanged among the processors, when the block border values of the distribution are to be accessed (the total amount of exchanged data scales as the square root of the number of processors).

The updating algorithm of the distribution function for a given value of the real space coordinates uses the values of the distribution function in the two neighboring cells for

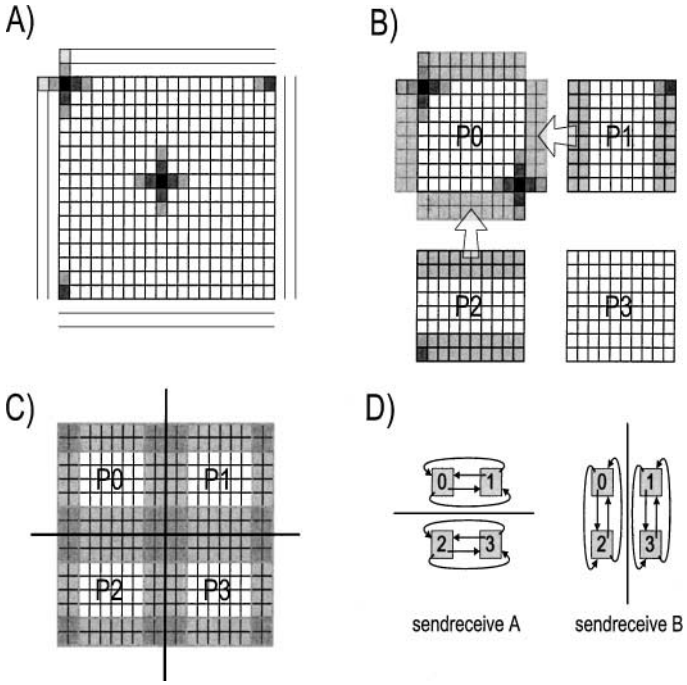

FIG. 7. Data distribution. (A) Discretized real space integration grid. Black squares represent cells being updated, while gray squares are cells used in the updating cycle (see Fig. 6). (B) Real space grid as it is distributed among four processors (P0, P1, P2, and P3). Gray rectangles represent the cells located on processors P2 and P1 and required to update the cells of processor P0. (C) Gray regions, cells communicated among processors. (D) Communication patterns between processors, implemented by the algorithm listed in Fig. 6.

each direction, which may or may not belong to the same block. This is illustrated in Fig. 7 where the physical space discretized on the numerical grid is displayed; panel B shows the partition for a four-processor distribution. Using this data distribution, the updating of the distribution function values is local (no communications required) for the cells with a distance larger than two (cells) from the local block border (i.e., black cell in the middle of the grid of Fig. 7A). The cells in the upper left corner, bottom left side, and upper right side of Fig. 7A illustrate how periodic boundary conditions are implemented.

The updating of the distribution function in a cell close to the border (black cells in Fig. 7B) requires the values of the distribution function stored on other processors (gray rectangles), so that any given processor needs the boundary data only from its neighboring processors in the x and y directions. Figure 7C shows the total area of data communicated in an updating cycle (gray region). Thanks to the block partitioning, the communications in the updating loop along a given direction are overlapped (see Fig. 7D, where processor 0 communicates with processor 1 while processor 2 communicates with processor 3) and it is possible to implement the communications of the block borders values in a single call using a send-and-receive subroutine (i.e., a single call to the MPI (message passing interface) subroutine MPI_SENDRECV). Finally, since data communication occurs only between processors that are logically first neighbors, it is possible on many supercomputing systems to associate a logical processor layout with a physical processor layout, with a consequent increase in the code performance.

```
FOR jz = −Nv_z, Nv_z
    FOR jy = −Nv_y, Nv_y
        FOR jx = −Nv_x, Nv_x
            CALL EXCHANGE_BORDER_VALUES( f )
            propagate along x
            FOR iy = 1, N_{y_local}
                FOR ix = 1, N_{x_local}
                    g_local(ix, iy) = Σ_c (A_x(c) * f^t_local(c))
                END
            END
            CALL EXCHANGE_BORDER_VALUES ( g )
            propagate along y
            FOR iy = 1, N_{y_local}
                FOR ix = 1, N_{x_local}
                    f^{t+Δt}_local(ix, iy, jx, jy, jz) = Σ_c (A_y(c) * g_local(c))
                END
            END
        END
    END
END
```

FIG. 8. Updating parallel loop using MPI library. Here f and g are explicitly distributed among processors. Each processor has a subset of the whole array of dimension ($N_{x_{local}}+4$, $N_{y_{local}}+4$, N_{v_x}, N_{v_y}, N_{v_z}), where $N_{x_{local}}$ and $N_{y_{local}}$ are the (x, y) dimensions of the local block of real space grid (see also Fig. 7B), while the $+4$ accounts for the cell of the blocks borders. A new subroutine performing the borders cells exchange (EXCHANGE_BORDER_VALUES) has been introduced in the code.

6.3. *Algorithm Implementation*

Two different parallel programming scheme, HPF (high performance Fortran) and MPI, have been used to implement the data distribution using the SPMD paradigm.

- HPF. This choice is related to the fact that one of our target platforms was the CRAY T3E with a highly optimized HPF compiler and because only a little recoding is required. The parallelization of the code reduces to the addition of a few HPF directives in the code source and the fortran statements remain unchanged with respect to the scalar code.
- MPI. The choice of the MPI library guarantees an optimal portability, but now a significant recoding is required and the communications must be handled explicitly. The resulting updating cycle is displayed in Fig. 8. To distribute the data grid, the processors have been organized on a two-dimensional logical mesh whose sizes are nprow and npcol (total number of processors used in a given run = nprow ∗ npcol). Each processor is identified by an MPI index (0 . . . nproc-1) and two mesh coordinates (myrow, mycol). On each processor the sizes in the x and y directions of the local block of data are nxl (nxl = nx/nprow) and nyl (nyl = ny/npcol); in this case a small extra amount of storage, with respect to the scalar case, is required in order to store the border cells.

6.4. *Other Data Structures and Input/Output*

All the other relevant data structures, the electric and magnetic fields, and the currents are distributed and parallelized in the same way. The calculations of the distribution function moments (see Eq. (7)) is local and thus perfectly parallelized. Particular attention has been paid to the code input/output (I/O) since the distribution f in a production run could

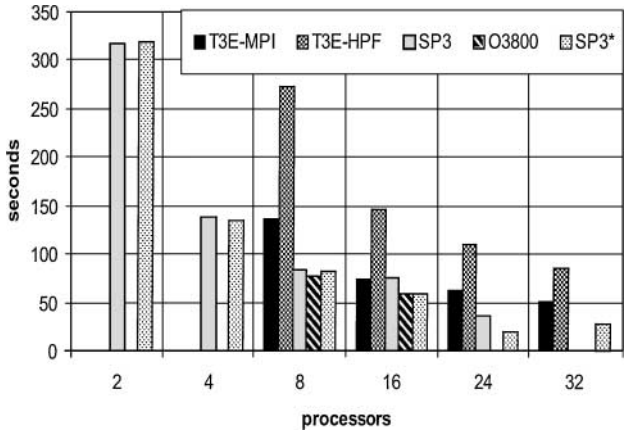

FIG. 9. Execution time, in seconds, of the code for a benchmark on a numerical grid of size $32^2 \times 61^3$ on a Cray T3E (using MPI and HPF), on an IBM SP3 (using MPI), and on an SGI Origin 3800 (using MPI) for different numbers of processors. The SP3* values were obtained by exchanging the communication step with the loop over the N_{vx}.

be as large as tens of gigabytes. Indeed, to save 10 Gbytes at 20 Mbytes per s (typical sustained speed of a fast I/O system) takes 500 s. In the scalar code, the I/O is implemented with a Fortran "write" instruction in a loop over all the elements. This straightforward implementation has very poor performances (less than 10 Mbytes per s), especially on supercomputers file systems where "RAID" technologies are used. We have improved the I/O of the code by writing the distribution f by data blocks of a given size, thus obtaining an increase in the performance up to 200 Mbytes per s.

6.5. *Code Benchmarks*

Two benchmark cases, with different communications/computations ratios, have been performed on a CRAY T3E 1200 with 256 processors, an IBM SP3 (NightawkII) with 128 processors, and an SGI Origin 3800 with 128 processors available at the CINECA supercomputing center (Bologna, Italy). The first benchmark reported in Fig. 9 was done on a numerical grid of size $32^2 \times 61^3$ with a high communications/computations ratio; in the second benchmark (see Fig. 10) the grid size is $64^2 \times 21^3$, with a lower communications/computations ratio.

On the T3E, where both HPF and MPI schemes are allowed, the MPI implementation is, on average, faster than the HPF one due to the hidden syncronizations introduced by HPF. The scalability of the MPI code in the second benchmark, where the communications/computations ratio is low, is good for all the architectures we have tested, while in the first benchmark the MPI code scales only on the T3E. The good performances of the T3E in the first benchmark is mainly due to the lower latency of the T3E network with respect to the networks of the other two machines, since the bandwidth of the three networks are comparable.

On the SP3, to test the relative weight of the latency with respect to the bandwidth, we made a small improvement to the algorithm by exchanging the loop over the first velocity component (v_x) with the cell boundary communications. In this case larger communication buffers are required. The resulting code communicates the same amount of data

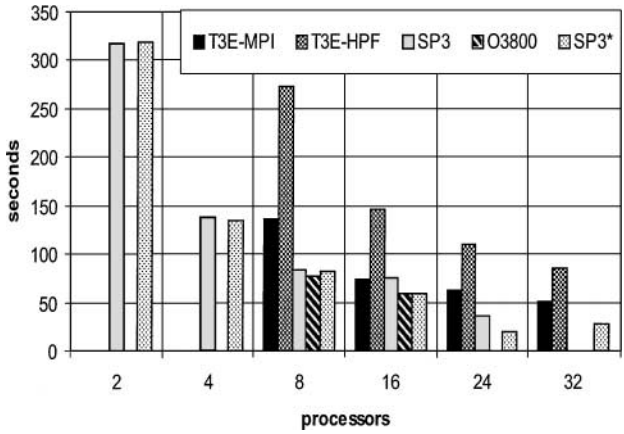

FIG. 10. Execution time, in seconds, of the code for a benchmark on a numerical grid of size $64^2 \times 21^3$ on a Cray T3E (using MPI and HPF), on an IBM SP3 (using MPI), and on an SGI Origin 3800 (using MPI) for different numbers of processors. The SP3* values were obtained by exchanging the communication step with the loop over the N_{vx}.

with fewer syncronization points. The scalability performance of the code with this new algorithm (see Figs. 9 and 10) is significantly better than the original one, confirming that the critical communication parameter in this architecture is the latency and not the bandwidth. Finally, note that for this code the SP3 and Origin 3800 machines have comparable performances.

7. AN EXAMPLE

In this section we discuss a specific example of the evolution of the current filamentation instability (know also as the Weibel instability), which is an electromagnetic plasma instability which combines two important points to be stressed here: (i) a significant 2d–3V test case both for the linear and for the nonlinear regime and (ii) a physical, up-to-date problem in plasma physics research of great interest both for laboratory and astrophysical plasmas. For example, a debate has been recently raised about the role of this instability for the plasma dynamics induced by an ultrastrong, ultrashort laser pulse in an overdense plasma [30].

7.1. *The Current Filamentation Instability*

The current filamentation instability [31, 32] (hereafter CF instability) is a 1d electromagnetic instability driven by the presence of electron momentum anisotropy generated, for example, by the breaking of large-amplitude plasma waves. The importance of this instability is that a dipolar magnetic field is generated during its linear development in the direction perpendicular to the plane of the wavevector and of the electron beams. Recently, this mechanism of evident interest for space plasmas has received much interest in the laser-plasma interaction context in order to explain the intense magnetic field occurring in the wake of an ultraintense, ultrashort laser pulse propagating in an underdense plasma [33]. The physical mechanism of the CF instability, similar to the Weibel instability [34]

in the case of electron temperature anisotropy, can be described as follows. When a beam of fast electrons is generated in the plasma, then a counterpropagating, compenetrating electron beam carried by the cold component of the plasma is immediately produced in order to maintain quasineutrality. Then, the total net current in the plasma is zero, but the two resulting electron currents are unstable since any transversal disturbance is reinforced by the repulsion of the two oppositely directed currents. As a result, the "free" kinetic energy stored in the electron beams can be partly converted into magnetic energy, giving rise to a nonpropagating magnetic field (i.e., the real part of the frequency is zero) which grows exponentially in time. When the speed of the electron beams approaches the velocity of light, the counterfilamentation instability has a growth rate comparable to the electron plasma frequency. Ions can thus be assumed to be immobile, providing a uniform neutralizing background.

By using a standard normal mode approach, the dispersion relation of the 1d CF instability was obtained in Ref. [35] (see Refs. [36] for the 2d case) in the framework of the two fluid electron equations. Here we recollect that in the 1d symmetric nonrelativistic case the dispersion relation is characterized by the fact that the growth rate increases linearly with the wavenumber in the small wavenumbers case (long wavelengths) and saturates when the inverse of the wavenumber becomes comparable to the electron skin depth.

In the following we discuss Vlasov–Maxwell simulations of the evolution of the CF instability driven by two counterstreaming electron beams directed along the z-axis of a Cartesian reference frame perturbed by an initial random noise in the (x, y) plane. No variations are assumed in the stream direction, i.e., $\partial/\partial z = 0$. This configuration has been studied in the relativistic limit in Ref. [37] by using particle-in-cell numerical simulations and recently applied to the problem of energy transport in overdense plasma with applications to the fast ignition concept [38].

7.2. *Numerical Simulations*

We normalize the Vlasov–Maxwell equations by using the speed of the light, the inverse of the electron plasma frequency, and the electron mass as characteristic velocity, time and mass, respectively, and we initialize the Vlasov–Maxwell code in the 2d (x, y) 3V (v_x, v_y, v_z) phase space with the distribution function

$$f_e(t=0) = \frac{1}{(\pi\beta)^{3/2}} e^{-(v_x^2+v_y^2)/\beta} \left[\frac{1}{2} e^{-(v_z-v_0)^2/\beta} + \frac{1}{2} e^{-(v_z+v_0)^2/\beta} \right], \tag{51}$$

where $\beta^{1/2}$ is the thermal velocity and v_0 is the velocity of the electron beams. The two initial electron beams are directed along the v_z direction and the initial density is uniform and equal to one, $\langle n \rangle(x, y, t=0) = 1$. In the simulations discussed here we have taken $\beta = 0.002$ and $v_0 = 0.2$. We have considered two different initial perturbations: (i) a monochromatic one with $\mathbf{E} = 0$ and $\mathbf{B} = \epsilon \sin(k_0 y) e_x$ (here $\epsilon = 10^{-3}$ is the amplitude and $k_0 = 1.0$ the wavevector) and (ii) a white random noises. Equations (1)–(3) are integrated in the (x, y) spatial domain of dimension $(2\pi \times 2\pi)$ with a velocity phase space interval given by $[-0.4, 0.4]$ in all directions on the numerical grid of size $16^2 \cdot 41^3$.

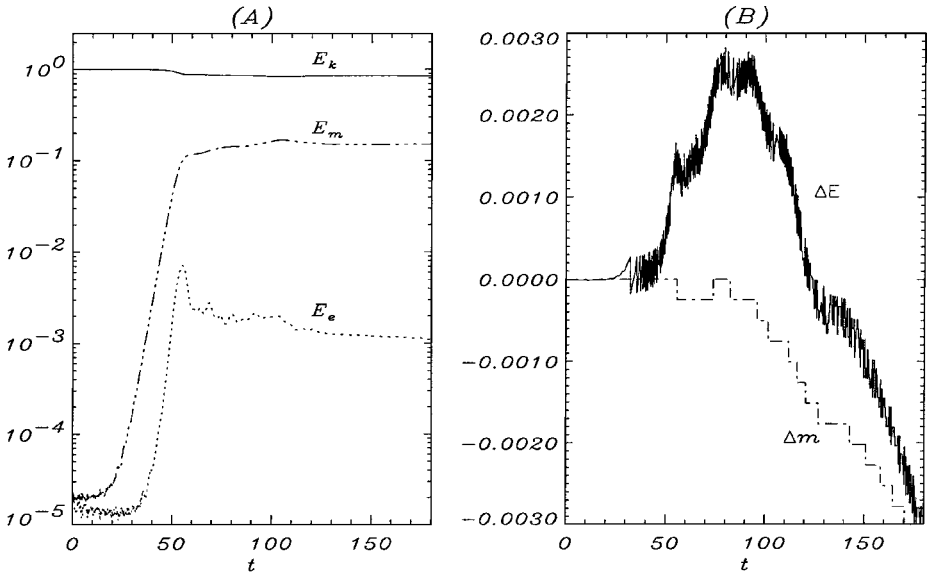

FIG. 11. (A) The kinetic, magnetic, and electric energies E_k (continuous line), E_m (dashed–dotted line), and E_e (dotted line), respectively, versus time. (B) The total energy and total charge density error, $\Delta E = 1 - E$, $\Delta m = 1 - M$, versus time.

In the following we discuss the results obtained with random initial conditions since the monchromatic case gives, as expected, results in excellent agreement with those obtained in Ref. [32] (run 4 in Table 1).

We used the algorithm summarized by Eq. (22) associated with a second-order Van Leer discretization of the x, y, v_x, v_y, v_z translations; periodic boundary conditions are used in the spatial directions. In Fig. 11A, we plot the time evolution of the kinetic, magnetic, and electric energy. This figure shows the exponential increase in the magnetic energy during the linear regime, $t < 50$, with a growth rate $\gamma = 0.138$ on the order of the homogeneous fluid growth rate with wavevector $k \simeq 1$. Then, the instability saturates in the nonlinear regime and the magnetic energy becomes more or less constant, $t > 70$. As discussed in Ref. [32], the saturation of the current filamentation instability is a direct consequence of the kinetic trapping of the electrons close to the peaks of the magnetic field (we recall that, on the other hand, in the fluid approximation a numerical divergence is observed as soon as the nonlinear phase starts [36]).

In Fig. 12 we show the magnetic field in the (x, y) plane at four different times. The initial random field is first organized into small-scale structures of vortex type which then merge into larger vortices and eventually end up in an X-point structure. The X point is also shown in Fig. 13A–C, where we show the shaded isocontours of the density, the isocontours of the current parallel to the electron beams, and the electric field in the (x, y) plane. On the other hand, the magnetic field B_z parallel to the electron beams (Fig. 13D) seems not to be directly correlated to the X-point structure.

The energy and charge density conservation of the numerical algorithm is shown in Fig. 11B, where we plot the deviation of the total energy and of the total charge density from the initial values. This figure shows that after more than 150 inverse of the plasma frequency, the energy and the charge density are very well conserved, with a maximum error on the order of the 0.3%.

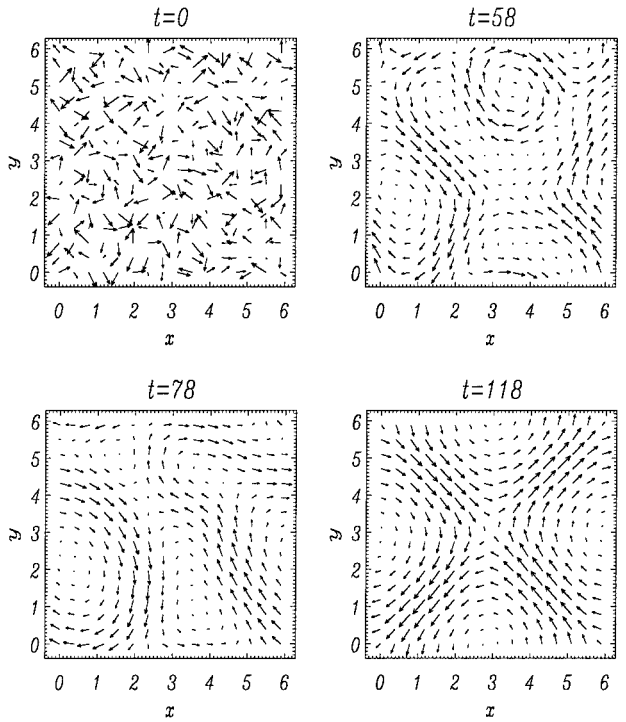

FIG. 12. The magnetic field in the (x, y) plane at $t = 0$, 58, 78, 118.

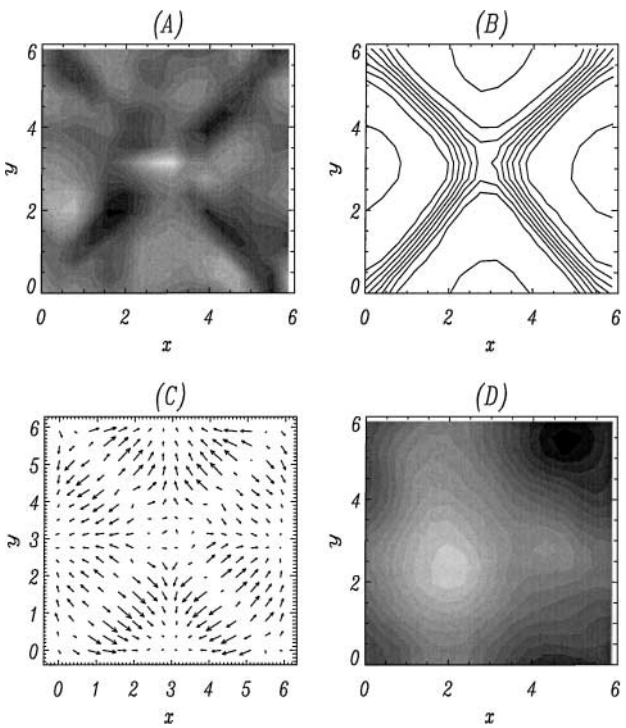

FIG. 13. (A) The density n, (B) the z component of the current j_z, (C) the electric field (E_x, E_y), and (D) the z component of the magnetic field B_z in the (x, y) plane at $t = 118$.

8. DISCUSSION AND CONCLUSIONS

A characteristic feature of the Vlasov equation is the tendency to produce small scales in phase space, even in the absence of any electromagnetic field, a phenomenon called "filamentation." This is immediately seen from the "free-streaming" equation

$$\frac{\partial f(x, v, t)}{\partial t} + v \frac{\partial f(x, v, t)}{\partial x} = 0 \tag{52}$$

starting at $t = 0$ with a distribution function which varies on the large spatial scales $L = 2\pi/k_0$ with a sinusoidally modulated density

$$f(x, v, t = 0) \propto [1 + a \sin(k_0 x)] \exp\left(- \frac{v^2}{2v_t^2} \right).$$

At times $t > 0$, the solution of Eq. (52) is

$$f(x, v, t) \propto (1 + a[\sin(k_0 x) \cos(k_0 v t) - \cos(k_0 x) \sin(k_0 v t)]) \exp\left(- \frac{v^2}{2v_t^2} \right).$$

The spatial scale is basically unchanged but the velocity variations are much more complicated, with ripples having a characteristic scale $v_* = 1/k_0 t$ which decreases as time goes on and reaches the numerical resolution Δv in velocity space at times on the order of $t_{fil} \sim 1/(k_0 \Delta v)$.

The filamentation process is affected in two ways by numerical schemes similar to those which were described here. First, for a "reasonable" numerical resolution, the small-scale structures associated with the filamentation are rapidly dissipated at the grid mesh size, i.e., $\gamma_{ss} t_{fil} > 1$, where γ_{ss} is the numerical damping rate associated with the largest wave numbers $k \sim k_{\max}$ (see, for example, Figs. 2 and 3). Second, some coarse graining is performed when projecting the solution of the Vlasov equation onto a finite-dimensional function space. This is well illustrated if one projects Eq. (52) onto the $V_{M,Nv}$ velocity space

$$f(x, v, t) = \sum_{j=-Nv}^{Nv} \sum_{l=0}^{M} b_{j,l}(x, t) \phi_{j,l}(v), \quad b_{j,l}(x, t) = \frac{1}{c_l^2} \int_{-1}^{1} P_l(\xi) f\left(x, \bar{v}_j + \frac{\Delta v}{2} \xi \right),$$

giving, for $l = 0$, i.e., the velocity cell average, an equation of the form

$$\frac{\partial b_{j,0}(x, t)}{\partial t} + \bar{v}_j \frac{\partial b_{j,0}(x, t)}{\partial x} + \frac{\Delta v}{2} \frac{\partial b_{j,1}(x, t)}{\partial x} = 0.$$

Using a truncature similar to (31) to express $b_{j,1}$ in terms of the velocity derivative of $b_{j,0}$ at lowest order in Δv one obtains for the coarse-grained distribution $b_{j,0}$ an equation of diffusive type,

$$\frac{\partial b_{j,0}(x, t)}{\partial t} + \bar{v}_j \frac{\partial b_{j,0}(x, t)}{\partial x} + \left(\frac{\Delta v}{2} \right)^2 \frac{\partial^2 b_{j,1}(x, t)}{\partial x \partial v} = 0, \tag{53}$$

similar to those which have been extensively studied in the gravitational case (see Tremaine

et al. [39]). The diffusive term is also very effective in the damping of the small velocity scales.

As a result, finite-difference numerical simulations of the Vlasov equation usually do not suffer from the numerical instabilities driven by filamentation.

However, the filamentation, or phase space mixing, is an important physical process. It has been argued by Klimas and Farrell [5] that the corresponding difficulties could be bypassed by applying some velocity averaging to the distribution function, in a spirit similar to what we have just presented, without changing the basic physics. Such an averaging does not influence the calculations of the electric charge and current densities, but the averaged distribution function obeys a more complicated equation, similar to Eq. (53) with a relaxation term containing a double derivative with respect to space and velocity. By solving exactly this equation, one could hope to follow the correct dynamics of the system, even if only the averaged distribution is known. For this to be true, it would be necessary to have a one-to-one relation between the "true" distribution function and the averaged one, which is obviously not the case: within the framework of the present paper, this would mean that the velocity dependence of the distribution function is fully described by the coefficients $b_{j,0}$, $b_{j,1}$.

Most probably, the filamentation problem cannot be properly studied within the strict Vlasov approximation, since the formation of smaller and smaller scales in velocity space can be understood as a manifestation of particle discreteness, which could be introduced in Vlasov simulations as subgrid effects; this will be the subject of future work.

Vlasov simulations provide an excellent description of small-scale phase space dynamics, although the filamentation mechanism is saturated by some artificial numerical dissipation and coarse graining. Very fine physical phenomena can be successfully studied, as, for example, the nonlinear echo phenomenon simulated by Nocera and Mangeney [40]. In the multidimensional case, Vlasov simulations require large numerical resources. On the other hand, due to the recent impressive evolution of massive parallel supercomputers, it is now possible to attack with a Vlasov code complicated problems of interest in plasma research which can be modeled in the 2d–2V or 2d–3V phase space, as, for example, collisionless magnetic reconnection [41], magnetic vortices generation [42], or the study of the evolution of the Weibel instability [32], which has attracted much attention in the last years in the laser plasma interaction context (see, for example, [30] and references therein). It is even expected that in the next few years, with the new generation of teramachines, the full 3d–3V case will be finally accessible to Vlasov simulations. Typical simulation dimensions which can be today run on large parallel supercomputers (Cray T3E, IBM SP3, etc.) are on the order of 64×64 in space and $61 \times 61 \times 61$ in velocity corresponding nearly to 8 Gbytes of memory. Therefore, it is now important to develop performing numerical codes able to take advantage of these new resources.

In this paper we have presented a numerical algorithm for the solution of the fully electromagnetic Vlasov–Maxwell system of equations; we have given a general discussion of the discretization and solution of the advection equation and given an original treatment of the boundary conditions for open systems.

The comparison of the different schemes for the advection equation discussed in this paper should be extended to the full Vlasov equation and will be the subject of a future publication. The splitting scheme used here should also be compared to more efficient symplectic schemes for the time advance. Similarly, other spatial discretization methods should be considered, such as the wavelet projection used by Benhadid [43].

APPENDIX I: DIMENSIONLESS EQUATIONS

In this section we discuss the characteristic parameters used in the Vlasov code to write the equations in dimensionless form.

Electrostatic

The Vlasov–Poisson system of equations is normalized by using the following characteristic quantities: the mass of the electron m_e, a characteristic density $\bar{n}$ and the Debye length L_{De}, and the inverse of the plasma frequency ω_{pe}^{-1} as a characteristic length and time scale, respectively, defined as

$$L_{De} = \left[\frac{\epsilon_0 m_e v_{th,e}^2}{\bar{n}e^2}\right]^{1/2}, \quad \omega_{pe} = \left[\frac{\bar{n}e^2}{\epsilon_0 m_e}\right]^{1/2},$$

where $v_{th,e}$ is the electron thermal speed. Finally, the characteristic electric field is defined as

$$\bar{E} = \frac{m_e \omega_{pe} v_{th,e}}{e}.$$

Concerning the protons, we note that only the bulk proton velocity is normalized to $v_{th,e}$, while the "microscopic" proton velocity is normalized to the proton thermal speed $v_{th,p}$. The dimensionless parameters λ_α and R_α which appear in the dimensionless Vlasov equation (4) are then defined as

$$\lambda_e = \zeta_e = 1, \quad \lambda_p = \frac{v_{th,p}}{v_{th,e}}, \quad \zeta_p = \frac{m_e v_{th,e}}{m_p v_{th,p}},$$

where m_p is the proton mass.

Electromagnetic

In the electromagnetic case, Eqs. (4)–(6), we introduce first the following reference characteristic quantities which must be then further specified according to the physical problem to be studied: $\bar{n}$, $\bar{m}$, $\bar{L}$, $\bar{E}$, and $\bar{B}$, which represent a characteristic density, mass, length, electric field, and magnetic field, respectively.

To those quantities, depending on the problem under consideration, are associated a number of different possible characteristic quantities; here we have chosen a "drift" speed $\bar{U} = \bar{E}/\bar{B}$, with the corresponding time scale $\bar{T} = \bar{L}/\bar{U}$ to be compared with typical physical scales as the speed of the light c, the characteristic cyclotron time $T_* = \bar{m}/e\bar{B}$, and the electron skin depth (or ion inertial length) $L_{in} = (\bar{m}c^2\epsilon_0/\bar{n}e^2)^{1/2}$. Furthermore, it is convenient to normalize the velocity coordinate for each particle species to its thermal velocity v_α. Then, we get the following dimensionless parameters:

$$\lambda_\alpha = \frac{v_\alpha}{U}, \quad R_E = \frac{c^2}{\bar{U}^2}, \quad \zeta_\alpha = \left(\frac{\bar{m}}{m_\alpha}\right)\left(\frac{\bar{U}}{v_\alpha}\right)\left(\frac{T_*}{\bar{T}}\right).$$

The total energy conservation reads

$$\frac{\partial}{\partial t}\left(\frac{B^2}{2} + \frac{1}{R_E}\left(\frac{E^2}{2}\right) + R_c \sum_\alpha K_\alpha\right) + \nabla \cdot \left(R_c \sum_\alpha \mathbf{Q}_\alpha + \mathbf{E} \times \mathbf{B}\right) = 0, \qquad (54)$$

where the energy density is measured in units of $\bar{B}^2/\mu_0$, and where the kinetic energy density K_α, the kinetic energy flux $\mathbf{Q}_\alpha$, and the dimensionless parameter R_c for each particle species are given by

$$K_\alpha = \frac{\lambda_\alpha}{\zeta_\alpha}\int \frac{v^2}{2} f_\alpha \, d\mathbf{v}, \quad \mathbf{Q}_\alpha = \frac{\lambda_\alpha^2}{\zeta_\alpha}\int \left(\frac{v^2}{2}\right)\mathbf{v} f_\alpha \, d\mathbf{v}, \quad R_c = \left(\frac{\bar{L}^2}{L_{in}^2}\right)\left(\frac{T_*}{\bar{T}}\right).$$

The expression of the energy density helps to explain the meaning of the parameters R_E and R_c; they measure the relative contribution of the electric field and of the particles to the energy density.

We consider here two examples: the high-frequency case, where the time and length scales are fixed by the electrons, and the low-frequency case, where the time and length scales are fixed by the protons.

In the first case, the natural units are the electron mass, $\bar{m} = m_e$, the inverse of the electron plasma frequency $\bar{T} = \omega_{pe}^{-1}$, and in the electrostatic case the electron thermal velocity $v_{th,e}$, the Debye length $L_D = v_{th,e}/\omega_{pe}$, and the electric field $\bar{E} = m_e v_{th,e}\omega_{pe}/e$, while in the electromagnetic case the speed of light c, the electron skin depth, $d_e = c/\omega_{pe}$, and the electric and magnetic field $\bar{E} = c\bar{B} = m_e c\omega_{pe}/e$. As a result, in the electrostatic case we get

$$\lambda_\alpha = \frac{v_\alpha}{v_{th,e}}, \quad \zeta_\alpha = \frac{m_e v_{th,e}}{m_\alpha v_\alpha}, \quad R_E = \frac{c^2}{v_{th,e}^2}, \quad R_c = R_E^{-1},$$

while in the electromagnetic case we get

$$\lambda_\alpha = \frac{v_\alpha}{c}, \quad \zeta_\alpha = \frac{m_e c}{m_\alpha v_\alpha}, \quad R_E = R_c = 1.$$

In the second, low-frequency, case we choose the proton mass as the characteristic mass, $\bar{m} = m_p$, the proton inertial length L_{in} as a characteristic length, the inverse of the proton cyclotron frequency as the characteristic time, $\bar{T} = m_p/e\bar{B}$, so that the characteristic speed $\bar{U}$ is the Alfvèn speed, $\bar{U} = v_A \equiv \bar{B}/(\mu_0 \overline{nm})^{1/2}$. The dimensionless parameters become

$$R_E = \frac{c^2}{\bar{v}_A^2}, \quad R_c = 1, \quad \lambda_\alpha = \frac{v_\alpha}{\bar{v}_A}, \quad \zeta_\alpha = \left(\frac{\bar{m}}{m_\alpha}\right)\left(\frac{\bar{v}_A}{v_\alpha}\right).$$

APPENDIX II: THE FIELD EQUATIONS

In this section we discuss the solution of the Poisson and the Maxwell equations with periodic boundary conditions in the y direction and open boundary conditions in the x direction.

The Poisson Equation

We consider the two-dimensional Poisson equation

$$\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} = -q(x, y) \tag{55}$$

in the two-dimensional domain $[0, L_x] \times [0, L_y]$. As usual, the electric field is defined as $E = -\nabla\phi$. We shall assume that the system is periodic in the y direction and that the boundary conditions are given by the (arbitrary) value of the potential at one side, for example $\phi(x = 0) = \phi_0$, and by the value of the potential at the other side, $\phi(x = 1) = \phi_1$, or by its derivative at the left or right boundary, $\phi'(x = 0) = \phi'_0$ or $\phi'(x = L_x) = \phi'_1$. By taking the Fourier transform along the y direction of Eq. (55) we get

$$\frac{\partial^2 \hat{\phi}}{\partial x^2} - k^2 \hat{\phi} = -\hat{q}, \tag{56}$$

where $k(m) = 2\pi m / L_y$ ($m = 0, N_y$) is the wave vector and $\hat{\phi} \equiv \hat{\phi}(x, k)$ and $\hat{q} \equiv \hat{q}(x, k)$ are the k component of the Fourier transform of the potential ϕ and charge density q, respectively.

In order to solve Eq. (56), we first recall that by using the compact finite-difference scheme of Eq. 2.2.6 in Ref. [44], the second-order derivative ϕ'' of a discretized function $[\phi_i]_{i=1,N}$ (on a regular equispaced mesh of size Δ) is given by the implicit relation

$$\alpha\phi''_{i-1} + \phi''_i + \alpha\phi''_{i+1} = \frac{a}{\Delta^2}(\phi_{i-1} - 2\phi_i + \phi_{i+1}), \tag{57}$$

with $\alpha = 1/10$ and $a = 6/5$. By summing Eq. (56) evaluated in point i and the same equation multiplied by α in points $i - 1$ and $i + 1$ and using the relation given by Eq. (57), we obtain

$$(\alpha\Delta^2 k^2 - a)\hat{\phi}_{i-1} + (\Delta^2 k^2 + 2a)\hat{\phi}_i + (\alpha\Delta^2 k^2 - a)\hat{\phi}_{i+1} = \Delta^2[\alpha\hat{q}_{i-1} + \hat{q}_i + \alpha\hat{q}_{i+1}].$$

This equation is valid in the interior of the domain, $i = 2, N - 1$; concerning the boundary conditions one must distinguish the case $k = 0$, where the value of the potential must be given at the left or right boundary (the precise value being unimportant since the potential is defined up to an arbitrary constant), while for $k \neq 0$ the two boundary conditions can be any combination of the potential or its derivative (in this case, for example, one can use the boundary formulas given in Ref. [44]) at the left and right boundary.

As a result, one has to solve for each wave vector k a tridiagonal system in order to recover the values of the potential $\hat{\phi}_i$ on the numerical grid. This scheme is much more accurate than directly solving the tridiagonal system resulting from Eq. (56) by using, for example, a three-point centered finite-difference scheme for $\hat{\phi}''_i$.

After getting the solution for the potential in the physical space by making an inverse Fourier transform, one finally gets the electric field by using any first-derivative numerical scheme.

Maxwell Equations

The Maxwell Eqs. (5) and (6) are integrated in the physical lower dimensionality space; therefore, the required computational effort is, in any case, negligible with respect to the integration of the Vlasov equation.

When the boundary conditions are periodic, these equations are integrated in the Fourier space by using a standard FFT algorithm coupled to the fourth-order Runge–Kutta scheme. On the other hand, in the nonperiodic 2d case, we have adopted the following scheme, which essentially consists of the integration of the equations along the characteristics. First of all,

we note that in 2d the electromagnetic fields split into two decoupled systems of equations, the first one involving only the E_x, E_y, and B_z fields and the second one concerning only the E_z, B_x, and B_y fields. Let us discuss the solution of the first system assuming periodic boundary conditions in the y direction. By defining

$$y^{\pm} = B_z \pm \frac{E_y}{R_E^{1/2}} \tag{58}$$

and by taking the Fourier transform along the y direction of the electric and magnetic fields, we end up with the equations

$$\frac{\partial \hat{Y}^{\pm}}{\partial t} \pm R_E^{1/2} \frac{\partial \hat{Y}^{\pm}}{\partial x} = ik\hat{E}_x \mp \frac{\hat{J}_y}{R_E^{1/2}}, \tag{59}$$

$$\frac{\partial \hat{E}_x}{\partial t} - ikR_E\hat{B}_z = -\hat{J}_x, \tag{60}$$

where $\hat{Y}^{\pm} \equiv \hat{Y}^{\pm}(x, k)$, and so forth, are the k component of the Fourier transform along y.

The first two equations given in Eq. (59) are solved by means of a finite-volume technique by advancing in time the cell-averaged quantities defined as

$$\bar{Z}^{\pm} = \frac{1}{dx} \int_{x_i}^{x_{i+1}} \hat{Z}^{\pm}(x)\, dx \tag{61}$$

so that Eq. (59) reduces to

$$\frac{\partial \bar{Y}^{\pm}}{\partial t} = \mp \frac{R_E^{1/2}}{dx} (\hat{Y}^{\pm}(x_{i+1}) - \hat{Y}^{\pm}(x_i)) + ik\bar{E}_x \mp \frac{\bar{J}_y}{R_E^{1/2}}. \tag{62}$$

Equations (60) and (62) are advanced in time by using the fourth-order Runge–Kutta scheme; in order to avoid the propagation inside the integration domain of numerical waves (varying on the grid length scale) generated artificially by an outgoing propagating signal at the two boundaries (the right and the left boundary for $\bar{Y}^{+}$ and $\bar{Y}^{-}$, respectively), we use a small-scale (fourth order) filter obtained numerically by a tridiagonal compact finite-difference scheme discussed in Ref. [44] (Eq. C.2.1 with $\beta = d = 0$ and Eqs. C.2.11.a/b at the boundaries). The nodal values of the fields E_y and B_z required for the advancement of the Vlasov equation are then obtained by means of a reconstruction algorithm based on the computation of the "primitive function" [21] defined by

$$\hat{T}^{\pm}(x) = \int_0^x \hat{Y}^{\pm}(x')\, dx' \rightarrow \hat{T}^{\pm}(x_i) = \sum_{n=1}^{i} \bar{Y}_i^{\pm}\, dx,$$

which finally gives the nodal values:

$$\hat{Y}^{\pm}(x_i) = \frac{\partial \hat{T}^{\pm}(x)}{\partial x} \bigg| x = x_i.$$

An inverse Fourier transform is used to get back into the physical (x, y) space.

APPENDIX III

In this appendix we present the explicit expressions used for the four schemes used to solve numerically the advection equation. We use the notation $\delta = |v\Delta t/\Delta x|$ and assume that $\delta < 1$.

Discontinuous Galerkin Method (VL2, VL3)

In the expression (29) of the matrix elements, the cell index j can take only two values (since $\delta < 1$), either $j = i, i-1$ for positive velocities or $j = i, i+1$ for negative velocities. Using the change of integration variable $x = \bar{x}_i + (\Delta x/2)\xi$ in the definition (29) we obtain

$$
\begin{aligned}
A^{i,l}_{ik}(v \geq 0) &= \frac{1}{c_l^2}\int_{-1+2\delta}^{1} d\xi\, P_l(\xi)P_k(\xi - 2\delta),\\
A^{i,l}_{i,k}(v < 0) &= \frac{1}{c_l^2}\int_{-1}^{1-2\delta} d\xi\, P_l(\xi)P_k(\xi + 2\delta),\\
A^{i,l}_{i-1k}(v \geq 0) &= \frac{1}{c_l^2}\int_{-1}^{1-2\delta} d\xi\, P_l(\xi)P_k(\xi + 2(1-\delta)),\\
A^{i,l}_{i+1k}(v < 0) &= \frac{1}{c_l^2}\int_{1-2\delta}^{1} d\xi\, P_l(\xi)P_k(\xi - 2(1-\delta)).
\end{aligned}
$$

A very useful symmetry relation is

$$
c_l^2 A^{i,l}_{i,k}(v<0) = c_k^2 A^{i,k}_{i,l}(v \geq 0), \quad c_l^2 A^{i,l}_{i+1,k}(v<0) = c_k^2 A^{i,k}_{i-1,l}(v \geq 0),
$$

which allows calculation of only the matrix coefficients for postive velocities.

Second Order ($M = 1$), $v > 0$

In this case, the only nonvanishing matrix elements are

$$
A^{i,0}_{i,0} = 1-\delta, \quad A^{i,0}_{i,1} = -\delta(1-\delta), \quad A^{i,1}_{i,0} = 3\delta(1-\delta), \quad A^{i,1}_{i,1} = (1-\delta)(1-2\delta-2\delta^2), \tag{63}
$$

$$
A^{i,0}_{i-1,0} = \delta, \quad A^{i,0}_{i-1,1} = \delta(1-\delta), \quad A^{i,1}_{i-1,0} = -3\delta(1-\delta), \quad A^{i,1}_{i-1,1} = -\delta(3-6\delta+2\delta^2). \tag{64}
$$

After the projection of Eq. (31),

$$
a^{t+\Delta t}_{i,0} = \sum_{j=i-1}^{j=i+1} A^{i,0}_{j,0}\, a_{j,0} + \sum_{j=i-1}^{j=i+1} A^{i,0}_{j,1}\, a_{j,1},
$$

we get the coefficients of the Van Leer's scheme (VL2):

$$
\begin{aligned}
A_{-2} &= -\frac{\delta}{4}(1-\delta), \quad A_{-1} = \delta + \frac{\delta}{4}(1-\delta),\\
A_0 &= (1-\delta)\left(1+\frac{\delta}{4}\right), \quad A_1 = -\frac{\delta}{4}(1-\delta), \quad A_2 = 0.
\end{aligned} \tag{65}
$$

Note that

$$\sum_{j=-2}^{j=1} A_j(\delta) = 1,$$

expressing the conservation of total mass.

For open boundaries, we have to modify the fluxes in the vicinity of the boundaries, so that the displacement matrix is somewhat modified and becomes

$$A^*_{i,j} = A_{j-i}(\delta), \quad v > 0, \quad A^*_{i,j+1} = A_{i-(j+1)}(\delta), \quad v \le 0,$$

for $2 < i < N-1$, $j = i-2, i+1$, while the matrix elements $A^*_{1,j}$, $A^*_{2,j}$, $A^*_{N,j}$ corresponding to the boundary cells are given, for $v \ge 0$, by

$$A^*_{1,1}(\delta) = (1-\delta)\left(1+\frac{3\delta}{4}\right), \quad A^*_{1,2}(\delta) = -\delta(1-\delta), \quad A^*_{1,3}(\delta) = \frac{\delta}{4}(1-\delta),$$

$$A^*_{2,1}(\delta) = \frac{\delta}{2}(1+\delta), \quad A^*_{2,2}(\delta) = (1-\delta)(1+\delta), \quad A^*_{2,3}(\delta) = -\frac{\delta}{2}(1-\delta),$$

$$A^*_{N,N-2}(\delta) = -\frac{\delta}{2}(1-\delta), \quad A^*_{N,N-1}(\delta) = \delta(2-\delta), \quad A^*_{N,N}(\delta) = (1-\delta)\left(1-\frac{\delta}{2}\right).$$

For $v \le 0$ the corresponding matrix elements are easily obtained by using the symmetry relations (47) and (48).

Third Order ($M = 2$), $v > 0$

In this case, the only nonvanishing matrix elements are those listed in Section 11.2 and

$$\begin{gathered} A^{i,2}_{i,0} = -5\delta(1-\delta)(1-2\delta), \quad A^{i,2}_{i,1} = 5\delta(1-\delta)(1-\delta-\delta^2), \\ A^{i,0}_{i,2} = -\delta(1-\delta)(1-2\delta), \quad A^{i,1}_{i,2} = -3\delta(1-\delta)(1-\delta-\delta^2), \\ A^{i,2}_{i,2} = (1-\delta)[1-2\delta(1+\delta)(2-3\delta^2)]; \end{gathered} \tag{66}$$

$$\begin{gathered} A^{i,2}_{i-1,0} = 5\delta(1-\delta)(1-2\delta), \quad A^{i,2}_{i-1,1} = -\delta(1-\delta)(1-3\delta+7\delta^2), \\ A^{i,0}_{i-1,2} = \delta(1-\delta)(1-2\delta), \quad A^{i,1}_{i-1,2} = -2\delta(1-\delta)(3-9\delta-2\delta^2), \\ A^{i,2}_{i-1,2} = \delta + 2\delta(1-\delta)(2-13\delta+12\delta^2-3\delta^3). \end{gathered} \tag{67}$$

After the projection (Eq. (32)) the matrix elements become

$$A_{-3} = \frac{\delta(1-\delta)}{24}\left(1-\frac{1}{6}(1-2\delta)\right), \quad A_{-2} = -\frac{\delta(1-\delta)}{4}\left(1+\frac{1}{2}\left[1-\frac{17}{18}(1-2\delta)\right]\right);$$

$$A_{-1} = \delta + \frac{\delta(1-\delta)}{4} + \frac{\delta(1-\delta)}{12}\left(1-\frac{23}{6}(1-2\delta)\right);$$

$$A_0 = (1-\delta)\left(1+\frac{\delta}{4}+\frac{\delta}{12}\left(1+\frac{23}{6}(1-2\delta)\right)\right);$$

$$A_1 = -\frac{\delta(1-\delta)}{4}\left(1+\frac{1}{2}\left(1+\frac{17}{18}(1-2\delta)\right)\right), \quad A_2 = \frac{\delta(1-\delta)}{24}\left(1+\frac{1}{6}(1-2\delta)\right).$$

For this scheme (VL3) the stencil is larger than for the VL2 scheme: six points instead of four.

Hermite Interpolation (HMN)

The elements $h_{i,k}(x)$ of the basis of the "smooth Hermite" spaces H_{NM} introduced in Section 3.3.2, Eq. (36), can be considered translated and rescaled versions, $\xi = (x - x_i)/\Delta x$, of functions $h_k(\xi)$ defined on the interval $[-1, 1]$. For $M = 1$ ($m = 0$), the basis reduces to the classical "hat" function:

$$h_0 = 1 - |\xi|, \quad -1 \le \xi \le 1.$$

We consider here only the periodic case; at both ends of the numerical domain the basis functions are extended by using the periodicity.

The next order is $M = 3$ ($m = 1$), and the basis functions are

$$h_0 = (1 - 2\xi)(1 - |\xi|^2), \quad h_1 = \xi(1 - |\xi|)^2, \quad -1 \le \xi \le 1,$$

with the corresponding modifications for the boundary grid points. The matrix elements of T_H, for the scheme HMN, Eq. (37), are then given by (for $v \ge 0$)

$$A^{i,0}_{i,0} = (1+2\delta)(1-\delta)^2, \quad A^{i,0}_{i,1} = -\delta(1-\delta)^2, \quad A^{i,0}_{i-1,0} = \delta^2(3-2\delta), \quad A^{i,0}_{i-1,1} = \delta^2(1-\delta);$$
$$A^{i,1}_{i,0} = 6\delta(1-\delta), \quad A^{i,1}_{i,1} = (1-3\delta)(1-\delta), \quad A^{i,1}_{i-1,0} = -6\delta(1-\delta), \quad A^{i,1}_{i-1,1} = -\delta(2-3\delta).$$

The symmetry relation which allows calculation of the matrix elements for $v < 0$ is now

$$A^{i,l}_{i,k}(v<0) = (-1)^{k+1} A^{i,k}_{i,l}(v \ge 0), \quad A^{i,l}_{i-1,k}(v<0) = (-1)^{k+1} A^{i,k}_{i+1,l}(v \ge 0).$$

Spline Interpolation (SPL)

In this scheme, SPL, the data are the values of the distribution function, $f_0, \ldots, f_N$, at the cell boundaries. To advance these data by a time step ΔT, one first determines the corresponding cubic spline interpolant $\tilde{f}_{spl}$ through standard procedures (see, for example, Rubin and Khosla [25]); the periodic case being a little more involved, we detail in Appendix IV the solution of the tridiagonal system with periodic boundary conditions which allows determination of the second derivatives $D^2 f_0, \ldots, D^2 f_N$ of the interpolant using the continuity of the first derivative. Once these second derivatives are obtained, one knows explicitly $\tilde{f}_{spl}$,

$$\tilde{f}_{spl}(x) = \frac{1}{6} D^2 f_{i-1}(1-\xi)^3 + \frac{1}{6} D^2 f_i \xi^3 + \left(f_{i-1} - \frac{1}{6} D^2 f_{i-1} \right)(1-\xi) + \left(f_i - \frac{1}{6} D^2 f_i \right)\xi, \tag{68}$$

for each cell $[x_{i-1}, x_i]$, with $\xi = (x - x_{i-1})/\Delta x$. Then the updated datas are simply given by

$$\{\tilde{f}_{spl}(x_i - vt)\}_{i=0,N}.$$

Note that the second derivative of the function f in the Eq. (68) must be calculated from the solution of a linear system and not calculated from the tabulated values of f (in other words, not from an explicit finite-difference scheme). The linear system comes out from the requirement of the first-derivative continuity across the boundary between two intervals.

APPENDIX IV

Periodic Boundary Conditions for a Tridiagonal Scheme

Let us assume that we must solve an algebraic system of equations $\mathbf{Ax} = \mathbf{b}$ of the form

$$\begin{Vmatrix} 1 & \alpha & 0 & 0 & \dots & & & \dots & 0 & \alpha \\ \alpha & 1 & \alpha & 0 & .. & & & & \dots & 0 \\ 0 & \alpha & 1 & \alpha & 0 & & & & \dots & 0 \\ 0 & \dots & \dots & \dots & & & & & \dots & 0 \\ 0 & \dots & \dots & \dots . & \dots . & & & & \dots & 0 \\ 0 & \dots & & \dots & \dots . & \dots & & & \dots & 0 \\ 0 & \dots & & & \dots & \dots . & \dots & & \dots & 0 \\ 0 & \dots & & & & 0 & \alpha & 1 & \alpha & 0 \\ 0 & \dots & & & & \dots & 0 & \alpha & 1 & \alpha \\ \alpha & 0 & \dots & & & \dots & 0 & 0 & \alpha & 1 \end{Vmatrix} \begin{Vmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ \dots \\ \dots \\ \dots \\ \dots \\ x_{N-2} \\ x_{N-1} \\ x_N \end{Vmatrix} = \begin{Vmatrix} b_1 \\ b_2 \\ b_3 \\ \dots \\ \dots \\ \dots \\ \dots \\ \dots \\ b_{N-2} \\ b_{N-1} \\ b_N \end{Vmatrix}, \tag{69}$$

which is, for example, the typical representation of a compact finite-difference scheme with periodic boundary conditions. It is immediately seen that the tridiagonal character of the matrix $\mathbf{A}_{N,N}$ is broken by the last and first element of the first and the last line, respectively. To avoid the numerical inversion of the full matrix, which would be too expensive both in memory storage and in CPU time, we can adopt the following strategy.

We define $\tilde{\mathbf{A}}_{N-1,N-1}$ as the submatrix of $\mathbf{A}$ ranging from $1 \le i \le N-1,\ 1 \le j \le N-1$ (in other words, $\mathbf{A}$ without the last line and last column). We also define $\tilde{\mathbf{x}}_{N-1}$ and $\tilde{\mathbf{b}}_{N-1}$ as the subvectors of $\mathbf{x}$ and $\mathbf{b}$ ranging from $1 \le i \le N-1$ and $\mathbf{r}_{N-1}$ as the vector $r_1 = r_{N-1} = 1$, $r_i = 0$ for $i = 2, 3, \dots, N-2$. Then, the system of Eqs. (69) can be cast in the form

$$\tilde{\mathbf{A}}\tilde{\mathbf{x}} + \alpha x_N \mathbf{r} = \tilde{\mathbf{b}}, \tag{70}$$

$$\alpha x_{N-1} + x_N + \alpha x_1 = b_N. \tag{71}$$

Multiplying Eq. (70) by $\tilde{\mathbf{A}}^{-1}$ we get

$$\tilde{\mathbf{x}} = \mathbf{Y} - \alpha x_N \mathbf{Z}, \tag{72}$$

where $\mathbf{Z} = \tilde{\mathbf{A}}^{-1}\mathbf{r}$ must be calculated only once, while $\mathbf{Y} = \tilde{\mathbf{A}}^{-1}\tilde{\mathbf{b}}$ is calculated at each time step (assuming $\mathbf{b}$ as a time-dependent vector). Then, from the first and the last equation of Eq. (72),

$$x_1 = Y_1 - \alpha Z_1 x_N, \quad x_{N-1} = Y_{N-1} - \alpha Z_{N-1} x_N,$$

and from Eq. (71), we can calculate the numerical values of $x_1,\ x_{N-1}$, and x_N. Finally,

$$x_i = Y_i \ - \alpha x_N Z_i; \quad i = 2, \dots, N-2.$$

APPENDIX V: VL2 AND SPL PERFORMANCE

We compare quantitatively the performance of the VL2 and SPL schemes by calculating the CPU time, the number of operations, and the mega flops of the two algorithms versus the number of points N_x in the case of the advection equation, Eq. (23). In particular, concerning the SPL algorithm, we have chosen a standard cubic spline interpolation [45], coupled with the Sherman–Morrison formula for cyclic tridiagonal systems [46]. We note that in this case, since there is no explicit time dependence of the translation operator, the interpolation coefficients are calculated once at the beginning of the time iteration.

In Fig. 14 we plot, for 101 iteration of the calculation of the translation operator of the advection equation, the CPU time (in seconds), the number of operations, and the corresponding mega flops versus the number of points N_x with a fixed number in the

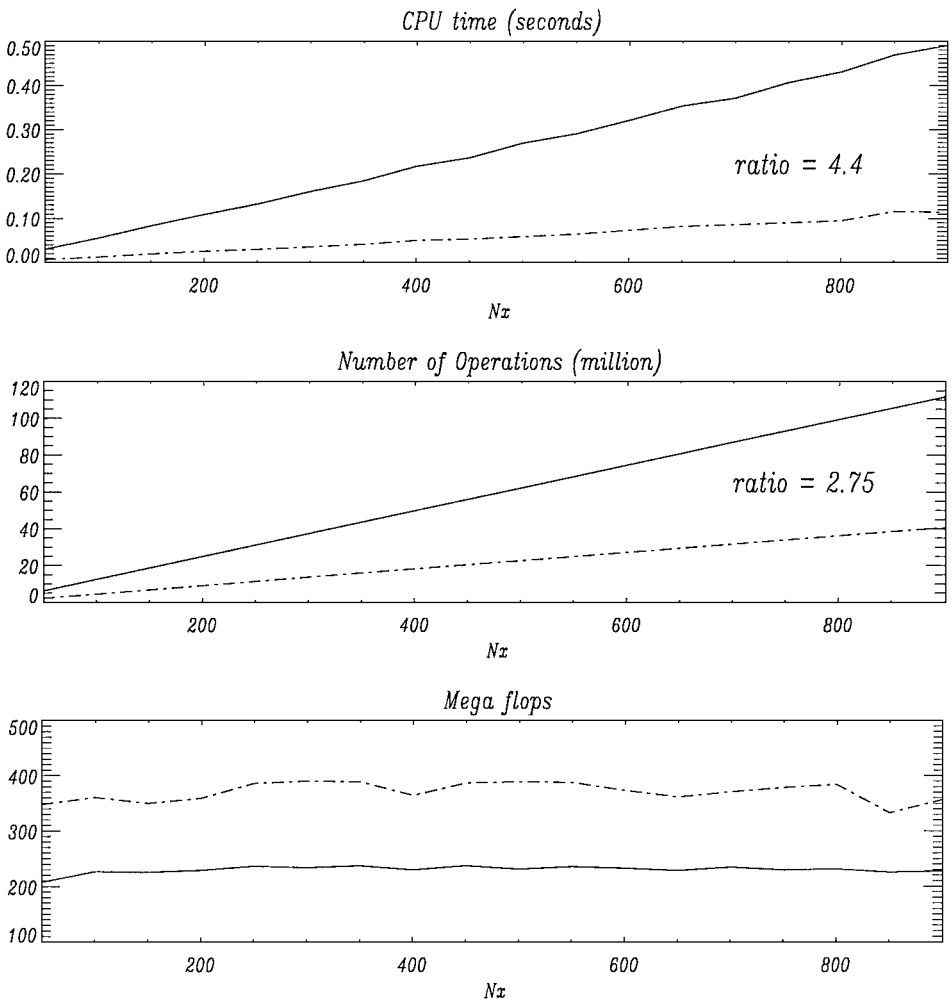

FIG. 14. The numerical performance of the SPL (continuous lines) and VL2 (dashed–dotted lines) algorithms for the calculation of the translation operator of the advection equation, Eq. (23). In this picture we plot the CPU time (in seconds) (first frame), the total number of operations (second frame), and the corresponding mega flops versus the number of grid points N_x with a fixed number in the velocity space, $N_v = 65$, and for 101 iterations.

velocity space, $N_v = 65$. The numerical calculations were performed on an IBM RISC processor Power 3 (375 MHz).

Looking at the execution time (first frame), the VL2 method is on average 4.4 times faster than SPL. Two reasons explain this difference. First, the VL2 algorithm uses on average a number of floating-point operations 2.7 times less than SPL (see the number of operations plot, middle frame). Therefore, unless a more performant SPL algorithm is used, the VL2 is faster than SPL by a factor that cannot be lower than 2.7 (in other words, to reduce the gap between the two methods, one needs a different approach for solving the tridiagonal system included in SPL). Second, the VL2 has been implemented in our test more efficiently since it runs on average between 350 and 400 Mflops (million of floating-point operations per second), while the SPL runs between 200 and 250 Mflops. With some effort in further optimizing the SPL code, this second cause could be almost completely eliminated.

In conclusion, concerning the algorithms presented in this paper, it seems resonable to take a factor of three in the execution time as a typical CPU time ratio between the VL2 and SPL algorithms.

ACKNOWLEDGMENTS

It is a pleasure to acknowledge Prof. F. Pegoraro and Prof. M. Shoucri for very fruitful discussions on the subject. This work was supported by the INFM Parallel Computing Initiative, by MURST, and, in part, by INTAS Contract 97-1612.

REFERENCES

1. A. I. Akhiezer, R. V. Polovin, and A. G. Sitenko, *Plasma Electrodynamics* (Pergamon, Oxford, 1975).
2. P. J. E. Peebles, *The Large Scale Structure of Universe* (Princeton Univ. Press, Princeton, NJ, 1980).
3. G. F. Bertsch and S. Das Gupta, A guide to microscopic models for intermediate energy heavy ion collisions, *Phys. Rep.* **160**, 190 (1988).
4. C. K. Birdsall and A. B. Langdon, *Plasma Physics Via Computer Simulation* (Inst. of Physics, Bristol, 1991).
5. A. J. Klimas and A. J. Farrell, A splitting algorithm for Vlasov simulation with filamentation filtration, *J. Comput. Phys.* **110**, 150 (1994).
6. C. G. Cheng and G. Knorr, The integration of the Vlasov equation in configuration space, *J. Comput. Phys.* **22**, 330 (1976).
7. Yu. V. Gott and E. I. Yurchenko, Five-parametrical group of scalings for transport coefficients in turbulent magnetized plasmas, in *Theory of Fusion Plasmas*, edited by J. W. Connor, E. Sindoni, and J. Vaclavik (SIF, Bologna, 1999), p. 31.
8. H. C. Andersen, A "velocity" version of the Shake algorithm for molecular dynamics calculations, *J. Comput. Phys.* **52**, 24 (1983).
9. H. Yoshida, Recent Progress in the Theory and Application of Symplectic Integrators, *Celest. Mech.* **56**, 27 (1993).
10. L. E. Johnson, Numerical model of plasma double layers using the Vlasov equation, *J. Plasma Phys.* **23**, 433 (1980).
11. C. G. Cheng, The integration of the Vlasov equation for a magnetized plasma, *J. Comput. Phys.* **24**, 348 (1977).
12. E. Fijalkow, Numerical solution to the Vlasov equation: The 1D code, *Comput. Phys. Commun.* **116**, 319 (1999).
13. J. Gazdag, Numerical solution of the Vlasov equation with the accurate space derivative method, *J. Comput. Phys.* **19**, 77 (1975).
14. R. Bermejo, Analysis of an algorithm for the Galerkin-characteristic method, *Numer. Math.* **60**, 163 (1991).

15. E. Sonnendrücker, J. Roche, P. Bertrand, and A. Ghizzo, The semi-Lagrangian method for the numerical resolution of Vlasov equations, *J. Comput. Phys.* **149**, 201 (1999).
16. S. Blanes and P. C. Moan, Splitting methods for non-autonomous Hamiltonian equations, *J. Comput. Phys.* **170**, 205 (2001).
17. E. Godlewski and P. A. Raviart, *Numerical Approximation of Hyperbolic Systems of Conservation Laws* (Springer-Verlag, Berlin, 1996).
18. H. D. Victory, Jr. and K. Ganguly, On finite difference methods for solving discrete ordinates transport equations, *SIAM J. Numer. Anal.* **23**, 78 (1986).
19. B. Cockburn, An introduction to the discontinuous Galerkin methods for convection dominated problems, in *Advanced Numerical Approximation of Nonlinear Hyperbolic Equations*, edited by A. Quarteroni (Springer-Verlag, Berlin/New York, 1997).
20. B. Van Leer, Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection, *J. Comput. Phys.* **23**, 263 (1977).
21. A. Harten, High resolution schemes for hyperbolic conservation laws, *J. Comput. Phys.* **135**, 260 (1997).
22. J. E. Fromm, *J. Comput. Phys.* **3**, 176 (1968).
23. P. G. Ciarlet, M. H. Schultz, and R. S. Varga, Numerical methods of high-order accuracy for nonlinear boundary value problems, *Numer. Math.* **9**, 394 (1967).
24. T. Nakamura and T. Yabe, Cubic interpolated propagation scheme for solving the hyper-dimensional Vlasov–Poisson equations inphase space, *Comput. Phys. Commun.* **120**, 122 (1999).
25. S. G. Rubin and P. K. Khosla, Polynomial interpolation methods for viscous flow calculations, *J. Comput. Phys.* **24**, 217 (1977).
26. M. Shoucri, A splitting scheme for the numerical solution of the Vlasov equation, *J. Comput. Phys.* **24**, 445 (1977).
27. S. Abarbanel and A. Chertock, Strict stability of high-order compact implicit finite-difference schemes: The role of boundary conditions for hyperbolic PDEs, *J. Comput. Phys.* **160**, 42 (2000).
28. M. Sabatier, N. Suh, P. Mineau, M. Feix, M. Shoucri, P. Bertrand, and E. Fijalkow, *Numerical Simulations of the Vlasov Equation Using a Flux Conservation Scheme; Comparison with the Cubic Pline Interpolation Code*, Internal Report 330e (Centre Canadian de Fusion Magnetique, 1990).
29. E. Pohn, M. Shoucri, and G. Kamelander, Study of the formation of a charge separation at a plasma edge. Part II. Comparison between three different interpolation methods for the solution of the kinetic Vlasov equation, *Comput. Phys. Commun.* **137**, 396 (2001).
30. B. Lasinski, A. Langdon, S. Hatchett, M. Key, and M. Tabak, Particle-in-cell simulations of ultra intense laser pulses propagating through overdense plasma for fast-ignitor and radiography applications, *Phys. Plasmas* **6**, 2041 (1999); Y. Sentoku, K. Mima, S. Kojima, and H. Ruhl, Magnetic instability by the relativistic laser pulses in overdense plasmas, *Phys. Plasmas* **7**, 689 (2000); T. Taguchi, T. Antonsen, Jr., C. S. Liu, and K. Mima, Structure Formation and Tearing of an MeV Cylindrical Electron Beam in a Laser-Produced Plasma, *Phys. Rev. Lett.* **86**, 5055 (2001).
31. R. C. Davidson, D. A. Hammer, I. Haber, and C. E. Wagner, Nonlinear development of electromagnetic instabilities in anisotropic plasmas, *Phys. Fluids* **15**, 317 (1972).
32. F. Califano, F. Pegoraro, S. Bulanov, and A. Mangeney, Kinetic saturation of the Weibel instability in a collisionless plasma, *Phys. Rev. E* **57**, 7048 (1998).
33. G. A. Askar'yan, S. V. Bulanov, F. Pegoraro, and A. M. Pukhov, Magnetic interaction of self-focusing channels and fluxes of electromagnetic radiation: their coalescence, the accumulation of energy, and the effect of external magnetic fields on them, *JETP Lett.* **60**, 251 (1994); T. V. Liseikina, F. Califano, V. A. Vshivkov, F. Pegoraro, and S. Bulanov, Small-scale electron density and magnetic-field structures in the wake of an ultraintense laser pulse, *Phys. Rev. E* **60**, 5991 (1999).
34. E. S. Weibel, Spontaneously growing transverse waves in a plasma due to an anisotropic velocity distribution, *Phys. Rev. Lett.* **2**, 83 (1959).
35. F. Pegoraro, S. Bulanov, F. Califano, and M. Lontano, Non-linear development of the Weibel instability and magnetic field generation in collisionless plasmas, *Phys. Scr.* **T63**, 262 (1996).
36. F. Califano, R. Prandi, F. Pegoraro, and S. V. Bulanov, Non-linear filamentation instability driven by an inhomogeneous current in a collisionless plasma, *Phys. Rev. E* **58**, 7837 (1998).

37. R. Lee and M. Lampe, Electromagnetic instabilities, filamentation, and focusing of relativistic electron beams, *Phys. Rev. Lett.* **31**, 1390 (1973).
38. M. Honda, J. Meyer-ter-Vehn, and A. Pukhov, Collective stopping and ion heating in relativistic-electron-beam transport for fast ignition, *Phys. Rev. Lett.* **85**, 2128 (2000).
39. S. Tremaine, M. Hénon, and D. Lynden-Bell, H-functions and mixing in violent relaxation, *J. Mon. Not. R. Astron. Soc.* **219**, 285 (1986).
40. L. Nocera and A. Mangeney, Evidence for second-order oscillations at the Best frequency in direct numerical simulations of the Vlasov equation, *Phys. Plasmas* **6**, 4559 (1999).
41. F. Califano, N. Attico, F. Pegoraro, G. Bertin, and S. V. Bulanov, Fast formation of magnetic islands in a plasma in the presence of counterstreaming electrons, *Phys. Rev. Lett.* **86**, 5293 (2001).
42. F. Califano, F. Pegoraro, and S. Bulanov, Impact of kinetic processes on the macroscopic nonlinear evolution of the electromagnetic beam plasma instability, *Phys. Rev. Lett.* **84**, 3602 (2000).
43. Y. Benhadid, Ph.D. thesis, Méthodes numériques pour léquation de Vlasov, (Orleans University, Orleans, France, 2001).
44. S. K. Lele, Compact finite difference schemes with spectral-like resolution, *J. Comput. Phys.* **103**, 16 (1992).
45. *Numerical Recipes in Fortran 77*, 2nd ed., edited by William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. (Cambridge Univ. Press, Cambridge, UK, 1999).
46. *Numerical Recipes in Fortran 77*, 2nd ed., edited by William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. (Cambridge Univ. Press, Cambridge, UK, 1999).