JOURNAL OF COMPUTATIONAL PHYSICS 112, 102–116 (1994)

Current Advance Method and Cyclic Leapfrog for 2D Multispecies Hybrid Plasma Simulations

ALAN P. MATTHEWS*

Département de Recherche Spatiale (DESPA), Observatoire de Paris-Meudon, 92190 Meudon, France

Received June 9, 1992; revised June 16, 1993

CAM-CL (current advance method and cyclic leapfrog) is a new algorithm for hybrid plasma simulations. In common with existing methods, its physical basis is a "hybrid" plasma model which treats the ions as particles and the electrons as a massless fluid. CAM-CL is distinguished from previous 2D hybrid algorithms by four main features: (1) *Multiple ion species may be treated with only a single computational pass through the particle data; this is achieved without extrapolation of the electric field in time.* The particles are advanced by a leapfrog procedure which requires the electric field to be a half time-step ahead of the particle velocities. The electric field depends on the ionic current density and hence the particle velocities. In order to avoid a time-consuming "pre-push" of the velocities, CAM advances the ionic current density a half-step with an appropriate equation of motion. This is similar in concept to the moment method (D. Winske and K. B. Quest, *J. Geophys. Res.* **93** (A9), 9681 (1988), Appendix A), except for the next two features: (2) *CAM advances the ionic current density, whereas the moment method advances the fluid velocity.* Consequently, multiple ion species may be easily treated. (3) *A free-streaming ionic current density is collected (velocities are collected at positions a half time-step ahead).* An equation of motion is then applied, in which the advective term and the ionic stress tensor in the moment method are not needed, since transport effects are included in the free-streaming current. (4) *CL is a leapfrog scheme for advancing the magnetic field, an adaptation of the modified midpoint method described by W. H. Press et al. (Numerical Recipes* (Cambridge Univ. Press, Cambridge, 1986)). It is stable and allows sub-stepping of the magnetic field (the magnetic field time-step may be different to the particle time-step). A two-dimensional version of the algorithm has been tested on a quiet plasma, MHD wave propagation, and ion beam instabilities, the results of which are discussed.

1. INTRODUCTION

Hybrid numerical codes are used to simulate plasma behaviour in which ion kinetic effects are important, whereas those of electrons may be neglected. This paper describes an algorithm (CAM-CL) which integrates the differential equations in a manner that is explicit in time and spatially local. CAM-CL refers to two features of the means by which the system is advanced in time: *current advance method* for the ions, and *cyclic leapfrog* for the magnetic field. A 2D implementation of the algorithm, a code called H2CAM, has much in common with three methods reviewed by Quest [8]: a predictor–corrector scheme described by Harned [4], a code used by Terasawa *et al.* [9], and especially a *moment method* described by Winske and Quest [14]. It is distinguished from these methods by four main features: (1) multiple species may be treated with only a single computational pass through the particle data; this is achieved without extrapolation of the electric field in time; (2) CAM advances the ionic current density, whereas the moment method advances the fluid velocity; (3) a "free-streaming" ionic current density is collected which is then advanced by an appropriate equation of motion, whereas in the moment method advective and ionic stress tensor terms are treated; (4) CL is an adaptation of the modified midpoint method [7] to magnetic field evolution.

1.1. *Hybrid Codes*

A hybrid model treats the various components of a plasma in a different manner; here the ions are modelled as particles and the electrons as a fluid (see Winske [12] for a review). The system is governed by the Vlasov-fluid equations, comprising the equations of motion for individual ions, and the electron fluid equations. The hybrid model used here neglects electron inertial effects; the electrons act as a massless fluid of constant and uniform temperature which serves to neutralise the plasma as well as to provide a pressure which acts on the ions. An important feature of any hybrid code is that the ion population is discretised into macroparticles, so there is always a noise level in the computed plasma density due to statistical fluctuations in the number of macroparticles per grid cell, even in a theoretically uniform plasma. The Vlasov-fluid equations and notation used in this paper are described in detail in Section 2.

In several respects, hybrid simulations lie between purely fluid, or magnetohydrodynamic simulations, and those run by particle codes, in which electrons as well as ions are

* Present address: Department of Physics, University of Natal, Durban 4001, South Africa.

0021-9991/94 $6.00

modelled as particles. First, the physical scales vary. An MHD model may be formulated for scales considerably greater than the ionic scales intrinsic to a hybrid model, although the spatial resolution places limits on the Reynold's number, the thickness of discontinuities, and the time-step. An explicit particle code must resolve electron behaviour at scales far smaller than for ions, so given the same computing resources a particle code models smaller regions for shorter times than a hybrid code. Second, modelling the particles requires much more computer memory (and consequently more time) than modelling a fluid. For each point in space where fluid variables are specified in an MHD code, a large number of macroparticles (usually between 10 and 100) is required to give even a coarse representation of their velocity distribution in a hybrid code. Thus for the same simulation grid, an MHD code uses much less computing power than a hybrid code, which in turn uses considerably less than a particle code.

Two-dimensional hybrid codes have been used to simulate ion beam instabilities [13, 6], shocks with ion reflection [10, 14], and various other instabilities ([4, 9] and [5] with a 3D code). The code described here is intended for simulating the solar wind, especially its bow shock with the earth and associated collisionless plasma physics.

1.2. *The Algorithm CAM-CL*

The plasma has two time-dependent components: ion macroparticles with positions and velocities and the magnetic field specified at the nodes of a regular, rectangular computing grid. Several species of ions with differing masses and charges may be modelled. The displacement current is neglected in Maxwell's equations, so there is no equation for the time-evolution of the electric field. Instead, it is determined by the requirement that the electric force balances the other forces acting on the massless electron fluid; the electric field is a function of the ion moments (interpolated at grid points from particle data), the magnetic field, and the electron temperature (constant and uniform). Bilinear interpolation is used for moment collection and evaluation of the Lorentz force at particle positions. An explicit form of the time-dependent differential equations is used. Implicit methods are generally more time-consuming than explicit methods, and they are global rather than local. Explicit methods are local and faster and are employed in all the 2D hybrid codes referred to in Section 1.1. Spatial derivatives are approximated by centred finite-differences.

CAM-CL is essentially a time-centred leapfrog algorithm for both particles and fields. This means that the time-derivative of a variable is evaluated at the midpoint of its time-step. Now if the time-derivative in question is itself a function of the variable to be advanced, then there must be some estimate of the variable a half time-step ahead. Two approaches to this problem are outlined by Press *et al.* [7]: the midpoint method and the modified midpoint method. The first approach is to make an initial half-step using the time-derivative at the beginning of the step. The second is to keep two solutions staggered a half time-step apart and to use each solution to evaluate the centred time-derivative of the other. The two solutions "leapfrog" over each other, and since they usually diverge, they may be periodically averaged. The difference between the solutions may serve as a check on their accuracy. The modified midpoint method as described by Press *et al.* [7] repeats this procedure with diminishing time-steps until the successive averaged solutions converge.

Cyclic leapfrog is the integration procedure of the modified midpoint method applied to the magnetic field. The label "cyclic leapfrog" has been chosen because it readily conveys the essence of the technique in the language of plasma simulation.

The particle positions are advanced by leapfrogging them with the velocities, which in turn are advanced by the midpoint method, using values of the electric and magnetic field interpolated at the particle positions a half time-step ahead. An initial half-step of the particle velocity is used to compute the time-centred acceleration. The whole procedure can be accomplished with a single computational pass through the particle data arrays—provided that the electric field is also known a half time-step ahead. Yet it depends on the ionic current density and hence the particle velocities, so at first sight a "pre-push" of velocities is needed to evaluate the time-centred electric field.

The *current advance method* resolves this problem by advancing the ionic current densities a half-step instead, with an appropriate equation of motion. This almost halves the total computing time compared with employing a pre-push of the particle velocities. By collecting appropriate moments of the particle distribution and applying an equation of motion, the ionic current density is solved to first order in the time-step.

An important feature of the algorithm is that the time-step for particle and field advance is different. In general, the magnetic field requires a smaller time-step, especially to resolve high-frequency behaviour (see Terasawa *et al.* [9]) such as dispersion.

1.3. *Comparison with Other Codes*

CAM-CL may be compared with other explicit multi-dimensional hybrid codes to place it in context theoretically, but a comparison of performances would require testing which is beyond the scope of this introduction. The codes chosen for comparison are: Harned's [4] predictor–corrector scheme, that of Terasawa *et al.* [9], the moment method described by Winske and Quest [14, Appendix A], and the 3D code QN3D presented by Horowitz *et al.* [5].

Harned's [4] predictor–corrector algorithm linearly extrapolates the electric field from two previous values to the midpoint of the particle velocity time-step. The particles are pushed once to obtain a final value of the fields, which in turn are used to push the particles a second time. Its main differences with CAM-CL are that pushing the particles twice requires more computer time, and the time-step for particles and fields is the same.

Horowitz *et al.*'s [5] three-dimensional code QN3D also has the same time-step for fields and particles. Functional iteration advances the fields: the average between the initial and the (iterated) final value is used to evaluate the time-centred derivative for the next iteration. The implicit equation for particle velocity advance is solved exactly; if terms above second order are dropped, the expression is equivalent to the midpoint method, so for a sufficiently small time-step these methods give very similar results. The essential difference with CAM-CL, without taking into consideration several optimisations in QN3D, is functional iteration versus sub-stepping of the fields.

Terasawa *et al.* [9] sub-step the fields for each particle step, as in cyclic leapfrog. A second-order rational Runge–Kutta algorithm [11] advances both fields and particles. CAM-CL adopts sub-stepping of the fields and adds the *current advance method.*

Winske and Quest [14, Appendix A] present a moment method, which Quest [8] describes as a variation of Terasawa *et al.*'s method. The fields are also sub-cycled, but with a fourth-order rational Runge–Kutta algorithm. The variation is the moment method: the plasma fluid velocity is advanced using an MHD equation which includes an advective term and an ionic stress tensor which must be collected.

The current advance method has several advantages compared with the moment method:

(1) multiple ion species may be modelled. The moment method is best suited to modelling a single species because it advances the fluid velocity (and hence would need to do this for each species to model multiple species), whereas in CAM the ionic current is easily advanced provided "free-streaming" moments are collected

(2) the numerically awkward advective term is absent. The collection of moments at the time-centred positions a half-step ahead includes the transport of particle momenta

(3) no ionic pressure tensor is collected (for the same reason as in (2)).

CAM-CL is in many respects an extension of the methods reviewed here. Harned [4] provides the structure of the computing grid, the electric field expression, and time- and space-centred finite differences. Terasawa *et al.* [9] introduce sub-stepping of the fields, and Winske and Quest [14] suggest advancing the moments directly to calculate the time-centred electric field instead of either a particle pre-push or functional iteration of the fields. CAM-CL takes a logical step from existing methods: cyclic leapfrog is a different way of sub-stepping, and the current advance method is the moment method applied to a particular moment, the ionic current density.

2. THE HYBRID MODEL

In terms of the hybrid model, the plasma is governed by the Vlasov-fluid equations

$$dx_s/dt = v_s \tag{1}$$

$$dv_s/dt = \frac{q_s}{m_s}(E + v_s \times B) \tag{2}$$

$$\partial B/\partial t = -\nabla \times E \tag{3}$$

$$\nabla \times B = \mu_0 J \tag{4}$$

$$n_e m_e \frac{du_e}{dt} = -n_e e E + J_e \times B - \nabla p_e \tag{5}$$

$$p_e = n_e \kappa T_e, \tag{6}$$

where the quantities and their symbols are ion position x_s, ion velocity v_s, ion mass m_s, ion charge q_s, electric field E, magnetic field B, magnetic permeability μ_0, current density J, electron number density n_e, electron mass m_e, electron fluid velocity u_e, magnitude of the electronic charge e, electronic current density $J_e = -n_e e u_e$, electron fluid pressure p_e, Boltzmann's constant κ, and electron temperature T_e. Equation (4) neglects the displacement current (the Darwin approximation), and (5) is the equation of motion for the electron fluid.

The subscript s may refer to individual particles which are members of an ion species of particle mass m_s and charge q_s, or to collective quantities evaluated for species s much as n_s and J_s defined with other quantities as

$$n_s = \int f_s(x_s, v_s)\, d^3 v_s \tag{7}$$

$$(nu)_s = \int v_s f_s(x_s, v_s)\, d^3 v_s \tag{8}$$

$$u_s = (nu)_s / n_s \tag{9}$$

$$\varrho_c = \sum_s n_s q_s \tag{10}$$

$$\varrho_m = \sum_s n_s m_s \tag{11}$$

$$J_s = q_s n_s u_s \tag{12}$$

$$J_i = \sum_s J_s \tag{13}$$

$$J = J_i + J_e, \tag{14}$$

where $f_s(x_s, v_s)$ is the species distribution function, and we denote species particle number density n_s, species "velocity density" $(nu)_s$, species fluid velocity u_s, charge density ϱ_c, mass density ϱ_m, species current density J_s, and ionic current density J_i.

The assumption that the electrons act as a massless, charge-neutralising fluid implies that $m_e = 0$ and $n_e e = \varrho_c$, so (5), together with (4) and (14), may be rearranged to give an expression for the electric field

$$E = -\frac{J_i \times B}{\varrho_c} + \frac{(\nabla \times B) \times B}{\mu_0 \varrho_c} - \frac{\nabla p_e}{\varrho_c} \tag{15}$$

so that $E = E(\varrho_c, J_i, B, T_e)$ is a state function. Substituting (15) into (3) gives

$$\frac{\partial B}{\partial t} = \nabla \times \frac{J_i \times B}{\varrho_c} - \nabla \times \frac{(\nabla \times B) \times B}{\mu_0 \varrho_c}. \tag{16}$$

The first term describes induction, the second, dispersion. The electron pressure in (6) does not influence magnetic field evolution.

Magnetic and ionic species (partial and combined) pressures

$$p_B = B^2/2\mu_0, \qquad p_s = \varrho_m^s (v_{th}^2)_s/2, \qquad p_i = \sum_s p_s,$$

where v_{th} is the ion thermal speed, are used to define the ratios

$$\beta_s = \frac{p_s}{p_B} = \frac{(v_{th}^2)_s}{v_A^2} \frac{\varrho_m^s}{\varrho_m},$$

$$\beta_i = \sum_s \beta_s,$$

$$\beta_e = \frac{p_e}{p_B} = \frac{2 n_e \kappa T_e}{\varrho_m v_A^2},$$

where v_A is the Alfvén speed given by $v_A^2 = B^2/\mu_0 \varrho_m$. The sound speed is given by

$$c_s^2 = \frac{(p_i + p_e)}{\varrho_m} = \frac{1}{2} (\beta_i + \beta_e) v_A^2.$$

3. CURRENT ADVANCE METHOD

3.1. *Optimising the Leapfrog Scheme*

The leapfrog scheme for particle advance is the midpoint method applied to Eq. (2), and a time-centred integration of Eq. (1). A time-step Δt is introduced, and quantities are evaluated at different time levels denoted by superscripts such that, relative to a time t_0,

$$x^k = x(t_k) = x(t_0 + k\,\Delta t).$$

The differential equations are rewritten as difference equations

$$x^{1/2} = x^{-1/2} + \Delta t v^0 \tag{17}$$

$$v^1 = v^0 + \Delta t \frac{q}{m} (E^{1/2}(x^{1/2}) + v^{1/2} \times B^{1/2}(x^{1/2})). \tag{18}$$

A primary objective in simulations is to minimise the number of operations required to push the particles, while simultaneously attaining acceptable accuracy in solving the equation of motion. A fundamental means of maximising speed is to find an algorithm that requires only one computational pass through the particle data arrays.

The problem with solving Eq. (18) is that it is implicit. Neither $v^{1/2}$ nor $E^{1/2}$ (which depends on all the $v^{1/2}$ via $J_i^{1/2}$) are known at the beginning of a time-step. The midpoint method solves Eq. (18) to second order in Δt by making a first-order half-step from v^0 to $v^{1/2}$,

$$v^{1/2} = v^0 + \frac{\Delta t}{2} \frac{q}{m} (E^{1/2} + v^0 \times B^{1/2}). \tag{19a}$$

If a suitable estimate of $E^{1/2}$ is made, then Eqs. (17), (18), and (19a) may be included in a single pass through the particle data arrays. The question is, how to obtain a suitable estimate of $E^{1/2}$?

First, a brief note on second-order solutions. Equation (2) may be written in the form $u' = au + b$, where $u' \equiv du/dt$, and $u(t)$, $a(t)$, and $b(t)$ are functions of time. If $u(0)$ is known then $u(h)$ may be approximated to second order in h by employing a "time-centred" derivative as follows:

$$\begin{aligned} u(h) &= u(0) + h u'(h/2) \\ &= u(0) + h \frac{d}{dt} \left[u(0) + \frac{h}{2} u'(0) \right] \\ &= u(0) + h u'(0) + \frac{h^2}{2} u''(0) + \text{error}(h^3). \end{aligned}$$

Thus $u(h/2)$ need only be approximated to first order in $h/2$ to give a second-order approximation for $u(h)$ in the expression above. It is now shown that the gradient $u'(*)$ at mixed time-levels (defined below) may be used to give a first-order accurate approximation for $u(h/2)$ when $a(h/2)$ and $b(h/2)$ are expanded to first order. First

$$\begin{aligned} u'(*) &= a(h/2)\, u(0) + b(h/2) \\ &= \left[a(0) + \frac{h}{2} a'(0) \right] u(0) + b(0) + \frac{h}{2} b'(0) \\ &= a(0)\, u(0) + b(0) + \frac{h}{2} [a'(0)\, u(0) + b'(0)] \\ &= u'(0) + \frac{h}{2} \varepsilon \end{aligned}$$

where $c = a'(0)\,u(0) + b'(0)$, and then

$$u(h/2) = u(0) + \frac{h}{2}u'(*)$$

$$= u(0) + \frac{h}{2}u'(0) + \text{error}(h^2)$$

This expression for $u(h/2)$ gives second-order accuracy for $u(h)$ in

$$u(h) = u(0) + h\frac{d}{dt}\left[u(0) + \frac{h}{2}u'(*)\right].$$

Returning to the question of a suitable estimate for $E^{1/2}$, it is clear that a straightforward answer is a "pre-push" of the particles to obtain $v^{1/2}$ and hence $J_i^{1/2}$. This solution requires two passes through the particle tables, the first of which (the "pre-push") employs a mixed time-level electric field $E^* = E(\varrho_c^{1/2}, J_i^0, B^{1/2}, T_e)$ in the equation

$$v^{1/2} = v^0 + \frac{\Delta t}{2}\frac{q}{m}(E^* + v^0 \times B^{1/2}). \tag{19b}$$

This equation is first-order accurate in Δt, as may be shown by consideration of the note on second-order solutions discussed above.

Two passes through the particle data arrays may be used to solve Eq. (18). However, the objective of a single pass requires a different approach:

Advance the ionic current density J_i^0 a half time-step to $J_i^{1/2}$ with an appropriate equation of motion.

Now only one pass through the particle velocity table is necessary to solve Eq. (18), since $E^{1/2}$ is computed as a function of $J_i^{1/2}$. This is called a *moment method*, which almost halves the computing time because of the small number of grid points involved in the computation of $J_i^{1/2}$ compared to the number of particles involved in a "pre-push" (typically 30 times more particles than grid points).

Since the ionic current density J_i is advanced, this method is termed the "current advance method" (CAM). Of course J_i is simply one kind of moment of the distribution function; another is the fluid velocity, which may also be advanced a half-step to obtain an estimate of $J_i^{1/2}$. However, the advantage of the current advance method is that it is straightforwardly applied to a multi-species plasma, whereas the fluid velocity approach would require treating each species separately. In addition, the advective and heat-flow terms are treated in a simple manner in CAM, as will be shown in the rest of Section 3.

3.2. *"Free-Streaming" Currents*

Current and charge densities are defined analytically by taking moments of the distribution function $f(x, v)$. In simulations they are determined at grid points by computing weighted sums over the particles. A weighting function $\phi_{sj} \equiv \phi(x_s, x_j)$ associates a particle position x_s with a grid point x_j [2], where $\phi_{sj} = \phi_{js}$, and $\sum_j \phi_{sj} = 1$ (i.e., each particle has unit weight). Ionic charge and current density are collected at grid points as

$$\varrho_c(x_j) = \sum_s \phi_{js} q_s \tag{20}$$

$$J_i(x_j) = \sum_s \phi_{js} q_s v_s. \tag{21}$$

It has been assumed that all quantities are specified at a given time. Now consider J_i evaluated in Eq. (21) with the particle variables given at different times: velocities at the beginning of the time-step v_s^0, and positions at its midpoint $x_s^{1/2}$. This is equivalent to neglecting the acceleration and only taking into account the transport of charge due to freely streaming particles, since the velocity of each particle has remained constant during the time interval $\Delta t/2$, whereas the particle position has changed. Applied to Eq. (20), the "free-streaming" charge density referred to by Friedman *et al.* [3] is obtained. Applying it to Eq. (21) yields the "free-streaming" ionic current density

$$J_i^*(x_s^{1/2}, v_s^0) = \sum_s \phi(x_s^{1/2})\, q_s v_s^0, \tag{22}$$

where the grid position x_j has been omitted in the notation, but the dependence of J_i on the particle positions and velocities has been emphasised. Note that in $J_i^*(x_s^{1/2}, v_s^0)$, x_s and v_s refer to all the particles.

3.3. *Advancing the Ionic Current Density*

An equation of motion is now derived for advancing J_i to the midpoint of the time-step. Consider ions at a grid point j (i.e., in a small neighbourhood of x_j). Multiply the "pre-push" equation (19b) by q_s (which may in general be different for each particle), and sum the contributions of the terms at the grid point, using weights $\phi_{sj}^{1/2} = \phi(x_s^{1/2}, x_j)$ evaluated at particle positions $x_s^{1/2} = x_s(t_0 + \Delta t/2)$:

$$\sum_s \phi_{sj}^{1/2} q_s v_s^{1/2} = \sum_s \phi_{sj}^{1/2} q_s v_s^0 + \frac{\Delta t}{2}\sum_s \phi_{sj}^{1/2}\frac{q_s^2}{m_s}(E^* + v_s^0 \times B^{1/2})$$

$$J_i^{1/2} = J_i^* + \frac{\Delta t}{2}(\Lambda E^* + \Gamma \times B^{1/2}) \tag{23}$$

$$\Lambda = \sum_s \phi_{sj}^{1/2}\frac{q_s^2}{m_s} \tag{24a}$$

$$\Gamma = \sum_s \phi_{sj}^{1/2}\frac{q_s^2}{m_s} v_s^0. \tag{24b}$$

Equation (23) is the equation of motion for the current advance method, giving the first-order time-advance of the ionic current density over a step $\Delta t/2$.

Thus an approximation to $J_i^{1/2}$ may be obtained by first collecting the "free-streaming" current $J_i^*(x_s^{1/2}, v_s^0)$, Λ and Γ during the loop that advances the particles, and then applying Eq. (23). This method does not require treatment of the advective and the heat-flow tensor terms found in the usual MHD formulation of the fluid equation of motion. Numerical difficulties in treating the advective term and the need to collect tensor components at each grid point (and for each species) are complications which CAM avoids.

The principal advantages of CAM are

(1) computing time is almost halved compared with a method requiring a particle "pre-push" to evaluate $J_i^{1/2}$

(2) multiple species may be modelled easily compared with a fluid velocity moment method in which each species would have to be treated separately

(3) complications of treating the advective term and heat flow tensor are avoided, since the "free-streaming" ionic current density, computed as a function of particle positions a half time-step ahead of the particle velocities, includes the contributions of advection and heat flow to $J_i^{1/2}$.

4. CYCLIC LEAPFROG AND TIME-STEP ESTIMATION

4.1. *Cyclic Leapfrog Applied to Magnetic Field Sub-stepping*

Magnetic field evolution is determined by Eq. (3), for which any one of a variety of numerical methods may be compatible with CAM. *Cyclic leapfrog* is an application of the modified midpoint method [7] to the magnetic field.

Two copies of the magnetic field are leapfrogged over each other and periodically averaged in a cycle of arbitrary length. In many situations the magnetic field requires a smaller time-step than the particles to resolve its evolution (see [9]), in particular the effect of dispersion. Cyclic leapfrog enables it to be sub-stepped through a cycle of smaller time-steps for each particle step. The electric field is evaluated as a function of the time-centred charge and current densities

$$E_p = E(\varrho_c^{1/2}, J_i^{1/2}, B_p, T_e)$$

and $B_0 \equiv B(t_0)$ is advanced from t_0 to $t_0 + H$ in a cycle of n substeps of size $h = H/n$, so that

$$B_p \equiv B(t_0 + ph)$$

as

$$\begin{aligned} B_1 &= B_0 - h\nabla \times E_0 \\ B_2 &= B_0 - 2h\nabla \times E_1 \\ &\vdots \\ B_{p+1} &= B_{p-1} - 2h\nabla \times E_p \\ &\vdots , \end{aligned}$$

where $p = 1, 2, \ldots, n-1$,

$$\begin{aligned} B_n &= B_{n-2} - 2h\nabla \times E_{n-1} \\ B_n^* &= B_{n-1} - h\nabla \times E_n \end{aligned}$$

and, finally,

$$B(t_0 + H) = \tfrac{1}{2}(B_n + B_n^*) \tag{25}$$

Computationally, this uses two copies of B, one for the "odd" solution (p odd, together with B_n^*) and one for the "even" solution (p even), which leapfrog over each other. After n steps as described in (25), the two solutions may either be averaged, or the cycle may be prolonged, starting with $B_{n+1} = B_n^* - h\nabla \times E(B_n^*)$. This may be prescribed by the algorithm, or alternatively the error between the two solutions B_n^* and B_n may be used as a criterion for averaging them and starting a new cycle, possibly with a different time-step.

4.2. *Time-step Estimation*

The time-step for both particles and magnetic field may be prescribed and kept constant for the duration of a simulation. On the other hand, a variable time-step may be estimated periodically or in the function of some criterion such as the error between two solutions in cyclic leapfrog B-advance. If a new time-step is chosen, then the particle positions and velocities must be synchronized.

The time-dependent differential equations are resolved if the time-step is some fraction of the inverse characteristic angular frequencies of the system. For Eqs. (1), (2), and (16) these frequencies are

$$\omega_L \sim kv, \qquad \omega_G \sim q_s B/m_s,$$
$$\omega_E \sim q_s E/m_s v, \qquad \omega_D \sim k^2 B/\mu_0 \varrho_c.$$

Here ω_L is a "linear" frequency characterized by a velocity v (which may be the maximal particle speed or the maximal fluid speed) and a maximal wave-number $k = \pi/\Delta x$ in a computing grid in which Δx is the cell size, ω_G is the ion gyrofrequency (also denoted by Ω_i), ω_E is related to electric field acceleration, and ω_D is the frequency of the dispersive effect in the magnetic field.

The dispersion frequency is usually higher than the other frequencies, especially at low densities, and it is this

high-frequency dispersion in the magnetic field which often requires that sub-stepping be employed.

5. NUMERICAL IMPLEMENTATION

5.1. *Normalisation and Units*

In simulations, given a multi-species plasma with unit mass density ϱ_{m0}, units are specified in terms of a reference plasma of protons of mass m_p and charge e, which has the same mass density ϱ_{m0} as the multi-species plasma. In this reference plasma, the proton number density $n_0 = \varrho_{m0}/m_p$. Dimensionless simulation values are related to physical values by $s = x/u$, where s is a simulation variable, x is the corresponding physical variable, and u is a physical unit value. Units are: *mass density* ϱ_{m0} (thus unit mass $= \varrho_{m0} \times$ (unit length)^3), *magnetic field* B_0 (which is an arbitrary value of a background field), *Alfvén speed* v_A (unit of speed), *ion inertial length* $c/\omega_{pi} = v_A/\Omega_i$ (unit of length), *cyclotron time* Ω_i^{-1} (unit of time), *charge density* $n_0 e$ (thus unit charge $= n_0 e \times$ (unit length)^3), *electric field* $v_A B_0$, and *unit of energy* $= \varrho_{m0}(c/\omega_{pi})^3 v_A^2$. In the above definitions, c is the speed of light, the gyrofrequency $\Omega_i = eB_0/m_p$, and the ion plasma frequency $\omega_{pi}^2 = n_0 e^2/\varepsilon_0 m_i$. Note that in terms of these units the proton charge-to-mass ratio is unity. In simulations the magnetic permeability, ionic and electronic betas, and sound speed respectively have values

$$\mu_0 = 1, \qquad \beta_s = \varrho_m^s (v_{th}^2)_s, \qquad \beta_e = 2\tau_e, \qquad c_s^2 = (\beta_i + \beta_e)/2,$$

where $\tau_e = \kappa T_e/e$ is a measure of the electron temperature, so $p_e = \varrho_c \tau_e$.

5.2. *Spatial Discretisation in 2D*

Spatial derivatives are approximated by finite-differences in a rectangular grid. The simulation is restricted to a rectangular plane which is divided into rectangular cells of size $(\Delta x, \Delta y)$. The fields are specified at the nodes of two interlaced grids:

- *Full-integer grid with nodes at cell vertices* $(n_x \Delta x, n_y \Delta y)$, $n_{x,y} = 1, 2, \ldots, N_{x,y}$, *at which* B, ϱ_c, J_i, *and* p_e *are specified*
- *Half-integer grid with nodes at cell centres* $((n_x + \frac{1}{2})\Delta x, (n_y + \frac{1}{2})\Delta y)$, *including a row of boundary cells just outside the simulation plane, at which* E *is specified.*

The electric field is on an interlaced grid because its curl is used to evaluate $\partial B/\partial t$, whereas p_e is on the same grid as B because ∇p_e is a term in the equation for E (Eq. (15)). The moments ϱ_c and J_i could, in principle, be specified on either grid, and here they are placed on the same grid as B because the $J_i \times B/\varrho_c$ term may be computed on the full-integer grid and then interpolated onto the E-grid. This interpolation serves the useful purpose of smoothing E a little. Another advantage of ϱ_c and J_i on the same grid as B is that they may be treated by the same routines in graphical and numerical diagnostics of results.

The spatial derivative $\partial/\partial x$ is approximated at a grid point by bilinear finite-differences on adjacent nodes on the interlaced grid. For example, at $x = (i + \frac{1}{2})\Delta x$, $y = (j + \frac{1}{2})\Delta y$,

$$\frac{\partial \psi}{\partial x}(i + \tfrac{1}{2}, j + \tfrac{1}{2}) = \frac{1}{2\Delta x}[\psi(i+1, j+1) - \psi(i, j+1) + \psi(i+1, j) - \psi(i, j)].$$

5.3. *Gridding and Interpolation*

Gridding (assigning values to fields at grid points by weighted sums over particles) has been discussed in Section 3.2 for the case of collection of moments at grid points x_j. In simulations, the electric and magnetic fields are specified at grid points on different grids and interpolated at particle positions x_s as

$$E(x_s) = \sum_j \phi(x_s, x_j^E)\, E(x_j^E)$$

$$B(x_s) = \sum_j \phi(x_s, x_j^B)\, B(x_j^B),$$

where x_j^E and x_j^B are positions of grid points in the E-grid (half-integer) and the B-grid (full-integer), respectively, and where $\phi(x_s, x_j)$ has been defined in Section 3.2.

Bilinear weighting is used for both gridding and interpolation. Given the normalised position $(x, y) \in [0, 1] \times [0, 1]$ in a grid cell, the bilinear weights at cell vertices are

$$\phi(0, 0) = (1 - x)(1 - y)$$

$$\phi(1, 0) = x(1 - y)$$

$$\phi(1, 1) = xy$$

$$\phi(0, 1) = (1 - x)\, y.$$

It is important to note that in simulations the "particles" are *not* ions, but *macroparticles*, each representing a very large number of ions. Therefore the distribution function is effectively discretised into a finite number (N) of ion clouds with centres x_s, where $s = 1, 2, \ldots, N$.

Bilinear weighting is equivalent to "binning" the ions in a macroparticle into cells enclosing grid points, where the macroparticle position is the centre of an ion cloud with the dimensions of a grid cell. The discretisation of phase space introduces noise into the moment arrays. A consequence is that there is always a noise level in the density and currents acting as a source of small perturbations. Therefore

"infinitesimal" disturbances cannot be simulated unless the number of particles is very great.

Smoothing of the moments may be used to damp the statistical noise. It also helps to stabilise the code against the influence of perturbations at the shortest wavelengths. Here a three-point smoothing stencil is applied in x and y, that is equivalent to

$$\psi(i)_{\text{smoothed}} = \tfrac{1}{4}\psi(i-1) + \tfrac{1}{2}\psi(i) + \tfrac{1}{4}\psi(i+1).$$

5.4. *Calculation of the Electric Field*

Following the notation of Section 5.3, let

$$\langle\psi\rangle \equiv \tfrac{1}{4}[\psi(0,0) + \psi(1,0) + \psi(0,1) + \psi(1,1)]$$

denote the cell-centred average of the field ψ, where $\psi(x_j^B)$ is sampled on the full-integer grid. Then Eq. (15) gives

$$E = -\frac{\langle J_i\rangle \times \langle B\rangle}{\langle \varrho_c\rangle} + \frac{(\nabla \times B)\times\langle B\rangle}{\mu_0\langle\varrho_c\rangle} - \frac{\nabla p_e}{\langle\varrho_c\rangle}.$$

It is evident that the $(\nabla \times B)\times B/\mu_0\varrho_c$ term renders the electric field very large at low densities relative to the background density ϱ_{c0}, which could make the code unstable. At low densities the dispersion frequency becomes too high for the (global) time-step to satisfy the Courant condition. Furthermore, the hybrid model by itself is not adapted to vacuums, since the displacement current is neglected (electric signals cannot propagate in free space) and the electric field is proportional to ϱ_c^{-1} (since E is derived from a balance of forces on the electron fluid; where there is no electron fluid, E is undefined). As a precautionary measure in situations when low relative densities may occur, a large number of magnetic field sub-steps are used and the electric field is set to zero in regions where the density is zero or below a minimum threshold.

5.5. *Initialisation and Boundary Conditions*

Initial conditions are specified by giving B, ϱ_m^s, and u_s (for each species s) as functions of x and y sampled at grid points x_j and also β_s and β_e, from which v_{th}^s and τ_e are obtained. The composition of a multi-species plasma is determined by the species charge-to-mass ratios $r_s = q_s/m_s$. A drifting Maxwellian is generated for each species with u_s and v_{th}^s, and particles are distributed randomly in the grid cells. The number of particles per grid cell (ng_s for species s) is one of the most important numerical parameters in the simulation, since it determines the resolution of phase space and largely governs the computer memory and time required for the simulation. The mass of a macroparticle is then $m_s = \varrho_m^s\,\Delta x\,\Delta y/ng_s$, and its charge is $q_s = r_s m_s$.

In the simulations presented here, periodic boundary conditions are applied to E in boundary cells, but not to B since it is obtained from E. Moments at opposite boundary points are added together, and particles crossing a boundary re-enter on the opposite side of the simulation plane.

5.6. *Time-Advance Algorithm*

In general, given $x^{1/2}$, v^0, B^0, ϱ_c^0, $\varrho_c^{1/2}$, J_i^0, Λ, Γ, and $J_i^+ = J_i^*(x^{1/2}, v^0)$ (with $T_e = \text{const}$), each step from t_0 to t_1 is made in a sequence

1. **advance B^0 to $B^{1/2}$, J_i^+ to $J_i^{1/2}$ and evaluate $E^{1/2}$**

$$B^{1/2} = B^0 - \int_0^{\Delta t/2} \nabla \times E(\varrho_c^0, J_i^0, B(t), T_e)\,dt$$

$$E^* = E(\varrho_c^{1/2}, J_i^0, B^{1/2}, T_e)$$

$$J_i^{1/2} = J_i^+ + \frac{\Delta t}{2}(\Lambda E^* + \Gamma \times B^{1/2})$$

$$E^{1/2} = E(\varrho_c^{1/2}, J_i^{1/2}, B^{1/2}, T_e)$$

2(a). **advance v^0 to v^1, $x^{1/2}$ to $x^{3/2}$**

$$v^{1/2} = v^0 + \frac{\Delta t}{2}\frac{q}{m}(E^{1/2} + v^0 \times B^{1/2})$$

$$v^1 = v^0 + \Delta t\frac{q}{m}(E^{1/2} + v^{1/2} \times B^{1/2})$$

$$x^{3/2} = x^{1/2} + \Delta t v^1$$

2(b). **collect moments in the same loop through the particles**

$$\varrho_c^{3/2} = \varrho_c(x^{3/2})$$

$$J_i^- = J_i^*(x^{1/2}, v^1)$$

$$J_i^+ = J_i^*(x^{3/2}, v^1)$$

$$\Lambda = \Lambda(x^{3/2}, v^1)$$

$$\Gamma = \Gamma(x^{3/2}, v^1)$$

3. **ϱ_c^1 and J_i^1 are obtained as averages, and $B^{1/2}$ is advanced to B^1**

$$\varrho_c^1 = \tfrac{1}{2}(\varrho_c^{1/2} + \varrho_c^{3/2})$$

$$J_i^1 = \tfrac{1}{2}(J_i^- + J_i^+)$$

$$B^1 = B^{1/2} - \int_{\Delta t/2}^{\Delta t} \nabla \times E(\varrho_c^1, J_i^1, B(t), T_e)\,dt$$

B is integrated in time by *cyclic leapfrog* (Eqs. (25)). The procedure is illustrated in Fig. 1.

At the beginning of a simulation, x, v, and B are known at time t_0, and at the end, a data set with synchronous

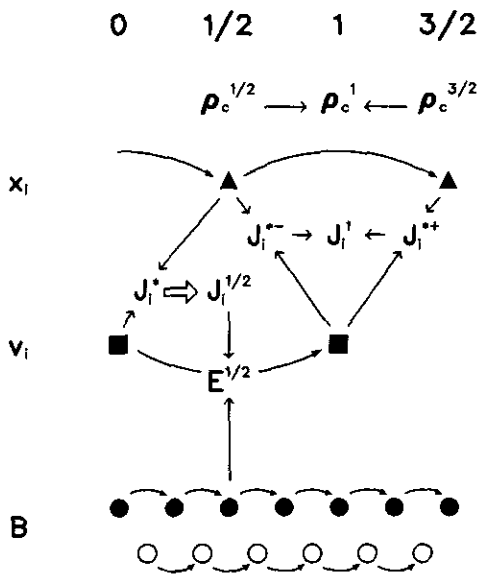

FIG. 1. Schematic diagram of the time advance scheme in CAM-CL. At the beginning of the step, x has been advanced to time level $\frac{1}{2}$ with v^0. Moments already collected are $\varrho_c^{1/2}$, the "free-streaming" ionic current density $J_i^*(x^{1/2}, v^0)$, as well as ϱ_c^0 and J_i^0. Two solutions of B are advanced by sub-steps (*cyclic leapfrog*) to time level $\frac{1}{2}$, with $E(\varrho_c^0, J_i^0, B, T_e)$. The *current advance method* advances J_i^* to $J_i^{1/2}$, with the fields $B^{1/2}$ and $E(\varrho_c^{1/2}, J_i^0, B^{1/2}, T_e)$. The time-centred (for v) electric field is now evaluated at time level $\frac{1}{2}$: $E^{1/2} = E(\varrho_c^{1/2}, J_i^{1/2}, B^{1/2}, T_e)$. The particles are pushed: $v^0 \to v^1$, $x^{1/2} \to x^{3/2}$, and moments collected: $\varrho_c(x^{3/2})$ from which ϱ_c^1 is obtained as an average of $\varrho_c^{1/2}$ and $\varrho_c^{3/2}$, and the backward and forward "free-streaming" currents $J_i^{*-}(x^{1/2}, v^1)$ and $J_i^{*+}(x^{3/2}, v^1)$, which are averaged to yield J_i^1. Finally $B^{1/2} \to B^1$.

quantities is desirable. Therefore a first and last step are necessary:

first step—given x^0, v^0, **and** B^0,

$$\varrho_c^0 = \varrho_c(x^0)$$
$$J_i^0 = J_i(x^0, v^0)$$
$$x^{1/2} = x^0 + \frac{\Delta t}{2} v^0$$
$$\varrho_c^{1/2} = \varrho_c(x^{1/2})$$
$$J_i^+ = J_i^*(x^{1/2}, v^0)$$
$$\Lambda = \Lambda(x^{1/2}, v^0)$$
$$\Gamma = \Gamma(x^{1/2}, v^0)$$

final step—retreat x **a half-step, collect** ϱ_m^1 **and** u_i^1,

$$x^1 = x^{3/2} - \frac{\Delta t}{2} v^1$$
$$\varrho_m^1 = \varrho_m(x^1)$$
$$u_i^1 = u_i(x^1, v^1)$$

The computation of the above sequences may be optimised. Moment collection, for instance, only requires the collection of species number density $n_s = \sum_i \phi_{ji}$ and "velocity density" $(nu)_s = \sum_i \phi_{ji} v_i$, where $i = 1, N_s$ is the particle index running through the particles of species s. The multiplication by the factors q_s, m_s, and q_s^2/m_s may be performed after a loop through the particles of each species:

2(b)(i). **collect species partial moments** n_s **and** $(nu)_s$,

$$n_s^{3/2} = \sum_s \phi(x_s^{3/2})$$
$$(nu)_s^+ = \sum_s \phi(x_s^{3/2})\, v_s^1$$
$$(nu)_s^- = \sum_s \phi(x_s^{1/2})\, v_s^1$$

2(b)(ii). **post-multiply and sum over species**,

$$\varrho_c^{3/2} = \sum_s q_s n_s^{3/2}$$
$$J_i^{\pm} = \sum_s q_s (nu)_s^{\pm}$$
$$\Lambda = \sum_s \frac{q_s^2}{m_s} n_s^{3/2}$$
$$\Gamma = \sum_s \frac{q_s^2}{m_s} (nu)_s^+$$

The price of this saving of time is that extra arrays are required to store the partial moments n_s and $(nu)_s$.

6. TESTS

A hybrid code should be able to model wave propagation and wave–particle interaction with reasonable accuracy and energy conservation. The tests which follow should give assurance that CAM-CL meets these criteria. All the simulations that follow were run on a VAX computer, so the dimensions of arrays were limited by the available memory. Boundary conditions were periodic. In all simulations four sub-steps of the magnetic field were made per particle step, the error between magnetic field solutions was calculated every five particle steps and was averaged if the error exceeded 10^{-4}. A constant particle time-step was chosen for each simulation. Simulation variables are used throughout this section. Units are defined in Section 5.1.

6.1. *Single Particle*

It is instructive to begin by considering energy conservation for a single particle in a uniform magnetic field. The particle energy remains constant. Standard leapfrog is a

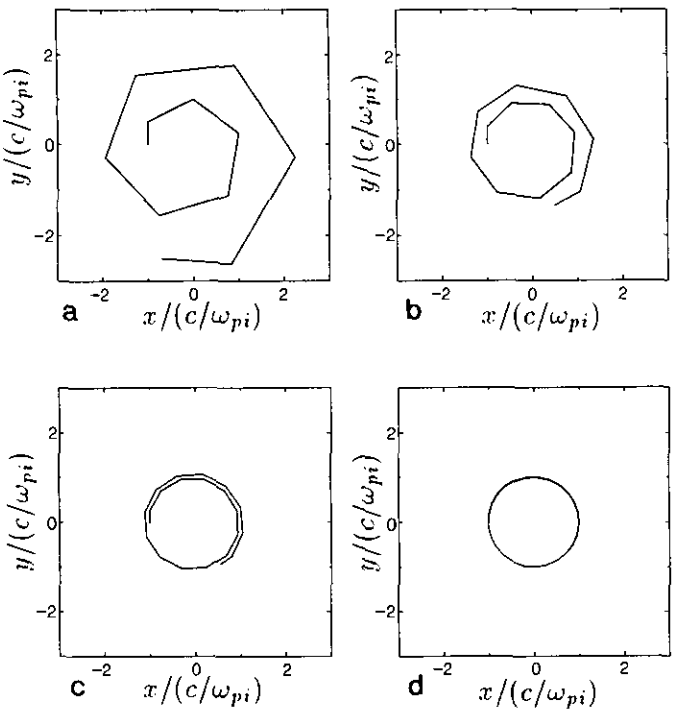

FIG. 2. Trajectories of single particles in the plane normal to a uniform magnetic field. Straight lines join positions computed with a leapfrog scheme over a time $t = 10\Omega_i^{-1}$ with various time-steps Δt: (a) 1, (b) 0.75, (c) 0.5, (d) 0.25. Initial speed $v_0 = v_A$.

second-order method for simulating the particle motion and introduces an error in the form of a gain in energy according to

$$E_n = E_0(1 + \Omega^4 \Delta t^4/4)^n,$$

where E_n is the kinetic energy after n steps, Ω is the gyrofrequency, Δt is the time-step, and the velocity is assumed to be perpendicular to B. Given $\Omega = 1$ and $\Delta t = 0.1$, the energy gain at $t = 100$ is 2.5%, and at $t = 300$ it is 7.8%. This sets a minimum level of energy conservation in a uniform plasma.

Figure 2 shows the trajectory of a single particle in a magnetic field $B = (0, 0, 1)$ with initial velocity $v_0 = (0, 1, 0)$. The particle gyrates for a time $t = 10$ with time steps Δt: 1, 0.75, 0.5, and 0.25. The largest step obeys the Courant–Lewy–Friedrichs condition for stability, but it is completely inaccurate. At $\Delta t = 0.25$ the trajectory is better resolved, but the energy is not well conserved, for at $t = 100$ the energy increase is almost 50%, whereas as the same time with $\Delta t = 0.1$, it is 2.5%, and for $\Delta t = 0.05$ it is 0.3%. An isotropic distribution of particles results in an energy gain

$$E_n = E_0\left(1 + \frac{2}{3}\frac{\Omega^4 \Delta t^4}{4}\right)^n$$

since $v_{||}$ is constant.

6.2. *Quiet Plasma*

A "quiet" or uniform plasma is a useful test that there are no obvious problems with the code such as programming bugs and algorithmic errors which render the plasma unstable or cause it to change in an unphysical way: a quiet plasma should remain quiet and unchanged, apart from a low level of turbulence due to statistical fluctuations in the particle distribution. If nothing at all happens that could also indicate a problem. This test is a useful way of setting bounds on values for numerical parameters such as the number of particles per cell, cell-size, and time-step, before setting up a simulation.

In conjunction with 2D tests, a series of 1D tests were run to investigate the influence of numerical parameters on energy conservation. One-dimensional simulations are much faster to run than their two-dimensional counterparts, and whereas it is recognised that 1D results do not necessarily hold in higher dimensions, they do give considerable insight and guidance in initialising more time-consuming 2D runs.

More than thirty 1D quiet plasma tests were run in which the following parameters were varied: number of particles per cell ng, cell size Δx, time step Δt, number of cells ncx, particle energy or thermal speed governed by β_i, uniform magnetic field direction (θ, ϕ in spherical polar coordinates), and smoothing of the electric field.

Results were compared with those of a reference case with rather bad energy conservation, with parameters $ncx = 16$, $\Delta x = 0.5$, $ng = 16$, $\Delta t = 0.1$, $\beta_i = 1$, $\beta_e = 0$, $B = B_0(\theta = \phi = 45°)$. β_i was chosen for approximate equipartition between the particles and the magnetic field, and $T_e = 0$, so as to eliminate the non-adiabatic influence of the electrons, which may act as a source or sink of heat. The magnetic field direction was chosen to have a component in each dimension. Particle, magnetic, and total energies were recorded at $t = 100$ (the time-scale within which many simulations are run) and $t = 300$ (primarily to test for long-term stability).

Now it is to be expected that energy is not conserved; after all, a simulation is essentially a polynomial (or rational function) extrapolation which by its nature diverges as t becomes large. What is important is that energy is conserved within reasonable limits within the time-scale of interest, that is to say that the energy increase is small compared with the total energy and small compared to the energy exchange within the system, e.g., between waves and particles in an ion–beam simulation.

The reference case had energy increases at $t = 100$ of 9% (total), 14% (particles), and 2% (magnetic field) and at $t = 300$, 47% (total), 77% (particles), and 4% (magnetic field). The bulk of the energy gain was in the particles, and this was the pattern in all the tests: the particles were being heated, but the magnetic field remained stable and relatively constant well beyond $t = 100$.

The following principal conclusions were drawn from the tests:

(1) Energy conservation improves most strongly with increased number of macroparticles per grid cell (ng). Varying only ng from the reference case resulted in total energy gains at $t = 300$ of 180% ($ng = 8$), 47% ($ng = 16$), 17% ($ng = 20$), and 14% ($ng = 32$). At $t = 100$, total energy gain was 4% for $ng = 32$, which is within acceptable bounds for a simulation.

(2) Cell-size (Δx) influences energy gain. Varying Δx from the reference case resulted in energy gains at $t = 300$ of 7% ($\Delta x = 0.25$, $ncx = 16$), 47% ($\Delta x = 0.5$, $ncx = 16$), 21% ($\Delta x = 1$, $ncx = 8$), and 10% ($\Delta x = 2$, $ncx = 8$). Energy conservation is best for small Δx ($\leqslant 0.5$), since spatial resolution is better, and for large Δx ($\geqslant 1$), since small-scale spatial fluctuations are suppressed. Small cell-size is preferable, since large cell-size also suppresses physical phenomena, notably dispersive waves in the magnetic field. An example of this is illustrated in Section 6.4.

(3) Time-step (Δt) had the least influence on energy conservation, provided that the time-scales of interest were well resolved. Halving Δt from 0.1 (reference case, total energy gain 47%, $t = 300$) to $\Delta t = 0.05$ resulted in a reduction of total energy gain to 36%. The fact that reducing Δt does not lead energy gain to approach the minimum level for standard leapfrog in a strictly uniform magnetic field, suggests that it is fluctuations in the field quantities which are responsible for further non-conservation.

Further conclusions were:

(4) Smoothing of the electric field (implemented by Winske and Quest [13]) improved total energy conservation markedly to 3.5% at $t = 100$ (9% for the reference case) but only 22% at $t = 300$ (47% for the reference case). This suggests that, whereas in the short-term smoothing E yields good energy conservation, in the longer term it is not sufficient by itself to maintain it. It has the disadvantage that small-wavelength physical structures may be damped.

(5) The angle of the magnetic field influences energy conservation; it is not isotropic. In 1D, the x-component is constant, and when it was set initially to zero, total energy gain varied between 3% and 6% at $t = 300$ (47% for the reference case). Other results were: 4% for $B = (1, 0, 0)$, 12% for $B = (1/\sqrt{2}, 1/\sqrt{2}, 0)$, and 14% for $B = (1/\sqrt{2}, 0, 1/\sqrt{2})$. It may be seen from the equations governing the magnetic field that the choice of initial orientation affects the way in which magnetic field fluctuations are excited and coupled, which in turn affects the spurious heating of particles.

(6) Small ion speeds improve energy conservation. This is not surprising, since fluctuations in the velocity field and, hence, the magnetic field are low as a consequence.

These 1D tests show that energy conservation is a complicated function of a number of variables, depending both on easily analysed errors in standard leapfrog and on less discernible influences of statistical fluctuations arising from the finite number of macroparticles per grid cell. In general, CAM-CL is most reliable and conserves energy best if the number of particles per grid cell is maximised, given that temporal and spatial scales are well-resolved (i.e., Δt and Δx should be substantially less than the thresholds of resolution, although reducing them further does not proportionally improve energy conservation). Recommended values are $ng \geqslant 30$, $\Delta t \leqslant 0.1$, and $\Delta x \leqslant 0.5$. These tests illustrate that pilot 1D runs of planned 2D simulations give considerable insight into optimising parameters and should be used as an integral tool in simulations in higher dimensions.

A two-dimensional quiet plasma simulation was run with the following parameters: a grid of 64×64 cells of size $\Delta x = \Delta y = 0.5$, 32 particles per cell, $\beta_i = 1$, $\beta_e = 0$, $B = B_0(\theta = \phi = 45°)$, and time step $\Delta t = 0.1$. At $t = 100$, energy gain was 2.6% (total), 3.8% (particles) and 0.7% (magnetic field). At $t = 300$ it was 14% (total), 23% (particles) and 1% (magnetic field). Halving cell-size made very little difference (total energy gain of 15% at $t = 300$), whereas decreasing number of particles per cell to 20 had a substantial effect (total energy gain 26% at $t = 300$).

These 2D tests underline the conclusions drawn from the 1D tests that energy conservation depends most strongly on number of particles per grid cell. Furthermore, it must be emphasized that although simulations with energy gains of 10% and more at $t = 300$ have limited physical validity, the tests demonstrate the important property that CAM-CL is stable up to times of at least $t = 300\Omega_i^{-1}$.

6.3. *MHD Wave Propagation*

The propagation of fast, slow and Alfvén waves in the non-dispersive limit is a feature of magnetohydrodynamics that a hybrid code should be capable of modelling. Figure 3 shows MHD wave velocities for two different values of β, 0.5 in Fig. 3a, and 4 in Fig. 3b (with β equally divided between electrons and protons). In each case slow, fast, and Alfvén sinusoidal waves of wavelength $\lambda = 50c/\omega_{pi}$ propagated in various directions to a uniform magnetic field; their speeds are plotted on a polar diagram and compared with the theoretical speeds in the non-dispersive limit. The parameters were: a grid of 100×10 cells of size $\Delta x = \Delta y = 0.5$, 30 particles per cell. Propagation angles were 15°, 30°, 45°, 60°, and 75°, with a parallel Alfvén wave as well. The simulations were run for a time $t = 10$ with a time step $\Delta t = 0.1$. The speeds were calculated using a least-squares fit with the theoretical oscillating magnetic field component in all cases except for the slow waves in the low-beta case where the density was used, because the signal-to-noise

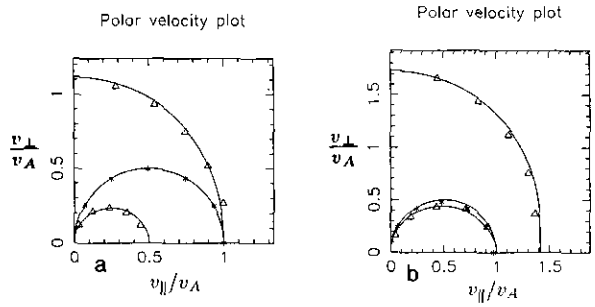

FIG. 3. Polar plots of MHD wave speeds at various angles to the magnetic field. Triangles represent fast and slow magnetosonic waves, and asterisks represent Alfvén waves. Panel (a) is for $\beta = 0.5$ and (b) is for $\beta = 4$.

ratio in the magnetic field was too low. This is seen in Fig. 4a for a slow wave and the corresponding situation in the $\beta = 4$ case, where the noise level is lower in the magnetic field than the density (Fig. 4b). The tests show good agreement between simulations and theory in the non-dispersive limit.

6.4. *The influence of Cell-size*

This test is an illustration of a numerical artefact rather than a test of the two-dimensional code. It consists of the evolution of a Gaussian density pulse in one dimension, with the other parameters satisfying the differential equations for a fast MHD simple wave given by Akhiezer (1975). In order to minimise the influence of statistical noise, the ions were cold (with 30 macroparticles per cell), whereas $\beta_e = 0.5$. A pulse of magnitude $\delta\varrho/\varrho_0 = 0.5$ and width $5c/\omega_{pi}$ was placed in a 1D box of length $100c/\omega_{pi}$, in a uniform magnetic field (locally perturbed by the MHD pulse). Three cases were simulated: (1) propagation at 30° to the

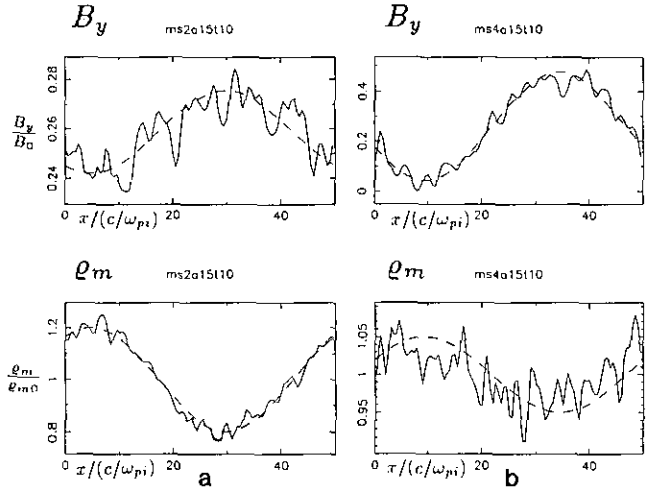

FIG. 4. Mass density and B_y profiles for two slow waves at $\theta = 15°$ in Fig. 4: (a) $\beta = 0.5$; (b) $\beta = 4$.

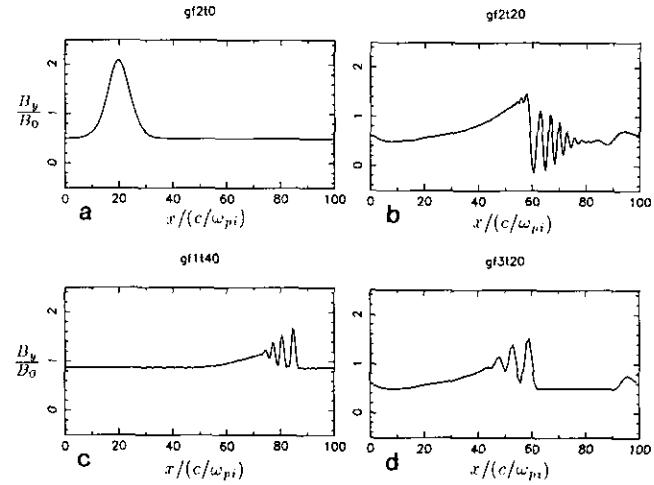

FIG. 5. B_y vs x profiles of an initially Gaussian fast MHD pulse, $\beta_i = 0$, $\beta_e = 0.5$, with (a) $\theta = 30°$, $\Delta x = 0.5$, $t = 0$, which evolves into (b) at $t = 20$, (c) $\theta = 60°$, $\Delta x = 0.5$, $t = 40$, (d) $\theta = 30°$. $\Delta x = 1$, $t = 20$.

undisturbed magnetic field, cell size $\Delta x = 0.5$, (2) propagation at 60°, $\Delta x = 0.5$, (3) propagation at 30°, $\Delta x = 1$.

The effect of low spatial resolution is obvious in the plots of B_y shown in Fig. 5. Figure 5a is the initial profile at $t = 0$ which evolves into Fig. 5b, resembling a whistler mode (simulation 1). Figure 5c is simulation (2) at $t = 40$, which resembles a train of solitons. Simulation (3) (Fig. 5d) has the same physical parameters as (1) (Fig. 5b), yet it resembles a train of solitons. It is clear that the lower spatial resolution in simulation (3) dramatically affects the evolution of the pulse. In particular, the length scale c/ω_{pi} is not resolved, so that dispersive effects are suppressed. At the lower resolution the whistler mode is unable to develop; the large cell size introduces a numerical viscosity which balances the steepening of the pulse and gives rise to a soliton-like structure.

6.5. *Ion Beam Instability*

A low-density ion beam interacting with a background plasma is a test of anisotropic wave–particle interaction which utilises the multi-species capacity of CAM-CL. Here two cases are studied for which the physical parameters follow Winske and Quest [13]: a resonant case and a non-resonant case.

In both cases a beam of ions of density n_b and speed $v_b = 10v_A$ propagated initially parallel to a uniform magnetic field B_0 through a core (background) plasma of density n_c. Both core and beam have thermal speeds $(v_{th})_{b,c} = v_A$. Electron temperature T_e was set to zero. The *resonant beam* had a density $n_b = 0.015n_c$, with core speed $v_c = -0.2v_A$, resulting in a plasma with zero momentum in the simulation frame. The *non-resonant beam* had $n_b = 0.1n_c$ and $v_c = -1.1v_A$.

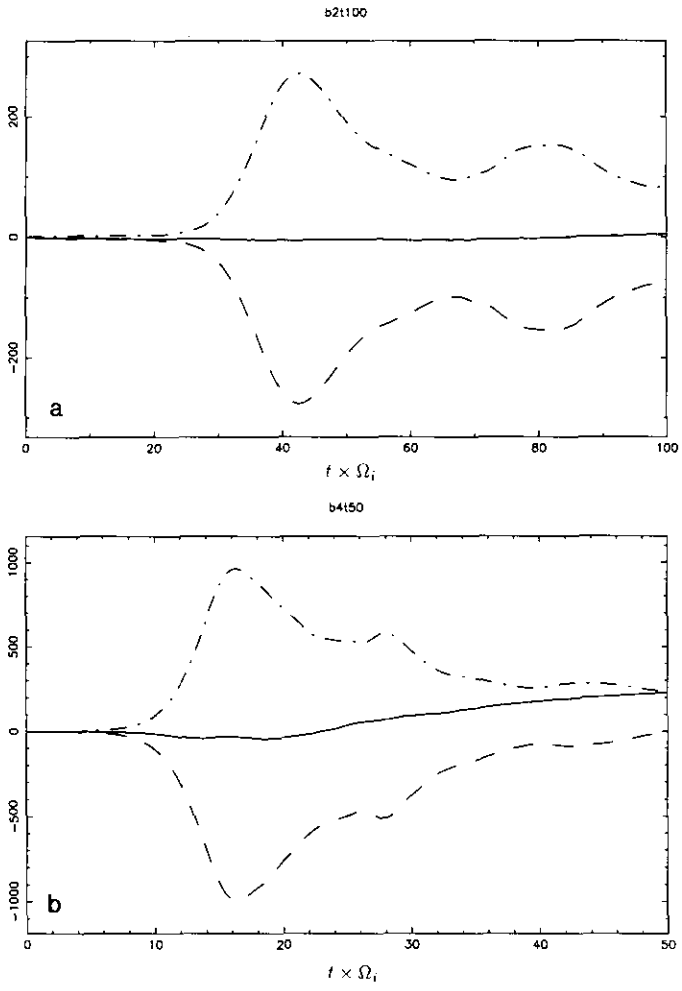

FIG. 6. Time histories of variations in total energy (solid line), particle energy (dashes), and magnetic field energy (dots and dashes), for ion beam simulations: (a) resonant case; (b) non-resonant case. Energy units are defined in Section 5.1.

The multi-species capacity of the CAM-CL algorithm was exploited to model the beam and core ions as separate species with 32 particles per grid cell, but with different macroparticle masses and charges.

A series of 1D resonant ion beam simulations were run which showed that the results vary with cell-size Δx. The peak of the curve of magnetic energy versus time varied by more than 10% about a mean in both magnitude and location on the time-axis. Tests of the 1D resonant beam showed that energy conservation improves with smaller cell size (11.4% at $t = 50$ for $\Delta x = 1$; 2.8% at $t = 50$ for $\Delta x = 0.5$).

Two-dimensional results are presented which were constrained by computer memory to simulation grids of 256×4 cells for the resonant case and 128×4 cells for the non-resonant case (i.e., the code was tested in its 2D implementation, but in quasi-1D scenarios). The cell size was $\Delta x = \Delta y = 1$ and the time-step was $\Delta t = 0.025$. The magnetic field was $B_0 = (1, 0, 0)$.

Time histories of the variation in total, particle, and magnetic energies are plotted in Fig. 6. The exchange of energy between particles and magnetic field is clearly illustrated, and the total energy remains constant in the resonant case (gain of 0.2% at $t = 100$) and reasonably constant in the non-resonant case (gain of 6% at $t = 50$; this energy gain is much less than the energy exchanged between particles and the magnetic field). One-dimensional tests discussed above suggest that energy conservation in the non-resonant case could be improved by reducing the cell-size. The magnetic energy curves compare well with those presented by Winske and Quest [13], bearing in mind the conclusions of 1D tests that variations of 10% in computed quantities due to numerical and algorithmic differences should not be surprising.

For further comparison the y-component of the magnetic field at various times is plotted in Fig. 7. The magnitude and wavelength of the excited modes are similar to those in

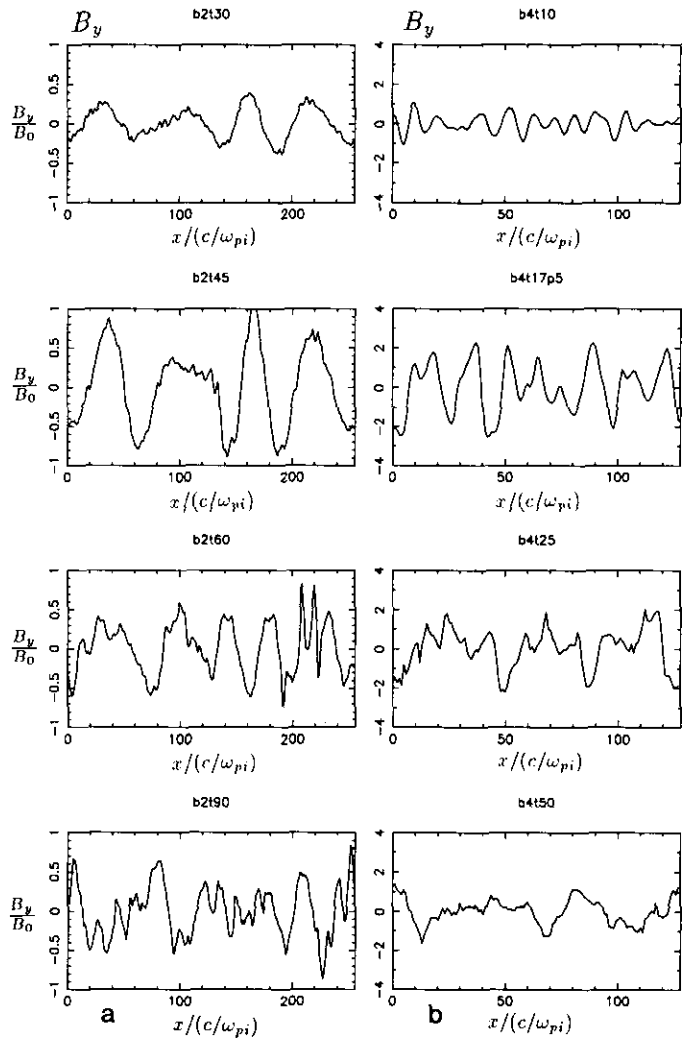

FIG. 7. Profiles of B_y vs x at various times for ion beam simulations: (a) resonant case; (b) non-resonant case.

Winske and Quest [13]. Finally, the beam phase space near the magnetic energy maximum in time is plotted in Fig. 8 (non-resonant case), showing the trapping of particles and the isotropisation of the beam.

A more comprehensive study of ion beams is beyond the scope of this section. Other quasi-1D tests were run, with $\beta_e = 1$, which showed very little difference to the tests presented here. The resonant case was run with $\Delta x = \Delta y = 2$ and showed an extra local maximum on the magnetic energy curve when compared with the corresponding simulation in Winske and Quest [13], indicating that CAM-CL (in its present implementation) may be most reliable when $\Delta x \leqslant 1$. Further investigation of ion beams in two dimensions at higher spatial resolution is planned.

The conclusion of this section is that the main features of Winske and Quest's [13] quasi-1D simulations were successfully modelled by CAM-CL, exploiting its multi-species capability with acceptable energy conservation.

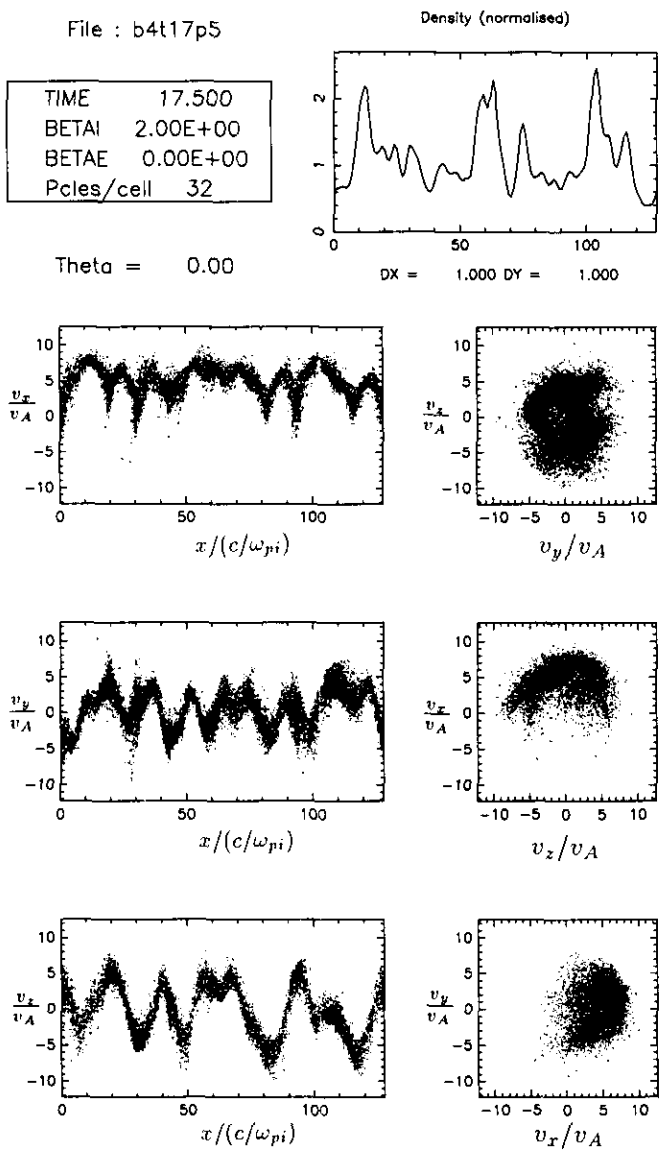

FIG. 8. Beam phase space for the non-resonant ion beam simulation.

7. CONCLUSION

CAM-CL (current advance method and cyclic leapfrog) is an algorithm for hybrid plasma simulations. In common with existing methods [4, 9, 14] its physical basis is a "hybrid" model in which ions are treated as particles and the electrons as a massless fluid.

Four main features distinguish CAM-CL from previous methods. First, it is capable of treating multiple ion species with only one pass through the particle data tables, and this is achieved without extrapolating the electric field forwards in time. Advancing the particle velocities through a timestep Δt ($v^0 \to v^1$) requires the time-centred electric field $E^{1/2}$ which depends on the ionic current density $J_i^{1/2} = \sum \phi q v^{1/2}$. At the beginning of a time-step, $v^{1/2}$ is not known, so CAM advances the ionic current density a half time-step ($J_i^0 \to J_i^{1/2}$). Then $E^{1/2}$ is evaluated as a function of $J_i^{1/2}$ rather than by an extrapolation in time, and $J_i^{1/2}$ is obtained without a time-consuming "pre-push" of the particle velocities ($v^0 \to v^{1/2}$). Hence only one pass through the particle velocity data is required.

The second distinguishing feature is the principal difference between CAM and the moment method [14, Appendix A]: CAM advances the ionic current density J_i, whereas the moment method advances the fluid velocity. This difference enables CAM to easily model multiple ion species. Another difference is presented here as the third distinguishing feature: CAM collects "free-streaming" currents $J_i^* = \sum \phi(x^{1/2})\, q v^0$ which are then advanced in time with an equation of motion. This avoids treatment of the advective and ionic stress tensor terms in the moment method.

Fourth, CL is an application of the modified midpoint method [7] to magnetic field evolution. Two solutions of the magnetic field are leapfrogged and periodically averaged after cycles of arbitrary length. CL is very stable and permits a smaller time-step for the magnetic field than for the particles, which is useful for resolving magnetic dispersion (prevalent in non-linear domains such as shocks) without pushing the particles too often.

CAM-CL is presented here as an algorithm applicable to hybrid simulations in one or more dimensions, rather than as an optimised code. Numerous improvements in the 2D code (H2CAM) used in the tests of CAM-CL are feasible; no claim is made that the current implementation is optimal. A variety of powerful techniques such as multi-gridding, particle sorting techniques, and perhaps higher-order spatial derivative evaluations could be applied, not to mention machine-dependent improvements. Also, particles could be advanced with a higher-order scheme. However, the objective here is to present the original aspects of the

algorithm, as well as tests of the method with a relatively simple code.

The tests provide evidence that the algorithm acceptably models the physics; a quiet plasma remains quiet and stable, with energy conserved to within 5% over $100\Omega_i$, which is a typical time-scale of interest. One-dimensional tests indicate that statistical noise from low numbers of macroparticles is a major factor in increasing the total energy. MHD waves propagate at the correct speed in the non-dispersive limit. Ion beams show instability behaviour similar to that in Winske and Quest [13], and tests on the evolution of a fast MHD pulse show that cell sizes above c/ω_{pi} can distort the physics and should be used with care.

Problems envisaged for application of CAM-CL are: turbulence in the solar wind, the evolution of large-amplitude waves, ion beam instabilities, and collisionless shocks.

ACKNOWLEDGMENTS

I thank André Mangeney, David Burgess, Filippo Pantellini, Marco Velli, Roland Grappin, Omar Elbachiri, and Michel Moncuquet for advice and assistance. The constructive comments of the referees are gratefully acknowledged. I am grateful for financial support from the French Ministry of Foreign Affairs, the French Ministry of Education, and the European Network for Numerical Simulation of Space Plasmas, EEC Contract SCIENCE-0468M.

REFERENCES

1. A. I. Akhiezer, I. A. Akheizer, R. V. Polovin, A. G. Sitenko, and K. N. Stepanov, *Plasma Electrodynamics. Vol. 1. Linear Theory* (Pergamon, Elmsford, NY, 1975).
2. C. K. Birdsall and A. B. Langdon, *Plasma Physics via Computer Simulation* (McGraw-Hill, New York, 1985).
3. A. Friedman, S. E. Parker, S. L. Ray, and C. K. Birdsall, *J. Comput. Phys.* **96**, 54 (1991).
4. D. S. Harned, *J. Comput. Phys.* **47**, 452 (1982).
5. E. J. Horowitz, D. E. Schumaker, and D. V. Anderson, *J. Comput. Phys.* **84**, 279 (1989).
6. A. Mankofsky, R. N. Sudan, and J. Denavit, *J. Comput. Phys.* **70**, 89 (1987).
7. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes* (Cambridge Univ. Press, Cambridge, 1986).
8. K. B. Quest, "Hybrid Simulation," in *Third International School for Space Simulation, Tutorial Courses*, June 15–20, 1987 (*CRPE*), Vol. 177 (Cepaudes-Editions, Toulouse, 1989).
9. T. Teresawa, M. Hoshino, J.-I. Sakai, and T. Hada, *J. Geophys. Res.* **91** (A4), 4171 (1986).
10. V. A. Thomas and S. H. Brecht, *J. Geophys. Res.* **92**, 3175 (1987).
11. A. Wambecq, *Computing* **20**, 333 (1978).
12. D. Winske, *Space Sci. Rev.* **42**, 53 (1985).
13. D. Winske and K. B. Quest, *J. Geophys. Res.* **91** (A8), 8789 (1986).
14. D. Winske and K. B. Quest, *J. Geophys. Res.* **93** (A9), 9681 (1988).