Journal of Computational Physics **183,** 470–485 (2002)
doi:10.1006/jcph.2002.7203

3D Hybrid Simulation Code Using Curvilinear Coordinates

T. Bagdonat and U. Motschmann

Institute for Theoretical Physics, Techn. Univ. of Braunschweig, Mendelssohnstr. 3, D-38106 Braunschweig, Germany
E-mail: t.bagdonat@tu-bs.de

Received October 29, 2001; revised September 5, 2002

A new simulation code using the hybrid approximation for modeling extraterrestrial plasma processes is described, which can be used in an arbitrary three-dimensional, ordered, hexahedral grid. Maxwell's equations are transformed using common tensor analysis and solved by a finite differencing scheme. A particle coalescing technique was adopted to account for differences in cell size. The numerical techniques and some results are presented. © 2002 Elsevier Science (USA)

Key Words: hybrid; plasma; simulation; 3D; curvilinear.

1. INTRODUCTION

The hybrid approximation of a plasma is a combination of a full particle approach and a fluid approach. It models the plasma dynamics by treating the ions as particles and the electrons as a charge-neutralizing fluid.

The earliest versions used massless electrons and an isothermal equation of state in one dimension [21], which were soon extended up to two spatial dimensions [4, 11, 12]. Furthermore, models taking the electron inertia into account have been developed in one [16] and two [13] dimensions.

Later several authors introduced the hybrid technique to the field of astrophysics and simulated collision-free shock waves [24, 32]. An introduction of a two-dimensional code can be found in [32, 33]. At this time also several improvements to the numerical algorithm were proposed, like the predictor–corrector scheme of Harned [11], the so-called "substepping" for the integration of the magnetic field equation [31], a "moment method" applying a fourth-order Runge–Kutta scheme in two dimensions [35], and implicit schemes for the calculation of the electric field with the three-dimensional code QN3D [14]. Brecht and Thomas extended Harned's algorithm to three spatial dimensions [3].

At the beginning of the last decade Matthews combined some of these ideas to develop a new method for the integration of the field equations, the so-called "current advancement

0021-9991/02 $35.00

method" (CAM) [22], and implemented this in a two-dimensional code with massless electrons. A more recent two-dimensional code with finite electron mass was developed by Shay *et al.* [29] and also Lipatov and Buechner [17], who furthermore uses an implicit scheme for solving the field equations, which was already extended to three dimensions [17].

A general introduction to the topic of hybrid code simulation can be found in [34].

All mentioned codes use a Cartesian or at least orthogonal grid for simulation. The only one which uses a nonorthogonal grid is the code by Swift [30], which is based mainly on the method of Harned [11]. To solve the field equations in a curvilinear grid, the method of Madsen [20] was adopted.

In view of the upcoming Rosetta mission, simulations of the interaction of solar wind plasma with the cometary coma seems desirable. It quickly became clear in earlier simulations using the fluid model in two dimensions [2, 19] that the draping of the magnetic field lines in front of the nucleus is not modeled correctly by two-dimensional models. Three-dimensional fluid simulations of this problem were done by Fischer [8] and hybrid simulations by Lipatov *et al.* [18]. These results suggest that the large tail structures may be influenced strongly by much smaller scale processes in the vicinity of the nucleus. Therefore it is necessary to increase the local resolution near the nucleus, which can only be done using a code with a nonorthogonal simulation grid. The technical details of this new code, which is mainly based upon the algorithm of Matthews [22] for time integration and the method of Eastwood *et al.* [7] for the spatial derivations, are described in the following sections.

2. BASIC EQUATIONS

2.1. Hybrid Model Equations

The hybrid model assumes the electrons to be a masseless, charge-neutralizing fluid, whereas the ions are simply treated as macroparticles, which models physical structures within a time scale below the inverse lower hybrid frequency.

The assumption of a massless electron fluid yields a momentum equation,

$$n_e m_e \frac{d\mathbf{u}_e}{dt} = 0 = -n_e e\mathbf{E} + \mathbf{J}_e \times \mathbf{B} - \nabla p_e + n_e m_e \nu_{ei}(\mathbf{u}_{\text{ion}} - \mathbf{u}_e), \tag{1}$$

where the l.h.s. is zero, due to the assumption of massless electrons ($m_e = 0$). $\mathbf{u}_e$ and $\mathbf{u}_{\text{ion}}$ are the bulk velocities of the electrons and ions, respectively. ν_{ei} is the anomalous ion–electron collision frequency, which may arise, for example, from microscopic wave-particle scattering. From (1) one obtains an expression for the electric field $\mathbf{E}$, and together with Faraday's law, the quasineutrality condition $n_i = n_e$ and $\mathbf{J} = \mathbf{J}_{\text{ion}} + \mathbf{J}_{\mathbf{e}} = n_i e\mathbf{u}_{\text{ion}} - n_e e\mathbf{u}_e$, one obtains a partial differential equation for the time development of the magnetic field in the form

$$\frac{\partial \mathbf{B}}{\partial t} = \underbrace{\text{curl}\frac{\mathbf{J}_{\text{ion}} \times \mathbf{B}}{\rho_c}}_{\text{kinetic term}} - \underbrace{\text{curl}\frac{\text{curl}\,\mathbf{B} \times \mathbf{B}}{\mu_0 \rho_c}}_{\text{Hall term}} - \underbrace{\eta\,\text{curl}\frac{\text{curl}\,\mathbf{B}}{\rho_c}}_{\text{resistive term}}, \tag{2}$$

where the displacement current is neglected. η is a parameter depending on ν_{ei}. ρ_c and $\mathbf{J}_{\text{ion}}$ are the charge and current density of the ions, respectively.

For the ions the equations of motion

$$\frac{d\mathbf{x}_p}{dt} = \mathbf{v}_p, \quad \frac{d\mathbf{v}_p}{dt} = \frac{q_p}{m_p}(\mathbf{E}' + \mathbf{v}_p \times \mathbf{B}) \tag{3}$$

have to be solved, where the index "p" runs through all particles which may have different charges q_p and masses m_p. To conserve momentum one has to use $\mathbf{E}' = \mathbf{E} - \eta/\rho_c \text{ curl } \mathbf{B}$ in the above equation.

To close this set of equations the kinetic electron pressure is assumed to be adiabatic; i.e.,

$$p_e = p_{e0}\left(\frac{\rho_e}{\rho_{e0}}\right)^{\kappa}, \tag{4}$$

where ρ_e is the electron charge density and κ is the adiabatic exponent. ρ_{e0} and p_{e0} are the initial background quantities.

2.2. Normalization Units

All quantities used in the simulation and in the following results are normalized; e.g.,

$$A^* = \frac{A}{A_0},$$

where A is some physical quantity, A^* is the dimensionless quantity used in the simulation, and A_0 is an appropriate normalization value.

In our code we use the following normalization values:

B_0	background magnetic field
n_0	background ion density
$t_0 = \Omega_i^{-1}$	Ω_i background ion gyrofrequency
$x_0 = c/\omega_{p,i}$	$\omega_{p,i}$ background ion plasma frequency
$v_0 = v_A$,	v_A background Alfvén velocity
$J_0 = e v_A n_0$	e electron charge ($e > 0$)
$p_0 = B_0^2/2\mu_0$	

3. NUMERICAL SCHEME

3.1. Time Integration Scheme

The time integration scheme for solving Eqs. (2) and (3) is mainly adopted by the code described in [22]. The main steps are briefly described here for reference. For given particle positions $\mathbf{x}_p^n$ at a time step n, for their velocities $\mathbf{v}_p^{n+1/2}$ at the half time step $n + 1/2$, and for the magnetic field $\mathbf{B}^n$ at time step n, one computing cycle proceeds as follows:

1. Calculate the charge densities ρ_c^n at each gridpoint from $\mathbf{x}_p^n$ and currents $\mathbf{J}_{\text{ion}}^{n+1/2}$ from $\mathbf{v}_p^{n+1/2}$.
2. Update positions $\mathbf{x}_p^n$ to $\mathbf{x}_p^{n+1}$ according to Eq. (3) via a leapfrog scheme using $\mathbf{v}_p^{n+1/2}$.
3. Calculate ρ_c^{n+1} from $\mathbf{x}_p^{n+1}$ and $\rho_c^{n+1/2} = 1/2(\rho_c^n + \rho_c^{n+1})$.
4. Update the magnetic field from $\mathbf{B}^n$ to $\mathbf{B}^{n+1}$ using $\rho_c^{n+1/2}$ and $\mathbf{J}_{\text{ion}}^{n+1/2}$ according to Eq. (2). This is done with a smaller time step and a cyclic leapfrog method, as described in [22].

5. Extrapolate $\mathbf{J}_{\text{ion}}^{n+1/2}$ to $\mathbf{J}_{\text{ion}}^{n+1}$ using the "current advancement method" (CAM) described in [22].

6. Calculate the electric field $\mathbf{E}^{n+1}$ from Eq. (1) and update velocities from $\mathbf{v}_p^{n+1/2}$ to $\mathbf{v}_p^{n+3/2}$ according to Eq. (3). To do so, one must of course interpolate the electromagnetic field quantities from the simulation grid onto the individual particle positions.

Steps 1, 2, 3, and 6 are described in Section 3.4 and step 4 is described in 3.2. See [22] for a discussion of step 5. There it is also shown that this whole scheme is of second-order accuracy.

3.2. Finite Differencing in Curvilinear Coordinates

The simulation code needs as input the location of each gridpoint in physical space. The grid has to be hexahedral (each cell has six faces) and ordered, so that each gridpoint can be described by the indices q, r, s. Let the grid point positions be $\mathbf{r}_{qrs}$.

For the formulation of the scheme described above in this arbitrary grid, a method similar to that used by [7] is adopted. In principal, all equations are rewritten coordinate independent using common tensor calculus.

First of all, internal coordinates $\tilde{x}^i$ are defined, where $i = 1, 2, 3$. The integer part $[\tilde{x}^i]$ corresponds to the cell indices q, r, s, in which this point is located. $\xi^i = \tilde{x}^i - [\tilde{x}^i]$ gives the relative position in this cell.

Any function defined on the grid points f_{qrs} (which may also be a vector function) can be linearly interpolated to any point inside the cell by

$$f(\tilde{x}^i) = \sum_{a=0}^{1}\sum_{b=0}^{1}\sum_{c=0}^{1} f_{q+a,r+b,s+c} w_a(\xi^1) w_b(\xi^2) w_c(\xi^3), \tag{5}$$

with

$$w_0(z) = 1 - z \quad \text{and} \quad w_1(z) = z. \tag{6}$$

The coordinate transformation $\mathbf{x}(\tilde{x}^i)$ is then given by Eq. (5) using $\mathbf{r}_{qrs}$ for f_{qrs}. Three covariant basis vectors are defined for each grid point either by

$$\mathbf{b}_{1,qrs} = \mathbf{r}_{q+1,r,s} - \mathbf{r}_{qrs}, \quad \mathbf{b}_{2,qrs} = \mathbf{r}_{q,r+1,s} - \mathbf{r}_{qrs}, \quad \mathbf{b}_{3,qrs} = \mathbf{r}_{q,r,s+1} - \mathbf{r}_{qrs} \tag{7}$$

or, if the $\mathbf{r}_{qrs}$ are given by some analytical formula $\mathbf{r}_{qrs} = \mathbf{f}(\tilde{x}^1, \tilde{x}^2, \tilde{x}^3)$, by

$$\mathbf{b}_i = \frac{\partial \mathbf{f}}{\partial \tilde{x}^i}. \tag{8}$$

From these $\mathbf{b}_i$ the contravariant basis vectors and the metric tensor elements

$$\mathbf{b}^i = \frac{\mathbf{b}_j \times \mathbf{b}_k}{\sqrt{g}}, \quad g_{ij} = \mathbf{b}_i \cdot \mathbf{b}_j, \quad g^{ij} = \mathbf{b}^i \cdot \mathbf{b}^j, \quad \sqrt{g} = \det g_{ij} \tag{9}$$

can be obtained for each grid point (the index qrs was suppressed). Whether all these quantities are precomputed before a simulation run or each time they are needed is a question of calculation cost and memory usage. It turned out to be a good compromise

to calculate at least the g_{ij} and $\mathbf{b}_i$ in advance, because the $\mathbf{b}^i$ can be computed relatively quickly.

Using these local basis vectors each physical vector can be expressed by means of its co- and contravariant components via

$$\mathbf{A} = A^i \mathbf{b}_i, \quad A^i = \mathbf{b}^i \cdot \mathbf{A} \quad \text{or } \mathbf{A} = A_i \mathbf{b}^i, \quad A_i = \mathbf{b}_i \cdot \mathbf{A}. \tag{10}$$

The common vector operations are then given by

$$(\mathbf{A} \times \mathbf{B})^i = \frac{\epsilon^{ijk}}{\sqrt{g}} A_j B_k, \quad (\mathbf{A} \times \mathbf{B})_i = \sqrt{g}\, \epsilon_{ijk} A^j B^k, \tag{11}$$

$$(\text{curl } \mathbf{A})^i = \frac{\epsilon^{ijk}}{\sqrt{g}} \frac{\partial}{\partial \tilde{x}^j} A_k, \tag{12}$$

with $\epsilon^{ijk} = \epsilon_{ijk}$ being the Levi–Civita symbol.

Applying this to Eq. (2) yields an expression for the time development of the contravariant components of $\mathbf{B}$:

$$\frac{\partial B^i}{\partial t} = \frac{\epsilon^{ijk}}{\sqrt{g}} \frac{\partial}{\partial \tilde{x}^j} \left(\frac{\sqrt{g}}{\rho_c} \epsilon_{klm} J^l_{\text{ion}} B^m - \frac{\epsilon_{klm} \epsilon^{lnp} (\partial B_p / \partial \tilde{x}^n) \cdot B^m}{\mu_0 \rho_c} - \eta \frac{g_{kl} (\epsilon^{lmn}/\sqrt{g}) (\partial B_m / \partial \tilde{x}^n)}{\rho_c} \right). \tag{13}$$

The r.h.s. can now be easily computed by common finite difference schemes. Note that both the co- and contravariant components of $\mathbf{B}$ have to be known at each integration step. In the current version, therefore, at the beginning of each calculation cycle, the physical components of $\mathbf{B}$ are transformed to the co- and contravariant components, Eq. (13) is integrated (where the covariant B_i is calculated from the updated B^i after each step), and, after this, the result is transformed back into physical components, which is advantageous for the calculation of the Lorentz force in Eq. (3). For the calculation of the electric field the same transformation in the local coordinate system applies.

3.3. Finite Differencing Scheme

For the discretization of Eq. (13) two methods may be applied. Usually one computes the bracket (it is mainly $\mathbf{E}$) on an interlaced grid. This applies particularly to full particle codes when the whole set of Maxwell's equations has to be solved. Then the interlaced gridpoints guarantee correct expressions for the second-order derivatives of $\mathbf{B}$.

In the approximation underlying Eq. (13), however, the equation for $\mathbf{B}$ can be solved directly without any interlaced grid points and with the same accuracy up to second order. To illustrate this, a brief discussion of the most complicated term in Eq. (13), the Hall term, will be given.

Neglecting $\sqrt{g}$ and ρ_c (which may be assumed constant for the following analysis), one component of the Hall term has the form

$$\frac{\partial}{\partial \tilde{x}^1} \left(\frac{\partial B_2}{\partial \tilde{x}^1} B^1 \right). \tag{14}$$

An accuracy analysis using $B^1 = \exp(ik_1\tilde{x}^1)$ and $B_2 = \exp(ik_2\tilde{x}^1)$ yields

$$\frac{\partial}{\partial\tilde{x}^1}\left(\frac{\partial B_2}{\partial\tilde{x}^1}B^1\right) = -k_2(k_1+k_2)\exp(ik_1\tilde{x}^1 + ik_2\tilde{x}^1) \tag{15}$$

as the analytical result.

Taking (14) as it is, calculating the inner derivative on interlaced grid points, multiplying with an interpolated B^1 on these points, and then calculating the outer derivative back on the integer grid points yields

$$[-k_2(k_1+k_2) + \mathcal{O}_4(k_1\Delta) + \mathcal{O}_4(k_2\Delta) + \mathcal{O}_4(k_1\Delta + k_2\Delta)]\exp(ik_1\tilde{x}^1 + ik_2\tilde{x}^1). \tag{16}$$

When not using interlaced grid points one has to manipulate (14) analytically and calculate

$$B^1\frac{\partial^2}{(\partial\tilde{x}^1)^2}B_2 + \frac{\partial}{\partial\tilde{x}^1}B^1 \cdot \frac{\partial}{\partial\tilde{x}^1}B_2. \tag{17}$$

Taking (17) and calculating all derivatives with the common centered differencing scheme yields

$$\left[-k_2(k_1+k_2) + \mathcal{O}_4(k_2\Delta) + k_1\Delta\frac{(k_2\Delta)^3}{6} + k_2\Delta\frac{(k_2\Delta)^3}{6} + \cdots\right]\exp(ik_1\tilde{x}^1 + ik_2\tilde{x}^1). \tag{18}$$

Comparison of (16) and (18) shows that the two methods have the same accuracy up to the fourth order in $k\Delta$, where Δ is the grid point distance. A similar statement holds for all components in Eq. (13). Therefore it is possible to solve Eq. (13) using no interlaced grid points without losing accuracy. This is advantageous because an interlaced grid would need much more storage capacity (e.g., for storing the g_{ij} components). Moreover it is not quite clear how to do the interpolation needed in obtaining (16) in curvilinear coordinates.

However, because using interlaced grid points has become such a standard technique, we tested both methods, which yielded no visible differences in the results, what confirms the above arguments.

3.4. Treatment of Particles

3.4.1. *Particle Movement*

The solution of the equations of motion (3) for each particle is based on the leapfrog scheme and consists of two steps, namely the position update and the velocity update. Both steps have to be done in the transformed coordinate system.

In the following the particle index p at $\tilde{x}^i$ and $\mathbf{v}$ is suppressed.

The position update is straightforward. For each particle the local coordinates $\tilde{x}^i$ are stored, as is the vector of its velocity $\mathbf{v}$ in physical components. From $\mathbf{v}$ the contravariant components can be computed with the contravariant local basis vectors at the position $\tilde{x}^i$. This local basis is obtained by Eq. (5) using the basis vectors from Eq. (7) at each grid point for f. With these contravariant components v^i the position update takes the form

$$\tilde{x}^{i,n+1} = \tilde{x}^{i,n} + v^{i,n+1/2}\cdot\Delta t. \tag{19}$$

The problem is to obtain the $v^{i,n+1/2} = \mathbf{b}^i(\tilde{x}^{i,n+1/2}) \cdot \mathbf{v}^{n+1/2}$. To do this one has to interpolate the basis vectors to the position $\tilde{x}^{i,n+1/2}$ at the half time step, which is not known. Three solutions can be applied. The most simple one is to take $\tilde{x}^{i,n}$ as an approximation for $\tilde{x}^{i,n+1/2}$. This of course introduces an error which becomes larger the more the grid differs from the Cartesian form. The second possibility is to solve Eq. (19) iteratively, which has a huge cost in the form of computing time. The third possibility is to calculate the physical coordinates of the particle position, update these coordinates using the physical components of the velocity, and find the new $\tilde{x}^{i,n+1}$ by inverting Eq. (5). This inversion can also be done only iteratively, but with less computing time than the second method.

The first and last method have been tested. Almost no difference for typical cases was found. Of course, one has to keep in mind the fact that the first (and fastest) method can cause trouble when either the curvature of the grid or the time step becomes large. For the results presented here, the last method was applied.

For the velocity update the electromagnetic field vectors have to be interpolated from the grid points to the actual particle position using the same method as for the basis vectors. To achieve second-order accuracy Eq. (3) is discretized in time by

$$\mathbf{v}^{n+1/2} = \mathbf{v}^{n-1/2} + \frac{q_p \Delta t}{m_p}(\mathbf{E}'^n + \mathbf{v}^n \times \mathbf{B}^n) \quad \text{with } \mathbf{v}^n = \frac{1}{2}\left(\mathbf{v}^{n-1/2} + \mathbf{v}^{n+1/2}\right). \tag{20}$$

This semi-implicit formula can be solved analytically for $\mathbf{v}^{n+1/2}$ (cf. [9]). Since the result is rather complicated, it is convenient to do the necessary calculations with the physical components of the corresponding vectors. This is the reason for transforming **B** back into physical components after each time step.

3.4.2. *Adaptive PIC Method*

The collection of the moments ϱ_{qrs} and $\mathbf{J}_{qrs}$ from the particle positions and velocities proceeds in the local coordinate system similarly to the common PIC method used in a Cartesian grid. For collecting the charge density one has to add

$$\Delta\rho_{q+a,r+b,s+c} = \frac{q_p}{\sqrt{g}_{qrs}} w_a(\xi^1) w_b(\xi^2) w_c(\xi^3), \quad (a, b, c = 0, 1), \tag{21}$$

to the eight corner grid points of the cell in which the particle is located. Note that the only difference to the Cartesian case is the factor $1/\sqrt{g}_{qrs}$. Of course, if the number density of the particles is constant in physical space, there are more particles in a larger cell than in a smaller one. Therefore, one has to divide by the cell volume to get constant values for ρ_{qrs}.

However, the main purpose of a curvilinear grid is to have a higher resolution in one part of the simulation area. Assume a grid like that shown in Fig. 1. One has to put a certain number of particles in the leftmost cells. The number density in physical space at the start of the simulation has to be constant. Therefore, in the larger cells a huge number of particles are located. This is even worse in three dimensions. Of course, this leads to a huge loss of computing efficiency. To overcome this, one can use particle coalescence as described in [15], or use particles with variable size in phase space (the "blob method") [6].

In [15] a detailed proof is given for the possibility of splitting one particle into two while preserving the first three moments of the distribution function when linear weighting is

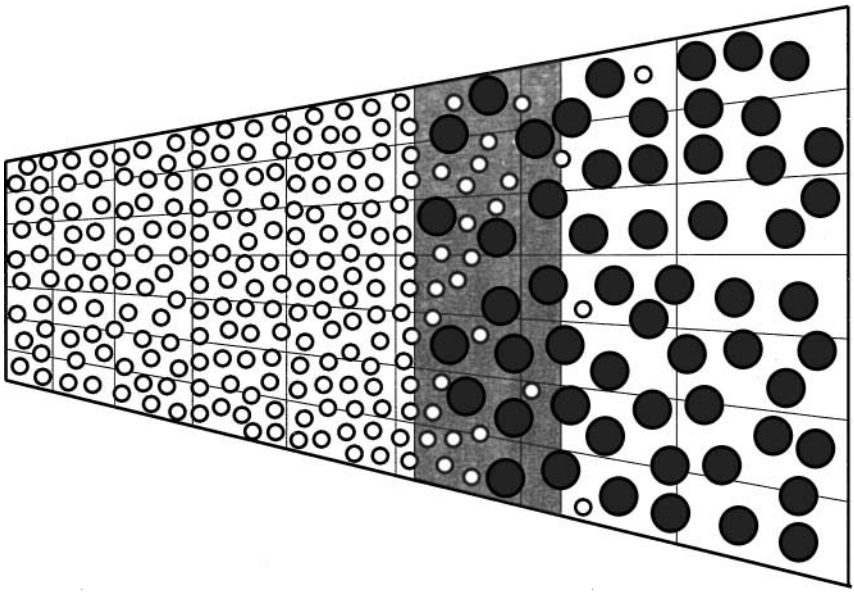

FIG. 1. Schematic sketch of the adaptive PIC method. In the grey "averaging zone" two incoming particles are replaced by one with twice the charge and mass. In an actual simulation more than one of these zones can be applied.

used, as was done here. Furthermore, it is mentioned there that this splitting can be exactly inverted (i.e., two particles coalesce into one, preserving energy and momentum) if they have equal velocity. For particles with similar velocity this can be still achieved approximately. This is demonstrated in [15].

As the purpose of our code is mainly to use a grid like that shown in Fig. 1, only the particle coalescence is needed to control the number of particles in a cell. We therefore use a method which is based upon the more general method mentioned above and optimized for the special geometry of Fig. 1.

Several "averaging areas" along the flow direction in the simulation box are chosen, as sketched in Fig. 1. If a particle enters such an area, a "partner" is searched for, which is located nearby in phase space (a more detailed investigation is given in the next subsection). If such a particle is found, these two are deleted, averaged in phase space, and replaced by a particle with twice the charge and mass, i.e., twice the "numerical weight." To keep this transition smooth, a probability for this process is introduced, which rises linearly from 0 to 1 along the averaging zone.

It would be a N^2 process to find a suitable partner for the averaging process if one searched over the whole particle list each time. Because this is too expensive, another search strategy is applied. If a particle enters the averaging zone it is put on a "search list." The next particle entering the same zone looks only on this search list for a partner. If no partner is found, it puts itself at the end of this search list. This is done during the particle position update cycle.

Generally this kind of particle coalescence introduces noise to the moments of the distribution function, i.e., for the densitites and currents collected on the grid. For linear PIC weighting this noise is proportional to $1/\sqrt{N}$, when N is the number of particles per cell. With no particle coalescence N is proportional to the cell volume $V = \sqrt{g}_{ijk}$. To keep a

noise level, which is at least smaller than that at the left boundary the "averaging zones" have to be placed at intervals where the cell volume rises by a factor of $\sqrt{2}$, because N is approximately halved in each "averaging zone."

It is clear that this whole method is only effective for a grid like the one in Fig. 1 if the particles move generally from left to right, which applies to a wide range of plasma flow problems and their interaction with obstacles.

3.4.3. *Energy Conservation*

The averaging process described above will be investigated in more detail here. Let $\mathbf{v}_1$ and $\mathbf{v}_2$ be the velocities of two particles with mass m. The resulting "combined" particle then has a mass of $2m$ and a velocity of $1/2\ (\mathbf{v}_1 + \mathbf{v}_2)$. This conserves momentum, but the overall energy decreases, since

$$2m\left[\frac{1}{2}(\mathbf{v}_1+\mathbf{v}_2)\right]^2 \le m\left(v_1^2+v_2^2\right) \Leftrightarrow \mathbf{v}_1\cdot\mathbf{v}_2 \le \frac{1}{2}v_1^2+\frac{1}{2}v_2^2,$$

which is always fulfilled. It is clear that this energy loss becomes smaller if the two velocity vectors differ less. Therefore, only particles are involved in the average process which fulfill the condition

$$|\mathbf{v}_1-\mathbf{v}_2|^2 < \alpha\cdot v_{\text{th}}^2 \quad \text{and} \quad |\mathbf{x}_1-\mathbf{x}_2| < \beta, \tag{22}$$

where v_{th}^2 is the initial mean average square of the velocity distribution and α, β are two parameters which have to be chosen properly.

β is normally taken to be one or two cell sizes. For a reasonable choice of α refer to Fig. 2. This figure shows (a) the energy loss and (b) the efficiency of the averaging process for two extreme values of α. The simulation was done in a 2D Cartesian grid with 200 particles per cell. For $\alpha = 0.1$, almost no energy loss occurs, but only about 60% of the particles are actually averaged. For $\alpha = 0.5$, almost all particles are averaged but there occurs a huge loss of energy, about 8%. Typically a value of $\alpha = 0.2$ was used in the simulations.

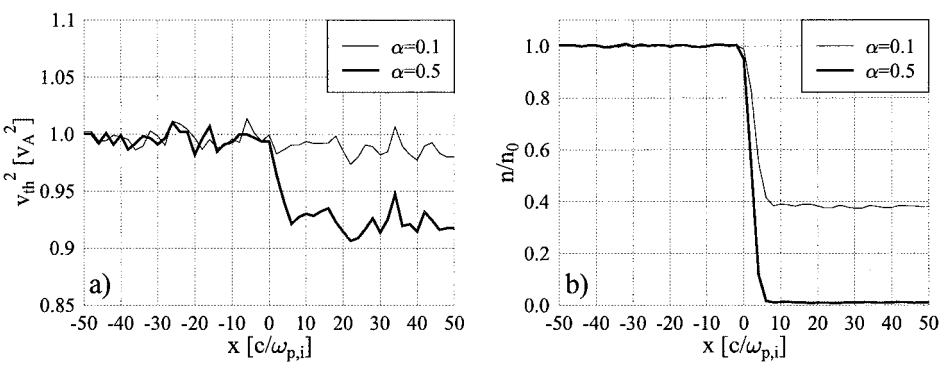

FIG. 2. Energy loss due to phase space averaging of particles. (a) Mean square of thermal velocity for two different values of α (see text). (b) Density of particles which are not involved in the averaging process.

3.4.4. *Particle Initialization*

To achieve a uniform particle density with the least possible noise, at the beginning each cell is filled with particles rather than the particles being distributed globally.

First, the code searches for the smallest cell with volume V_0. In this cell P_0 particles should be placed. P_0 is mostly chosen to be about 20 for 3D and 50 in 2D. From this the charge and mass of these superparticles can be calculated to yield the desired charge density. In all other cells $P_{qrs} = P \cdot (\sqrt{g}_{qrs}/V_0)$ particles should be placed. In general P_{qrs} is a real number, which has to be rounded up or down to yield just P_{qrs} in the average.

The particles in a particular cell are equally distributed with respect to the internal coordinates $\tilde{x}^i$. This, of course, yields a nonuniform distribution in physical space if the cell is nonorthogonal. However, this error is completely negligible compared to the noise introduced by the particle method itself.

For the particles' velocities a three-dimensional Maxwellian distribution is applied. This defines an ion temperature by the mean thermal velocity square v_{th}^2, which is related to the usual plasma ion beta via $\beta_{\text{ion}} = v_{\text{th}}^2/v_A^2$. The electron temperature is defined by the parameter p_{e0} in Eq. (4), which is just $\beta_e/2$ in normalized units.

3.5. Boundary Conditions

For each boundary three different types of behavior can be chosen: periodic, inflow, and outflow conditions.

A periodic boundary is only possible for two parallel boundaries of the same size. It is applied in the same way as for a Cartesian grid.

At an outflow boundary the crossing particles are simply deleted and the field values are extrapolated. In the present version no absorbing boundaries are included, because the investigated problems always dealt with high Mach number flows, where no waves can travel upstream.

At an inflow boundary the field values are kept constant. New particles have to be inserted at each time step. For an orthogonal cell this is simple, since the volume which has to be filled is a slice of this cell, which is again simply an orthogonal volume. For a nonorthogonal cell this volume is more complicated and it is difficult to place the particles inside this volume with the correct distribution function. Therefore, after each time step all particles inside an inflow cell are deleted and the whole cell is filled with new particles.

4. RESULTS

4.1. Test Run—Wave Propagation

The purpose of this section is to demonstrate that the new code presented here reproduces well-known results from linear MHD wave theory. Therefore a number of 1D simulations with periodic boundary conditions have been performed, each with a different angle θ between the initial magnetic field and the x-axis.

The simulations used 200 cells of length $\Delta x = 0.2c/\omega_{p,i}$ and ran for 1500 time steps with $\Delta t = 0.2\Omega_i^{-1}$. The initial magnetic field was chosen to be

$$\mathbf{B} = (B_0 \cos(\theta),\, B_0 \sin(\theta),\, 0), \tag{23}$$

with $B_0 = 1$ (in normalized units). A harmonic perturbance with an amplitude of 10% of the

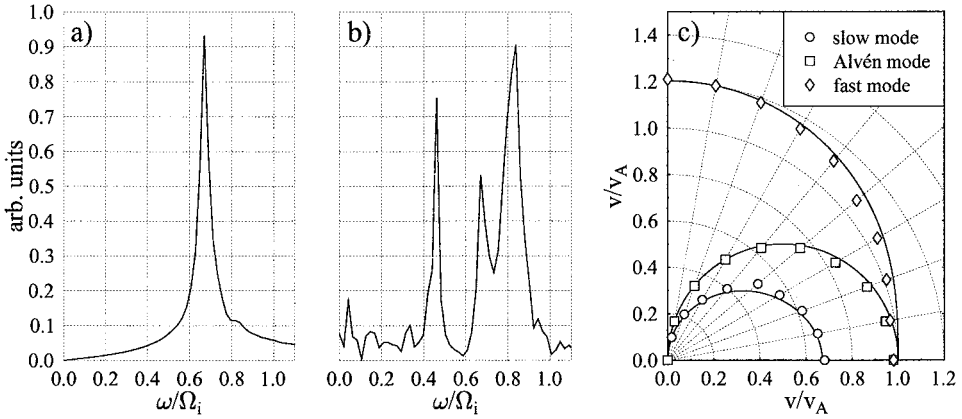

FIG. 3. Test of linear wave propagation. (a) Fourier power spectrum of transverse magnetic field component B_z for $\theta = 30°$. (b) Same as (a) for ion density ρ_c. (c) Friedrichs diagram for the resulting propagation velocities calculated from spectra like that in (a) and (b) for different angles θ (markers) and the results of linear wave theory (solid lines). See text for more details and parameters.

background field was superimposed in the z-component of **B** with a wavelength according to $k = \pi/4$. For the plasma betas, $\beta_i = 0.1$ and $\beta_e = 0.8$ were chosen, yielding a sound velocity of

$$c_s = \sqrt{\frac{\beta_i + \beta_e}{2}} v_A \approx 0.67 v_A. \tag{24}$$

The z-component of **B** and the ion density ρ_c were recorded for all cells at each time step. After the run finished a Fourier analysis was applied to both data sets, yielding a power spectrum for the chosen k-value, as shown in Figs. 3a and 3b. Three different peaks can be clearly distinguished. The two magnetosonic wave modes are visible mainly in the density fluctuations, whereas the Alfvén mode is very pronounced in the transverse magnetic field fluctuations, as one would expect. From these spectra the frequencies of each mode were measured and the linear propagation velocity was calculated, and plotted as a Friedrichs diagram in Fig. 3c. The solid lines in this plot represent the prediction of linear wave theory (cf. [5]). The agreement is reasonable with some small statistical error, probably due to the rather low resolution in ω-space.

4.2. Comet Wirtanen

As an example of the results obtained with the new code we present a 2D simulation run for comet Wirtanen at 3AU. The purpose is to show the new possibilities which can be achieved by using the described numerical techniques. Detailed physical interpretation and more results can be found in [1].

4.2.1. *Model*

The plasma surrounding of a comet is formed by the interaction of the cometary ions (produced by UV ionization of the neutral gas coma) with the solar wind.

The solar wind is determined by its proton number density n_0, the background magnetic field strength B_0, the ratio of thermal pressure to magnetic pressure of the ions and electrons

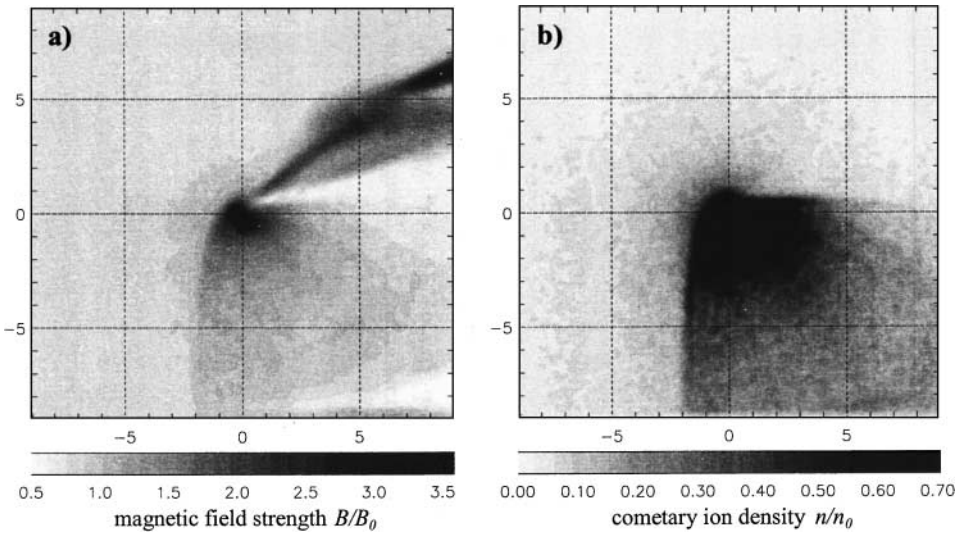

FIG. 4. Results from a 2D simulation of the plasma environment near the nucleus of comet Wirtanen at 3AU. The solar wind flows from left to right, with a background magnetic field perpendicular to the shown plane (pointing inward). (a) Magnetic field strength B/B_0, $B_0 = 1.2$ nT. (b) Cometary ion density n/n_0, $n_0 = 0.8$ cm^{-3}. (Parameters) Alfvénic Mach number $M_A = 10$, $\beta_i = \beta_e = 0.3$, time step $\Delta t = 0.01\Omega_i^{-1}$. Simulation box size is $L_x = L_y = 18x_0$ ($x_0 \approx 250$ km) with 100×100 cells. Each cell is initialized with $P_0 = 40$ superparticles. The picture was taken after 2000 simulation steps ($t = 20\Omega_i^{-1}$, $\Omega_i^{-1} \approx 9$ s). See text for parameters of the comet.

β_i, β_e, and the Alfvénic Mach number M_A. The parameters at 3AU were obtained by fitting the measurements of Voyager 2 and IMP-8 [26, 27] using the Parker model [25]. The parameters are given in the caption of Fig. 4.

In our simulations we choose a frame where the comet is at rest. The solar wind flows along the x-axis from left to right. The background magnetic field is directed perpendicular to the simulation plane (pointing inward), which is a good assumption for 3AU. The solar wind protons are described by a number of superparticles having an initial velocity M_A in x-direction and a superimposed thermal velocity distribution according to β_i. At the start the simulation box is filled with these particles. During the run new particles are inserted at the inflow boundary to the left, whereas particles crossing the outflow boundary at the right are deleted. The upper and lower boundary are also treated as inflow boundaries, which turned out to be the most simple and stable boundary condition in this case.

The comet is modeled as a source of heavy ions. The largest contribution arises from the UV ionization of sublimated water molecules, yielding mainly O^+ ions. Assuming a spherical symmetric neutral gas coma which extends with a velocity v_0, one can derive the spatial charge density distribution $\varrho_h(r)$ of newly produced ions per time unit (cf. [1, 18]); i.e.,

$$\frac{d\varrho_h(r)}{dt} = \frac{e\nu G}{4\pi r^2 v_0} \exp\left(-\frac{\nu}{v_0}(r-a)\right), \tag{25}$$

where r, G, ν, and a are the distance from the nucleus, the neutral gas production rate, the ionization rate, and the radius of the cometary nucleus, respectively. The parameter a is chosen larger than in reality ($a = 0.1x_0 \approx 25$ km), because otherwise the $1/r^2$ dependence in two dimensions yields gradients too steep in the center of the coma, leading to unphysical results.

During the simulation run each time step $N_h = 600$ superparticles are placed inside the simulation box to yield the charge distribution gain given above. For the O^+ ions under consideration, the charge to mass ratio is 16 in normalized simulation units (where $m_p = e = 1$). The physical parameters for comet Wirtanen used in the simulations are $G \approx 3 \times 10^{25}$ s^{-1} [28], $\nu = 10^{-6}$ s^{-1} [10, 23], and $v_0 = 1$ km/s (thermal velocity at sublimation temperature of ice).

4.2.2. *Grids*

Two simulation runs with the same physical parameters but different scales and grids were performed.

The first run (Fig. 4) uses a Cartesian grid with 100×100 cells and a box size of $L_x = L_y = 18x_0 \approx 4500$ km.

For the second run (Fig. 5) all physical parameters are the same, but the simulation box was chosen as sketched in Fig. 1. The whole simulation box consists also of 100×100 cells. The outer, rectangular box has the dimensions $L_x = L_y = 60x_0 \approx 15{,}000$ km. However, the resolution near the nuclues is about the same as in the first run.

The position of the grid points are given by

$$\mathbf{r}_{ijk} = \left\{ X + (L_x - X) \cdot s, \left(\frac{j \cdot L_y}{g_y} - \frac{L_y}{2} \right) \cdot s, \left(\frac{k \cdot L_z}{g_z} - \frac{L_z}{2} \right) \cdot s \right\}, \tag{26}$$

with

$$s = s_0 \cdot \alpha^i, \quad \alpha = s_0^{-1/g_x}, \quad s_0 = \frac{X}{X - L_x}, \tag{27}$$

where g_x, g_y, g_z are the number of cells in each direction and X is the x-coordinate, where all grid lines would intersect.

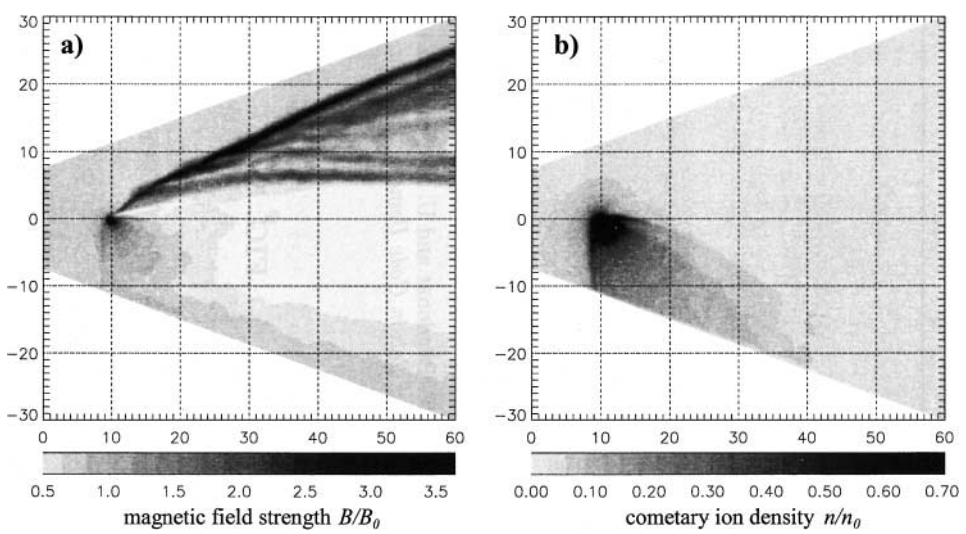

FIG. 5. Same as Fig. 4 but for a nonorthogonal grid and a much larger scale. The particle coalescence method was used with three equidistant averaging zones along the x-axis. Note that all details from Fig. 4 are well resolved in the vicinity of the nucleus, but also the large-scale structure of the nonlinear, split Mach cones is obtained.

Along the x-axis three equidistant "averaging zones" (each three cells in width) are placed. Both runs use $P_0 = 40$ particles per cell, which yields about 4 million particles for the first run and about 5 million particles for the second run. The computational effort for both runs is therefore almost the same. However, the numerical overhead for the particle coalescence process slightly increases the computation time for the second run.

4.2.3. *Results*

As already stated, only a short physical description of the simulation results is given here. More details can be found in [1].

First of all, the cometary ions are picked up by the solar wind due to the Lorentz force. This force acts perpendicularly to the magnetic field and the solar wind flow direction, forcing the cometary ions on a cycloidal trajectory moving "downward" as can be seen in Figs. 4b and 5b. Although this type of plasma tail, typical for weak comets, has not yet been observed, it is a well-known fact from bi-ion fluid simulations [2, 18, 19].

The magnetic field lines drape around the obstacle, which leads to an increase in the magnetic flux density directly in front of and in the coma. For an even weaker comet a classical Mach cone would be formed behind the obstacle, with a depression of the magnetic field below and a compression above the obstacle. In the case shown in Figs. 4a and 5a, nonlinear phenomena give rise to a formation of multiple compression cones. The physical details of this process are also discussed in [1]. Such a Mach cone splitting has also been obtained from other hybrid simulations [18] but not in MHD simulations, implying that this is a kinetic effect.

This phenomena can only be resolved well if the resolution near the cometary nucleus is high enough. However, then only the onset of this splitting can be obtained, as in Fig. 4a; the large scale structure is not accessible, because the box has to be kept small. With the new code presented here we can resolve the processes near the nucleus well enough and also get a picture of the large-scale structure (Fig. 5a). To get the same picture with a Cartesian grid one would need approximately $(60/18)^2 \approx 10$ times more cells and particles and, therefore, storage capacity.

5. SUMMARY

A new 3D hybrid simulation code was described, which is capable of using arbitrary, curvilinear grids. The code uses the approximations of massless electrons and quasineutrality and neglects displacement currents. The time integration scheme for Maxwell's equations and the particle movement is based on Matthews' scheme described in [22].

The solution of Maxwell's equations is done by a transformation using common tensor analysis and a finite differencing scheme.

Because in an arbitrary grid the cell sizes may differ, the known PIC method has the disadvantage of large differences in the total number of particles in each cell. For a special type of grid this problem has been overcome by introducing a so-called averaging zone, where particles are collected in pairs to yield a new particle with twice the numerical weight to reduce the overall number of particles in areas with a larger cell size. This method has the disadvantage of not conserving energy exactly, but it was shown that the numerical error can be kept small.

The advantage of using a non-Cartesian grid is to enhance the resolution where necessary and leave out regions where the background plasma is undisturbed, which saves a lot in computing costs. This has been demonstrated for the example of comet Wirtanen at a distance of 3AU from the sun. However, the numerical overhead produced by the more complicated equations and methods reduces the profit. Therefore, the overall computing time is not reduced significantly, whereas the overall storage capacity needed decreases drastically, which is an important factor in 3D simulations.

The code was tested in various ways for correct numerical and physical behavior. The comparison between a small-scale Cartesian and a large-scale curvilinear grid using the full capability of the code, as well as some lD results for linear wave propagation, have been presented. Meanwhile the code has been applied to various problems in the field of solar wind interaction with small ionospheric obstacles, especially comets. From these runs we experienced the code to be stable and accurate. The results will be presented in further publications.

ACKNOWLEDGMENT

This work is supported by the Deutsche Forschungsgemeinschaft through Grant MO 539/10-1.

REFERENCES

1. T. Bagdonat and U. Motschmann, From a weak to a strong comet—3D global hybrid simulation studies, *Earth, Moon, Planets*, in press.
2. A. Bogdanov, K. Sauer, K. Baumgärtel, and K. Srivastava, Plasma structures at weakly outgasing comets—results from bi-ion fluid analysis, *Planet. Space Sci.* **44**(6), 519 (1996).
3. S. H. Brecht and V. A. Thomas, Multidimensional simulations using hybrid particle codes, *Comput. Phys. Commun.* **48**, 135 (1988).
4. J. A. Byers, B. I. Cohen, W. C. Condit, and J. D. Hanson, Hybrid simulations of quasineutral phenomena in magnetized plasma, *J. Comput. Phys.* **27**, 363 (1978).
5. P. Colella, Multidimensional upwind methods for hyperbolic conservation laws, *J. Comput. Phys.* **87**, 171 (1990).
6. G. G. M. Coppa, G. Lapenta, G. Dellapiana, F. Donato, and V. Riccardo, Blob method for kinetic plasma simulation with variable-size particles, *J. Comput. Phys.* **127**, 268 (1996).
7. J. W. Eastwood, W. Arter, N. J. Brealey, and R. W. Hockney, Body-fitted electromagnetic pic software for use on parallel computers, *Comput. Phys. Commun.* **87**, 155 (1995).
8. C. T. Fischer, *Beobachtung und Simulation von Strukturen im Plasmaschweif eines Kometen*, Ph.D. thesis (Techn. Univ. of Braunschweig, 1999).
9. D. W. Forslund, Fundamentals of plasma simulation, *Space Sci. Rev.* **42**, 3 (1985).
10. T. I. Gombosi, D. L. DeZeeuw, and R. M. Häberli, Three-dimensional multiscale MHD model of cometary plasma environments, *J. Geophys. Res.* **101**(A7), 15233 (1996).
11. D. S. Harned, Quasineutral hybrid simulation of macroscopic plasma phenomena, *J. Comput. Phys.* **47**, 452 (1982).
12. D. W. Hewett, A global method of solving the electron-field equations in a zero-inertia-electron-hybrid plasma simulation code, *J. Comput. Phys.* **38**, 378 (1980).
13. D. W. Hewett and C. W. Nielson, A multidimensional quasineutral plasma simulation model, *J. Comput. Phys.* **29**, 219 (1978).
14. E. J. Horowitz, D. E. Shumaker, and V. Anderson, QN3D: A three-dimensional quasi-neutral hybrid particle-in cell code with applications to the tilt mode instability in field reversed configurations, *J. Comput. Phys.* **84**, 279 (1989).

15. G. Lapenta and J. U. Brackbill, Dynamic and selective control of the number of particles in kinetic plasma simulations, *J. Comput. Phys.* **115**, 213 (1994).
16. P. C. Liewer, Numerical studies of ion reflection in collisionless theta-pinch implosions using a hybrid Vlasov-fluid model, *Nucl. Fusion* **16**, 817 (1976).
17. A. S. Lipatov and J. Buechner, On the influence of nonstationary electron inertia on collisionless reconnection, submitted for publication.
18. A. S. Lipatov, U. Motschmann, and T. Bagdonat, 3D hybrid simulation of the solar wind-weak comet interaction, *Planet. Space Sci.* **50**, 403 (2002).
19. A. S. Lipatov, K. Sauer, and K. Baumgärtel, 2.5D hybrid code simulation of the solar wind interaction with weak comets and related objects, *Adv. Space Res.* **20**(2), 279 (1997).
20. N. K. Madsen, Divergence preserving discrete surface integral methods for maxwell's curl equations using non-orthogonal unstructured grids, *J. Comput. Phys.* **119**, 34 (1995).
21. R. J. Mason, Computer simulation of ion-acoustic shocks: The diaphragm problem, *Phys. Fluids* **14**, 1943 (1971).
22. A. P. Matthews, Current advance method and cyclic leapfrog for 2D multispecies hybrid plasma simulations, *J. Comput. Phys.* **112**, 102 (1994).
23. D. A. Mendis, H. L. F. Houpis, and M. L. Marconi, The physics of comets, *Fundam. Cosmic Phys.* **10**, 1 (1985).
24. U. Motschmann, K. Sauer, and T. Roatsch, Subcritical multiple-ion shocks, *J. Geophys. Res.* **96**(A8), 13841 (1991).
25. E. N. Parker, Dynamics of the interplanetary gas and magnetic fields, *Astrophys. J.* **128**, 664 (1958).
26. J. D. Richardson, J. W. Belcherand, A. J. Lazarus, K. I. Paularena, and P. R. Gazis, Statistical properties of the solar wind, *AIP Conf. Proc.* **382**, 483 (1996).
27. J. D. Richardson, K. I. Paularena, A. J. Lazarus, and J. W. Belcher, Radial evoultion of the solar wind from imp 8 to voyager 2, *Geophys. Res. Lett.* **22**, 1469 (1995).
28. G. Schwehm and R. Schulz, Coma composition and evolution of Rosetta target comet Wirtanen, *Space Sci. Rev.* **90**, 321 (1999).
29. M. A. Shay, J. F. Drake, R. E. Denton, and D. Biskamp, Structure of the dissipation region during collisionless magnetic reconnection, *J. Geophys. Res.* **103**(A5), 9165 (1998).
30. D. W. Swift, Use of a hybrid code for global-scale plasma simulation, preprint (1996).
31. T. Terasawa, M. Hoshino, J.-I. Sakai, and T. Hada, Decay instability of finite-amplitude circularly polarized Alvén waves: A numerical simulation of stimulated Brillouin scattering, *J. Geophys. Res.* **91**(A4), 4171 (1986).
32. D. Winske and M. M. Leroy, *Hybrid Simulation Techniques Applied to the Earths Bow Shock* (Terra, Tokyo, 1985).
33. D. Winske and N. Omidi, *Hybrid Codes: Methods and Applications* (Terra, Tokyo, 1993).
34. D. Winske and N. Omidi, A nonspecialist's guide to kinetic simulations of space plasmas, *J. Geophys. Res.* **101**(A8), 17287 (1996).
35. D. Winske and K. B. Quest, Magnetic field and density fluctuations at perpendicular supercritical collisionless shocks, *J. Geophys. Res.* **93**(A9), 9681 (1988).