JOURNAL OF COMPUTATIONAL PHYSICS **126,** 109–121 (1996)
ARTICLE NO. 0124

Use of a Hybrid Code for Global-Scale Plasma Simulation

DANIEL W. SWIFT

Geophysical Institute, University of Alaska, Fairbanks, Alaska 88775-7320

This paper presents a demonstration of the use of a hybrid code to model the Earth's magnetosphere on a global scale. The typical hybrid code calculates the interaction of fully kinetic ions and a massless electron fluid with the magnetic field. This code also includes a fluid ion component to approximate the cold ionospheric plasma that spatially overlaps with the discrete particle component. Other innovative features of the code include a numerically generated curvilinear coordinate system and subcycling of the magnetic field update to the particle push. These innovations allow the code to accommodate disparate time and distance scales. The demonstration is a simulation of the noon meridian plane of the magnetosphere. The code exhibits the formation of fast and slow-mode shocks and tearing reconnection at the magnetopause. New results include particle acceleration in the cusp and nearly field aligned currents linking the cusp and polar ionosphere. The paper also describes a density depletion instability and measures to avoid it.

1. INTRODUCTION

In the 35 years of space exploration we have learned much about the Earth's magnetosphere. Yet, we do not understand it, even though the fundamental physical laws governing plasma behavior in the neighborhood of the Earth are well known. A major difficulty is that the dimensions of the magnetosphere are vast, measuring many tens of Earth radii, yet anybody who has had an opportunity to observe the aurora will see size scales the order of a tenth of a kilometer. Ion and electron streams are important and transition regions the order of an ion gyroradius are observed, so kinetic processes are important. Global-scale MHD codes to simulate the magnetosphere have been in existence for about 10 years [1, 2]. They have proven successful in reproducing the gross shape and dynamics of the magnetosphere, but still an understanding of the aurora and the substorm [3] eludes us. There is a basic need for a code that will accommodate disparate size scales and include kinetic processes.

The most complete description of a plasma is contained in the full electromagnetic code [4]. One difficulty with this approach is that the electromagnetic code contains the electrostatic interaction. For explicit codes this requires resolution of Debye lengths and temporal resolutions of the electron plasma period. Another serious time step constraint is due to the Courant condition with respect to the propagation of light, which in concert with the Debye length spatial resolution would require time steps of a small fraction of a microsecond. The electromagnetic code would require too many grid points and too many time steps to be feasible for large-scale simulations.

The hybrid code is offered as a practical compromise. The hybrid code assumes quasi-neutrality, which eliminates the need to solve the Poisson equation and the attendant constraints on the time step and grid size. The code is nonradiative, so there is no Courant condition with respect to the propagation of light.

There have been a variety of one-dimensional hybrid codes proposed. For a review see Winske [5]. There are, however, three types of multidimensional hybrid codes. In one type, proposed by Hewett and Nielson [6], the electrons must have finite inertia, and the longitudinal electric field is computed from the quasi-neutrality condition. The magnetic field is computed from Ampere's law. This code requires the solution of elliptic equations for the magnetic field, and transverse and longitudinal electric fields. Solution of elliptic equations can be difficult in a curvilinear coordinate system. Moreover, a momentum transport equation must be solved for the electron fluid, which imposes a Courant condition with respect to the electron flow speed. This effectively requires an unrealistically large electron-to-ion mass ratio lest the electron flow speed become very large in some regions.

Another type of hybrid code was proposed by Byers *et al.* [7]. In this code, the electrons are assumed massless, and the electric field is calculated from the electron momentum equation. The magnetic field is calculated from the magnetic vector potential, which is calculated by integrating in time the expression for the transverse electric field. The transverse electric field is calculated by subtracting the longitudinal electric field from the electric field derived from the electron momentum equation. The longitudinal field is derived from the quasi-neutrality condition. This also requires the solution of an elliptic equation for the electrostatic potential.

The hybrid code, which is the subject of this paper, was originally demonstrated by Harned [8] and subsequently used by others [9–11]. Brechet and Thomas [10] have used hybrid code applications customarily reserved to MHD and Brecht and Ferrante [12] have used in a global scale

0021-9991/96 $18.00

simulation of a planetary magnetosphere, as have Thomas and Winske [13]. In this version, the magnetic field is advanced in time from Faraday's law, and the electric field is also calculated algebraically from the electron momentum balance equation. The entire code is explicit, so there is no requirement for solving an elliptic equation. This makes the implementation of the code in a curvilinear coordinate system straightforward.

The purpose of this paper is to describe a number of innovations that make the basic hybrid code suitable for simulation of the Earth's magnetosphere and surrounding magnetosheath on a global scale. One is the use of generalized curvilinear coordinates. This makes it possible to achieve high spatial resolution in regions near the Earth and near the magnetopause, while using a much lower density of grid points in the magnetotail lobes where the resolution is not needed. Another innovation is the use of a fluid approximation to the ions, as well as the fully kinetic implementation. The fluid approximation is used to simulate the much denser ionospheric component near the earth, where kinetic effects are unimportant. This is an enormous saving in computational effort. The final innovation is the subcycling of the magnetic field and fluid update to the particle push. The hybrid code has a Courant condition with respect to the whistler mode if the time step is shorter than the inverse ion gyrofrequency and with respect to the Alfven modes if the time step is longer. The subcycling makes it possible to update the fluid and magnetic fields with a small enough time step to avoid numerical instability, while taking fewer of the more expensive particle push time steps.

The application described by Harned [8] contained vacuum regions. He circumvented the stability problem by matching the magnetic field within the plasma to a vacuum field through an iterative process. The addition of an MHD fluid component represents another way of avoiding the requirement of having to fill the entire simulation domain with an adequate density of particles. Because the particle and fluid populations are allowed to co-mingle, there is no necessity for setting up any matching conditions.

The next section, 2, will outline the method for implementing the hybrid code for a particle and fluid ion mixture in the presence of frictional dissipation terms. This will be followed by a description of the time stepping scheme in Section 3, including the subcycling of the magnetic field and fluid update. Section 4 will describe the implementation of the curvilinear coordinates. The following section, 5, will describe the boundary conditions, and Section 6 will present results from a simulation of the interaction of the dayside magnetosphere and the solar wind advecting a southward interplanetary magnetic field. Section 6 will also describe a density depletion instability encountered during code development. The final section, 7, will discuss future extensions of the code.

2. THE ALGORITHMS

This section presents the equations that will be used as a basis of the simulation. The equations are then reduced to a form convenient for introduction of the time stepping and differencing algorithms to be described in later sections. The code involves the solutions of four basic equations: One is for the advance of discrete particles by Newton's law and the Lorentz force. Another is the electron momentum balance equation, which is used to determine the electric field. The electron velocity is calculated from Ampere's law. The third equation is used to advance the fluid approximation to the ions. Finally, the magnetic field is advanced through Faraday's law. The equations for the particle and fluid ion populations are formulated in a way that both components intermingle in any proportion. This eliminates the need for formulation of special boundary or jump conditions. These equations contain ion–electron frictional dissipation terms. Such terms are found to be necessary to stabilize the code against a nonlinear density depletion instability to be described later. At the end of this section, modifications used to include electron inertia will be described.

The equation for ion particle motion is given by

$$\frac{d\mathbf{v}}{dt} = \mathbf{E} + \mathbf{v} \times \mathbf{B} - \nu(\mathbf{u}_\mathrm{p} - \mathbf{u}_\mathrm{e}), \tag{1}$$

where $\mathbf{E}$ is the electric field in units of ion acceleration, $\mathbf{B}$ is the magnetic field in units of the ion gyrofrequency, ν is the collision frequency, $\mathbf{u}_\mathrm{p}$ is the particle ion bulk flow velocity, and $\mathbf{u}_\mathrm{e}$ is the electron flow velocity. The collision frequency, as can be seen, applies to the bulk flow with respect to the electron fluid, rather than to the individual particles. Application of friction to individual particles would lead to rapid cooling of the ions. Equation (1) must be solved for each ion present. Throughout this paper, the symbol $\mathbf{v}$ will be reserved for the individual particle velocity, while the symbol $\mathbf{u}$ will be used for the bulk flow velocity. The symbol $\mathbf{v}$ will often be subscripted by an index numbering the individual particles.

The electron momentum equation is written in the form

$$\mathbf{E} = -\mathbf{u}_\mathrm{e} \times \mathbf{B} - \nu(\mathbf{u}_\mathrm{e} - \mathbf{u}_\mathrm{i}) - r\frac{d\mathbf{u}_\mathrm{e}}{dt}, \tag{2}$$

where $\mathbf{u}_\mathrm{i}$ is the total ion bulk flow speed, given by

$$\mathbf{u}_\mathrm{i} = \frac{n_\mathrm{p}}{n}\mathbf{u}_\mathrm{p} + \frac{n_\mathrm{f}}{n}\mathbf{u}_\mathrm{f}, \tag{3}$$

where the subscripts p and f stand for discrete particle and fluid, respectively, and the n's are the densities. Last

term on the r.h.s. of (2) is the electron inertial term [14] and r is the electron-to-ion mass ratio. The electron inertial term is evaluated by taking the time derivative of the expression for the electron flow velocity. For the time being, the mass ratio will be set to zero, to simplify the exposition. Other authors [8–10] have included effects of finite electron pressure, which also requires inclusion of electron energy transport in the code. The pressure term will be neglected here.

The electron flow speed is evaluated from Ampere's law

$$\mathbf{u}_e = \mathbf{u}_i - \frac{\nabla \times \mathbf{B}}{\alpha n}, \tag{4}$$

where in the units used in the simulation, $\alpha = (4\pi e^2/m_i c^2)$. Note that $(\alpha n)^{-1/2}$ is the ion inertial length. It also turns out that α, the charge coupling constant, always multiplies the density, and the density always multiplies α. The value of α is used to scale the density to the physical situation under consideration.

In this code, we consider the ion fluid density as being given, so there is no ion density to transport. This is a reasonable approximation for ionospheric plasma as part of a magnetospheric system. The important role played by ionospheric plasma is conduction of field-aligned currents and transmission of Alfven waves between the outer magnetosphere and the conducting ionospheric shell at low altitudes. This requires solution of a transport equation for the ion fluid velocity. The equation for the fluid ion velocity is given by

$$\frac{d\mathbf{u}_f}{dt} = \mathbf{E} + \mathbf{u}_f \times \mathbf{B} - \nu(\mathbf{u}_f - \mathbf{u}_e). \tag{5}$$

Again, the pressure term has been neglected. This approximation is particularly valid in the magnetospheric context, where the fluid approximation is used for the cold and comparatively dense ionospheric plasma. Actually, the time derivative on the left-hand side could be replaced by partial time derivative, because for flow velocities small in comparison to relevant wave speeds, the $\mathbf{u} \cdot \nabla\mathbf{u}$ term is small. However, as will be discussed later, the code indicated noise buildup in the equatorial ionosphere at low altitudes which was effectively suppressed by the numerical dissipation [15] provided by a first-order transport scheme. The set of Eqs. (1), (2), and (5) is momentum conserving.

Assuming for the moment that $\mathbf{B}$ is known, we rewrite the (1) and (5), making use of the electron momentum equation and Ampere's law, in a form useful for later computation. With the use of (2), (3), and (4), Eq. (1) may be written in a form more convenient for coding the time-stepping algorithm

$$\frac{d\mathbf{v}}{dt} = \mathbf{E}_p + \mathbf{v} \times \mathbf{B}, \tag{6}$$

where

$$\mathbf{E}_p = \left(\frac{\nabla \times \mathbf{B}}{\alpha n} - \mathbf{u}_i\right) \times \mathbf{B} + \nu \frac{n_f}{n}(\mathbf{u}_f - \mathbf{u}_p). \tag{7}$$

Here, $\mathbf{E}_p$ is simply a convenient grouping of terms and is not exactly the same as the electric field that appears in (1). Similarly, the ion fluid momentum equation (5) can be rewritten in the form

$$\frac{\partial \mathbf{u}_f}{\partial t} = -\mathbf{u}_f \cdot \nabla\mathbf{u}_f + \mathbf{E}_f + \frac{n_p}{n}\mathbf{u}_f \times \mathbf{B} - \nu\frac{n_p}{n}\mathbf{u}_f, \tag{8}$$

where

$$\mathbf{E}_f = \left(\frac{\nabla \times \mathbf{B}}{\alpha n} - \frac{n_p}{n}\mathbf{u}_p\right) \times \mathbf{B} + \nu\frac{n_p}{n}\mathbf{u}_p \tag{9}$$

is another convenient grouping of terms.

Finally, we use Faraday's law

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E} \tag{10}$$

to update the magnetic field, which, upon making use of (2), gives

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \left[\left(\frac{\nabla \times \mathbf{B}}{\alpha n} - \mathbf{u}_i\right) \times \mathbf{B}\right] - \nabla \times \left(\nu \frac{\nabla \times \mathbf{B}}{\alpha n}\right). \tag{11}$$

The last term on the r.h.s. is diffusive and is responsible for smoothing out fluctuations in the magnetic field. The $\nabla \times \mathbf{B}/\alpha n$ in the first term on the r.h.s. is responsible for propagation of the whistler mode in the limit that the propagation frequency is much less than the gyrofrequency. The term containing the $\mathbf{u}_i$, in concert with the particle/fluid equations, propagates the Alfven modes.

The main effect of electron inertia is to add electron polarization drift to modify (11) as

$$\frac{\partial}{\partial t}\left[\mathbf{B} + r\nabla\times\left(\frac{\nabla\times\mathbf{B}}{\alpha n} - \mathbf{u}_\mathrm{i}\right)\right] = r\nabla\times(\mathbf{u}_\mathrm{e}\cdot\nabla\mathbf{u}_\mathrm{e}) - \nabla\times\left[\left(\frac{\nabla\times\mathbf{B}}{\alpha n} - \mathbf{u}_\mathrm{i}\right)\times\mathbf{B}\right] - \nabla\times\left(\nu\frac{\nabla\times\mathbf{B}}{\alpha n}\right), \quad (12)$$

where the electron drift $\mathbf{u}_\mathrm{e}$ is obtained from Ampere's law (4). The effect of the electron polarization drift in the ion equations (7) and (9) is neglected, as the correction is only the order of the electron-to-ion mass ratio. Equation (12) can be written in the more computationally convenient form

$$\frac{\partial}{\partial t}\mathbf{B}' = -r\nabla\times[\mathbf{u}_\mathrm{e}\times(\nabla\times\mathbf{u}_\mathrm{e})] - \nabla\times\left[\left(\frac{\nabla\times\mathbf{B}}{\alpha n} - \mathbf{u}_\mathrm{i}\right)\times\mathbf{B}\times\nu\frac{\nabla\times\mathbf{B}}{\alpha n}\right], \quad (13)$$

where

$$\mathbf{B} = \mathbf{B}' - r\nabla\times\left(\frac{\nabla\times\mathbf{B}}{\alpha n} - \mathbf{u}_\mathrm{i}\right). \quad (14)$$

This is solved for $\mathbf{B}$ by iteration. For r small enough to maintain an electron inertial length much less than the distance between grid points only one iteration is needed. Mandt and Drake [14] solve for $\mathbf{B}$ by inversion of the differential operator. It can be seen that the electron inertia also leads to smoothing of the field.

In many problems there is a large curl-free field, such as the Earth's dipole field in the magnetospheric problem. In order to avoid inaccuracy in the computation of $\nabla\times\mathbf{B}$ due to small differencing errors on large curl-free fields, the magnetic field is split into a time independent, curl-free portion, $\mathbf{B}_0$ and variable field $\mathbf{B}_1$

$$\mathbf{B} = \mathbf{B}_0 + \mathbf{B}_1,$$

where only $\mathbf{B}_1$ is updated.

3. TIME STEPPING ALGORITHM

As mentioned above, the particle equations of motion are advanced with one time step, and the fluid and field equations are advanced using a time step that is an integral fraction of the particle time step. This section first describes the particle update, assuming the velocities are known at the half time step and the positions are at the even time step. The magnetic field, $\mathbf{B}$, and the pseudo electric field $\mathbf{E}_\mathrm{p}$ given in (7) are also assumed known at the even time step. Next, the subcycle update of the magnetic and fluid fields, using the leapfrog technique, are described. This section concludes with a description of how the two time-stepping schemes are articulated.

In the particle velocity update, we assume that $\mathbf{B}$ and $\mathbf{u}_\mathrm{f}$ are known at the whole time step. $\mathbf{u}_\mathrm{p}$ as it appears in $\mathbf{E}_\mathrm{p}$ of (7), is calculated from moments derived from $\mathbf{v}_k$, where k is the particle index. However, $\mathbf{u}_\mathrm{p}$ is known at the previous half time step. The procedure is to make a provisional extrapolation to the whole time step

$$\mathbf{u}_\mathrm{p}^n = 1.5\mathbf{u}_\mathrm{p}^{n-1/2} - 0.5\mathbf{u}_\mathrm{p}^{n-3/2},$$

where the superscript n stands for the time level. Now, $\mathbf{E}_\mathrm{p}$ can be taken as known at the whole time step.

By making the substitutions

$$\mathbf{v}^\pm = \mathbf{v}^{n\pm1/2} \mp \frac{\Delta t}{2}\mathbf{E}_\mathrm{p} \quad (15)$$

in (6), the equation for the second-order velocity advance is written

$$\mathbf{v}^+ = \mathbf{v}^- + \frac{\Delta t}{2}(\mathbf{v}^+ + \mathbf{v}^-)\times\mathbf{B}^n \quad (16)$$

which is solved for $\mathbf{v}^+$. However, before back solving for $\mathbf{v}^{n+1/2}$, the partial results for $\mathbf{v}^\pm$ are used to obtain

$$\mathbf{v}^n = \tfrac{1}{2}(\mathbf{v}^+ + \mathbf{v}^-) \quad (17)$$

from which an improved $\mathbf{u}_\mathrm{p}$ can be obtained at the even time step. Since $\mathbf{u}_\mathrm{f}$ was calculated from (8) and (9) based on an old value of $\mathbf{u}_\mathrm{p}$, $\mathbf{u}_\mathrm{f}$ is recalculated based on the improved value of $\mathbf{u}_\mathrm{p}$, advancing $\mathbf{u}_\mathrm{f}$ by $\Delta t/2$ using (8). These updated results are incorporated into $\mathbf{E}_\mathrm{p}$, and the procedure indicated in (15) and (16) is repeated to obtain a final value for $\mathbf{v}^{n+1/2}$.

The next step is the update of $\mathbf{u}_\mathrm{f}$ from (8) and (9) which is done on the subcycle time step, with $\mathbf{u}_\mathrm{f}$ taken to be at the subcycle half time step. The fluid velocity, $\mathbf{u}_\mathrm{f}$ is updated using the subcycle time step, because near the Earth, where $B\Delta t > 1$ the Courant condition is small with respect to the Whistler mode, the Alfven mode phase velocity is large. The update of $\mathbf{u}_\mathrm{f}$ uses the same procedure as is used in the advancement of the discrete particle velocity

$$\mathbf{u}_f^\pm = \mathbf{u}_f^{m\pm1/2} \mp \frac{\delta t}{2}\mathbf{E}_\mathrm{f},$$

where m is the subcycle time level index and δt is the subcycle time step. The update of $\mathbf{u}_\mathrm{f}$ is calculated from

$$\mathbf{u}_f^+ = \mathbf{u}_f^- + \frac{n_p \delta t}{2n}(\mathbf{u}_f^+ + \mathbf{u}_f^-) \times \mathbf{B} - \nu \frac{n_p \delta t}{2n}(\mathbf{u}_f^+ + \mathbf{u}_f^-). \quad (18)$$

Note that in the limit of vanishing discrete particle density, the complicating implicit terms disappear. For a mixed discrete and fluid species, considerations of stability require the term involving the magnetic field be handled implicitly.

The magnetic field update of $\mathbf{B}_1$ uses a predictor–corrector, or leapfrog trapezoidal, scheme [16–17]. The predictor step using Faraday's law is

$$\tilde{\mathbf{B}}^{m+1} = \mathbf{B}^{m-1} - 2\delta t(\nabla \times \mathbf{E})^m, \quad (19)$$

where $\mathbf{E}^m$ is evaluated using an average of $\mathbf{u}_f$ at the old and new half time steps. The corrector step is then

$$\mathbf{B}^{m+1} = \mathbf{B}^m - \delta t(\nabla \times \mathbf{E})^{m+1/2} \quad (20)$$

with $\frac{1}{2}(\tilde{\mathbf{B}}^{m+1} + \mathbf{B}^m)$ used in the computation of $\mathbf{E}$ at the half time step. The simple second-order Runge–Kutta method was found to be unstable.

A couple of different methods were used in the articulation of the main and subcycling time step. Say, for example, that 10 subcycling time steps were used for each main step. In one method, the subcycling time step was run for 15 time steps and the value of $\mathbf{B}_1$ averaged over the last 10 time steps was used to advance the particles. Also, interpolation and extrapolation using $\mathbf{u}_p^{n-1/2}$ and $\mathbf{u}_p^{n+1/2}$ were used to center the particle flow at the correct second-order subcycle time step. The other method used the last value of $\mathbf{B}_1$ calculated in the 10-step subcycle loop to advance the particles and assumed a constant $\mathbf{u}_p^{n+1/2}$ for the duration of the subcycle loop. There was negligible difference between the two methods.

4. CURVILINEAR COORDINATES

Most emphasis on use of curvilinear coordinates has been in application to electromagnetic codes. Holland [18] described methods for integration of the Maxwell equations in a generalized curvilinear coordinate system. More recently Eastwood *et al.* [19] have used a multiple patch particle code with curvilinear body-fitting coordinates to simulate microwave devices, while Madsen [20] has developed a source-free finite-element electromagnetic code. This section describes the adoption of the basic methods for the much more complex hybrid code. In addition, this section will describe how the necessary geometrical coefficients can be calculated from a table giving the Cartesian positions of the curvilinear grid points. This will make it possible to change a coordinate system by changing the table. This feature could later be exploited to implement a dynamically adaptive grid system. The mesh will be *structured* and have quadrilateral, for 2D, or hexahedral, for 3D, elements. This will still give great flexibility, but be considerably simpler than the *unstructured* mesh consisting of triangular or tetrahedral elements. The differencing algorithms will leave $\mathbf{B}_1$ exactly divergenceless. An innovative particle pusher is also described. For reasons to be described below, the particle velocities are kept in Cartesian components, but the particle position is given in curvilinear coordinates. The indices pointing to the grid points defining the cell in which the particle resides are determined simply by truncating the curvilinear position.

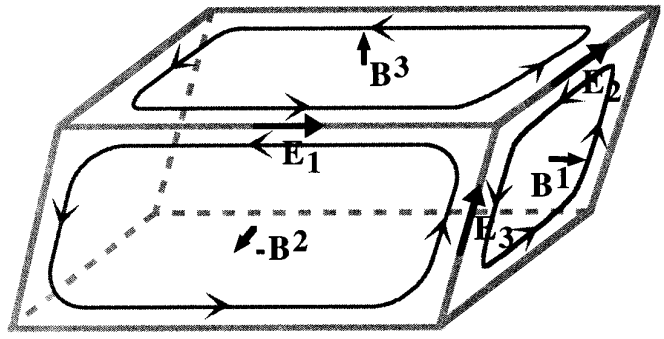

FIG. 1. A coordinate cell showing the position and orientation of curvilinear components of the magnetic and electric fields.

The basic method is most easily illustrated by reference to Fig. 1 which shows a parallelepiped cell in a nonorthogonal coordinate system. Let the center of the cell be at the grid point i, j, k, while the corner points are at $i + \frac{1}{2}, j + \frac{1}{2}, k + \frac{1}{2}$ and $i - \frac{1}{2}, j + \frac{1}{2}, k + \frac{1}{2}$, etc. The magnetic field components are shown as vectors on cell faces with the components pointing normal to the faces, while the electric field resides on the cell edges, with components parallel to the cell edges. In terms of classical tensor notation, the magnetic field is represented in *contravariant* components, while the electric field is given in *covariant* components. The magnetic field components are found by taking the scalar product of $\mathbf{B}$ with the unit vector normal to the cell surface, while the components of $\mathbf{E}$ are found by taking the scalar product with the unit vector parallel to the cell edge.

The curvilinear algorithm for the advancement of $\mathbf{B}$ is derived by applying Stokes' theorem to Faraday's law

$$\int \frac{\partial \mathbf{B}}{\partial t} \cdot d\mathbf{A} = \oint \mathbf{E} \cdot dl. \quad (21)$$

Now let us apply this to the right-facing face of the parallelepiped centered at $(i + \frac{1}{2}, j, k)$. The discretization of (21) gives

$$\left[\frac{(B^1)^{n+1}-(B^1)^n}{\Delta t}\frac{\mathbf{l}_1}{l_1}\cdot\mathbf{A}^1\right]_{i+1/2,j,k}$$
$$=(\mathbf{E}^{n+1/2}\cdot\mathbf{l}_3)_{i+1/2,j-1/2,k}-(\mathbf{E}^{n+1/2}\cdot\mathbf{l}_3)_{i+1/2,j+1/2,k}$$
$$+(\mathbf{E}^{n+1/2}\cdot\mathbf{l}_2)_{i+1/2,j,k+1/2}-(\mathbf{E}^{n+1/2}\cdot\mathbf{l}_2)_{i+1/2,j,k-1/2}, \tag{22}$$

where $l_{2,3}$ are lengths of the cell edges and reside at the center of the edges. The l's are lengths of tangent vectors which are found by taking the difference between coordinate points specified in the table, previously mentioned, i.e.,

$$(\mathbf{l}_2)_{i+1/2,j,k+1/2}=\mathbf{r}_{i+1/2,j+1/2,k+1/2}-\mathbf{r}_{i+1/2,j-1/2,k+1/2}, \tag{23}$$

where the $\mathbf{r}$'s are the positions of the coordinate points , which defines the coordinate system. $\mathbf{A}^1=\mathbf{l}_2\times\mathbf{l}_3$ is the area of the cell face, and this quantity resides at the center of the cell face. $\mathbf{l}_1$ is in this case a dual cell tangent vector, which will be defined below. This procedure is repeated for all the cell faces to advance all components of $\mathbf{B}$. The r.h.s. of (22) shows the curl operation computed without the need to explicitly differentiate any of the metric coefficients. The cell face areas used in the code is an average of taking the cross product between the tangent vectors at two opposite corners of the parallelogram.

The tangent vectors $\mathbf{l}_1,\mathbf{l}_2,\mathbf{l}_3$ in fact determine everything that is needed to do a simulation in a curvilinear coordinate system. We also observe that $g_{i,j}=\mathbf{l}_i\cdot\mathbf{l}_f$ is the metric tensor. The volume of the cell is

$$\sqrt{g}=\mathbf{l}_1\cdot(\mathbf{l}_2\times\mathbf{l}_3). \tag{24}$$

The divergence of $\mathbf{B}$ resides at the cell center (i, j, k), and it may be computed taking differences across the cell between normal field components multiplied by the respective cell face areas. It can be shown by simple geometric arguments that if $\mathbf{B}$ is initially divergenceless, it will remain *exactly* divergenceless. This is because, as can be seen from the circulation arrows in Fig. 1, the sum of the line integrals around the edges of the cell faces exactly cancel.

Ampere's law also requires that we take the curl of $\mathbf{B}$. However, the scheme outlined above for taking the curl only works if $\mathbf{B}$ is a tangent vector on cell edges. This also requires that we define a *dual* cell. The dual cells have centers at the half grid points, $i+\frac{1}{2}, j+\frac{1}{2}, k+\frac{1}{2}$ and corners at the whole grid points, i, j, k. This requires that the table specifying the coordinate system also specify the location of the dual/half-integer, as well as the main/whole-integer points. The components of $\mathbf{B}$ reside at the center of cell edges of the dual cell. If the coordinate system is orthogonal, the face normal components on the main cell are the tangent components on the dual cell. So, the curl operation for Ampere's law is done exactly as outlined above for Faraday's law, except that it is done on the dual cell with a shifting in the indices by $\frac{1}{2}$.

If a coordinate system must satisfy a number of requirements, it is not always possible to guarantee that it will be everywhere orthogonal. In this case it is necessary to have a subroutine that will convert between covariant and covariant components. I will briefly describe how this is done and how to convert between the true tensor components and the physical components used in the differencing algorithm described above. The tensor components will be denoted by a carat, i.e., $\hat{B}^1$ or $\hat{B}_1$. The superscript stands for contravariant and the subscript stands for covariant. A physical vector in these two representations is given by

$$\mathbf{B}=\mathbf{l}_1\hat{B}^1+\mathbf{l}_2\hat{B}^2+\mathbf{l}_3\hat{B}^3=\boldsymbol{\omega}^1\hat{B}_1+\boldsymbol{\omega}^2\hat{B}_2+\boldsymbol{\omega}^3\hat{B}_3, \tag{25}$$

where the $\boldsymbol{\omega}$'s are reciprocal basis vectors with the property that $\boldsymbol{\omega}^i\cdot\mathbf{l}_j=\delta^1_j$. These are readily computed from the tangent vectors, for example,

$$\mathbf{w}^1=\frac{\mathbf{l}_2\times\mathbf{l}_3}{\sqrt{g}} \tag{26}$$

with $\sqrt{g}$ defined in (24). The magnetic field can also be specified in terms of the "physical" components, i.e.,

$$\mathbf{B}=\mathbf{e}_1B^1+\mathbf{e}_2B^2+\mathbf{e}_3B^3=\mathbf{e}^1B_1+\mathbf{e}^2B_2+\mathbf{e}^3B_3, \tag{27}$$

where $\mathbf{e}^1=\mathbf{w}^1/|\mathbf{w}^1|$ and $\mathbf{e}_1=\mathbf{l}_1/|\mathbf{l}_1|$. The components $\hat{B}^i$ and $\hat{B}_i$ in terms of the covariant/contravariant components can be obtained by taking the scalar product of (25) with $\mathbf{w}^i$ or $\mathbf{l}_i$ and making use of the orthogonality relation between the basis tangent vectors or one-forms. Comparison between (25) and (27) indicates the scale factors to convert from the tensor to physical components. When it is all put together, the relation between the face normal and cell edge components is a linear relationship involving the angles between the tangent vectors. All of the required relationships are computed from the tangent vectors, which are computed from the displacement vectors between neighboring grid points. In a nonorthogonal grid, B_1 is, in general, computed from all three components of B^i, but only B^1 is in the correct location. The other components must be interpolated from other parts of the cell to the location of B^1. The interpolation introduces some inaccuracy, so errors are minimized by keeping the coordinate system as nearly orthogonal as possible.

The most troublesome operations are in the cross product in $(\nabla\times\mathbf{B})\times\mathbf{B}$ that appear in (7), (9), and (11). The reason is that $\nabla\times\mathbf{B}$ is dual cell contravariant, $\mathbf{B}$ is main cell contravariant, and the product must be main cell covariant. The cross product between the covariant basis vectors is

a one-form, which takes covariant vector components. The problem is that the location of the two input vectors do not match and the positions of neither input vector matches the locations of the components of the output vector. Interpolation from one point to another on the cell must be used. Two strategies have been used. One has been to interpolate the components of the input vector to the positions of the output vector. The other has been to interpolate all components to of the input vector to the dual cell center, form the cross product, then interpolate the result to the positions required for the output vector. The results in a running code are not significantly different. The second method is conceptually simpler, and the result is exactly perpendicular to **B**.

There is a subtle point in regard to the placement of the dual grid points. For a uniform grid, the obvious point is to locate the dual points in the geometrical center of the cell whose vertices are the main cell grid points. However, in a nonuniform grid the main cell grid points will no longer be geometrically centered with respect to the cell formed by the dual grid points. Monk and Süli [20] have shown that such a grid point layout is also second-order accurate with respect to simulations of the full Maxwell equations. The other choice for grid point layout is to topologically center the grid points. For example, let q be a curvilinear coordinate and let it be defined with respect to the uniform grid variable x by $x(q)$. Let the curvilinear grid points be at $x_i = x(q_i)$, $q_i = i\delta q$ for uniform spacing between the q's. Then the dual grid points are spaced at $x_{i+1/2} = x(q_{i+1/2})$, $q_{i+1/2} = (i + \frac{1}{2})\delta q$. Thus the grid points and their dual are mutually nested at half intervals from each other. Both methods of nesting were used in the example to be presented later. There was negligible difference in the results, except that the former method seemed to require somewhat less dissipation for stability.

The curvilinear coordinates are laid out such that the topological distance between grid points is always unity, i.e., the grid points are spaced at unit intervals of the curvilinear coordinates, q^i. This can be done without loss of generality, as the physical scale lengths are determined by the tangent vector lengths, l_i.

Methods for the advancement of the field equations in curvilinear coordinates have been described which do not involve explicit differentiation of the metric coefficients, i.e., without the need to introduce the Christoffel coefficients [22]. However, the expression for the acceleration in (6) in curvilinear coordinates requires explicit time differentiation of the tangent vectors, as $\mathbf{v} = \sum_{i=1}^{3} \mathbf{l}_i \dot{q}^i$. If the tangent vectors lengths and direction do not vary smoothly, the differencing may lead to large peaks in the particle acceleration. This is avoided by resolving the velocity in Cartesian components, i.e., $\mathbf{v} = \mathbf{1}_x v_x + \mathbf{1}_y v_y + \mathbf{1}_z v_z$. This requires the electric and magnetic fields as used in (6) and (7) be converted into Cartesian components. The conversion is readily derived from (23), since the grid point positions are specified on a Cartesian grid. Before the components are converted, the curvilinear components are interpolated to the dual center at $\mathbf{r}_{i+1/2,j+1/2,k+1/2}$. PIC weighting is used with weights derived from the curvilinear position, as if the particle resided in a unit cube. Conversion to a common point also guarantees that the cross product of two vectors will be orthogonal to the two vectors. For consistency, the particle density and flux moments are also calculated on the dual cell center. The flux will be resolved in Cartesian components. The flux is then divided by the density to obtain the ion flow speed $\mathbf{u}_p$, which is converted to contravariant components and interpolated to the proper dual cell contravariant positions.

The particle position must be specified in terms of the curvilinear components, q^1, q^2, q^3, as these coordinates serve as pointers to the grid points defining the cell in which the particle resides. The cell indices are calculated by truncation of the curvilinear coordinates. The particle position is updated from

$$(q^i)^{n+1} = (q^i)^n + \Delta t \mathbf{M}^i \cdot \mathbf{v}^{n+1/2},$$

where the vectors $\mathbf{M}^i$ convert $\mathbf{v}$ into tensor contravariant components $\dot{q}^i$. These vectors are themselves position dependent, so they must be interpolated from the grid to the particle position using PIC weighting. Second-order accuracy requires that $\mathbf{M}^i$ be evaluated at the position $(q^i)^{n+1/2}$. This is evaluated by taking a first-order half time step advance of q^i.

Finally, $\mathbf{u}_f$ is advanced from (8) and (9) in contravariant components.

5. BOUNDARY CONDITIONS

The exposition to this point has been largely independent of the specific application. The boundary conditions to a large measure drive the internal processes. Although some of the methods for imposing boundary conditions on the two-dimensional simulation to be described in the next section may not particularly innovative, they will be presented here for the sake of completeness.

Figure 2 shows the coordinate system to be used. The top of the page faces sunward. The solar wind convecting an interplanetary magnetic field flows downward across this boundary. The straight line boundary segments are the north and south polar axes and represent an outflow boundary. The inner semicircle represents the surface of the earth and the conducting ionospheric layer. All boundaries lie along dual cell grid surfaces, as defined in Fig. 1. On these boundaries, the tangential component of the electric field and normal component of the magnetic field are specified, as well as the normal particle and fluid flux.

The inner semi-circle is the simplest of the boundaries.

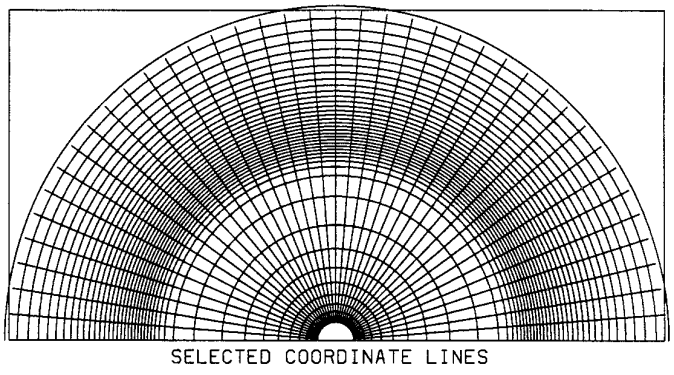

FIG. 2. The coordinate system used to simulate the noon meridian plane of the magnetosphere. The outer radius is 18 R_E while the inner circle is at 1 R_E. For the sake of clarity only a third of the coordinate lines are shown.

It is assumed to be rigid and perfectly conducting, so the tangential component of the electric field and the normal component of the fluid flow are assumed to vanish. The normal component of the magnetic field is held at the initial value. There are no discrete particles near the Earth's surface, so particle boundary conditions are unnecessary.

The value of the normal component of the magnetic field on the outflow boundary is determined by the condition that the divergence of **B** at the main cell grid point just inside the boundary remain zero. The tangential electric field on the outflow boundary is in effect determined by specification of the tangential components of the magnetic field on the main cell grid points a half step outside the boundary. There the radial component of the magnetic field is held to the initial value and the normal derivative of the ignorable component, i.e., the component into the page, set to zero. This says there are no currents in the radial direction. The normal component of the fluid flow is assumed to be continuous across the outflow boundaries. The particle flow is assumed to highly supersonic in the antisunward direction on the outflow boundary, so any particle which crosses this boundary is simply removed from the simulation.

The entire system is driven by flow across the upstream boundary. The normal component of the magnetic field is simply the normal component of the interplanetary magnetic field (IMF) at that particular point and that particular time. The IMF is allowed to vary in time. The tangential electric field is derived from the cross product of the IMF and the antisunward plasma velocity vector. The fluid density is assumed exponentially small at the upstream boundary, so the only boundary condition assumed is that the normal flow component be continuous.

The boundary conditions are imposed by a separate subroutine which is called after each update of the magnetic field or fluid flow velocity and prior to calculating the curl of the electric field. This modularity provides some flexibility in imposition of the boundary conditions and facilitates the imposition of time-dependent boundary conditions.

Particles continuously flow across the upstream boundary. The particle boundary condition is handled by placing a buffer region just outside the upstream boundary. At each time step particles are distributed in this buffer region and their phase space positions are initialized so that the buffer population has the prescribed density, velocity, and temperature. These particles are moved a time step. Those that cross the upstream boundary into the simulation domain become part of the simulation. Those that do not are returned to the inactive reservoir. The particle boundary conditions are imposed in the subroutine that advances q using (28).

The simple procedures described for the upstream and outflow boundaries work, so long as there are few particles that would want to orbit back and forth across the boundary. Situations in which particles cross the boundary in both directions require an additional buffer region to allow particles to gyrate [23] in order to avoid spurious boundary currents due to orbit interruption [24].

6. TEST RUNS

6.1. *Test Runs with a Rectangular Grid*

During the development and checkout stages, test runs were done on meshes with rectangular elements in which the tangent vector lengths varied in only one direction. The major anomaly we uncovered was a tendency for the code to go whistler mode unstable even though the simulation was initialized with particle densities more than adequate to assure stability. The location where the instability initiated was invariably a region where the density had fallen low enough that the whistler mode phase speed exceeded $\Delta x/\delta t$. Since Eq. (11) for **B** is nonlinear, numerical instability will grow explosively with little advance warning.

We traced the evolution of the density depletion and found that it began as a region of slightly enhanced noise and particle temperature.[1] This acted to enhance the depletion. In a quiescent, magnetized plasma these regions tended to remain stationary, so it was possible to observe them throughout the run. If the code was initialized with a higher density or run with a shorter time step, the onset of the instability was only delayed. Adding a cold fluid plasma also only served to delay the onset, unless the

[1] Other workers have reported through private communications similar problems with hybrid codes. Although the problem is not new, it deserves to be reported in the open literature in order to inform new users of hybrid codes of its existence.

density of the fluid plasma was itself sufficient to maintain stability. Electron inertia was added to the code with ion-to-electron mass ratios of 200, 500, and 1000. Mandt and Drake [14] maintain that electron inertia will stabilize the code. Our experience was that electron inertia alone would not stabilize the code, but it could make the difference between the code going unstable and not for some settings of the frictional coefficient, ν. However, the code was always operated in the regime where the grid spacing was much greater than the electron inertial length, whereas Mandt and Drake operated their code with a grid spacing comparable to the electron inertial length, where the stabilizing effects of the $\nabla \times [(\nabla \times \mathbf{B})/(\alpha n)]$ term in (14) would be much more significant.

The only thing that seemed capable of guaranteeing stability was the collision frequency term ν appearing in Eqs. (1) and (2). The ν appears as a coefficient of a diffusive term in (11), so the effect is to dampen short wavelength fluctuations in the magnetic field. The collision frequency was applied uniformly, i.e., ν was taken to be a constant in the preliminary runs with the rectangular grid. Increasing the value of ν would prolong the duration of the run before it "bombed." Once ν was increased beyond a certain threshold, the code would run indefinitely. Since the instability involves a nonlinear interaction between the particles and fields, we can offer no theoretical guidance on how large ν must be to guarantee stability. One effect of finite ν is to cause the plasma to slowly lose energy. We tracked the total energy, including taking account of all the energy fluxes crossing the domain boundaries, using the energy theorem outlined in Appendix A. If $\nu = 0$, there was a slow increase in total energy. With finite ν, the total energy would slowly decrease, losing about 5% in about 500 particle and 5000 field subcycle time steps.

Kazeminezhad *et al.* [25] have also overcome the instability and noise problem by strongly filtering the short wavelength modes. In order to overcome the lack of energy conservation introduced by the filtering, they used a two-step particle pusher. The method conserved energy very well, but at the expense of having to boost the particles twice; however, they were still left with strongly diffused fields. Plasma in the Earth's magnetosphere is to a large extent dissipationless, and we desire to obtain the maximum spatial resolution possible. We therefore strive to use the minimum dissipation that will maintain stability.

6.2. *Run Using a Curvilinear Grid*

In this section I present results of a 2D simulation in the noon meridian plane of the response to a southward turning of the IMF. Figure 2 shows, for clarity of presentation, a small fraction of the 121 × 123 grid used in the simulation. The inner circle represents the ionospheric boundary, which is assumed rigid and conducting. Note that the higher grid density at about 10 R_E. to capture the magnetopause and bow shock. The typical grid separation in the magnetopause/magnetosheath region is 600 km. The coordinate system was numerically specified, even though in this simple case the metric coefficients could have been analytically derived. Because of the very inhomogeneous nature of this run, the frictional collision coefficient ν was made to be proportional to the current density, or $\nabla \times \mathbf{B}$. Again, the chosen values were the minimum to ensure stability.

At $t = 0$ the solar wind is assumed uniform and there is no IMF inside 18 R_E. The geomagnetic field is compressed so there is an approximate balance between the solar wind ram pressure and the confined geomagnetic field. At the Earth's surface, the dipole field reaches a value of 500 s^{-1}, or 8×10^4 nT. The magnetic field is expressed in units of the O^+ gyrofrequency. Beginning at $t = 0$ a southward IMF is convected inward with the solar wind, ramping up to a constant value over a distance of 1 R_E. The solar wind speed is about 1 R_E/s, or 6000 km/s. This reason for this high value is that the two-dimensional dipole field falls off to $\frac{1}{100}$ of its near Earth value, instead of the $\frac{1}{000}$ for a three-dimensional dipole field. This choice of wind speed also has the advantage of being able to complete the simulation run in a shorter time. In the units used here, the southward IMF has a value of 3 s^{-1}, or 500 nT, which gives a nominal ion gyroradius of about 2000 km. The solar wind density is chosen such that the ion inertial length is 1000 km. This compares with the actual value of 100 km in a solar wind density of 5 H^+ ions/cm^3. The nominal Alfven velocity is 0.5 R_E/s, while the thermal velocity assumed is 0.1 R_E/s. The particles were updated with a 0.15 s time step, and the fields were updated with a 0.015 s time step. The Alfven Mach number is about 2.

Figure 3 displays results of the simulation shortly after the southward IMF encountered the subsolar magnetopause. The sun is toward the top of the figures. Panel (a) shows the magnetic field vectors, which indicates tearing going on at the magnetopause. The bow shock can also be seen in the figure as the sharp deflection of the field lines. The magnitude of the field on a logarithmic scale is shown in panel (b). Detailed plots of the magnetic field intensity were run along the radial line pointing toward the sun and 45° from that direction, which show the field in the strong shock front ramping up within two grid points. The profiles showed a somewhat thicker shock front at 45° from the Earth–Sun line, but the difference is not sufficient to be apparent on the figure. The field value is less than unity in the region of the dashed contours. As can be seen in succeeding panels, the solid crescent-shaped contour in the upstream region shows the locations of both fast and slow shock fronts. Slow mode transitions have been observed in the magnetosheath by Song *et al.* [26], and MHD simulations of Yan and Lee [27] also indicated the formation of

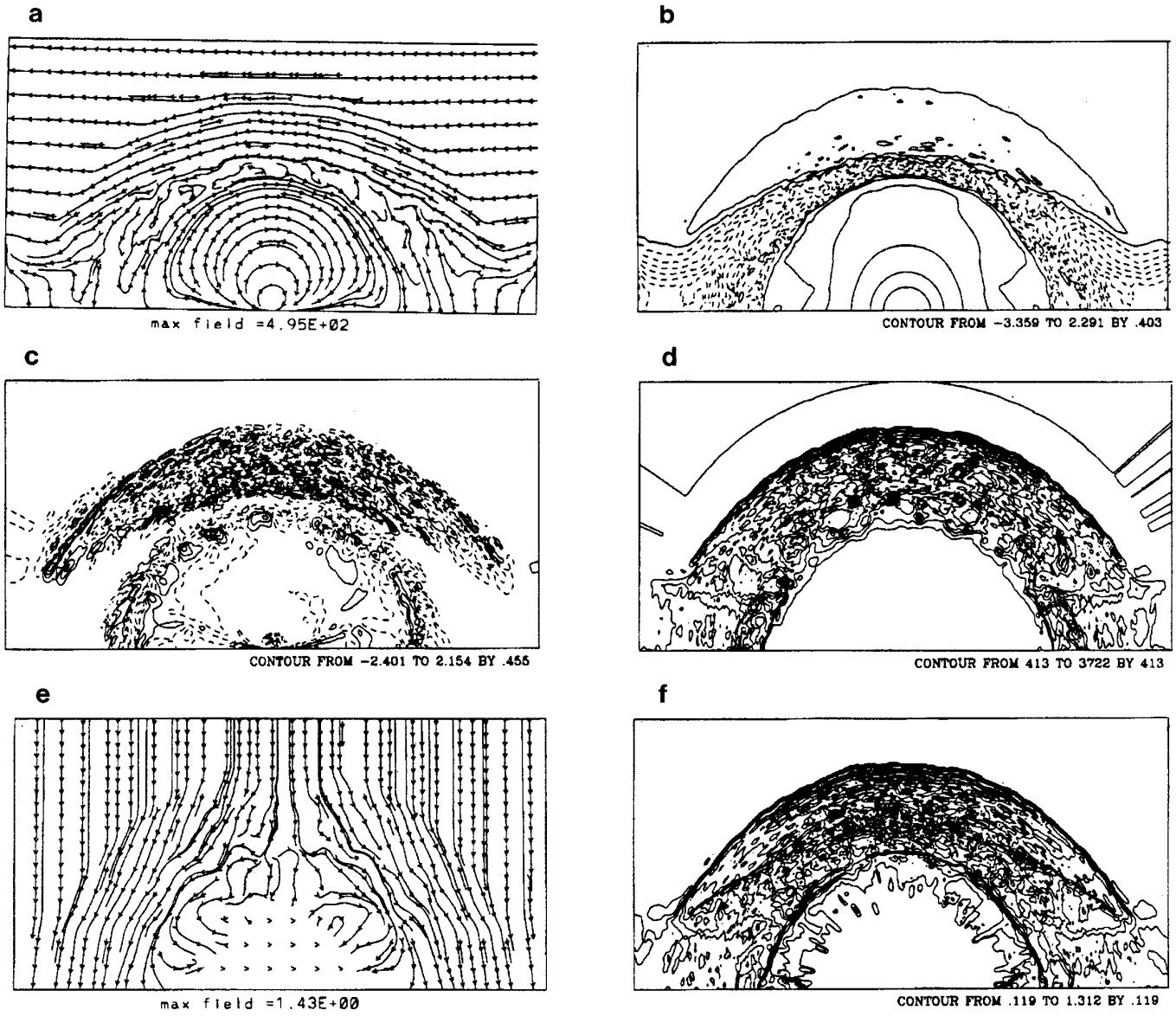

FIG. 3. Magnetic field vectors (a), logarithm of the magnetic field intensity (b), the *y*-component of the magnetic field (c), the plasma density (d), the plasma flow vectors (e), and the magnitude of the plasma flow speed (f). Each of the panels measures $18 \times 36\ R_E$.

a transient slow mode shock behind the bow shock. Panel (c) shows the *y*-component of the magnetic field. The little islands near the magnetopause provide further evidence of tearing at the magnetopause. Note the B_y contours extending from the cusp region to the high-latitude ionosphere. This is significant in that it indicates the presence of high-latitude field-aligned currents. These currents appear early in the simulation, so they do not depend on the presence of the IMF. They are likely due to differential motion between ions and electrons due to demagnetization of the ions in the cusp region. Panel (d) shows the density of the kinetic particles. Note the density pileup behind the fast shock, and the clumping and plasma blobs intruding into the magnetopause. Panel (e) shows the plasma flow vectors and the flow deflection at the location of the fast and slow shocks. Finally, panel (f) shows contours of the plasma flow magnitude. Again, note the sharp velocity decrease at the bow shock and the plasma flow jump at the slow shock.

Figure 4 shows the plasma and field configuration at the end of the run. The run was stopped because the bow shock had expanded to near the upstream boundary of the simulation domain. This continued outward expansion of the shock is due to the fact that in a two dimensions there

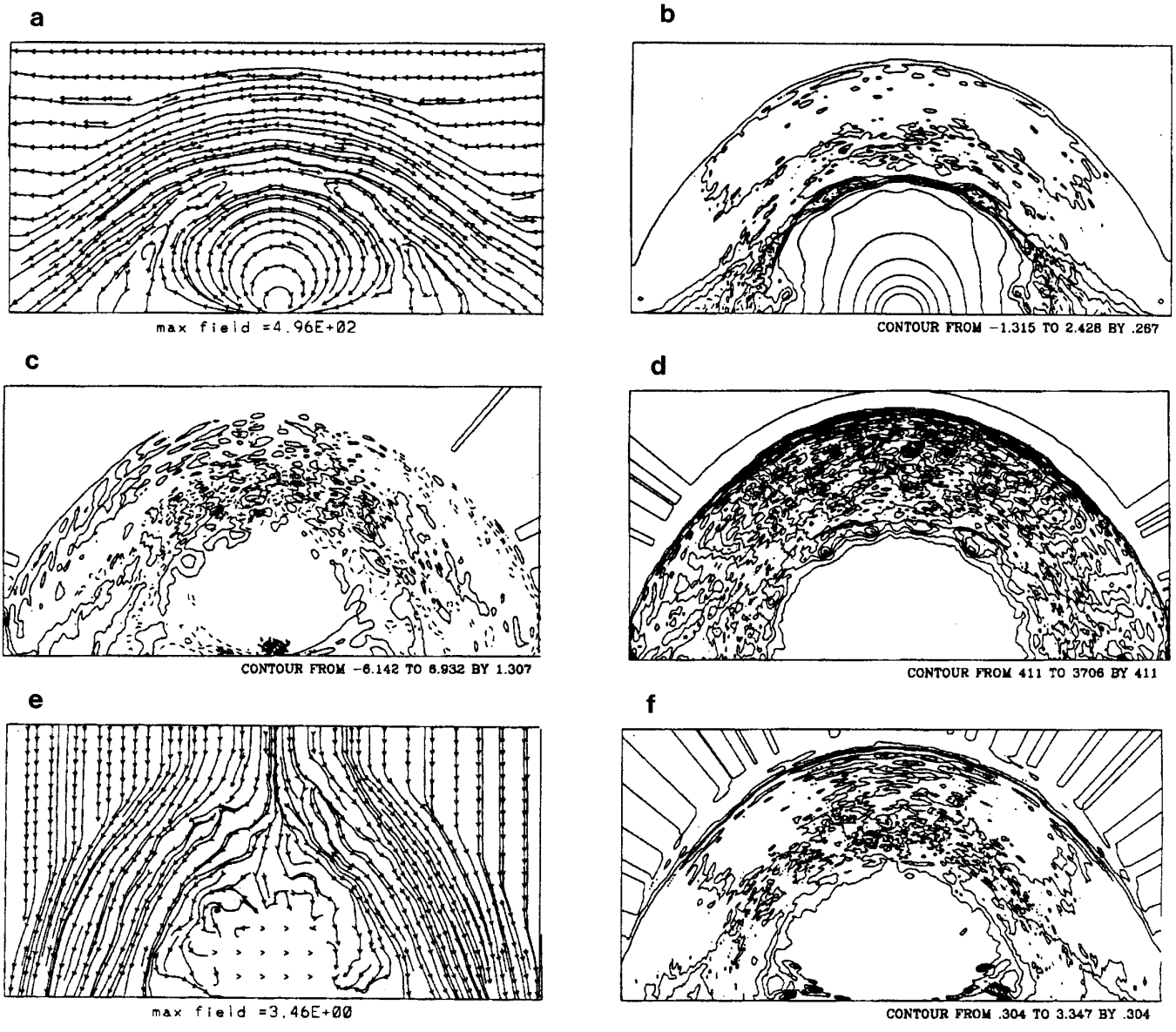

FIG. 4. Same as Fig. 3, except at the end of the run.

is no way for southward-directed magnetic field lines to pass around the magnetosphere, in the absence of greatly enhanced reconnection. Panel (a) shows a much thinner magnetopause and the appearance of a cusp in the polar regions, while panel (b) shows reconnected field lines. Panel (c) again shows field-aligned currents between the cusp and polar ionosphere. Panel (d) shows the discrete particle density. Notice the plasma clumping at the low latitude magnetopause that is indicative of magnetic island formation from tearing. Panels (e) and (f) show the penetration and acceleration of ions into the cusp. The accelerated particles appear late in the simulation after reconnection has taken place, so it is likely they were accelerated as a result of reconnection. The slow shock has all but disappeared.

The code, like the rectangular grid version described above was also subject to the nonlinear density depletion instability. This time, the frictional collision coefficient was made proportional to the relative ion–electron streaming speed and a coefficient was adjusted to make the code marginally stable against the density depletion instability, so that it ran with a minimum of dissipation. The code was also run with twice the value of the dissipation coefficient used in the run displayed in Figs. 3 and 4. The results showed a 1.3% field energy decrease and a 5.2% particle energy decrease at $t = 12$. At $t = 18$, the magnetic field

energy was also 1.3% less, while the particle energy was down by 12.0% from the case displayed. The added dissipation had little effect on the shock profile or on the position of the magnetopause or the slow shock. The amplitude of the B_y fluctuations was decreased, however. The major effect of the enhanced dissipation was on the position of the bow shock, which was displaced inward from the low density case by 0.5 $R_{\mathbf{E}}$ and 0.6 $R_{\mathbf{E}}$ at $t = 12$ and 18, respectively. The enhanced dissipation cools the particles in regions of high field fluctuation, which appears to slow the expansion of the bow shock front.

The code had another problem. The bottom center of the B_y plots, in the low altitude equatorial region, shows a region of excessive noise. The level of noise appeared to increase steadily with time. This is likely due to Alfven wave turbulence propagating in from the cusp and becoming trapped on low latitude closed field lines. This was not easily suppressed by increasing the friction coefficient ν. The most effective control agent was the use of a low-order advective transport scheme in Eq. (5) for the fluid ions. The fact that the noise is suppressed through dissipation in the fluid ions and appears in the y-component of the magnetic field supports the conjecture that the noise is due to shear Alfven waves.

7. FUTURE EXTENSIONS

The simulation has shown the two MHD shocks and evidence of reconnection. Two kinetic features emerged, namely the acceleration of ions in the cusp and the generation of the field-aligned currents due to ion demagnetization in the cusp. Perhaps more important, the code has demonstrated the application of a curvilinear coordinate system in a hybrid code and the use of subcycling the magnetic field and fluid update. The ability to simulate coexisting discrete and fluid plasma populations should prove very useful for simulation of planetary and astrophysical plasmas. An important extension, not yet implemented, would be an ability to accommodate gyrokinetic particles. Particles trapped in a planetary magnetosphere will spend part of their time in regions where the magnetic moment is conserved and time stepping to resolve the particle gyromotion is expensive. It should be possible to convert the particle's phase space coordinates back and forth from ($\mathbf{x}$, $\mathbf{v}$) to ($\mathbf{x}_{\mathrm{gc}}$, μ, ϕ, $v_{\parallel}$), the particle guiding center position, magnetic moment, gyrophase, and parallel velocity. The magnetization and guiding center drift currents can be accounted for directly in Ampere's law, and the polarization currents can be included with the electron inertial currents.

The code is entirely explicit and should readily scaleable to three dimensions and should adapt well to massively parallel computers. The major disappointment is that the hybrid code is not as robust as had originally been hoped. It does appear that the density depletion instability can be effectively controlled by the introduction of electron–ion friction. Even though the friction acts to diffuse the magnetic field fluctuations, the major impact is the cooling of the ions in turbulent regions, like behind the bow shock, where there is a rapid exchange of energy between the thermal and bulk flow particle motions and the magnetic field. With additional experience it may be possible to locate anomalous density depressions and locally increase processes that selectively damp short wavelength fluctuations and cause the depletions to be filled in. The hybrid code appears to hold considerable hope for enhancing our ability to model planetary magnetospheres.

APPENDIX A: AN ENERGY THEOREM FOR THE HYBRID CODE

The starting point is the nondissipative equation for ion motion

$$\frac{d\mathbf{v}_k}{dt} = \mathbf{E}(\mathbf{x}_k) + \mathbf{v}_k \times \mathbf{B}(\mathbf{x}_k), \tag{A1}$$

where

$$\mathbf{E}(\mathbf{x}_k) = \int S(\mathbf{x} - \mathbf{x}_k)\mathbf{E}(\mathbf{x})\, d^3x \tag{A2}$$

and S is the particle shape function. The next step is to take the scalar product of (A1) with $\mathbf{v}_k$ and sum over all the particles. This gives us an expression for the rate of change of the kinetic energy, $\mathscr{E}$ of the plasma

$$\frac{d\mathscr{E}}{dt} = \int \mathbf{E}(\mathbf{x}) \cdot \mathbf{f} d^3x, \tag{A3}$$

where $\mathbf{f}$ is the ion flux moment defined previously. The ion flux is determined from Ampere's law,

$$\mathbf{f} = n\mathbf{u}_{\mathrm{e}} + \frac{\nabla \times \mathbf{B}}{\alpha}. \tag{A4}$$

To lowest order $\mathbf{u}_{\mathrm{e}}$ is perpendicular to $\mathbf{E}$, so it does not contribute to the r.h.s. of (A3). Integration by parts casts the curl operation on the expression for the electric field. Upon making use of Faraday's law and Green's theorem, we obtain the desired conservation expression

$$\frac{d}{dt}\left[\mathscr{E} + \frac{1}{\alpha}\int B_1^2 d^3x\right] = \int \frac{\mathbf{E} \times \mathbf{B}_1}{\alpha} \cdot d\mathbf{S}, \tag{A5}$$

where the term on the right is recognized as the Poynting flux. Notice that there is no electric field energy density in

the expression. The energy conservation expression used in the code also contains terms for the ion fluid kinetic energy and the particle energy flux through the boundary surfaces.

ACKNOWLEDGMENTS

The codes were run at the Arctic Region Supercomputer Center at the University of Alaska supported by the Strategic Environmental Defense Research Program. The preparation of this paper was supported by AFOSR Grant-F49620-94-0218. I also thank Yu Lin for her assistance in investigation of the density depletion instability and Lou-Chuang Lee for useful discussions.

REFERENCES

1. J. A. Fedder and J. G. Lyon, *Geophys. Res. Lett.* **14,** 880 (1987).
2. T. Ogino and R. J. Walker, *J. Geophys. Res.* **91,** 5624 (1985).
3. S.-I. Akasofu, *Planet Space Sci.* **12,** 273 (1964).
4. C. K. Birdsall and A. B. Langdon, *Plasma Physics Via Computer Simulation* (McGraw–Hill, New York, 1985), p. 351.
5. D. Winske, *Space Sci. Rev.* **42,** 53 (1985).
6. D. W. Hewett and C. W. Nielson, *J. Comput. Phys.* **29,** 219 (1978).
7. J. A. Byers, B. I. Cohen, W. C. Condit, and J. D. Hanson, *J. Comput. Phys.* **27,** 363 (1978).
8. D. S. Harned, *J. Comput. Phys.* **47,** 452 (1982).
9. A. Mankofsky, R. N. Sudan, and J. Denavit, *J. Comput. Phys.* **70,** 89 (1987).
10. S. H. Brecht and V. A. Thomas, *Comput. Phys. Commun.* **48,** 135 (1988).
11. E. J. Horowitz, D. E. Shumaker, and D. A. Anderson, *J. Comput. Phys.* **84,** 279 (1989).
12. S. H. Brechet and J. P. Ferrante, *J. Geophys. Res.* **96,** 11209 (1991).
13. V. A. Thomas and D. Winske, *Geophys. Res. Lett.* **17,** 1243 (1990).
14. M. E. Mandt and J. F. Drake, *EOS, Trans. Am. Geophys. Union* **47,** 515 (1993).
15. P. J. Roache, *Computational fluid Dynamics* (Hermosa, Albuquerque, NM, 1982), p. 42.
16. S. T. Zalesak, *J. Comput. Phys.* **31,** 335 (1979).
17. L. G. Kelly, *Handbook of Numerical Methods and Applications* (Addison–Wesley, Reading, MA, 1966).
18. R. Holland, *IEEE Trans. Nucl. Sci.* **NS-30,** 4589 (1983).
19. J. W. Eastwood, W. Arter, N. J. Brealey, and R. W. Hockney, *Comput. Phys. Commun.* **87,** 155 (1995).
20. N. K. Madsen, *J. Comput. Phys.* **119,** 34 (1995).
21. P. Monk and E. Süli, *SIAM J. Numer. Anal.* **31,** 393 (1994).
22. J. Mathews and R. L. Walker, *Mathematical Methods of Physics* (Addison–Wesley, Reading, MA, 1970), p. 418.
23. D. W. Swift and C. Allen, *J. Geophys. Res.* **92,** 10,015 (1987).
24. H. Naitou, S. Tokuda, and T. Kamimura, *J. Comput. Phys.* **33,** 86 (1979).
25. F. Kazeminezhad, J. M. Dawson, J. N. Leboeuf, R. Sydora, and D. Holland, *J. Comput. Phys.* **102,** 277 (1992).
26. P. Song, C. T. Russell, and M. F. Thomsen, *J. Geophys. Res.* **97,** 8295 (1992).
27. M. Yan and L. C. Lee, *Geophys. Res. Lett.* **21,** 629 (1994).