JOURNAL OF COMPUTATIONAL PHYSICS 17, 246–275 (1975)

# Axisymmetric Multifluid Simulation of High Beta Plasmas with Anisotropic Transport Using a Moving Flux Coordinate Grid

D. V. ANDERSON*

*Naval Research Laboratory, Washington, DC 20375*

Received April 18, 1974

A method for solving the complete set of multifluid MHD equations is given. The algorithm SLIDE, which is derived from flux-corrected transport methods, is used to solve the fluid equations on a moving 2D axisymmetric coordinate grid in which one set of grid lines represents magnetic flux surfaces while a second set is constructed orthogonally. Preservation of the diagonal form of transport tensors allows accurate representation of anisotropic transport. Results are shown for a problem in theta-pinch geometry and are contrasted to the results of a more conventional $(r, z)$ code.

## I. INTRODUCTION

The modeling of plasma dynamic flows by a multifluid representation is of interest in several physical problems. Some of these include high beta[1] disturbances of the ionosphere, laser produced plasmas, and several plasma confinement problems such as those encountered in $\theta$-pinches, cusp devices, and mirror confinement schemes. Specifically we address the problem of simulating a plasma fluid which is characterized both by large anisotropies and by high beta convection of the magnetic field.

Early attempts to compute the MHD flow were computed by Lagrangian methods in which the grid points were constrained to lie on flux tube surfaces. For theta-pinch modeling Hain [1] used a scheme which orthogonalized the grid every time step and Hertweck and Schneider [2] reported a similar scheme. Both of these methods assumed ideal MHD in the direction across the magnetic field and hence no relative motion of the fluid to the field was permitted. The article by Roberts and Potter [3] gives a good review of several other MHD numerical models and discusses some of the difficulties encountered in the solution of them.

* Present address: Lawrence Livermore Laboratory L-388, P. O. Box 808 Livermore, CA 94550.

[1] For our purposes we use the conventional definition of the plasma beta ($\beta$) as the ratio of plasma pressure to the magnetic pressure. That is, $\beta = 8\pi P/B^2$.

A more recent Eulerian model for solution of MHD equations in theta-pinch geometry with anisotropic heat conductivity has been presented by Lindemuth and Killeen [4]. The representation of the heat flow is good until the field lines become oblique to the $(r, z)$ grid cells at which point the numerical effects cause the heat flow tensor to isotropise locally.

Also Chu and Johansson [5], Chu, Morton, and Roberts [6], and Morris and Nicoll [7] have studied anisotropic heat flow problems where the grid is at rest and for which the conductivity tensors are diagonal or diagonally dominant; their methods may then apply to the low-beta plasmas where distortion of the magnetic field lines is small.

Many authors are currently interested in the use of grids which are neither purely Lagrangian nor Eulerian. They use terms like "semi-Lagrangian," "quasi-Lagrangian," "pseudo-Lagrangian," "mixed Eulerian–Lagrangian," "observer grid," and others. For toroidal plasma confinement problems Grimm and Johnson [8] have developed a quasistatic MHD model in natural coordinates closely related to toroidal coordinates. Brackbill and Pracht [9], extending a method developed by Hirt, Amsden, and Cook [10] have used a semi-Lagrangian mesh on which they solve the MHD equations for a theta-pinch device; for large convective velocities the grid tends to be Lagrangian but for small velocities the grid relaxes to an Eulerian prescription. Their algorithm is first order in time and presently does not include facility for anisotropic transport. An artificial viscosity term is included for reducing numerical Gibbs phenomena and a smoothing coefficient is introduced to stabilize the rezone algorithm; this is in sharp contrast to the flux-corrected transport (FCT) algorithms [11, 12] where the numerical Gibbs phenomena is eliminated optimally with minimum numerical diffusion. Because Brackbill [9] raises a question regarding an inappropriate constraint in the FCT methods we stress that the FCT algorithms do not impose nonphysical constraints. When the density near a steep density gradient is nearly zero, the numerical Gibbs phenomena could produce a negative result. Suppression of this phenomena guarantees positivity as well as giving reasonable profiles for shock and contact discontinuties.

The strict use of Lagrangian grids in multidimensional fluid flows is well known to cause severe difficulties associated with the large distortions of initially rectangular grid cells; these problems are particularly nasty in sheared flows. A popular remedy is to follow contours of some important physical quantity and identify them as a set of primary grid lines. A second set of grid lines may then be chosen orthogonal as was done in the references [1, 8, 13] and is also the alternative used by the author. Others have chosen the second set of grid lines to be contours of a secondary physical quantity and while these are not necessarily orthogonal they may lead to tractable results. Methods for orthogonalizing the grid are given by Potter and Tuttle [13] and by Anderson and Clark [14], and we use the latter method.

Here we present a method using the SLIDE algorithm [15] (also described in Appendix C) which rezones the grid at every timestep in such a manner so that the points follow the flux surfaces; this preserves the diagonal form of the transport tensors. The rezoning is an integral part of the FCT fluid algorithm and is performed in such a manner as to reduce the numerical diffusion compared to an Eulerian treatment.

Except in the most symmetric plasma flows the magnetic flux tube surfaces will distort such that the metric of a moving magnetic coordinate system will be an explicit function of time. The SLIDE algorithm properly accounts for the effect of this time-dependent metric.

A major restriction on the transformation algorithm we present here is that it must be nonsingular. Furthermore it is assumed that problems in two or three dimensions are solved by a method of time-step splitting in the orthogonal coordinates, that is, the multidimensional problem is decomposed into a succession of one-dimensional equations to be solved. For each 1D equation the appropriate transformation is made to use the Cartesian algorithm SHASTZ of Boris and Gardner [11]. Any rezoning is done automatically by the SHASTZ algorithm in such a manner to reduce, rather than increase, the numerical diffusion. The SLIDE algorithm we use is a method for solving the split form of a general fluid-like equation along an orthogonal coordinate line.

For the sake of simplicity the method presented here assumes axisymmetry and open flux tubes. These restrictions could be removed in a more complicated model based on essentially the same ideas. Otherwise many plasma effects are included such as drag and heating produced by beaming instabilities, thermal-electric generation of magnetic fields, anisotropic thermal conductivity, and a quite general Ohm's law. A much simpler MHD model would have sufficed for this demonstration except that an $(r, z)$ code had been developed [21] which already had included these many plasma phenomena. This $(r, z)$ code has been used then as a control or benchmark for the development of the flux code described here and it proved easy to carry the details of the physics, practically intact, from one code to the other.

In the next section the flux tube coordinates are compared with $(r, z)$ and flux line coordinates. Section III presents the multifluid equations and gives the "split' forms to be used in the orthogonal flux tube coordinates. The Lagrangian equations of flux tube motion are derived in Section IV and the metric scale factors are found. In Section V the continuously rezonable numerical algorithm for the time-split form of the equations is presented. Finally, in Section VI we show results for a very high beta plasma dynamic flow in theta-pinch geometry where the heat flow anisotropy for electrons is $\sim 10^5$. Details of the algorithms SHASTA, SHASTZ, and SLIDE are presented in the appendices.

## II. Flux Tube Coordinate Representation

To show the advantage of flux coordinates we compare the κ and ν tensors for heat conductivity and collisional momentum transfer between fluids in various coordinate systems. Using the usual two-independent component tensor frequently used in plasma physics we may express a tensor A as:

$$\mathsf{A} = A_\perp \mathsf{I} + [(A_\parallel - A_\perp)/B^2]\,\mathbf{BB}. \tag{2.1}$$

First we consider axisymmetric systems and single out the conventional $(r, z)$ cylindrical system and compare it to a flux-tube $(\Psi, X)$ system. The matrix representation of A in $(r, z)$ is given

$$\mathsf{A} = A_\perp \begin{pmatrix} 100 \\ 010 \\ 001 \end{pmatrix} + \frac{(A_\parallel - A_\perp)}{B^2} \begin{pmatrix} B_r^2 & B_\theta B_r & B_z B_r \\ B_\theta B_r & B_\theta^2 & B_z B_\theta \\ B_z B_r & B_z B_\theta & B_z^2 \end{pmatrix} \tag{2.2}$$

Whenever the flux lines are oblique to the $(r, z)$ grid lines it is clear that the off-diagonal terms will be significant.

The axisymmetric flux tube coordinates are described by surfaces of constant $\Psi$ (the flux function) and constructed normal surfaces of constant $X$. $\theta$ is again the third and ignorable coordinate. Details of this coordinate system are presented in Appendix E.

Let us define $\mathbf{B}'$ as the projection of $\mathbf{B}$ on to the $(r, z)$ plane. It should be noted that in the flux tube coordinates the normal to the $\chi$ surface points in the $\mathbf{B}'$ direction (having no $\theta$ component) which is not necessarily the $\mathbf{B}$ direction. When $B_\theta \neq 0$ then $\mathbf{B} \neq \mathbf{B}'$ and the flux lines will form helix-like structures, but due to their axisymmetry still define the flux tubes uniquely. Here the matrix version of the tensor is

$$A = A_\perp \begin{pmatrix} 100 \\ 010 \\ 001 \end{pmatrix} + \frac{(A_\parallel - A_\perp)}{B^2} \begin{pmatrix} 0 & 0 & 0 \\ 0 & B_\theta^2 & B_\theta B_X \\ 0 & B_X B_\theta & B_X^2 \end{pmatrix}. \tag{2.3}$$

Two off-diagonal elements remain but their adverse effect, unlike the $(r, z)$ case, will be shown to be negligible.

To make A completely diagonal the coordinate lines must be the flux lines themselves. Unfortunately we lose the explicit ignorable coordinate and end up in a 3D system. Figure 1 shows pictorially what the coordinate systems look like; note the screwlike appearance of the flux line coordinates. Let $\Psi$ be the usual flux coordinate, $\beta$ be a coordinate along $\mathbf{B}$ and set $\alpha$ as the mutually orthogonal coordinate. Then in terms of this system $(\Psi, \alpha, \beta)$ the tensor A becomes

$$\mathsf{A} = \begin{pmatrix} A_\perp & 0 & 0 \\ 0 & A_\perp & 0 \\ 0 & 0 & A_\parallel \end{pmatrix}. \tag{2.4}$$

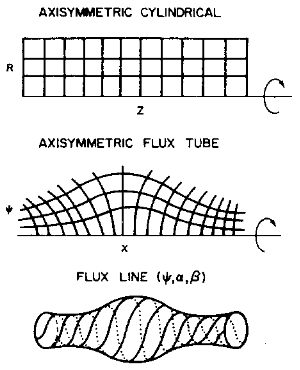

FIG. 1. Possible coordinate grids are shown. To keep the algorithm two-dimensional, the axisymmetric flux tube coordinate system was chosen over the more general flux line grid.

Choosing the flux tube coordinates with A given in (2.3) can be justified by understanding the effect of the two remaining off-diagonal elements. They couple the parallel flow along $\chi$ with flow along $\theta$, the ignorable coordinate. Since all $\partial/\partial\theta \equiv 0$ no heat will flow along $\chi$ due to gradients in $\theta$. Likewise, gradients in $\chi$ producing $\theta$ heat flow can have no effect due to the constancy of $T$ in $\theta$. These off-diagonal elements in the tensor for collisional momentum transfer are nonzero. They allow $\theta$ drifts to cause drag in the $\chi$ direction, but it does not appear that this is unphysical.

As an example of the advantage of the flux tube formulation we examine the heat flow terms. For the $(r, z)$ coordinates we get

$$\nabla \cdot (\kappa \cdot \nabla T) = \frac{1}{r}\frac{\partial}{\partial r}\left\{r\left[\kappa_\perp \frac{\partial T}{\partial r} + \left(\frac{\kappa_\parallel - \kappa_\perp}{B^2}\right)\left(\frac{\partial T}{\partial r} B_r^2 + \frac{\partial T}{\partial z} B_r B_z\right)\right]\right\}$$
$$+ \frac{\partial}{\partial z}\left\{\kappa_\perp \frac{\partial T}{\partial z} + \left(\frac{\kappa_\parallel - \kappa_\perp}{B^2}\right)\left(\frac{\partial T}{\partial r} B_r B_z + \frac{\partial T}{\partial z} B_z^2\right)\right\}. \quad (2.5)$$

Since a splitting method is used in our numerical solution, a problem is caused by the mixed derivatives here. Further there is no way to neatly separate the effects of the $\parallel$ and $\perp$ heat flow. In the flux tube system, the heat flow term is

$$\nabla \cdot (\kappa \cdot \nabla T) = \frac{1}{h_\Psi h_\chi h_\theta}\left\{\frac{\partial}{\partial \Psi}\left(\frac{h_\chi h_\theta}{h_\Psi} \kappa_\perp \frac{\partial T}{\partial \Psi}\right)\right.$$
$$\left. + \frac{\partial}{\partial \chi}\left(\frac{h_\Psi h_\theta}{h_\chi}\left[\kappa_\perp + \left(\frac{\kappa_\parallel - \kappa_\perp}{B^2}\right) B_\chi^2\right]\frac{\partial T}{\partial \chi}\right)\right\}. \quad (2.6)$$

The flux tube version exhibits no mixed derivatives and the cross-field heat flow is completely separated and will depend only on $\kappa_\perp$. Heat flow along $\chi$ of course is still mixed as it should be when $B_\theta \neq 0$. If $B_\theta \equiv 0$ then heat flow along $\chi$ will depend on $\kappa_\parallel$ only.

A related example has been worked out where we studied tensor heat flow equations on a 2D square Cartesian grid where the magnetic field line makes a 45° angle with the grid lines. A simple algorithm based on straightforward explicit differencing of Eq. (2.5) has been used to solve the heat flow on this domain. We look at the case where $\kappa_\perp = 0$ and $\kappa_\parallel \neq 0$. Figure 2 shows the initial profile and

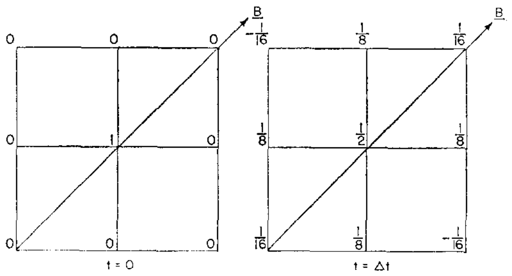

FIG. 2. Temperature field on a square mesh before and after first timestep. The extreme anisotropy is not well represented, even after one timestep ($\kappa_\perp = 0$; $\kappa_\parallel \neq 0$).

the result after one time step. It is clear the heat must flow in the $\perp$ direction along the grid lines even when $\kappa_\perp = 0$. After 35 cycles the temperature contours in Fig. 3 were obtained. The effective cross-field heat flow is about $\frac{1}{4}$ the longitudinal flow so we conclude that conventional Eulerian fixed grid attempts on the heat

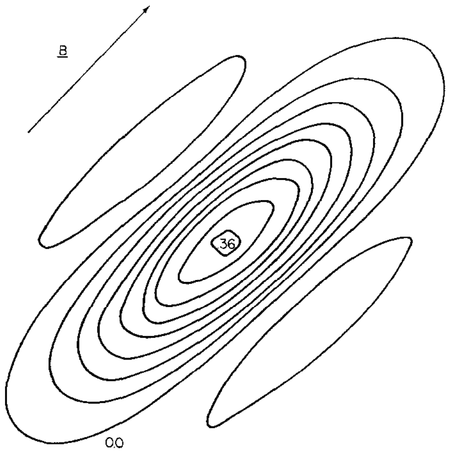

FIG. 3. Temperature field on the square mesh after 35 cycles. The effective anisotropy ratio seems to be $\kappa_\parallel/\kappa_\perp \sim 4$; $\kappa_\parallel \neq 0$; $\kappa_\perp = 0$.

conduction equation can handle only small anisotropies in $\kappa$, certainly no more than a factor of $\sim 3$ to 4 can be tolerated.

The above example used a point source. If one repeats the computation for a distributed hot spot (say a circle) then greater anisotropies can be represented. That is to say very fine grids can improve the result of attempted simulation of large anisotropies. At great expense one could represent ratios ($\sim$ say $(K_\parallel/K_\perp) \cong 100$) by a 25-fold increase in resolution; for a 2D grid this means $(25)^2 = 625$ fold increase in the number of grid points. Alternatively in the flux coordinates infinite anisotropies ($K_\parallel/K_\perp \sim \infty$) are representable even on a coarse mesh.

## III. Multifluid Equations

The full set of multifluid equations with tensor transport but with scalar pressure is written

$$(\partial \rho_i/\partial t) + \nabla \cdot (\mathbf{V}_i \rho_i) = 0, \tag{3.1}$$

$$\frac{\partial}{\partial t}(\rho_i \mathbf{V}_i) + \nabla \cdot (\rho_i \mathbf{V}_i \mathbf{V}_i) = -\nabla P_i + q_i\left(\mathbf{E} + \frac{\mathbf{V}_i \times \mathbf{B}}{c}\right) + \rho_i \sum_j \nu_{ij} \cdot (\mathbf{V}_j - \mathbf{V}_i), \tag{3.2}$$

$$\begin{aligned}\frac{\partial}{\partial t}(T_i) + \nabla \cdot (\mathbf{V}_i T_i) = &-(\gamma_i \cdot 2)\, T_i(\nabla \cdot \mathbf{V}_i) + \frac{m_i(\gamma - 1)}{k\rho_i} \nabla \cdot (\kappa_i \cdot \nabla T_i) \\ &+ \frac{\gamma - 1}{k} m_i \sum_j [(\mathbf{V}_{ph_{ij}} - \mathbf{V}_i) \cdot \nu_{ij} \cdot (\mathbf{V}_j - \mathbf{V}_i)] \\ &+ \sum_j |\nu_{ij}| (T_j - T_i) \cdot \left(\frac{m_i}{m_i + m_j}\right),\end{aligned} \tag{3.3}$$

$$\partial \mathbf{B}/\partial t = -c\nabla \times \mathbf{E}, \tag{3.4}$$

where the various symbols are defined:

| | |
|---|---|
| $\rho_i$ | mass density of $i$th fluid, |
| $\mathbf{V}_i$ | velocity of $i$th fluid, |
| $P_i = n_i k T_i$ | scalar pressure of $i$th fluid, |
| $\mathbf{E}$, $\mathbf{B}$, $c$ | electric field, magnetic field, and light speed, |
| $\nu_{ij}$ | collision frequency tensor of particle $i$ by species $j$, |
| $T_i$ | temperature of $i$th fluid, |
| $\gamma_i$ | thermodynamic ratio of specific heats, |
| $\kappa_i$ | heat conduction tensor for $i$th fluid, |
| $m_i$ | mass of particle of species $i$, |
| $n_i$ | number density of species $i$, |
| $\mathbf{V}_{ph_{ij}}$ | phase velocity for energy partition between species $j$ and $i$. |

For each ion or neutral species these fluid equations are used as given. Electrons, however, are treated in the quasineutrality limit. We retain Eq. (32) with the inertial terms on the left-hand side set to zero and use this equation to determine the electric field. Equation (33) is solved as is for the electron temperature. Electron density is obtained from $en_e = \sum_i q_i n_i$. Ampere's law $\mathbf{J} = (c/4\pi)(\nabla \times \mathbf{B})$ is used to determine $\mathbf{v}_e$ and as such is not solved as a partial differential equation. Furthermore, an ideal gas law $P = nkT$ is assumed.

The method we present is not sensitive to the precise set of multifluid equations picked nor on the method used to close the set. The ones used here were taken from Manheimer [16]. Other sets of equations such as those of Braginskii [17] could be solved by the algorithm. Equation (3.4) is sometimes written in the form of a fluid equation and solved accordingly in many fixed grid Eulerian methods. In flux coordinates, however, $B_r$ and $B_z$ are trivially evaluated while a fluid-like equation is still used to find $B_\theta$.

All of the MHD multifluid equations are of the form

$$(\partial f/\partial t) + \nabla \cdot (\mathbf{V}f) = \nabla \cdot \mathbf{g} + \mathbf{d} \cdot \nabla a + e. \tag{3.5}$$

When this is expressed in terms of an arbitrary orthogonal curvilinear coordinate system, we get

$$\begin{aligned}\frac{\partial f}{\partial t} &+ \frac{1}{h_1h_2h_3}\left[\frac{\partial}{\partial q_1}(V_1 f h_2 h_3) + \frac{\partial}{\partial q_2}(V_2 f h_1 h_3) + \frac{\partial}{\partial q_3}(V_3 f h_1 h_2)\right] \\ &= \frac{1}{h_1h_2h_3}\left[\frac{\partial}{\partial q_1}(g_1 h_2 h_3) + \frac{\partial}{\partial q_2}(g_2 h_1 h_3) + \frac{\partial}{\partial q_3}(g_3 h_1 h_2)\right] \\ &\quad + \frac{d_1}{h_1}\frac{\partial a}{\partial q_1} + \frac{d_2}{h_2}\frac{\partial a}{\partial q_2} + \frac{d_3}{h_3}\frac{\partial a}{\partial q_3} + e.\end{aligned} \tag{3.6}$$

The scale factors $h_i$ are defined in terms of elements of arc length by $h_i\, dq_i = ds_i$. In the previous section the orthogonal flux tube coordinates $(\Psi, X)$ were introduced. In terms of them Eq. (3.6) takes the form

$$\begin{aligned}\frac{\partial f}{\partial t} &+ \frac{1}{h_\Psi h_X h_\theta}\left[\frac{\partial}{\partial \Psi}(V_\Psi f h_X h_\theta) + \frac{\partial}{\partial X}(V_X f h_\Psi h_\theta)\right] \\ &= \frac{1}{h_\Psi h_X h_\theta}\left[\frac{\partial}{\partial \Psi}(g_\Psi h_X h_\theta) + \frac{\partial}{\partial X}(g_X h_\Psi h_\theta)\right] \\ &\quad + \frac{d_\Psi}{h_\Psi}\frac{\partial a}{\partial \Psi} + \frac{d_X}{h_X}\frac{\partial a}{\partial X} + e.\end{aligned} \tag{3.7}$$

We intend to solve Eq. (3.7) by the method of splitting. Two separate equations (one involving the $\Psi$ derivatives, the other the $X$ derivative terms) are fashioned such that their sum represents Eq. (3.7). This way we obtain one-dimensional equations.

The split forms of Eq. (3.7) are then

$$\frac{\partial f}{\partial t} + \frac{1}{h_\Psi h_\chi h_\theta}\frac{\partial}{\partial \Psi}(V_\Psi f h_\chi h_\theta) = \frac{1}{h_\Psi h_\chi h_\theta}\frac{\partial}{\partial \Psi}(g_\Psi h_\chi h_\theta) + \frac{d_\Psi}{h_\Psi}\frac{\partial a}{\partial \Psi} + \frac{e}{2}, \tag{3.8}$$

and

$$\frac{\partial f}{\partial t} + \frac{1}{h_\Psi h_\chi h_\theta}\frac{\partial}{\partial X}(V_\chi f h_\Psi h_\theta) = \frac{1}{h_\Psi h_\chi h_\theta}\frac{\partial}{\partial X}(g_\chi h_\Psi h_\theta) + \frac{d_\chi}{h_\chi}\frac{\partial a}{\partial X} + \frac{e}{2}. \tag{3.9}$$

Equations like these are the ones to be differenced in the scheme presented here. Appendix C details the procedure for doing this.

## IV. Lagrangian Motion of Flux Tubes

Before the fluid equations can be solved on the moving flux grid we must know the location and motion of the grid in space. The important motion is that normal to the flux tube surface. Let $s$ be a laboratory coordinate normal to a flux tube having dimensions of length, and let an observer ride on this surface; he will see

$$d\Psi = 0 = (\partial\Psi/\partial s)\,ds + (\partial\Psi/\partial t)\,dt. \tag{4.1}$$

The normal velocity is

$$V_n = ds/dt = -(\partial\Psi/\partial t)/(\partial\Psi/\partial s). \tag{4.2}$$

But from the metric scale factor definition

$$h_\Psi = 1/(\partial\Psi/\partial s) = 1/rB', \tag{4.3}$$

and from Maxwell's equation $\partial\mathbf{B}/\partial t = -c\nabla \times \mathbf{E}$ we have

$$\partial\Psi/\partial t = -rcE_\theta\,. \tag{4.4}$$

So it follows

$$V_n = cE_\theta/B' \tag{4.5}$$

which is just the component of the $\mathbf{E} \times \mathbf{B}'$ drift normal to the flux surface. Although the flux tubes have exactly this motion, material will drift relative to the tubes. Electrons, due to their small inertial forces (here set zero), will closely follow the flux tubes except for the effects of resistivity and electron pressure gradients. Taking $r$ and $z$ components we get the desired Lagrangian motion of the grid

$$dr/dt = cE_\theta B_z/(B')^2; \qquad dz/dt = -cE_\theta B_r/(B')^2. \tag{4.6}$$

Prior to the fluid timestep we use these equations to get the new array of positions to be found at the end of the time step and from these compute new values for $B_r$, $B_z$, $B'$, $h_\Psi$, $h_\theta$, and $h_\chi$ using Eqs. (2.5)–(2.10).

In the time-split method of solving the fluid equations we will first solve the equations along the coordinate $\Psi$ perpendicular to the flux surfaces with the grid moving as $c(\mathbf{E} \times \mathbf{B}'/B^2)$. The second time-split step will solve the split equations in the coordinate $\chi$ in the $\mathbf{B}'$ direction along the flux tube with the grid points moving along $\mathbf{B}'$ by a prescription depending on the dynamic plasma motion but constrained by orthogonality and by not allowing the flux tube to move.

For each step the algorithm SLIDE [15] will require not only the metric factors $h_\chi$, $h_\Psi$, $h_\theta$ both at the end and beginning of the timestep but will also need the effect of their time dependence

$$\partial\beta/\partial t = (\partial/\partial t)(h_\chi h_\Psi h_\theta). \tag{4.7}$$

The total derivative $d\beta/dt$ is easily computed from the strict time difference at the moving grid point. So to get $\partial\beta/\partial t$ we can write

$$\partial\beta/\partial t = (d\beta/dt) - \mathbf{V}_{\text{grid}} \cdot \nabla\beta. \tag{4.8}$$

In principle, Eq. (4.8) could be solved for $\partial\beta/\partial t$. When this is done numerical errors associated with spatial interpolation occur and produce a small but annoying anomaly. That is, a constant density force free fluid will get gradually distorted by the moving dilating or contracting coordinate grid. Of course it is the inclusion of the $\partial\beta/\partial t$ term which is to prevent such incorrect distortions, but errors in $\partial\beta/\partial t$ will allow smaller distortions to occur. Instead we use a method given in Appendix D which uses the properties of an ideal force free fluid to compute $\partial\beta/\partial t$ exactly down to round off.

It should then be clear that before the fluid timestep is performed one is equipped with knowledge of the new grid to be found at the end of the time step including the new metric factors and $\partial\beta/\partial t$.

## V. Moving Observer Grid Numerical Method

The code we employ, SLIDE, is derived from the sliding zone SHASTZ algorithm [11] which in turn is a generalization of the SHASTA algorithm [12]. SHASTA and SHASTZ are reviewed in Appendices A and B, respectively. SLIDE has several remarkable properties needed to solve the fluid equations in the orthogonal flux coordinates. The moving observer solves the fluid equations in the laboratory frame. The coordinates at the beginning of the timestep are frozen and used in the lab solution, call them $(\Psi_i^0, \chi_j^0)$. In this frozen system the equations are solved either for motions along $\hat{\Psi}_i^0$ or along $\hat{\chi}_j^0$ by the method of splitting. So let us analyze the split timestep where motion is across the field lines in the $\hat{\Psi}^0$ direction. The Lagrangian equations have given the new positions of the grid in $(r, z)$ to be found at $t + \delta t$ and these correspond to new values $\Psi_i^0 + \delta\Psi_i^0$ and $\chi_j^0 + \delta\chi_j^0$ at the old time $t$ as shown in Fig. 4. The algorithm has a rezone capability which is

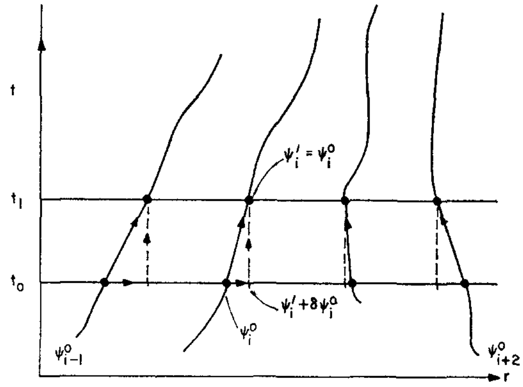

FIG. 4. Flux grid point characteristics. Trajectories of the grid points are plotted in the $(r, t)$ plane. We chose $z = 0$ for purposes of illustration. It is important to separate displacements in space from the more general displacements, along a trajectory, in space and time. Since the algorithm solves the equations in a rest laboratory frame, it is the spatial grid displacements which are to be used in the numerical rezone.

used to interpolate the result on $(\Psi_i^0 + \delta\Psi_i^0)$. Now the answers are computed at the new grid point locations, but we are still located in the frozen coordinate system belonging to the old time, the beginning of the timestep. To get updated we merely note the total derivative $d\Psi/dt = 0$ which means $\Psi_i = \Psi_i^0$. So the new grid points are relabeled with the old names. This is clearly required in the flux space where the grid is invariant so the labels of the grid line intersections remain the same.

A further refinement is allowed which complicates the situation but may be very useful in practice. One may wish to rezone in the flux space itself in which case the labels of the grid points will change from one timestep to the next. In the frozen coordinates the grid point moves to

$$\begin{aligned} \Psi_i^0 &\rightarrow \Psi_i^0 + \delta\Psi_i^0 + \Delta\Psi_i^0, \\ \chi_j^0 &\rightarrow \chi_j^0 + \delta\chi_j^0 + \Delta\chi_j^0, \end{aligned} \tag{5.1}$$

where $\delta$ is again the increment due to the spatial motion of the flux tubes. But $\Delta$

represents the increment-in the flux space. But after displacements both in space and in time corresponding to the moving coordinates

$$\begin{aligned} \Psi_i &\to \Psi_i + \Delta\Psi_i, \\ \chi_j &\to \chi_j + \Delta\chi_j, \end{aligned} \tag{5.2}$$

gives the reassignment of grid names in the flux space at the new time $t + \delta t$. In performing this more general rezone equation one must be careful to take this extra term into account in the computation of $\partial\beta/\partial t$ which is described in Appendix D.

The algorithm SLIDE, described in detail in Appendix C, was used to solve the fluid equations on the moving orthogonal grid. First a half-timestep is performed in the $\hat{\Psi}$ direction followed by a full-timestep in $\hat{\Psi}$. The half-timestep makes the fluid step second order in time by determining many of the source terms at a time-centered level. Then a half- and full-timestep along $\hat{\chi}$ are performed. These four steps have then advanced the values forward for one full-timestep.

The important features of SLIDE include:

(A) Rezone of grid at each timestep with reduction of numerical diffusion,

(B) Physically positive quantities such as density are kept positive,

(C) Steep gradients are preserved and Gibb's-like phenomena suppressed by FCT (Flux Corrected Transport),

(D) Transformation to a formally Cartesian system solvable by SHASTZ [11] with allowance for time-dependent metrics.

Incidentally the SLIDE algorithm can be applied to other problems involving fluid equations and quasi-Lagrangian grids (for example where $\Psi$ is the velocity stream function).

## VI. Multifluid MHD Test Problems in Theta-Pinch Geometry

A test problem was chosen which would demonstrate most of the features built into the model. An expanding very hot (10 keV) hydrogen plasma slug in a background plasma with an initially uniform 10kG magnetic field $B_z$ was modeled. The initial parameters are shown in Fig. 5. Both the flux code ($\Psi$, $X$) and the $(r, z)$ code were used to simulate the problem and the results of the two codes are compared.

The handling of boundary conditions is quite general and for each equation they may be posed independently. General inhomogeneous Dirichlet, Neumann, or mixed specifications as well as the periodic boundary condition are among those

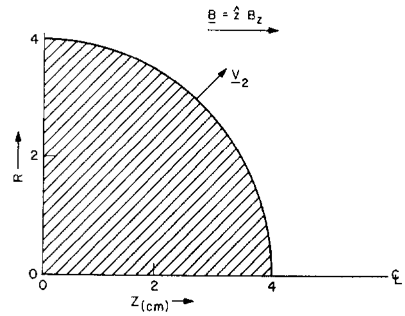

FIG. 5. Initial profile for expanding plasma slug. Fluid 1 is initially at rest with uniform density $n_1 = 10^{14}\,\text{cm}^{-3}$. Fluid 2 is the spherical slug with uniform density $n_2 = 10^{18}\,\text{cm}^{-3}$ inside the sphere. It is uniformly expanding with velocity $4 \times 10^8$ cm/sec at the surface. Electrons are started quasineutral and the uniform magnetic field is $10^4$ gauss. Temperatures are set at $T_1 = 10$ eV, $T_2 = 10$ keV, $T_e = 0.1$ keV; $\mathbf{B} = \hat{z}B_z$.

representable. In Table I is a list of the boundary values used in the $(r, z)$ code. A nearly identical set was adopted for the flux code. As evident in the table most of the boundary conditions are homogeneous Neumann. Of interest to the results shown here are the Neumann boundary conditions imposed on the temperature equations which correspond to the physical process of placing thermal insulation at the boundaries.

The heat flow terms in the flux code representation separate nicely, and it becomes possible to solve the cross-field transport explicitly while the transport along the flux tube is done implicitly. Since the parallel heat flow represents the fastest diffusion speed in the problem, its implicit solution allows us to predicate the timestep value upon slower speeds, notably the magneto-acoustic speed, the cross-field thermal diffusion speed, or the fluid velocities.

Using the anisotropic heat conduction tensor is straightforward in the flux code but it is not clear how to proceed in the $(r, z)$ code. If we simply use the components $\kappa_\perp$ and $\kappa_\parallel$ given by the physics, the large value for $\kappa_\parallel$ will severely limit the timestep in an explicit calculation. If we attempt to solve the heat flow on a separate implicit time-split step, we find that the mixed derivatives in Eq. (2.13) will destroy the stability properties of the implicit method. So if we are willing to use the small explicit timestep, we can proceed with the calculation. The isotropization produced by the numerics described in Section II will dominate, and all the heat will rapidly conduct out of the system both along and across field lines.

A second choice would be to limit $\kappa_\parallel$, to say $4\kappa_\perp$, so that an explicit calculation would have both a reasonably large timestep and would keep the anisotropy ratio representable by the numerics. Then the heat flow across the field will be accurate but along the field it will be much too low. Unlike the first choice where a plasma

TABLE I

Boundary Conditions on the $(r, z)$ Rectangle for the $(r, z)$ Code[a]

| Physical variable | $r = 0$ | $r = r_{\max}$ | $z = 0$ | $z = z_{\max}$ |
|---|---|---|---|---|
| $T_1$ | $N$ | $N$ | $N$ | $N$ |
| $T_2$ | $N$ | $N$ | $N$ | $N$ |
| $T_e$ | $N$ | $N$ | $N$ | $N$ |
| $n_1$ | $N$ | $D$ | $N$ | $D$ |
| $n_2$ | $N$ | $N$ | $N$ | $N$ |
| $V_{r_1}$ | 0 | 0 | $N$ | 0 |
| $V_{r_2}$ | 0 | $N$ | $N$ | $N$ |
| $V_{\theta_1}$ | 0 | 0 | $N$ | 0 |
| $V_{\theta_2}$ | 0 | $N$ | $N$ | $N$ |
| $V_{z_1}$ | $C$ | 0 | $N = -1$ | 0 |
| $V_{z_2}$ | $C$ | $N$ | $N = -1$ | $N$ |
| $B_r$ | 0 | 0 | $N = -1$ | 0 |
| $B_\theta$ | 0 | 0 | $N = -1$ | 0 |
| $B_z$ | $N$ | $D$ | $N$ | $D$ |

[a] The conditions in the flux code are analogous. Great care must be exercised in choosing a set of boundary values to keep the problem well posed and stable. The designation $N$ refers to the homogeneous Neumann boundary condition. $D$ and 0 refer to inhomogeneous and homogeneous Dirichlet boundary values. $N = -1$ is a special inhomogeneous Neumann condition such that $f_n = -f_{n-1}$. Finally the designation $C$ refers to the conservative boundary condition which generates a boundary value guaranteed to conserve the integral of the physical quantity.

hot spot cools off too fast, here it will not cool fast enough due to the suppression of $\kappa_\parallel$.

Finally a compromise can be found as follows. The value of $\kappa_\parallel$ is ignored for the purposes of determining the allowed timestep. Then $\kappa_\parallel$ is limited such that it does not violate this timestep condition. If cross-field heat flow $\kappa_\perp$ gives the fastest speed in the determination of the timestep, then $\kappa_\parallel = \kappa_\perp$. However if some other speed such as the magneto–acoustic velocity determines the timestep, then $\kappa_\parallel \gg \kappa_\perp$ may be permitted if the thermal diffusion speed is much less than, say, the magneto–acoustic speed. For many problems $\kappa_\parallel$ will be restricted orders of magnitude, and the overall effect of this compromise method will be to keep the plasma too hot. Although this choice may not appear to be satisfactory, neither are the other options described above. For the sake of comparison with the flux code some model of heat flow must be used in the $(r, z)$ code, and so we use this one which limits $\kappa_{\parallel}$ to the timestep criterion.

After a very few computer cycles the effect of the heat flow is very pronounced. Figure 6 shows electron temperature contours plotted with the flux lines in the

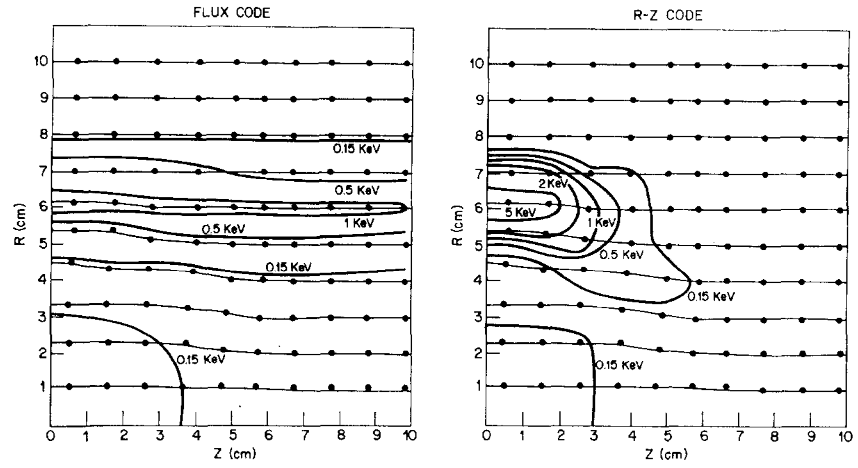

FIG. 6. Electron temperature contours in the $(r, z)$ plane. Results of the flux code and $(r, z)$ code are compared after three timesteps. The lines and dots in the background represent the flux grid (———, $T_e$ contours; –●–●–●–, Flux lines).

background. This particular run was made on a very coarse mesh (11 × 11), and yet the results are quite intelligible. Already the heat confinement displayed by the flux code is much less than in the $(r, z)$ code, with a factor of 5 difference in maximum temperature, and probably much more realistic.

On a larger mesh (21 × 21) more extensive runs have been made. The collision frequency tensor $\nu$ incorporates both classical Spitzer [18] collisions as well as effects of various beaming instabilities including the isotropic beam cyclotron [19], the cross-field modified two-stream [20], and magnetized ion–ion [21] models.

It is not the purpose of this paper to treat the theory of these instabilities nor to even discuss the numerical implementation of them; suffice it to say they determine the values of $\nu$ and $\mathbf{V}_{ph}$ to be used in Eqs. (3.2) and (3.3) which we solve here. The paper by Wagner and Anderson [22] gives a detailed treatment of the numerics regarding these instabilities, and the method adopted there is the one used here. In fact the code described there was built in concert with and is very similar to the $(r, z)$ version used here.

In Figs. 7 and 8 the flux tubes and the electron temperature contours are shown for the time 4.1 ns. Later at 5.1 ns shown in Figs. 9 and 10, the flux tubes have compressed more and show a tendency to compress greatly at the midplane. We

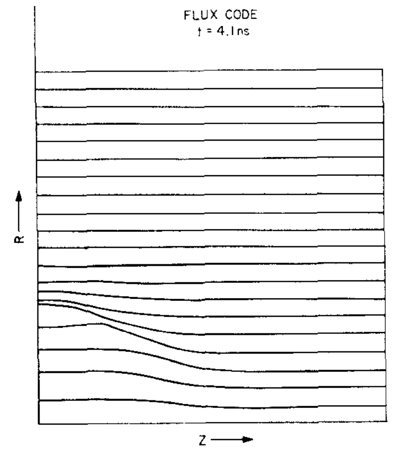

FIG. 7. Flux tubes at 4.1 ns. Distortion is mainly in the region of the expanding plasma slug.

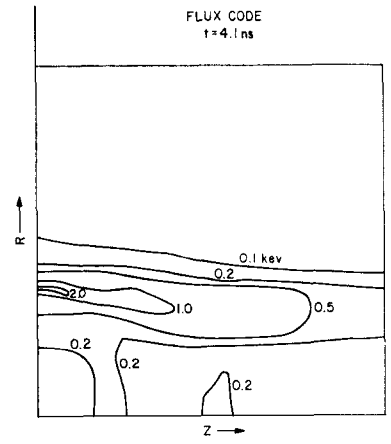

FIG. 8. Electron temperature contours from flux code at 4.1 ns.

note that the region of greatest flux density (large $B'$) is just the region of very large thermal gradients. This is expected as the perpendicular thermal conductivity goes like $1/B'$; as is well known, a large magnetic field should be a good insulator capable of supporting a large temperature gradient. Discontinuities in the $B$ gradient, evident in Figs. 7 and 9, correctly indicate the presence of a current sheet induced near the surface of the expanding spherical slug. Extensive printouts show where the various instability mechanisms turn on and to what extent they contribute to the components $\nu_{\perp}$ and $\nu_{\parallel}$ of the collision tensors. It is found that the values of

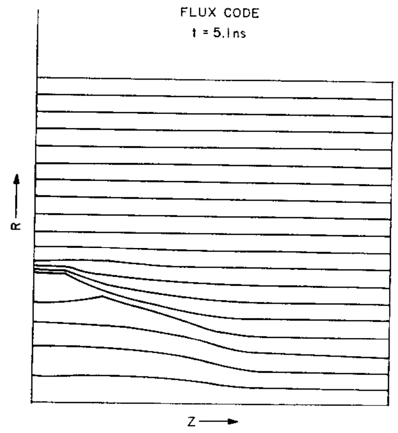

FIG. 9. Flux tubes at 5.1 ns.

these did not maximize at $z = 0$ but instead found their largest values off the midplane. It can be said that the large $| \nu |$ there gave an anomalous resistivity which prevented large gradients in $\boldsymbol{B}'$ and hence led to a magnetic diffusion of $\boldsymbol{B}'$ there. The effects of magnetic diffusion was less at $z = 0$, and hence the large compression of flux there was not diffused away as much. All of the instabilities included did operate at one time or another in some part of the domain. The magnetized ion–ion did operate at $z = 0$ predominantly but did not heat electrons nor produce any resistivity (to speak of). Hence it would not interfere with the above-mentioned large compressions at $z = 0$. Beam cyclotron instabilities did occur near $r = 0$ at finite $z$ and heated the electrons; this is evident in Fig. 10 where a 1-keV contour pokes up in the neighborhood of $r = 0$ for $z = 8$ cm. Most prevalent of the instabilities was the modified two stream which operated mostly for some $r \neq 0$. The fact that the 1.0-keV contour, which extends from $z = 0$, bulges out for $z > 0$ is attributed to this modified two-stream instability; as mentioned above, this prevented the magnetic compression one would otherwise expect with just Spitzer-like collisions.

Once the magnetic field lines become flared, the process of adiabatic cooling becomes evident. Contours which were previously open at very early times when the field was still quite straight (see the .5-keV contour in Fig. 6 are now closed within the domain due to this cooling process.

One odd feature of the flux tubes evident in Figs. 7 and 9 shows a positive radial magnetic field when normally a spherically expanding slug should carry the field with it in such a way to keep the radial field negative. The modified two-stream instability, as described above, operated predominantly off of the midplane and in the region where the positive radial field is seen. Intense heating of the electrons by

ev Tr tcc2470609(= ) Ti Tr 28.2668 0 T055 Tw (finite ) T 0.1298 Twctront

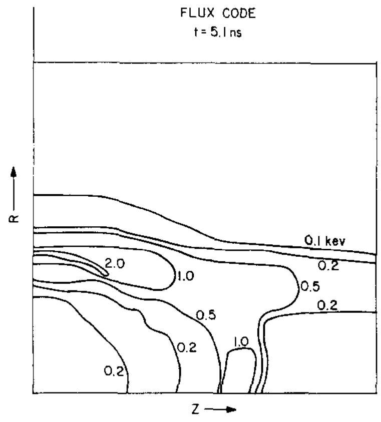

FIG. 10. Electron temperature contours from flux code at 5.1 ns.

this instability provided a large radial pressure gradient in which ablation does occur, that is, the electrons spurt out radially at a velocity larger than that of the expanding slug. This fast flow of the electron fluid carries the magnetic field along and produces the local positive radial field.

Figure 11 gives the electron temperature contours for the $(r, z)$ code when run to 5.1 ns. The choice of $\kappa_\parallel$ limited by the timestep has restricted the heat flow in the **B** direction and results in keeping the heat source region (heat is produced both by instabilities and interspecial heat conduction) quite hot; comparing Figs. 10 and 11 we see a factor of $\sim$10 between the maximum electron temperatures.

The second noteworthy difference between the two codes is evident by comparing the cross-field thermal gradients. Comparison of Fig. 11 with the flux lines shown in Fig. 9 shows that the $(r, z)$ code has pushed the region of steep thermal gradients out ahead of the region of magnetic field compression computed by the flux code. This incorrect flow of thermal energy across the flux lines is due to the effect of numerical isotropization discussed in Section II, Although not shown here, the $(r, z)$ code incorrectly pushes the region of magnetic flux compression ahead of the result given by the flux code in Fig. 9. A naive calculation of the location of the flux compression, assuming a constant radial velocity equal to the initial surface radial velocity of the pellet plasma, gives a result in close agreement with the flux code and in contradiction to the results of the $(r, z)$ code.

So for the anisotropic heat flow the $(r, z)$ code allows far too much heat to cross flux tubes and does not allow enough heat to flow along flux tubes. Now we turn to some additional benefits in the flux model which involve the timestep calculation (time resolution) and the flux compression (space resolution). For the $(r, z)$ code the hotter plasma ions give a larger magneto-acoustic speed which in turn helps deter-

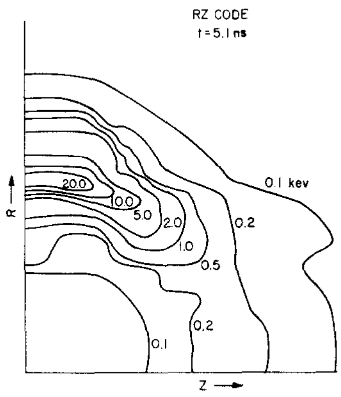

FIG. 11. Electron temperature contours from $(r, z)$ code at 5.1 ns.

mine the timestep. A modification of the Courant–Friedrichs–Lewy (CFL) timestep condition as described in Richtmeyer and Morton [23] applies here:

$$(V + C_{\mathrm{ma}})\,\delta t/\delta x \leqslant \tfrac{1}{2},$$

where $V$ is the fluid velocity and $C_{\mathrm{ma}}$ is the magneto–acoustic velocity. In short the lower temperatures and hence lower thermal speeds allow one to take bigger timesteps which means one is not required to maintain too fine a temporal resolution, and of course it means a reduction in the cost of running the code.

The spatial resolution is in large part determined by considerations of the important physical effects we wish to resolve. Usually the grid spacing should be less than the length scale associated with the physical gradients of interest. When these gradients scale like $1/B'$, then the flux coordinate mesh gives optimal resolution since the grid point density is also proportional to $1/B'$. So an initially coarser mesh may still give good resolution when and where needed. Then having fewer grid points also leads to less expense.

For a grid $(21 \times 21)$ we have found the flux code requires 30% more computer time per timestep compared to the $(r, z)$ version. This ratio approaches 1, as the number of fluids is increased but would be larger for fewer fluids. For two fluids the factor might be 100%. Reductions in running time allowed by use of coarser meshes and larger timesteps can overcome the higher expense per grid point per timestep.

One conclusion of the foregoing discussion is that even in the case of isotropic transport it may be cost effective to use the flux code at an expense comparable to using the $(r, z)$ code. But a stronger conclusion is that for anisotropic transport only enormous amounts of computer time associated with very fine grid resolution will give a physically reasonable result using the $(r, z)$ code. Only the flux code (or one like it) can give proper treatment to the anisotropic transport phenomena for reasonable or even coarse grids. So the flux code, for the anisotropic case, can run on a computer budget comparable to that required by a $(r, z)$ code and at the same time produce believable results rather than the numerical isotropisation associated with the latter.

In its present form the code we have described here has several limitations. A limitation of the present flux code is that even for equations in conservation form the quantities are conserved not to roundoff but only to a truncation error level. This is inherent in the changing metric from step to step.

Formally one can remove this small nonconservation by placing an appropriate constraint on the derivation of $\partial\beta/\partial t$. Further if the appropriate difference formulas are used to derive $\partial\beta/\partial t$, we also still get the correct treatment of the force free uniform fluids. It should be noted that $\partial\beta/\partial t$ will then be slightly different for each fluid equation solved because it will depend on the $f$'s. In a future version of the code we contemplate using such $\partial\beta/\partial t$ terms that will lead to exact conservation. For the $(r, z)$ code with time-independent metric there is no difficulty and conserves quantities (posed in conservation form) down to roundoff. Several other limitations were built in the flux code to make the tasks involved more manageable. Assumptions such as open flux tubes, scalar pressure, first-order Lagrangian grid transport, midplane symmetry and an ignorable $\theta$ coordinate, are seen to be surmountable in a numerical model more complex than this one but involving the same basic methods. It is not clear whether both the favorable features of FCT and of a completely implicit treatment could be combined to build a code capable of operating over very large timescales. What is feasible is to use mixed implicit–explicit methods to isolate the fast phenomena in the implicit steps but even here the relative motion of the fluids to the moving grid will always require explicit treatment in the SLIDE algorithm.

Many advantages have been discussed above and include accurate representation of plasma fluids with $\kappa_{\parallel} \gg \kappa_{\perp}$ (also for tensor $\nu$), no restriction on high-beta plasma flows, and perhaps an optimal packing of grid points. Some of the very desirable features of the FCT algorithms, especially preservation of physically positive quantities and minimal numerical diffusion, are unique here and distinguish this method from other quasi-Lagrangian methods mentioned in Section I.

On the matter of not getting conservation errors to roundoff level we must note that here in the flux code $\nabla \cdot \mathbf{B} \equiv 0$ exactly, merely by the assumption of flux tube coordinates. In other words we conserve the number of magnetic monopoles to be identically zero. This is not the case in the $(r, z)$ code nor in some other MHD codes where $\nabla \cdot \mathbf{B} \neq 0$ and where $\nabla \cdot \mathbf{B}$ can grow to unacceptable magnitudes unless ad hoc numerical suppression or damping techniques are introduced to force $\nabla \cdot \mathbf{B}$ back near zero.

In the field of plasma physics several problems of interest (with dynamic plasma flows) occur in large magnetic fields where one nearly always finds a large anisotropy in the heat conduction. Studies of energy confinement and plasma confinement as well may require numerical models like the one described here. It may be possible, for example, to represent the time history of a wetwood burner [24, 25] fusion scheme (either tokamak or mirror configurations) with a multifluid model like this one.

## APPENDIX A: Review of SHASTA Algorithm

The Boris–Book [12] SHASTA flux corrected transport (FCT) fluid solver for a 1D-Cartesian problem is briefly reviewed here. We review it here because we wish to emphasize its importance in the derivation of the SLIDE method. The SHASTA FCT algorithms were developed to overcome certain nasty features of several popular fluid algorithms including Lax–Wendroff, Leap-frog, and upstream–downstream where steep gradients could lead to numerical dispersion and Gibbs-like phenomena. Associated with these effects was the possibility of producing negative values for positive definite physical quantities. A well-known method of adding artificial viscosity near steep gradients [22] does overcome the above physical defects but adds too much unphysical numerical diffusion. The SHASTA algorithm also adds numerical diffusion but only enough to prevent the Gibbs-like phenomena.

Since all of the fluid equations resemble the continuity equation if the source terms are neglected we shall review just 1D-Cartesian continuity equation of the form.

$$(\partial\rho/\partial t) + (\partial/\partial x)(V\rho) = 0, \tag{A.1}$$

where $\rho$ is the density and $V$ the velocity of the fluid. This is a special case of a more general Cartesian fluid equation with source terms

$$(\partial f/\partial t) + (\partial/\partial x)(Vf) = (\partial/\partial x)(g) + d(\partial a/\partial x) + e. \tag{A.2}$$

The SHASTA algorithm has three important steps in the course of advancing the density values one timestep.

*Step* 1. Lagrangian conservative transport of the trapezoidal representation. Refering to Fig. 12 the solid line represents the density profile at $t = 0$. The transported trapezoids at $t = 0 + \delta t$ have been convected such that the cell boundaries are displaced $x_i' = x_i + v_i\delta t$ and are kept conservative (of constant area) by assigning the new density $\rho_i = \rho_i R_i$ where $R_i$ is the ratio of the old trapezoid base to the new trapezoid base. The dashed line in Fig. 12 represents the transported trapezoids. Source terms, if any, are added in here.

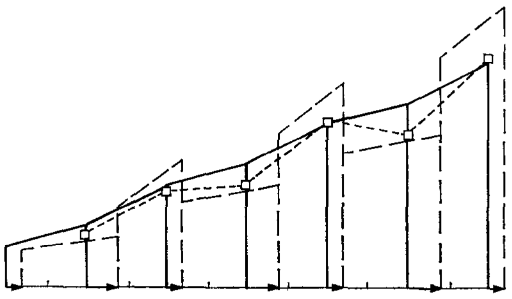

FIG. 12. Trapezoidal representation of density for the continuity equation by the SHASTA algorithm. The solid line shows the trapezoids of fluid at $t = t_0$. The convected trapezoids at $t = t_0 + \delta t$ are drawn with the dashed lines. The dotted line gives the result of area weighted interpolation and represents the solution prior to the antidiffusion procedure. The indicated center lines show grid cell center locations to be used in area weighting (———, $t = 0$ trapezoids; – – –, $t = \Delta t$ transported trapezoids; □– – – – –□, $t = \Delta t$ area weight interpolated result; →, displacement of fluid due to convection).

*Step* 2. The density representation given by the dashed line in Fig. 12 is now interpolated by area weighting back onto the original grid locations. This interpolation conserves the mass. The result of this interpolation is given by the dotted line in Fig. 12.

*Step* 3. The interpolation of Step 2. is extremely numerically diffusive and most of it is removed by a process of anitdiffusion. The antidiffusion is performed by determining a set of fluxes whose divergence is the numerical diffusion of Step 2. These fluxes are reduced in magnitude where ever the antidiffusion process would produce new maxima or minima not found in the result of Step 2.

## APPENDIX B: SHASTZ

Many details of the SHASTA and SHASTZ algorithms are omitted here and the reader is refered to the work of Boris and Book [12] and of Boris and Gardner [11], respectively, for further information.

The important feature of SHASTA which is relevant to the problem of continuous rezoning is Step 2. The interpolation by area weight need not be performed on the original grid but may be performed on a displaced grid. Again an antidiffusion step is well defined and can be used to eliminate all but a residual amount of numerical diffusion. If the grid displacement moves with the fluid in the Lagrangian sense, then the numerical diffusion of area weighting vanishes. Sometimes the grid is displaced in the same direction as the fluid motion but not as far. Then the numerical diffusion of Step 2 is less than that in the fixed Eulerian grid case. Only if the grid speed relative to the fluid exceeds the Eulerian speed of the fluid will the numerical diffusion increase. Hence a remarkable feature of the SHASTA rezone becomes apparent, that is, if the observer grid tends to move with the fluid, the numerical diffusion is reduced and is less than solving the problem without rezoning. It should also be clear that no separate rezoning algorithm is used since Step 2 performs it. The version of SHASTA which is able to rezone at every time step (continuous rezoning) is SHASTZ and it solves a Cartesian-1D form of the fluid equations. It was set up to solve the representation given in Eq. (2.2).

Other continuously rezoned algorithms, such as that used by Hirt, Amsden, and Cook [10], also reduce numerical diffusion but not as much as that obtained with the SHASTZ algorithm. Of course, purely Lagrangian algorithms may be used but then the grid moves exactly with the fluid quantity. Yet even in the case of a one-fluid problem, the energy will in general tend to propagate at a different speed than the density so the problem will not be solvable by a Lagrangian algorithm for all quantities, and those quantities which move relative to the Lagrangian grid will be diffused numerically by the algorithm. In the case of a multiple-fluid model it should be clear that most of the fluids will have motion relative to any moving grid. SHASTZ is a Cartesian algorithm which may be used as purely Lagrangian, purely Eulerian, or may use an arbitrarily moving observer grid (which may be chosen optimally to simplify the physics). For fluid problems and particularly multi-fluid problems with several distinct convection speeds the SHASTZ FCT algorithm is well suited to solve all the fluid equations from a single observer grid and give results which minimize numerical diffusion and remedy the Gibbs-like phenomena.

Of course the major restriction is that SHASTZ requires a Cartesian coordinate system. To get around this requirement we show how to transform the fluid equations from moving orthogonal coordinate systems into a formally Cartesian system; Appendix C will show how this is done.

## APPENDIX C: THE TRANSFORMATION ALGORITHM SLIDE

Our goal is to solve fluid or fluid-like equations on arbitrary orthogonal coordinates where the grid may move arbitrarily (but less than one half a grid spacing per time step) so the metric may change with time. We can cast a typical fluid equation as

$$\begin{aligned}\frac{\partial f}{\partial t} + \frac{1}{h_1h_2h_3}\left[\frac{\partial}{\partial q_1}(fV_1h_2h_3) + \frac{\partial}{\partial q_2}(fV_2h_3h_1) + \frac{\partial}{\partial q_3}(fV_3h_1h_2)\right] \\ = \frac{1}{h_1h_2h_3}\left[\frac{\partial}{\partial q_1}(g_1h_2h_3) + \frac{\partial}{\partial q_2}(g_2h_3h_1) + \frac{\partial}{\partial q_3}(g_3h_1h_2)\right] \\ + \frac{d_1}{h_1}\frac{\partial a}{\partial q_1} + \frac{d_2}{h_2}\frac{\partial a}{\partial q_2} + \frac{d_3}{h_3}\frac{\partial a}{\partial q_3} + e,\end{aligned} \tag{C.1}$$

where $f$ is the fluid quantity, the $q$'s and $h$'s are the coordinates and metric factors, the $V$'s the convection velocity, and the other factors ($a$, $d$, $e$, and $g$) are arbitrary source functions. If the right-hand side is zero, then the usual conservation form (continuity equation)

$$(\partial f/\partial t) + \nabla \cdot (f\mathbf{V}) = 0$$

is obtained. The right-hand side of Eq. (C.1) is written to contain an arbitrary divergence, the scalar product of a vector and a gradient, and finally a scalar source term. Almost any conceivable fluid equation can be cast in this form and the algorithm we shall describe solves this general form. Operators such as $\nabla^2$ are either represented explicitly as $\nabla \cdot \nabla$ in this formalism or they are frequently solved implicitly on a separate split step.

It is often customary, and we adopt this practice, to solve Eq. (C.1) by the method of time-step splitting; that is, we successively update $f$ by solving the sequence of the following three equations.

$$\frac{\partial f}{\partial t} + \frac{1}{h_1h_2h_3}\frac{\partial}{\partial q_1}(fV_1h_2h_3) = \frac{1}{h_1h_2h_3}\frac{\partial}{\partial q_1}(g_1h_2h_3) + \frac{d_1}{h_1}\frac{\partial a}{\partial q_1} + \frac{e}{3}, \tag{C.2a}$$

$$\frac{\partial f}{\partial t} + \frac{1}{h_1h_2h_3}\frac{\partial}{\partial q_2}(fV_2h_3h_1) = \frac{1}{h_1h_2h_3}\frac{\partial}{\partial q_2}(g_2h_3h_1) + \frac{d_2}{h_2}\frac{\partial}{\partial q_2} + \frac{e}{3}, \tag{C.2b}$$

$$\frac{\partial f}{\partial t} + \frac{1}{h_1h_2h_3}\frac{\partial}{\partial q_3}(fV_3h_1h_2) = \frac{1}{h_1h_2h_3}\frac{\partial}{\partial q_3}(g_3h_1h_2) + \frac{d_3}{h_3}\frac{\partial}{\partial q_3} + \frac{e}{3}, \tag{C.2c}$$

The factor of $\frac{1}{3}$ on the $e$ term accounts for the fact that this symmetric term appears three times in Eqs. (C.2). If we were solving a 2D problem by the method of time-step splitting, this factor would be $\frac{1}{2}$ instead. The set of equations in (C.2) are more easily solved than Eq. (C.1) as they are each 1D equations in their respective orthogonal coordinate. All of the equations in (C.2) are of the same form, and if we let $\beta = h_1h_2h_3$ and $\alpha = h_1$, $= h_2$, or $= h_3$, then any one of them can be written

$$\frac{\partial f}{\partial t} + \frac{1}{\beta}\frac{\partial}{\partial q}\left(\frac{\beta f V}{\alpha}\right) = \frac{1}{\beta}\frac{\partial}{\partial q}\left(\frac{\beta g}{\alpha}\right) + \frac{d}{\alpha}\frac{\partial a}{\partial q} + \frac{e}{3}. \tag{C.3}$$

Now apply the transformation

$$\begin{aligned} A &= a, \\ F &= f\beta, \\ U &= V/\alpha, \\ D &= d\beta/\alpha, \\ G &= g\beta/\alpha, \\ E &= (e\beta/3) + f(\partial\beta/\partial t), \end{aligned} \tag{C.4}$$

which must be nonsingular. So our split fluid equation becomes

$$(\partial F/\partial t) + (\partial/\partial q)(UF) = (\partial G/\partial q) + D(\partial A/\partial q) + E, \tag{C.5}$$

which is the standard Cartesian form of the fluid equation solvable by the SHASTZ algorithm. This is evident by comparing Eq. (C.5) with Eq. (C.2). Even when the transformation becomes singular or when its inverse becomes singular at a point, the method may be used at the other points while giving special treatment to the singular points. Cylindrical coordinates, for example, display such a point at $r = 0$ and in the next section we show how the singularity is "removed" by use of a higher-order algorithm.

## APPENDIX D: Computation of $\partial\beta/\partial t$

In (C.4) the transformations are straightforward but the $E$ transformation deserves comment. A time-dependent metric factor $\beta$ adds an extra term. The value of this term must be known prior to the fluid timestep as there is no simple way to compute it simultaneously with the fluid algorithm. Since the SHASTZ algorithm requires knowledge of the grid locations both at $t$ and at $t + \delta t$, it is clear that $\beta$ will be known both at the old and new times before performance of the fluid step. Since $\beta$ represents the Jacobian of the $q$ coordinates relative to Cartesian ones, its time derivative represents the rate at which the volume of the grid cells expand or [illegible]

physically when it should keep constant density. Inclusion of the $\partial\beta/\partial t$ term prevents this unphysical effect and the force free fluid remains intact keeping its constant density. One may employ several approximations to compute $\partial\beta/\partial t$. One could write

$$\partial\beta/\partial t = (d\beta/dt) - \mathbf{V}_{\text{grid}} \cdot \nabla\beta, \tag{D.1}$$

and then since we know the total derivative $d\beta/dt$ from the strict time difference at the moving grid point, we need only approximate $\mathbf{V}_{\text{grid}} \cdot \nabla\beta$ to compute $\partial\beta/\partial t$. It is this approximation which is difficult. How should one numerically do the interpolations required in evaluating $\nabla\beta$ in order to compute $\partial\beta/\partial t$ from (D.1)?

Instead of using (D.1) to compute it, we use a very important property of the numerical representation of $\partial\beta/\partial t$ to compute it. That is, as mentioned above, the numerical algorithm and the motion of the grid together with the accompanying change in the metric should have no effect on a uniform force free fluid.

So we employ a force free fluid of unit density and solve its continuity equation with the SLIDE algorithm. If primed quantities represent the solution after one timestep and unprimed the initial quantities and if lower case and upper case $f$'s represent the density and transformed density, respectively, then we can pose the problem as follows. Reference to the transformation Eqs. (C.4) shows that if we require our algorithm to leave $f$ invariant, then

$$\begin{aligned} f &= 1, \qquad F = \beta, \\ f' &= 1, \qquad F' = \beta', \end{aligned} \tag{D.2}$$

must be satisfied. Let the operator $L'$ represent the SHASTZ operation ($F' = L'(F)$) which in this case will be more general than the Cartesian continuity equation due to the term

$$\delta t E = \delta t (\partial\beta/\partial t) f = (\partial\beta/\partial t)\,\delta t. \tag{D.3}$$

Further let the operator $L^*$ and $F^*$ represent the SHASTZ operator with $E = 0$ and the result of that operation, respectively; that is,

$$\begin{aligned} F^* &= L^*(F), \\ \beta^* &= L^*(\beta) = L^*(F). \end{aligned} \tag{D.4}$$

From Eqs. (D.3) and (D.4) we may write

$$F' = L'(F) = L^*(F) + (\partial\beta/\partial t) f\,\delta t,$$

or

$$\beta' = \beta^* + (\partial\beta/\partial t)\,\delta t. \tag{D.5}$$

If we solve this for $\partial\beta/\partial t$, we obtain

$$\partial\beta/\partial t = (\beta' - \beta^*)/\delta t. \tag{D.6}$$

This gives us a prescription for $\partial\beta/\partial t$ which will have the advantageous numerical property of no interaction of the grid with any force free fluids (down to the round-off error of $\partial\beta/\partial t$).

## APPENDIX E: EXAMPLES OF COORDINATE SYSTEMS

We first present the transformations and metrics for several familiar coordinate systems with time-independent metric. Then we present an example from MHD theory where the coordinates are defined by magnetic flux surfaces and in which the metric varies in time. In the examples given for 3D problems it should be noted that the existence of one or two ignorable coordinates allows reduction of the problem to 2D or 1D, and the transformations shown are still used on the remaining coordinate lines. We note again that the $\frac{1}{3}$ factor in Eq. (C.4) becomes $\frac{1}{2}$ or 1 when the problem is reduced from 3D to 2D or 1D, respectively.

*Cylindrical*

Here the coordinates and metric factors are

$$\begin{aligned} r &= q_1, & h_1 &= 1, \\ \theta &= q_2, & h_2 &= r, \\ z &= q_3, & h_3 &= 1. \end{aligned} \tag{E.1}$$

A time-split method for a 3D problem would have a radial, an azimuthal, and an axial step. At the point $r = 0$ the inverse transformation of Eq. (C.4) is singular and this point is excluded from the domain of the inverse transformation. For the case of azimuthal symmetry ($\theta$ an ignorable coordinate) the point $r = 0$ causes trouble only on the radial step. At this singular point two basic methods have been used to supply a value for $f$. The simplest way is to prescribe a boundary condition usually $f = 0$ or $\partial f/\partial r = 0$. Alternatively a straightforward differencing of the radial part of the fluid equation can be used to supply a value for $f$. One choice is

$$\begin{aligned} f'(0) = f(0) &- 2\delta t\, f(\delta r)\, v(\delta r)/\delta r + \delta t\, d(0)(a(\delta r) + a(0))/\delta r + 2\delta t\, g(\delta r)/\delta r \\ &+ \delta t\, e(0), \end{aligned} \tag{E.2}$$

where $\delta r = r_2$, $0 = r_1$, i.e., $\delta r = r_2 - r_1$.
Knowing the value of $f$ at the singular point saves one from applying the inverse transformation at the singular point and the accompanying undefined result which is nonsense.

*Spherical*

Now the coordinates and metric factors are

$$\begin{aligned} r &= q_1, & h_1 &= 1, \\ \theta &= q_2, & h_2 &= r, \\ \phi &= q_3, & h_3 &= r \sin\theta. \end{aligned} \tag{E.3}$$

Solving a $3D$-fluid problem in spherical coordinates would have a radial, a polar, and azimuthal step. Again certain points where $r = 0$, $\theta = 0$, or $\theta = \pi$ must be excluded from the transformation but one can again patch these values either by use of boundary conditions or by the implementation of a higher-order algorithm at the singular points much in the manner described for the radial cylindrical case.

## *Other Standard Coordinate Systems*

The 13 standard orthogonal coordinate systems described by Morse and Feshbach [26] may also be used to solve problems by this method. They all are stationary systems, so $\partial\beta/\partial t = 0$. The general definitions given in Appendix C for $\alpha$ and $\beta$ allow one to implement any of these systems in a manner analagous to the cylindrical and spherical cases shown above.

## *Axisymmetric Magnetic Flux*

Here $\theta$ is an ignorable coordinate, so we have a 2D system in which the time splitting is performed. The coordinates together with the metric factors are

$$\begin{aligned} \Psi &= q_1, & h_\Psi &= h_1 = 1/rB', \\ \theta &= q_2(\text{ignorable}), & h_\theta &= h_2 = r, \\ \chi &= q_3, & h_\chi &= h_3 \equiv ((\partial r/\partial\chi)^2 + (\partial z/\partial\chi)^2)^{1/2}, \end{aligned} \tag{E.4}$$

where $B' = (B_r{}^2 + B_z{}^2)^{1/2}$ with $r$, $z$ the cylindrical coordinates. The $\Psi$ values are found by the usual definition for the flux function

$$\Psi = rA_\theta,$$

where $A_\theta$ is the theta component of the magnetic vector potential. Since $\mathbf{B} = \nabla \times \mathbf{A}$, we can write

$$\begin{aligned} rB_r &= -\partial\Psi/\partial z, \\ rB_z &= \partial\Psi/\partial r. \end{aligned} \tag{E.6}$$

The grid (i.e., the point $\Psi_i$, $\chi_i$) moves in $(r, z)$ space with the $c(\mathbf{E} \times \mathbf{B}')/B'^2$ drift velocity. Clearly if $B'$ is time dependent, then

$$\partial\beta/\partial t = (\partial/\partial t)(h_\Psi h_\theta h_\chi) = (\partial/\partial t)(h_\chi/B') \neq 0. \tag{E.7}$$

In addition to the SLIDE algorithm special cases of it relating to cylindrical radial and spherical radial grids have been written to include the "removal" of the singularities and other features useful in these special coordinates. Subroutines by the names SLIDER and SLIDES, respectively, are available to perform the numerics in those systems. The more general SLIDE algorithm, it should be noted, does not include any remedy for singular points, and any prospective users are warned they must provide the patch work to fix any singular points.

Many fluid algorithms claim conservative treatment for equations in conservation form. For some the conservation property fails if the mesh is not spatially uniform. Others fail when the results are rezoned. Some such as the stretched coordinate algorithms [27] are conservative in a transformed coordinate system but the transformation itself is not conservative. In some sense the SLIDE algorithm is of the stretched coordinate variety, except instead of transforming the coordinate values we have transformed the dependent variables, the physical quantities. The SLIDE algorithm in its present form is not conservative exactly, although the conservation error should be kept small. We have derived a different version of SLIDE using a modified value for $\partial\beta/\partial t$ such that exact conservation is recovered. It has not been treated yet. In many special cases the SLIDE algorithm is conservative. For example, both the SLIDER and SLIDES versions of it are conservative even with arbitrary rezoning. It is the changing metric which changes the conservation errors from roundoff levels to truncation levels; hence we find adjustment of the $\partial\beta/\partial t$ term will remedy this nonconservation in future versions of the code.

## Acknowledgments

We thank David Book, Timothy Coffey, and Niels Winsor for their encouragement and impetus they lent to the project reported on here; without their aid and optimism it is unlikely the work would have been attempted. Jay Boris, David Book, and John Gardner deserve special thanks for their pioneering work on Flux Corrected Transport and the SHASTA algorithms; these are fundamental to the results presented here. To Carl Wagner and Robert Clark we are indebted for many discussions of the relevant numerical representations of the many plasma physics effects included. This work was supported by the Defense Nuclear Agency.

## References

1. K. Hain, Numerical Solution for 1.5-dimensional, time dependent magnetohydrodynamic problems, *in* "Proc. of Symposium on Computer Simulation of Plasmas and Many-Body Problems," Cosponsored by NASA Langley Research Center and The College of William and Mary, held April 19–21, 1967, at The College of William and Mary, Williamsburg, VA.
2. F. Hertweck and W. Schneider, A two-dimensional computer program for end losses from a theta-pinch, *in* "Second European Conference on Controlled Fusion and Plasma Physics," Stockholm, 1967.
3. K. V. Roberts and D. E. Potter, Magnetohydrodynamic calculations, *in* "Methods in Computational Physics," Vol. 9, p. 339, Academic Press, New York, 1970.
4. I. Lindemuth and J. Killeen, *J. Computational Phys.* **13** (1973), 181.
5. C. K. Chu and G. Johansson, Numerical studies of the heat conduction equation with highly anisotropic tensor conductivity. II, Uppsala University, Dept. of Computer Sciences, Report No. 40, 1972.
6. C. K. Chu, K. W. Morton, and K. V. Roberts, Numerical studies of the heat conduction equation with highly anisotropic tensor conductivity, I, *in* "Proceedings of 3rd International Symposium on Numerical Fluid Dynamics (Paris 1972)," Springer–Verlag, Berlin/New York, 1973.
7. J. Ll. Morris and I. F. Nichol, *J. Computational Phys.* **13** (1973), 316.

8. R. C. Grimm and J. L. Johnson, Axisymmetric toroidal fluid simulations using natural coordinates, *in* "Proceedings of the 6th Conference on Numerical Simulation of Plasmas (Berkeley 1973)," Sponsored by Lawrence Livermore Laboratory, Conference Report No. 730804, 1973.
9. J. U. Brackbill and W. E. Pracht, *J. Computational Phys.* **13** (1973), 455.
10. C. W. Hirt, A. A. Amsden, and J. L. Cook, *J. Computational Phys.* **14** (1974), 227.
11. J. P. Boris and J. H. Gardner, "SHASTZ: A variable zone, Lagrangian and semi-Lagrangian flux corrected transport algorithm," Naval Research Laboratory Memorandum Report, to appear.
12. J. P. Boris and D. L. Book, *J. Computational Phys.* **11** (1973), 38.
13. D. E. Potter and G. H. Tuttle, *J. Computational Phys.* **13** (1973), 483.
14. D. V. Anderson and R. E. Clark, "ORTHOL: An explicit grid orthogonalizer for pseudo-Lagrangian numerical schemes," Naval Research Laboratory Memorandum Report No. 2849, 1974.
15. D. V. Anderson, "A Continuously rezonable fluid solver in moving orthogonal coordinates with time dependent metric," Naval Research Laboratory Memorandum Report No. 2769, 1974.
16. W. M. Manheimer, unpublished notes, Naval Research Laboratory, Washington, DC.
17. S. I. Braginskii, Transport processes in a plasma, *in* "Reviews of Plasma Physics," Vol. 1, p. 205, Consultants Bureau Enterprises, New York, 1965.
18. L. Spitzer, Jr., "Physics of Fully Ionized Gases," 2nd ed., Wiley (Interscience), New York, 1962.
19. M. Lampe, W. M. Manheimer, J. B. McBride, J. H. Orens, K. Papadopoulos, R. Shanny, and R. N. Sudan, *Phys. Fluids* **15** (1972), 662.
20. J. B. McBride, E. Ott, J. P. Boris, and J. H. Orens, *Phys. Fluids* **15** (1972), 2367.
21. K. Papadopoulos, R. C. Davidson, J. M. Dawson, I. Haber, D. A. Hammer, N. A. Krall, and R. Shanny, *Phys. Fluids* **14** (1971), 849.
22. C. E. Wagner and D. V. Anderson, Naval Research Laboratory Memorandum Report, to appear.
23. R. D. Richtmyer and K. W. Morton, "Difference Methods for Initial-Value Problems," 2nd ed., Wiley (Interscience), New York, 1967.
24. J. M. Dawson, H. P. Furth, and F. H. Tenney, *Phys. Rev. Lett.* **26** (1971), 1156.
25. R. F. Post, T. K. Fowler, J. Killeen, and A. A. Mirin, *Phys. Rev. Lett.* **31** (1973), 280.
26. P. M. Morse and H. Feshbach, "Method of Theoretical Physics," Vol. 1., p. 655, McGraw-Hill, New York, 1953.
27. E. K. deRivas, *J. Computational Phys.* **10** (1972), 202.