UCRL-LR-125050

Field Simulation of Axisymmetric Plasma Screw Pinches By Alternating-Direction-Implicit Methods

Michael Allen Lambert

PhD Thesis

June 1996

Lawrence Livermore National Laboratory

Work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract W-7405-ENG-48.

UCRL-LR-125050
Distribution Category UC-705

Field Simulation of Axisymmetric Plasma Screw Pinches By Alternating-Direction-Implicit Methods

Michael Allen Lambert

Doctor of Philosophy
Thesis

Manuscript date: June 1996

HH

LAWRENCE LIVERMORE NATIONAL LABORATORY
University of California • Livermore, California • 94551

Field Simulation of Axisymmetric Plasma Screw Pinches By Alternating-Direction-Implicit Methods

By

MICHAEL ALLEN LAMBERT

B.S. (Oregon State University) 1989

M.S. (University of California, Davis) 1991

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ENGINEERING - APPLIED SCIENCE

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Garry Rodrigue

[illegible]

Dennis W. Hewett

Committee in Charge

1996

Michael Allen Lambert

June 1996

Engineering-Applied Science

Field Simulation of Axisymmetric Plasma Screw Pinches

By Alternating-Direction-Implicit Methods

Abstract

An axisymmetric plasma screw pinch is an axisymmetric column of ionized gaseous plasma radially confined by forces from axial and azimuthal currents driven in the plasma and its surroundings. This dissertation is a contribution to detailed, high resolution computer simulation of dynamic plasma screw pinches in 2-d *rz*-coordinates.

The simulation algorithm combines electron fluid and particle-in-cell (PIC) ion models to represent the plasma in a hybrid fashion. The plasma is assumed to be quasineutral; along with the Darwin approximation to the Maxwell equations, this implies application of Ampere's law without displacement current. Electron inertia is assumed negligible so that advective terms in the electron momentum equation are ignored. Electrons and ions have seperate scalar temperatures, and a scalar plasma electrical resistivity is assumed. Alternating-direction-implicit (ADI) methods are used to advance the electron fluid drift velocity and the magnetic fields in the simulation. The ADI methods allow time steps larger than allowed by explicit methods. Spatial regions where vacuum field equations have validity are determined by a cutoff density that invokes the quasineutral vacuum Maxwell equations (Darwin approximation).

In this dissertation, the algorithm was first checked against ideal MHD stability theory, and agreement was nicely demonstrated. However, such agreement is not a new contribution to the research field. Contributions to the research field include new treatments of the fields in vacuum regions of the pinch simulation. The new field treatments fit within the framework of ADI and predict a level of magnetohydrodynamic turbulence near the bulk plasma surface that is higher than predicted by other methods, perhaps leading to more accurate calculation of turbulence. New treatments were motivated by inaccuracy of previous vacuum field mathematical models.

For higher resolution, pinch simulations can be implemented on larger parallel computers. Therefore, steps were taken toward parallel implementation of the simulation. First, a parallel ADI method was developed for solution of the steady-state diffusion equation; good parallel scalability was demonstrated. The parallel ADI can be used for a variety of physical simulations, including those of screw pinches. Finally, a parallel pinch simulation algorithm was outlined.

Thesis adviser: Garry H. Rodrigue

Contents

List of Figures

List of Tables

Glossary of Terms

ADI method alternating-direction-implicit method

bulk plasma region any group of points on the simulation grid for which the ion density after advance of ion quantities is above a cutoff density, ρ_{cutoff}. Quasineutral zero-electron-inertia electron fluid-field equations are used to calculate electron and field quantities in such regions.

DADI method dynamic alternating-direction-implicit method

DSCG method diagonall scaled conjugate gradient method

DSDADI method diagonally scaled alternating-direction-implicit method

EMPRE estimated maximum pointwise relative error

MHD magnetohydrodynamic(s)

model problem solution of the Laplace equation on the unit square with Dirichlet zero boundary conditions

N-form arrangement of a system of linear equations such that the coefficients form an image of the letter N

NN communication nearest-neighbor communication

PIC ions particle-in-cell ions

PR ADI method Peaceman-Rachford ADI method

relaxation parameter name for $\omega^n \equiv 2 / \Delta t^n$

Robbins boundary condition linear combination of Dirichlet and Neumann boudary conditions at a boundary point

RN residual norm

SDD spatial domain decomposition

SR pair send-receive pair

screw pinch plasma pinch that in cylindrical coordinates, (r, θ, z), has significant current flow in the azimuthal or θ-direction and in the axial, or z- or zeta-direction

theta pinch plasma pinch that in cylindrical coordinates, (r, θ, z), has significant current flow only in the azimuthal or θ-direction

vacuum region any group of points in the simulation grid for which the ion density after advance of ion quantities is less than a cutoff density, ρ_{cutoff}. Vacuum field equations are used to calculate fields in such regions.

Z-pinch zeta-pinch; plasma pinch that in cylindrical coordinates, (r, θ, z), has significant current flow only in the axial, or z- or zeta-direction

Acknowledgements

I wish to thank Dennis Hewett for supporting my doctoral research. He offers the freedom necessary for the research to be enjoyable. He is one of the few who tolerates my obsessive habits in scientific problem solving. He likes to share humorous science stories, and is serious only when necessary. In particular, I would like to thank him for permission to copy figures from his 1980 paper (Figures 2.15 and 2.19 of this dissertation).

I would like to thank Garry Rodrigue for the time spent in his office pondering tridiagonal systems of equations and the convergence of ADI methods. It was he and Tim Williams who planted the creative seeds for the tridiagonal solution techniques that were implemented in parallel DADI.

I would like to thank John DeGroot for his lectures on plasma physics in 1990 and 1991. He removed my fears of the subject and his enthusiasm was contagious.

I would like to thank William Bateson for his experience at setting up personal computers on which the pinch simulations were performed. I would like to thank NERSC and LCC personnel who have supported my efforts in parallel computations.

Last, but not least, I would like to thank my family for support during my trying journey through graduate school. My family does not envy me, but they would be very happy for something positive to come out of all of the delayed gratification.

Preface

In this dissertation, the words "pinch simulation" usually refer to the algorithm and associated FORTRAN computer code written and developed for the research in this dissertation. Although the pinch simulation is based on earlier work by Hewett [1980], it is unique in several respects.

The dissertation is organized so as to present the most physical aspects of the pinch simulation first. In the first two chapters, the mathematical models of the simulation are presented, including new contributions by this dissertation. Then the details of the ADI method underlying the simulation, including an extension to parallel computation, are covered in the last two chapters. In this way, the dissertation progresses from the viewpoint of an applied mathematical physicist to that of a computer programmer of high-performance numerical methods.

Two unit systems are used for the formulas in this dissertation, SI units and Gaussian units. All of Chapter 1 expresses the mathematical models of the pinch simulation algorithm in Gaussian units. In Chapter 2, SI units are used in development of ideal MHD stability theory, but when actual simulation parameters are tabled there, they are expressed in Gaussian units. An exception to the use of SI and Gaussian units involves thermal particle energies kT in eV (k is suppressed in formulas).

None of the appendices are recommended reading for a first sitting. They are support material for statements and comments made in the chapter bodies, and as such they are filled with derivations that are more detailed than necessary to understand the mesage of the dissertation. Mainly, that message is: "Here is a hybrid simulation method to which some significant improvements have been made. The simulation uses a stable ADI method that is general and simple to implement on most computational platforms, including parallel computers."

Chapter 1
Mathematical Models of the Screw Pinch Simulation

1.1 Background on screw pinch simulation

An axisymmetric plasma screw pinch is an axisymmetric column of ionized gaseous plasma that is radially confined by forces from axial and azimuthal currents that are driven in the plasma and its surroundings. Pinches with just axial currents, now called zeta-pinches (Z-pinches), have an early history associated with researchers R.W. Wood, R.A. Millikan, and W.H. Bennett. Wood reported properties of a high-current vacuum spark Z-pinch and possible uses of it [Wood 1897]. R.A. Millikan used the vacuum sparks for spectroscopic studies of light elements after World War I [Millikan 1918, 1924]. Later, in a landmark paper, Bennett derived general relations describing equilibria of high-electrical current streams in plasma in order to understand cathode cratering in vacuum tubes [Bennett 1934]; he derived an equilibrium Z-pinch radial density profile and critical current that is often cited in the literature. All of these plasma pinches consisted of a plasma column along which electrical current was flowing longitudinally so as to confine the plasma in directions perpendicular to the column.

Early high temperature plasma pinch experiments were performed in the 1950s [Kurchatov 1956; Andrianov *et al.* 1958; Curran *et al.* 1958; Tuck 1958]. In these experiments, high currents were intentionally driven through a plasma column in an attempt to compress and heat it; there was much enthusiasm surrounding possible thermonuclear applications. Unfortunately, photographs taken in a few experiments indicated that pinch performance was likely to be limited by plasma surface instabilities [Kurchatov 1956; Curzon *et al.* 1960].

For the most part, analysis of the instabilities began in the 1950s, [Lundquist 1951; Chandresekhar 1952, 1953; Kruskal and Schwarzschild 1954; Rosenbluth and Longmire 1957; Tayler 1957]. A general magnetohydrodynamic (MHD) stability analysis assuming an ideal Ohm's law was put forth late in the decade [Hain and Lüst 1960]; in Ohm's law, infinite conductivity was assumed, the Hall effect was neglected, and effects of electron pressure were ignored. Hain and Lüst reduced the stability analysis to the solution of an eigenmode equation. Energy formulations of stability analysis became widely known in the mid 1960s, with names such as V.D. Shafranov, B.B. Kadomtsev, and W.A. Newcomb permeating the related literature. Stability studies have become increasingly detailed since then. Effects of finite conductivity, the Hall effect, nonlinearities, vacuum field regions, and radiation effects have now been considered. Analytically, most of the effects have been considered separately, or in small groups, and have been restricted to lower spatial dimensions. Fortunately, by the mid 1970s it became a simple matter to numerically solve the eigenmode equation of Hain and Lüst, including versions that consider resistivity [Freidberg and Hewett 1980]. As computational power has increased, simulations approaching stability from the standpoint of an initial value problem have modeled more of the effects simultaneously.

Plasma simulations which attempt to attain as much realism as possible, whithout narrowly focussing on linear stability or other ideal situations, began in about 1960 [Hain 1960]. Since then, such simulations have proceeded to higher levels of complexity in many different ways. Calculations have proceeded from one spatial dimension to three. Transport coefficients such as resistivity and thermal conductivity have been generalized from scalars to tensors. At first, effects of radiation on energy balance and charge state were neglected, but detailed radiation transport has now been included in some simulations. In most instances, compressible plasmas are assumed, while a few earlier investigations invoked incompressibility. MHD simulations often neglected the Hall term and electron pressure term (electron diamagnetic drift) in Ohm's law, but these terms have now been included in simulations of strongly driven compressional pinches. It was often convenient to assume a fully ionized plasma, but it has become important to consider more general cases of mixed arbitrarily charged species. Because the collisionality of pinches can be small over much of the pinch lifetime, it has become necessary to extend the treatment of ions from a fluid treatment to a particle (Vlasov) treatment. In some instances a need has arisen for anisotropic temperatures, although most studies use scalar temperatures. The list of improvements/advances in pinch simulation will no doubt grow as computer power increases and as experiments add to our knowledge.

In his 1971 dissertation, Lindemuth reported one of the first uses of the alternating-direction-implicit (ADI) method for the time evolution of two-dimensional, two-fluid MHD model equations describing screw pinches [Lindemuth 1971]. He reported good numeric stability for time steps larger than allowed by earlier explicit schemes. He used a tensor electrical resistivity and thermal conductivity, but assumed scalar temperatures for each fluid. The Hall term and diamagnetic drift term in Ohm's law were neglected. In the late 1970s, Lindemuth addressed issues of plasma separation from pinch chamber walls with a general "background plasma" boundary condition that more realistically convected magnetic flux across low density plasma regions [Lindemuth 1977]. This was one of the early fixes for problems with vacuum regions in simulations. Previously, low density regions in MHD fluid calculations often developed unrealistic currents near chamber walls that shielded the bulk of the plasma from applied magnetic fields.

Early in the 1970s, a few of the first hybrid plasma simulations were reported. Such hybrid simulations modeled the plasmas with a combination of fluid electron equations and particle (Vlasov) ions subject to the Lorentz force. Forslund and Freidberg [1971], and Chodura [1975] used one-dimensional (1-d) hybrid simulations to model plasma shocks. A 1-d zero-electron-inertia hybrid model including all field components and a tensor resistivity was used to simulate the ZT-1 experiment at Los Alamos [Sgro and Nielson 1974]. Hamasaki and coworkers used 1-d hybrid codes with self-consistent turbulence effects to model theta pinches [Hamasaki and Krall 1977; Hamasaki *et al.* 1977]. In 1979, benchmark results of a 2-d zero-electron-inertia hybrid method with all field components were obtained [Hewett 1980]; this utilized ADI methods, incorporated the Hall and diamagnetic drift terms in a generalized Ohm's law, and used a simple and robust density cutoff technique to address vacuum regions of the simulation containing few particles. The Hall terms and the robust treatment of vacuum regions became recognized as key to fast and accurate simulation of compression in dynamic plasma pinches. More analytic understanding of Hall term effects on stability came with a one-dimensional study aimed specifically at inclusion of the term in Ohm's law [Coppins et al. 1984].

All of the hybrid simulations made it clear that a single-fluid treatment of ions fails to properly model strongly driven compressional pinches; because of low collisionality, ions reflected off of the "magetic piston" into the quiescent "internal" plasma give the ion distribution two significant peaks in velocity space. A single fluid ion theory is not meant to model two comparable peaks. It also became apparent that hybrid simulations were necessary to model the effects of large ion gyroradii or finite

Larmor radius (FLR) on plasma stability. One early experiment that attempted to directly measure FLR stabilization was a gas-puff Z-pinch experiment [Struve 1980]; the experiment was difficult to interpret, but scaling of instabilities was found to be consistent with theoretical scaling of FLR effects.

Numerical simulation of radiation transport in 1-d screw pinches became practical in the late 1970s during attempts to explain localized sources of X-rays and neutrons observed in plasma focus experiments [Shearer 1976; Vikhrev 1977, 1978]. It was not long before the early LASNEX code at Lawrence Livermore National Lab (LLNL) was used to investigate 1-d radiative collapse of Z-pinches that might explain the localized X-rays [Nielsen 1981]; this was based on earlier understanding that bremstrahlung and electron synchrotron radiation should cause radial collapse of an optically thin high-current Z-pinch to very small dimensions if the current exceeded what became known as the Pease-Braginskii current [Pease 1957; Braginskii 1958; Lawson 1959]. From the simulations and experiments it became clear that radiation has a major effect in time-dynamic pinches. By 1982, Lindemuth had added bremstrahlung radiation, shock physics, and deuteron-deuteron collisions to his two-fluid algorithms in order to simulate a dense plasma focus in one spatial dimension [Lindemuth and Freeman 1982].

Lindemuth continues to play a major role in development of pinch simulations. With the advent of solid deuterium fiber Z-pinches, two-fluid MHD calculations incorporating atomic data bases and equations of state, but not radiative effects, were quickly performed [Lindemuth *et al.* 1989]. By 1990, the solid deuterium fiber simulations progressed to two dimensions, and included radiation effects and neutron production [Lindemuth 1990]. By 1992, with influence from Lindemuth, solid fiber Z-pinch simulations in 2-d were compared in detail with Los Alamos HDZP-I and HDZP-II experiments [Sheehey *et. al.* 1992]; simulation and experiment both suggested rapid development of sausage instabilities after full ionization of the fiber. Curiously, the 1992 simulation did not include Hall and diamagnetic drift terms in Ohm's law, which make the quantitive results questionable.

Three dimensional studies have been confined mainly to MHD equilibrium or quasistatic studies. Nonlinear resistive 3-d MHD calculations were made in the early 1980s [Mirin and Killeen 1983]. Because of the large problem sizes in 3-d MHD, parallel computational methods have been developed [Anderson 1989]. One of the few 3-d electrodynamic plasma simulation codes was reported in 1993 [Larson 1993], but it is better suited to high-frequency plasma-radiation interactions.

The field simulation algorithm presented in this dissertation is a modified version of the algorithm of Hewett [1980]. The algorithm combines an electron fluid and particle-in-cell (PIC) ions to represent the plasma in a hybrid fashion. The plasma is assumed to be quasineutral and displacement current is ignored in Ampere's law. A zero-electron-inertia assumption is used in the electron momentum equation, which yields a generalized Ohm's law that retains the Hall term and diamagnetic drift term. Scalar temperatures are assumed for the electrons and ions, as well as a scalar electron fluid resistivity. The algorithm is implemented in two spatial dimensions, namely axisymmetric cylindrical coordinates. Alternating-direction-implicit (ADI) methods are used to advance the electron fluid drift velocity and the magnetic fields in the simulation. Low density regions of the plasma, where there are essentially no PIC ions, are determined by a cutoff density that invokes solution of the vacuum field equations.

Treatments of vacuum fields and the plasma-vacuum interface differ significantly from the method used by Hewett [1980], and these are the most significant contributions of this dissertation, beside parallelization of ADI. The following vacuum field and plasma-vacuum interface treatments were newly implemented:

1) In *extrained* vacuum regions (connected to the maximum radius), the azimuthal magnetic field B_θ is directly determined by the current, I_z, driven through the enclosed plasma. That field is simply given by $B_\theta = 2I_z/cr$. The equation $(\nabla^2 \bar{B})_\theta = 0$ is not used in extrained regions as by Hewett [1980] because it requires more computational work for relaxation; it also erroneously enforces a curl-free current density at the plasma-vacuum interface.

2) Update of B_θ in *entrained* vacuum regions (isolated from the maximum radius by plasma) is based on the integral form of Faraday's Law. This explicit update also avoids erroneous enforcement of a curl-free current density at the plasma-vacuum interfaces.

3) The electric field $\bar{E}$ is explicitly calculated in the vacuum regions by neglecting the resistive term in the generalized Ohm's law for the dense plasma. This permits calculation of $\bar{E}$ without the relaxation on $\nabla^2 \bar{E} = 0$ used by Hewett [1980].

4) Finite-difference grid effects that generate noise in $\bar{E}$ are ameliorated by an interpolated resistivity in low density plasma regions. This is in contrast to spatial density smoothing by Hewett [1980] and has a more physical basis.

Relaxation on the equations $(\nabla^2 \bar{B})_\theta = 0$ and $\nabla^2 \bar{E} = 0$ would be detrimental to parallel scalability of simulations as well. The new alternatives for evaluation of B_θ and $\bar{E}$ will execute more efficiently on large massively parallel computers.

1.2 Governing equations

The spatial region of a plasma pinch naturally divides into regions of low and high density. Physically, low density regions can be nonneutral and can exhibit long mean free paths and $\bar{E} \times \bar{B}$ charged particle drift, as long as the density of neutral species is sufficiently low. In contrast, high density regions of the pinch are generally quasineutral, collisional for charged particles, and dominated by Ohmic drift. Between the low and high density regions might be nonneutral transition regions in which drift of charged particles has significant amounts of both Ohmic and $\bar{E} \times \bar{B}$ contributions. Modeling in detail a space containing both low and high density regions can be a daunting task. If all spatial scales need resolving, including spatial variation in transition regions, the resolution would have to be on the order of a Debye length in the high-density region. Such spatial scales would be so small that numerical grids for the governing partial differential equations (PDEs) would have too many unknowns to fit in the memory of any computer. It is for this reason that resolution of the transition region is not presently practical for a pinch simulation.

Instead, so as to mimic physical reality, regions of the pinch simulation are classified as *plasma* or *vacuum* depending on whether the ion number density is above or below a cutoff density, ρ_{cutoff}. The interface between the quasineutral plasma and lower density vacuum is treated so as to retain the important physics in the plasma. The model of the plasma can be relatively detailed, the model for the vacuum can be relatively gross, and details of the transition region can be ignored almost completely. This is the spirit of the screw pinch simulation developed in this dissertation.

Besides reduction of the simulation region to plasma and vacuum regions, two other assumptions greatly simplify the pinch simulation. The assumptions are that both the solenoidal and irrotational parts of the displacement current are negligible in

Ampere's law. Neglect of the solenoidal part of the displacement current implies a Darwin approximation to the Maxwell equations. Neglect of the irrotational part is equivalent to an assumption of quasineutrality. In this dissertation, Ampere's law excludes displacement current unless explicitly stated otherwise.

Because ions have a larger mass than electrons, plasma inertia is dominated by ionic contributions. The practical effect is that the ion dynamics are calculated by explicit methods. For hybrid pinch simulation the ions are treated as particles with motion determined by the Lorentz force law. Standard explicit particle-in-cell (PIC) techniques are used for the particle representation. For a fluid ion simulation, the ion quantites are updated explicitly through standard fluid ion mass, momentum and energy equations. Regardless of the representation of the ions, the ions are subject to electric and magnetic fields that are seperately determined by electron fluid dynamics and quasineutral Darwin Maxwell equations. Therefore a time step in the plasma simulation consists of one update of ion quantities followed by updates of electron fluid quantities and electric and magnetic fields.

The update of the electron fluid and the electric and magnetic fields are the main subject of this chapter and the general subject of this dissertation. For the electron fluid and field updates, the updated ion density and ion drift velocity are assumed to be known, and the electrons are assumed to be inertialess.

1.2.1 Governing equations specific to plasma regions

Plasma regions are groups of points on the finite-difference grid for which the electron number density ρ from advancement of ion quantities is greater than a cutoff density ρ_{cutoff}. The cutoff density is always less than a few 10^{12} per cubic centimeter; which for 10eV electrons makes the mean electron-ion collision time τ_e at the plasma-vacuum interfaces on the order of a few hundred nanoseconds, the time scale of a complete pinch simulation. Such a cutoff density confines the dominant physical effects to declared plasma regions that support significant electrical currents due to collisionality.

In this section, the equations governing the dynamics of the plasma regions are itemized. The electron fluid momentum equation in the plasma regions, ignoring electron inertia is (Appendix 1.A1, Gaussian units):

$$\bar{E} = -\frac{\bar{\nabla}(\rho T_e)}{e\rho} - \frac{\bar{u}_e \times \bar{B}}{c} + \bar{\bar{\eta}} \cdot \bar{J}. \qquad (1.1)$$

Equation 1.1 is sometimes called the *generalized Ohm's law* for the plasma. It can be used to find the updated electric field $\bar{E}$ given an electron number density ρ, electron temperature T_e, electron velocity $\bar{u}_e$, magnetic field $\bar{B}$, and current density $\bar{J}$. The term $-\bar{u}_e \times \bar{B}/c$ contains the *Hall term*, $\bar{J} \times \bar{B}/ce\rho$. The term $-\bar{\nabla}(\rho T_e)/e\rho$ is often called the *diamagnetic drift term*, or electron pressure term.

With the use of $\bar{B} = \bar{\nabla} \times \bar{A}$ and the Coulomb guage $\bar{\nabla} \cdot \bar{A} \equiv 0$, the electric field can be decomposed into irrotational and solenoidal parts (same as longitudinal and transverse) as in Eq. 1.2.

$$\bar{E} = -\bar{\nabla}\varphi - \dot{\bar{A}}/c = \bar{E}_{irr} - \dot{\bar{A}}/c. \qquad (1.2)$$

As long as coherent electromagnetic radiation effects are negligible, the fields in the plasma obey to a good approximation the Darwin limit of Maxwell's equations. The Darwin limit together with the assumption of quasineutrality leads to a simplified Ampere's law: $\bar{\nabla} \times \bar{B} = \bar{\nabla} \times \bar{\nabla} \times \bar{A} = 4\pi\bar{J}/c$. Equations 1.1 and 1.2 along with Ampere's law then yield Eq. 1.3 for the time rate of change of $\bar{A}$ (Gaussian units).

$$\dot{\bar{A}} = c\bar{E}_{irr} + \frac{c\bar{\nabla}(\rho T_e)}{e\rho} + \bar{u}_e \times \bar{\nabla} \times \bar{A} - \frac{c^2 \bar{\bar{\eta}} \cdot \bar{\nabla} \times \bar{\nabla} \times \bar{A}}{4\pi}. \tag{1.3}$$

Equations 1.1 and 1.2 along with Faraday's Law, $\dot{\bar{B}} = -c\bar{\nabla} \times \bar{E}$, and Ampere's Law, give Eq. 1.4 for the time-rate of change of $\bar{B}$.

$$\dot{\bar{B}} = \frac{-c\bar{\nabla}\rho \times \nabla(\rho T_e)}{e\rho^2} + \bar{\nabla} \times \left(\bar{u}_e \times \bar{B}\right) - \frac{c^2 \bar{\nabla} \times \left(\bar{\bar{\eta}} \cdot \bar{\nabla} \times \bar{B}\right)}{4\pi}. \tag{1.4}$$

In this dissertation, the most general resistivity considered has the form of a diagonal tensor, Eq. 1.5. This is useful for Z-pinches , for which $\eta_\theta \neq \eta_r = \eta_z$ due to the presence of a large $\bar{B} = \hat{\theta}B_\theta$. It is not accurate for general screw pinches.

$$\bar{\bar{\eta}} = \hat{r}\hat{r}\eta_r + \hat{\theta}\hat{\theta}\eta_\theta + \hat{z}\hat{z}\eta_z. \tag{1.5}$$

For axisymmetric cylindrical coordinates, the angular component of the irrotational electric field must be zero. In other words, $E_{irr,\theta} \equiv -\partial\varphi/\partial\theta = 0$, since all $\partial/\partial\theta \equiv 0$. The angular component of Eq. 1.3 then gives Eq. 1.6.

$$r\dot{A}_\theta = \left[\left(\frac{c^2\eta_\theta}{4\pi} r \frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r}\right) - u_{er}\frac{\partial}{\partial r}\right) + \left(\frac{c^2\eta_\theta}{4\pi}\frac{\partial^2}{\partial z^2} - u_{ez}\frac{\partial}{\partial z}\right)\right] rA_\theta. \tag{1.6}$$

Extracting the angular component of Eq. 1.4 gives Eq. 1.7.

$$\begin{aligned}\dot{B}_\theta = &\left[\left(\frac{c^2}{4\pi}\frac{\partial}{\partial r}\left(\frac{\eta_z}{r}\frac{\partial}{\partial r} r\right) - \frac{\partial}{\partial r}u_{er}\right) + \left(\frac{c^2}{4\pi}\frac{\partial}{\partial z}\left(\eta_r \frac{\partial}{\partial z}\right) - \frac{\partial}{\partial z}u_{ez}\right)\right] B_\theta \\ &+ \frac{\partial}{\partial z}\left(u_{e\theta}B_z\right) + \frac{\partial}{\partial r}\left(u_{e\theta}B_r\right) + \frac{c}{e\rho}\left(\frac{\partial\rho}{\partial r}\frac{\partial T_e}{\partial z} - \frac{\partial\rho}{\partial z}\frac{\partial T_e}{\partial r}\right).\end{aligned} \tag{1.7}$$

Given the density ρ, temperature T_e, resistivity $\bar{\bar{\eta}}$, and ion drift velocity $\bar{u}_i$, Eqs. 1.6 and 1.7 determine the advanced fields in the simulation. Note that Eqs. 1.6 and 1.7 are coupled nonlinear equations. This is because the electron drift velocity is a function of A_θ and B_θ. Again, Ampere's Law gives $\bar{u}_e = \bar{u}_i - c\bar{\nabla}\times\bar{B}/4\pi e\rho$, which has components expressed in Eqs. 1.8.

$$u_{er} = u_{ir} + \frac{c}{4\pi e\rho}\frac{\partial B_\theta}{\partial z} \tag{1.8a}$$

$$u_{e\theta} = u_{i\theta} - \frac{c}{4\pi e\rho}\left(\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r}\right)+\frac{1}{r}\frac{\partial^2}{\partial z^2}\right)(rA_\theta) \tag{1.8b}$$

$$u_{ez} = u_{iz} - \frac{c}{4\pi e\rho}\frac{1}{r}\frac{\partial}{\partial r}(rB_\theta) \tag{1.8c}$$

So, Equations 1.6 and 1.7 are actually quite complicated. In practice, the electron drift velocity $\bar{u}_e$ is fixed in Eqs. 1.6 and 1.7, and can be updated through Eq. 1.8. Fixing $\bar{u}_e$ in this way allows the field advancement to proceed by linear methods at each time step.

1.2.2 Governing equations specific to vacuum regions

Vacuum regions are groups of points on the finite-difference grid for which the ion number density ρ from advancement of ion quantities is less than a cutoff density ρ_{cutoff}. In these vacuum regions, electric and magnetic fields are computed according to source-free quasineutral Darwin Maxwell equations. Current density is assumed to be zero exactly, so that Ampere's law reduces to $\bar{\nabla}\times\bar{B} = 0$.

Ignoring neutral species, the charged particles in the vacuum region might be assumed to be collisionless, executing simple $\bar{E}\times\bar{B}$ drift. Positive and negative charges drift in the same direction so that the electrical current is zero, assuming quasineutrality. Ampere's Law then gives $\bar{\nabla}\times\bar{\nabla}\times\bar{A} = 0$. In the Coulomb guage the angular component is conveniently reexpressed as $(\nabla^2\bar{A})_\theta = 0$, an elliptic PDE obeyed by the angular component of the magnetic vector potential:

$$\left(\nabla^2 \bar{A}\right)_\theta = \left(\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r}\right) + \frac{1}{r}\frac{\partial^2}{\partial z^2}\right)(rA_\theta) = 0. \tag{1.9}$$

Equation 1.9, subject to appropriate boundary conditions, is an accurate description of the spatial behavior of A_θ. But appropriate boundary conditions, especially across plasma-vacuum boundaries, are not immediately obvious. More will be written on this when the update over the whole spatial region is described below.

The behavior of the angular magnetic field in the vacuum is straightforward conceptually. The field must have a $1/r$ dependence in each vacuum region. This is a simple result of Ampere's law without displacement current, which gives Eq. 1.10.

$$B_\theta(r,z) = \frac{2I_z(r,z)}{cr}. \tag{1.10}$$

The quantity $I_z(r,z)$ is the Z-current enclosed by the loop $(r,\theta \in [0,2\pi),z)$ at (r,z). At this point it is useful to distinguish two types of vacuum regions, as in Figure 1.1. An *extrained* vacuum region is a vacuum region in contact with the external boundary at maximum radius. If the plasma is completely seperated from the external boundary, then there is only one extrained vacuum region. Since the vacuum electric current is negligible, B_θ in an extrained vacuum region is determined by I_z flowing in the whole enclosed plasma cross-section. When I_z for the plasma is a driving function for the simulation $I_z(r,z)$ is automatically known at every extrained vacuum grid point, which is very convenient.

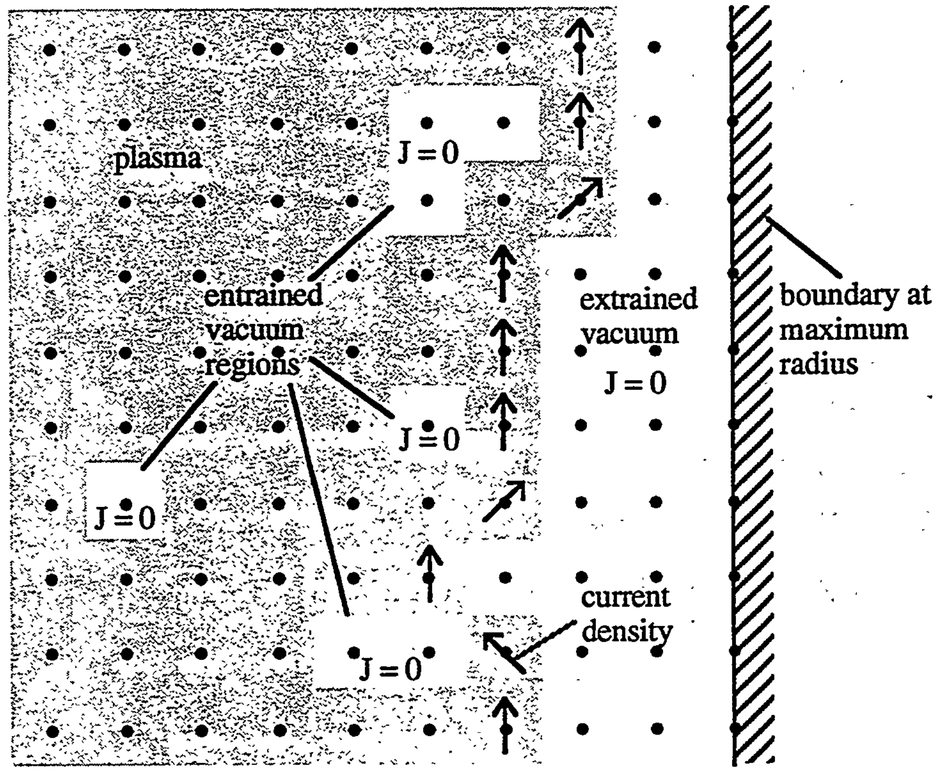

Figure 1.1. Illustration of extrained and entrained vacuum regions in the pinch simulation. There is only one entrained vacuum region in contact with the boundary at maximum radius.

First method for update of B_θ in entrained vacuum regions

Entrained vacuum regions are isolated from the extrained vacuum region by plasma. In entrained regions, $I_z(r,z)$ is not a directly driven quantity, but is determined by plasma dynamics. To effectively find $I_z(r,z)$ in entrained regions, the integral form of Faraday's law can be used, Eq. 1.11 (∂S in Figure 1.2).

$$\int_S dr\,dz\,\dot{B}_\theta \;=\; \frac{2\dot{I}_z}{c}\int_S dr\,dz\,r^{-1} \;=\; c\oint_{\partial S} d\bar{\ell}\cdot\bar{E} \tag{1.11}$$

This formula allows advancement of B_θ in the entrained vacuum regions after evaluation of the line integrals along the plasma-vacuum boundary of each region.

Briefly, the entrained B_θ update can be implemented as described by Eqs. 1.12. Because the void might first appear at time level n, a quantity $\bar{k}^n$ is defined which is the average over the void of the rB_θ product:

$$\bar{k}^n = \int_S drdz\, rB_\theta^n \Big/ \int_S drdz. \tag{1.12a}$$

Then B_θ at time level n is replaced by $B_\theta^n = \bar{k}^n/r$. The B_θ at time level $n+1$ is given by $B_\theta^{n+1} = k^{n+1}/r$, where

$$k^{n+1} = \bar{k}^n + \dot{k}^n \Delta t, \quad \dot{k}^n = c \oint_{\partial S} d\bar{\ell} \cdot \bar{E}^n \Big/ \int_S drdz\, r^{-1}. \tag{1.12b,c}$$

All of the spatial integrals are discretized for the grid. Note that this method for entrained B_θ is first-order accurate in time, at best. A quick method for evaluating line integrals around an arbitrary number of voids of arbitrary shapes was necessary for this scheme, and it was implemented for periodic boundary conditions. Note that Faraday's law gives the B_θ rate equations in both the plasma and the entrained vacuum, with the integral form in the vacuum.

Second method for update of B_θ in entrained vacuum regions

A method for advancing B_θ in the vacuum regions that avoids evaluation of line integrals is based on the equation $\bar{\nabla} \times \bar{\nabla} \times \bar{B} = 4\pi \bar{\nabla} \times \bar{J}/c$. For vacuum points that have no plasma points as nearest neighbors, $\bar{\nabla} \times \bar{J} = 0$. Since $\bar{\nabla} \cdot \bar{B} = 0$, B_θ should obey $(\nabla^2 \bar{B})_\theta = 0$ at such points. An approximation can be made that $\bar{\nabla} \times \bar{J} = 0$ even for

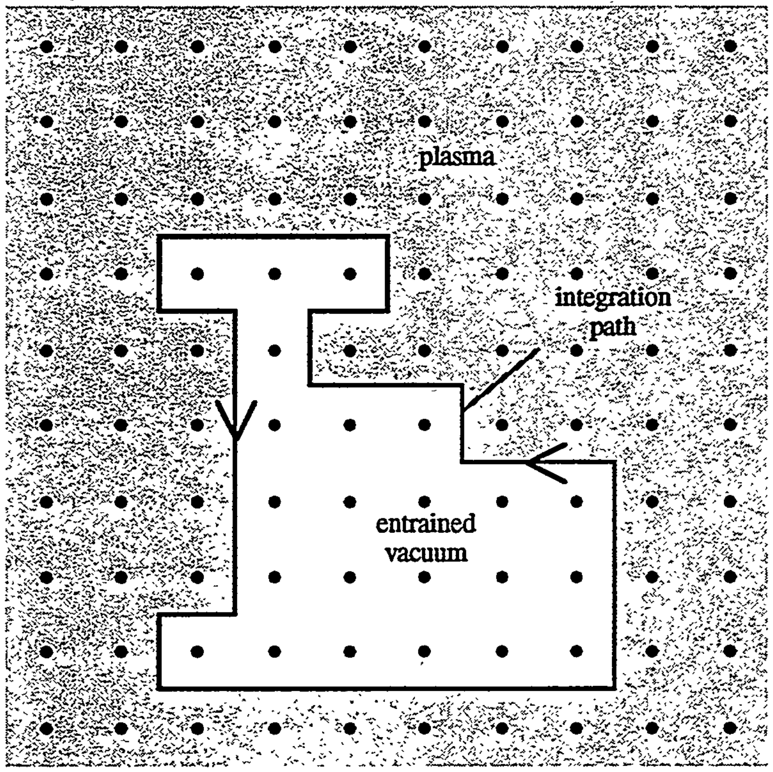

Figure 1.2. A vacuum void entrained inside plasma and isolated from external boundaries. In axisymmetric cylindrical coordinates, it is reasonable to update B_θ with the aid of the integral form of Faraday's law, integrating around the curve on the void boundary, as shown.

vacuum points adjacent to plasma, such as the lettered points in Figure 1.3. Then the points in the plasma can be taken as Dirichlet points for the solution of $(\nabla^2\bar{B})_\theta = 0$ in the vacuum [Hewett 1980]. This approach efficiently uses the ADI techniques elsewhere in the algorithm, but it presents obvious questions of accuracy; it is supported in the simulation algorithm, and was compared with the line integration technique to determine the effects on the simulation behavior.

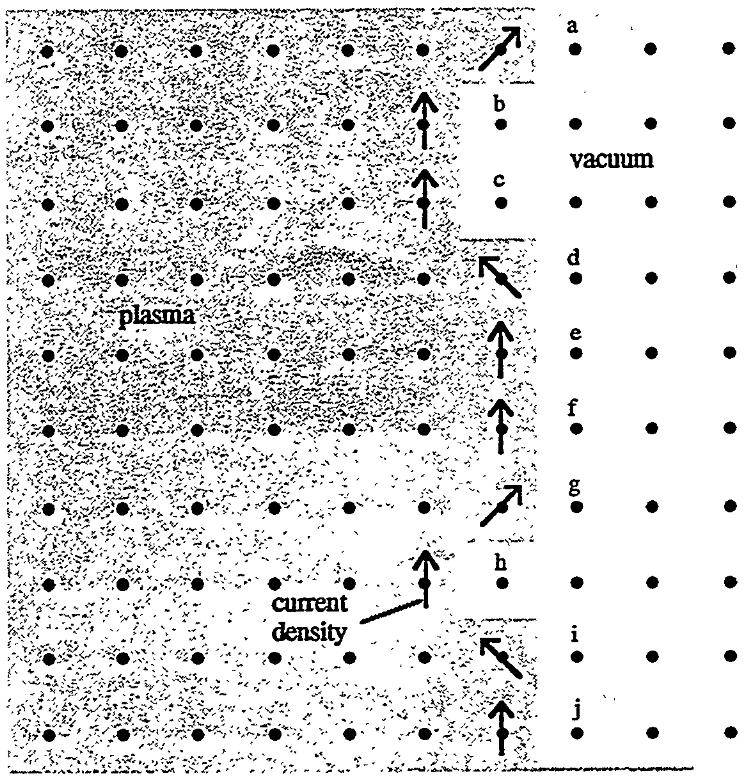

Figure 1.3. Figure illustrating vacuum points at which $\bar{\nabla}\times\bar{J}\neq 0$ in general. Such points are lettered a-j.

1.2.3 Advancement of the angular magnetic vector potential over the whole spatial region

If one were to proceed with Eqs. 1.6 and 1.9 in order to advance A_θ each electron time step, a natural way to pose the problem would be as follows:

> Given Robbins boundary conditions on A_θ along the outer closed boundary $S=\partial R$ of spatial region $R\equiv R_1\cup R_2$, where $R_1\cap R_2=\varnothing$ and the location of a second closed boundary $S_1=\partial R_1$ inside or on ∂R seperating R_1 and R_2, find the A_θ which satisfies Eq. 1.6 with initial condition $A_\theta=A_0$ in region R_1 and which satisfies Eq. 1.9 in region R_2.

Robbins boundary conditions along S specify the value of a linear combination of A_θ and the normal derivative of A_θ at each point on S [Lapidus and Pinder 1982]. Using ϕ instead of A_θ for the unknown, the problem can briefly be posed as follows:

$$\dot{\phi} = X_1\phi + f_1 \text{ and } \phi(t=0) = \phi_0 \text{ in } R_1,$$

$$X_2\phi = 0 \text{ in } R_2,$$

$$a\phi + b\hat{n}\cdot\bar{\nabla}\phi = c \text{ on } S \text{ (Robbins boundary condition).}$$

Here, X_1 and X_2 represent linear second order spatial differential operators, and ϕ is the scalar unknown function of position. All of X_1, X_2, f_1, a, b, and c are spatially dependent, in general.

The situation is illustrated in Figure 1.4. The parabolic equation is obeyed by points $j = 1-7$, which are subject to special linear constraints at points $i = 1-11$ determined by the elliptic equation in region R_2 and boundary conditions at S. An evolution method that proceeds along these lines is given in Appendix 1.A5. Unfortunately, such a technique of evolving ϕ is impractical for fast numerical computation, so that the approximate technique of Hewett [1980] was adopted.

Approximate solution

A computationally inexpensive method uses Eq. 1.6 with modified terms in the vacuum [Hewett 1980]. This early method is based on the observation that the solution to the elliptic PDE $(\nabla^2\bar{A})_\theta = 0$ is the time-asymptotic solution of the parabolic PDE in Eq. 1.13.

$$r\dot{A}_\theta = r\left(\nabla^2\bar{A}\right)_\theta = \left(r\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r}\right) + \frac{\partial^2}{\partial z^2}\right)(rA_\theta) \quad (1.13)$$

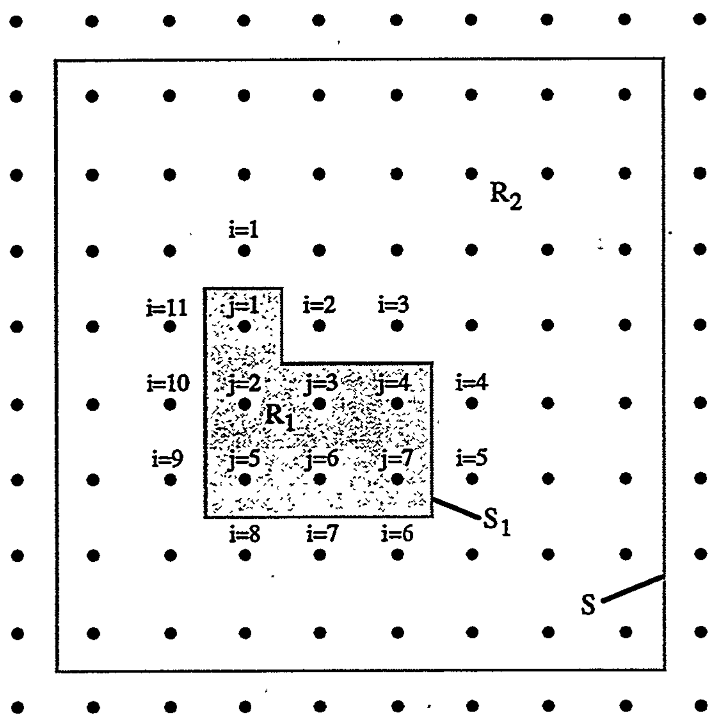

Figure 1.4. Illustration of the initial/boundary value problem for advancement of A_θ over a general spatial grid with both plasma and vacuum regions. A_θ obeys a parabolic rate equation in the plasma and an elliptic equation in the vacuum. Plasma and vacuum are coupled across internal surfaces such as $S_1 = \partial R_1$.

This is the case if both equations are subject to the same time-independent boundary conditions. This latter PDE fits easily into the ADI scheme used to advance A_θ in the plasma. In fact, this equation is the same as the equation used in the plasma region with the term $\bar{u}_e \times \bar{\nabla} \times \bar{A}$ thrown out, and with $c^2\eta_\theta/4\pi = 1$.

This suggests that Eq. 1.6 can be used over the whole grid, as long as $\bar{u}_e \times \bar{\nabla} \times \bar{A}$ is thrown out in the vacuum regions, and as long as η_θ is chosen in the vacuum regions so that A_θ is driven to steady state in a few electron time steps. Toward this end, $\eta_\theta = 4\pi/c^2$ is NOT the right value to use in the vacuum. Rather, $\eta_\theta \approx 4\pi/c^2\Delta t$ would be a better choice. After a time advancement like this, the vacuum equation might be violated to such a degree that it is necessary to solve $(\nabla^2 \bar{A})_\theta = 0$ in the vacuum while fixing A_θ in the plasma. This is essentially the same as the earlier approach of Hewett [1980]. One ADI double-pass is used to advance A_θ in the plasma, simultaneously with the vacuum, and then a few ADI double-passes are used to relax on $(\nabla^2 \bar{A})_\theta = 0$ in the vacuum alone. The total number of double-passes for this is typically less than 10. In the simultaneous plasma-vacuum ADI double pass for advancement of A_θ from t to $t + \Delta t$, linear interpolation in time can be used to set Dirichlet values of A_θ at time $t + \Delta t / 2$. This is not the most accurate treatment of the intermediate Dirichlet values of A_θ [Fairweather and Mitchell 1967], but it is adequate.

1.2.4 Advancement of the angular magnetic flux density over the whole spatial region

Advancement of B_θ naturally breaks into two steps. The first is the advancement of B_θ in vacuum regions. The second is the advancement of B_θ in the plasma according to Eq. 1.7 with the advanced vacuum B_θ used as a set of Dirichlet conditions. For extrained-vacuum regions, the advanced B_θ is given by Eq. 1.10 with I_z a given driving function for the simulation. For entrained-vacuum regions I_z can be found with the aid of Faraday's Law, Eq. 1.11. Using ADI, B_θ is then advanced in the plasma as the second step. For an ADI double-pass to advance B_θ in the plasma from time t to time $t + \Delta t$, linear interpolation in time can be used to set Dirichlet values of B_θ at time $t + \Delta t / 2$. This is not the most accurate treatment of the intermediate Dirichlet values of B_θ [Fairweather and Mitchell 1967], but it is adequate. The advancement outlined in this paragraph is used for the new simulations presented in this dissertation.

There is a previously developed method of advancing B_θ [Hewett 1980]. As long as the same time-independent boundary conditions are used, the solution to $(\nabla^2 \bar{B})_\theta = 0$ for the vacuum regions is the same as the time asymptotic solution to

$$\dot{B}_\theta = (\nabla^2 \bar{B})_\theta = \left(\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r} r \right) + \frac{\partial^2}{\partial z^2} \right) B_\theta . \tag{1.14}$$

This equation is the same as Eq. 1.7, with terms containing $\bar{\nabla} \times \left(\bar{u}_e \times \bar{B} \right)$ and $\bar{\nabla}(\rho T_e)$ thrown out, and with $c^2 \eta_\theta / 4\pi = 1$. Then, just as in the case for A_θ, Eq. 1.7 can be used over the whole grid by zeroing proper terms in the vacuum regions and choosing η_θ in the vacuum regions to drive B_θ closer to the steady state of Eq. 1.14.

1.2.5 Advancement of the poloidal magnetic flux density

Advancing B_r and B_z in the simulation is a simple matter of using $\bar{B} = \bar{\nabla} \times \bar{A}$ over the whole spatial region. As long as the boundary conditions have been properly incorporated in the advancement of A_θ, no boundary conditions need to be directly applied to B_r or B_z.

1.2.6 Advancement of the electron fluid drift velocity

Once A_θ and B_θ have been advanced over the whole grid, it is a trivial matter to take derivatives to get all of the components of electron drift velocity. Second order central differencing is used to calculate the derivatives from Eqs. 1.15:

$$u_{er} = u_{ir} + \frac{c}{4\pi e \rho} \frac{\partial B_\theta}{\partial z}, \tag{1.15a}$$

$$u_{e\theta} = u_{i\theta} + \frac{c}{4\pi e \rho} \left(\frac{\partial B_z}{\partial r} - \frac{\partial B_r}{\partial z} \right), \tag{1.15b}$$

$$u_{ez} = u_{iz} - \frac{c}{4\pi e\rho}\frac{1}{r}\frac{\partial}{\partial r}\left(rB_\theta\right). \tag{1.15c}$$

If $u_e \equiv |\bar{u}_e|$ is a significant fraction of the speed of light for more than a few per cent of the grid points, then the field advance will be invalid. But if only one or two grid points have relativistic electron velocities, the field advance might be acceptable. In this case the major effect of the fast electron fluid is the decreased time step necessary to obey the global electron fluid Courant condition.

When the electron fluid velocity is relativistic at one or two points in the simulation, it is useful to bound the velocity to avoid an exceedingly small time step. Special relativity provides a physical basis for the bound. Note that the drift velocity in the Lorentz term $\bar{u}_e \times \bar{B}$ is the ordinary velocity, and not the proper velocity, so it becomes reasonable to alter the calculation of $\bar{u}_e$ in the relativistic case. Recall the relativistic expression for the current density: $\bar{J} = e(\rho_i\bar{u}_i - \rho_e\bar{u}_e)$. Since the ions are massive relative to the electrons, $u_i \equiv |\bar{u}_i| << c$ in the laboratory frame, and to a good approximation $\rho_{i,lab} = \rho_{i,rest}$. On the other hand, the electrons might flow with sufficient velocity that $\rho_{e,lab} >> \rho_{e,rest}$. The rest frame of the electrons is not the rest frame of the ion species, in general, so the validity of quasineutrality suddenly comes under scrutiny in regions of fast electron flow.

In regions of fast electron flow it is reasonable to assume quasineutrality in the rest frame of the electrons. Since the electrons are lighter, they tend to neutralize any ion charge density that exists in the electron rest frame. The neutralization is aided by the diffusive effects of electron-ion collisions. At a given point in the plasma, the ion density in the rest frame of the electrons will be $\gamma_e\rho_i$, where ρ_i is the ion density in the lab frame and $\gamma_e = (1-u_e^2/c^2)^{-1/2}$. With the working hypothesis above, $\rho_{e,rest} = \gamma_e\rho_i$.

Transforming back to the lab frame then gives $\rho_{e,lab} = \gamma_e \rho_{e,rest} = \gamma_e^2 \rho_i$. In the lab frame then, it might be reasonable to assume $\bar{J} = e\rho_i(\bar{u}_i - \gamma_e^2 \bar{u}_e)$ for a few points of the simulation. The general equations for the updated electron drift velocity become:

$$\gamma_e^2 u_{er} = u_{ir} + \frac{c}{4\pi e\rho}\frac{\partial B_\theta}{\partial z} \tag{1.16a}$$

$$\gamma_e^2 u_{e\theta} = u_{i\theta} + \frac{c}{4\pi e\rho}\left(\frac{\partial B_z}{\partial r} - \frac{\partial B_r}{\partial z}\right) \tag{1.16b}$$

$$\gamma_e^2 u_{ez} = u_{iz} - \frac{c}{4\pi e\rho}\frac{1}{r}\frac{\partial}{\partial r}(rB_\theta) \tag{1.16c}$$

Note that when $u_e << c$, these are the same as the previous equations. Given any $\bar{u}_i$, A_θ, and B_θ, these three equations can be used to compute $\bar{\alpha} \equiv \gamma_e^2 \bar{u}_e$. With $u_e^2 = c^2(1 - 1/\gamma_e^2)$, one gets a quadratic that determines γ_e^2: $\alpha^2 = c^2(\gamma_e^4 - \gamma_e^2)$. If a simulation is to proceed in obeyance of these equations, all of the components of $\bar{u}_e$ should be updated whenever any of them are updated.

1.2.7 Advancement of the electric field

Once $\bar{B}$ and $\bar{u}_e$ have been advanced in the plasma, Eq. 1.1 can be used in the plasma to find the advanced $\bar{E}$. Then, to advance the electric field in the vacuum regions, it is tempting to find an expression for $\nabla^2 \bar{E}$ that is an elliptic PDE governing $\bar{E}$. This equation can then be solved using the advanced $\bar{E}$ in the plasma and on the driving electrodes as Dirichlet boundary conditions. The solution to the equation would be smooth and would have extrema on the vacuum boundary. Toward this end, one can use the identity $\nabla^2 \bar{E} = \bar{\nabla}(\bar{\nabla} \cdot \bar{E}) - \bar{\nabla} \times \bar{\nabla} \times \bar{E}$ to get Eq. 1.17.

$$\nabla^2 \bar{E} = 4\pi \bar{\nabla}\rho + \frac{1}{c^2}\left(4\pi \dot{\bar{J}} + \ddot{\bar{E}}_{irr}\right) \tag{1.17}$$

At first it is tempting to assume $\bar{\nabla}\cdot\bar{E}=0$ everywhere in space. But this is not generally valid at the plasma-vacuum interface, so the assumption should be discarded. Satisfactory use of Eq. 1.17 is therefore difficult at best. By definition, a quasineutral model has no straightforward knowledge of any net charge density ρ or charge density gradient $\bar{\nabla}\rho$. If one assumed a surface charge representation, then one would have to deal with calculation of that surface charge and incorporate it as a jump condition on the perpendicular electric field at the interface. This latter approach might be impossible. Further difficulties reside in the first and second time derivatives; implicit incorporation of these terms would be computationally expensive and explicit incorporation might degrade the stability of the algorithm. This goes without mentioning that the quasineutral model at hand leaves $\bar{E}_{irr}$ unisolated from $\bar{E}_{sol}$.

The simplest approach to calculating $\bar{E}$ in vacuum regions requires only a variation of the electron fluid momentum equation, Eq. 1.1, used in the plasma. Since the vacuum region of the spatial domain should have very low electron-ion collisionality, the resistive term should be negligible there. Throwing out the resistive term gives Eq. 1.18.

$$\bar{E} = -\frac{\bar{\nabla}(\rho T_e)}{e\rho} - \frac{\bar{u}_e\times\bar{B}}{c} \tag{1.18}$$

Calculating the vacuum $\bar{E}$ in this way is computationally cheap, and it ignores all of the interface questions arising with Eq. 1.17, keeping with the spirit of the algorithm. For vacuum points without nearest neighbor plasma points, $\bar{u}_i$ is numerically zeroed, and $\bar{u}_e \doteq \bar{u}_i$ within second-order discretization error, making the numerical $\bar{E}$ depend on the first term. This $\bar{E}$ is not physically accurate near any actual nonneutral plasma-vacuum interface, but it is perhaps sufficiently accurate for this quasineutral simulation. Inaccuracies in $\bar{E}$ in the vacuum away from the interfaces are of little concern. Only those particles which escape into the vacuum regions would see an inaccurate $\bar{E}$ there; as long as the number of escaped particles is relatively small, they will have a negligible effect on the evolution of the plasma regions. For vacuum points with nearest neighbor plasma points, merely enforcing $\bar{\nabla} \times \bar{B} = 0$ gives $u_{er,z} = u_{ir,z}$. The ion drift velocity is not zeroed at such points, so that there can be a contribution from both terms in Eq. 1.18.

The only time Eq. 1.17 might have application in this quasineutral simulation is when the right-hand side is relatively small; something that must be considered on a case-by-case basis. When the right-hand side is negligible, the component equations of $\nabla^2 \bar{E} = 0$ can be solved [Hewett 1980]:

$$\left(\nabla^2 \bar{E}\right)_r = \frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r}(rE_r)\right) + \frac{\partial^2 E_r}{\partial z^2} = 0 \qquad (1.19a)$$

$$\left(\nabla^2 \bar{E}\right)_\theta = \frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r}(rE_\theta)\right) + \frac{\partial^2 E_\theta}{\partial z^2} = 0 \qquad (1.19b)$$

$$\left(\nabla^2 \bar{E}\right)_z = \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial E_z}{\partial r}\right) + \frac{\partial^2 E_z}{\partial z^2} = 0 \qquad (1.19c)$$

Each one of these equations suggests a relaxation on a Laplace equation for each electron time step, with the plasma and the driven boundaries Dirichlet points in E_r, E_θ, and E_z For each solve, the initial guess for the ADI method is relatively good, and the number of iterations for an accurate result is small. This approach essentially loads a smooth weighted average of $\bar{E}$ from interfacial plasma points into nearby interfacial vacuum points. Since the relaxation requires more work, it was not used in the pinch simulations presented in Chapter 2.

1.3 Ion component of the simulation

The pinch algorithm of this dissertation has been developed with the intent of simulating the ion kinetics in strongly driven screw pinches. Because the kinetics require a particle representation of the ions, the electron fluid/field algorithm interfaces with a particle-in-cell (PIC) ion representation. The particle push of the PIC algorithm is presented in Section 1.3.2. Unfortunately, when testing the electron fluid/field algorithm on plasmas initialized in MHD equilibrium, the PIC representation of the ions is too crude for a reasonable number of PIC ions. The statistical noise in a few hundred thousand PIC particles obscures linear MHD growth rates, of interest for benchmarking, yet more particles will not fit in computer memory.

In order to benchmark the pinch simulation against linear MHD stability theory, it became necessary to use the electron/fluid algorithm with a fluid ion representation. The fluid ion representation leaves out ion kinetics important for strongly driven pinches, but it is all that is necessary for checking the stability of MHD equilibria. The fluid representation is free of the statistical noise of the PIC algorithm, requires less computer memory, and has a reasonably short execution time.

1.3.1 Fluid ions

The fluid ion representation is accurate when the ions at each point in space are well localized in velocity space. Localization in velocity space allows the ion velocities to be accurately described by a mean drift velocity $\bar{u}_i$ and higher order velocity moments. The equations governing each species of fluid ions are the typical mass, momentum, and energy conservation relations (Gaussian units):

$$\frac{\partial \rho}{\partial t} + \bar{\nabla}\cdot\left(\rho\,\bar{u}_i\right) = 0, \tag{1.20}$$

$$\frac{\partial}{\partial t}(\rho \,\bar{u}_i) + \bar{\nabla}\cdot(\rho \,\bar{u}_i\bar{u}_i) = -\frac{\bar{\nabla}(\rho T_i)}{m_i} + \frac{q_i\rho}{m_i}\left(\bar{E} + \frac{\bar{u}_i}{c}\times\bar{B}\right), \tag{1.21}$$

$$\frac{\partial T_i}{\partial t} + \bar{\nabla}\cdot(T_i\,\bar{u}_i) = \frac{1}{3}T_i\bar{\nabla}\cdot\bar{u}_i + \frac{1}{\rho}\bar{\nabla}\cdot\kappa\bar{\nabla}T_i + \frac{2m_e}{m_i\tau_e}(T_e - T_i). \tag{1.22}$$

In the pinch simulations, quasineutrality is assumed, so that the density $\rho \equiv \rho_e = \rho_i$ is not subscripted. The quantities m_i q_i, T_i, κ, and τ_e are the ion species mass, charge, temperature, thermal conductivity, and mean collision time with electrons. The speed of light is c; $\bar{E}$ and $\bar{B}$ are the electric and magnetic fields; and T_e is the electron temperature. For benchmarking the pinch simulation against linear MHD theory, it was convenient to take $\kappa \equiv 0$ and $T_e = T_i$. Here, temperatures are expressed in energy units (ergs). Equations 1.20-1.22 are finite-differenced in space and time in order to implement an explicit scheme with donor cell stabilization and conservative boundary conditions.

1.3.2 Particle-In-Cell (PIC) ions

A standard Boris push was used to advance the velocities and positions of the PIC ions [Boris 1970]. Given the jth particle velocity at time level -1/2, the particle velocity at time level +1/2 is given by Eq. 1.23. The jth particle position is then advanced from the 0th time level to the 1th time level according to Eq. 1.24.

$$\bar{v}_j^{n+1/2} = \bar{v}_j^{n-1/2} + \frac{q_j\Delta t}{m_j}\left(\bar{E}_j^n + \frac{\bar{v}_j^{n+1/2} + \bar{v}_j^{n-1/2}}{2c}\times\bar{B}_j^n\right), \tag{1.23}$$

$$\bar{x}_j^{n+1} = \bar{x}_j^n + \Delta t\bar{v}_j^{n+1/2}. \tag{1.24}$$

The $\bar{x}_j^{n+1}$ are then weighted to the grid and summed over particles to get the plasma particle density at the +1 time level, ρ_i^{n+1}. The formula for the velocity advance can be rewritten to facilite explicit calculation:

$$\bar{v}_j^{n+1/2} = \bar{\bar{R}}_j^n(\Delta t)\cdot\bar{v}_j^{n-1/2} + \frac{q_j\Delta t}{2m_j}\left(\bar{\bar{I}} + \bar{\bar{R}}_j^n(\Delta t)\right)\cdot\bar{E}_j^n. \tag{1.25}$$

Here, the tensor $\bar{\bar{R}}_j^n(\Delta t)$ is determined by the formula

$$\bar{\bar{R}}_j^n(\Delta t) \equiv \frac{(1-(\Theta_j^n)^2)\bar{\bar{I}} - 2\bar{\Theta}_j^n\times\bar{\bar{I}} + 2\bar{\Theta}_j^n\bar{\Theta}_j^n}{1+(\Theta_j^n)^2}, \text{ with } \bar{\Theta}_j^n(\Delta t)\equiv\frac{q_j\Delta t}{2m_jc}\bar{B}_j^n. \tag{1.26}$$

The electric field $\bar{E}_j^n$ and magnetic field $\bar{B}_j^n$ are interpolated from the grid to the jth particle, and are at the intermediate time level n. This is just the Boris push, a second-order accurate explicit leap-frog method [Boris 1970].

To get $\bar{v}_j^{n+1}$, the same velocity formula is used with half the time step and with the electric and magnetic fields lagged three quarters of a time step (from the second-order accurate Boris push formula):

$$\bar{v}_j^{n+1} = \bar{\bar{R}}_j^n(\tfrac{\Delta t}{2})\cdot\bar{v}_j^{n+1/2} + \frac{q_j\Delta t}{4m_j}\left(\bar{\bar{I}} + \bar{\bar{R}}_j^n(\tfrac{\Delta t}{2})\right)\cdot\bar{E}_j^n. \tag{1.27}$$

The $\bar{v}_j^{n+1}$ are then weighed to the grid and summed over particles to get the mean ion drift velocity at the +1 time level, $\bar{u}_i^{n+1}$. Note that $\bar{v}_j^{n+1}$ is not used in the next particle velocity advance.

The formulas for interpolating the fields from the grid to a particle and for weighing a particle to the grid use standard linear shape functions with weights involving only the four grid points nearest to the particle. Effects of the weighing scheme have been extensively studied elsewhere [Langdon 1970; Okuda 1972; Birdsall 1985].

Appendix 1.A1
Electron mass, momentum, and energy equation derivations

The plasma kinetic equation for the electrons in Gaussian units is Equation 1.28.

$$\frac{\partial f_e}{\partial t} + \bar{v}\cdot\bar{\nabla}_{\bar{x}} f_e - \frac{e}{m_e}\left(\bar{E}+\frac{1}{c}\bar{v}\times\bar{B}\right)\cdot\bar{\nabla}_{\bar{v}} f_e = \left(\frac{\partial f_e}{\partial t}\right)_c . \tag{1.28}$$

Here $f_e(\bar{x},\bar{v},t)$ is the time-dependent phase-space distribution of electrons, and

$$\left(\frac{\partial f_e}{\partial t}\right)_c = \frac{e}{m_e}\left\langle\left(\delta\bar{E}+\frac{\bar{v}\times\delta\bar{B}}{c}\right)\cdot\bar{\nabla}_{\bar{v}}\delta N_e\right\rangle \tag{1.29}$$

is the rate of change of f_e due to collisions in the plasma. The distrubution f_e is a smooth function obtained by averaging the Klimontovich density $N_e(\bar{x},\bar{v},t)$ over an appropriate volume, and $\delta N_e \equiv N_e - f_e$.

Mass conservation equation

We multiply the equation by $d\bar{v}$ and integrate term by term over all $\bar{v}$-space:

$$\frac{\partial}{\partial t}\int_{all\,\bar{v}} d\bar{v}\, f_e \equiv \frac{\partial n_e(\bar{x},t)}{\partial t}, \tag{1.30}$$

$$\int_{all\,\bar{v}} d\bar{v}\,\bar{v}\cdot\bar{\nabla}_{\bar{x}} f_e = \int_{all\,\bar{v}} d\bar{v}\,\bar{\nabla}_{\bar{x}}\cdot f_e\bar{v} = \bar{\nabla}_{\bar{x}}\cdot\int_{all\,\bar{v}} d\bar{v}\, f_e\,\bar{v} \equiv \bar{\nabla}_{\bar{x}}\cdot n_e(\bar{x},t)\bar{u}_e(\bar{x},t), \tag{1.31}$$

$$\int_{all\,\bar{v}} d\bar{v}\,\bar{E}\cdot\bar{\nabla}_{\bar{v}} f_e = \bar{E}\cdot\int_{all\,\bar{v}} d\bar{v}\,\bar{\nabla}_{\bar{v}} f_e = \bar{E}\cdot\oint_{v=\infty} ds_{\bar{v}}\,\hat{n}_{\bar{v}} f_e = 0, \tag{1.32}$$

$$\int_{all\,\bar{v}} d\bar{v}\,\bar{v}\times\bar{B}\cdot\bar{\nabla}_{\bar{v}} f_e = -\bar{B}\cdot\int_{all\,\bar{v}} d\bar{v}\,\bar{v}\times\bar{\nabla}_{\bar{v}} f_e = \bar{B}\cdot\int_{all\,\bar{v}} d\bar{v}\,\bar{\nabla}_{\bar{v}}\times f_e\bar{v} = \bar{B}\cdot\oint_{v=\infty} ds_{\bar{v}}\,\hat{n}\times f_e\bar{v} = 0. \tag{1.33}$$

The surface integrals are zero because the particle velocities are bounded. The following vector identities are useful:

$$\bar{\nabla}_{\bar{x}} \cdot f_e \bar{v} = \frac{\partial}{\partial x_i}(f_e v_i) = \frac{\partial f_e}{\partial x_i} v_i + f_e \frac{\partial v_i}{\partial x_i} = v_i \frac{\partial f_e}{\partial x_i} = \bar{v} \cdot \bar{\nabla}_{\bar{x}} f_e, \tag{1.34}$$

$$\bar{v} \times \bar{B} \cdot \bar{\nabla}_{\bar{v}} f_e = \varepsilon_{ijk} v_j B_k \frac{\partial f_e}{\partial v_i} = -B_k \varepsilon_{kji} v_j \frac{\partial f_e}{\partial v_i} = -\bar{B} \cdot \bar{v} \times \bar{\nabla}_{\bar{v}} f_e, \tag{1.35}$$

$$\begin{aligned} \bar{\nabla}_{\bar{v}} \times f_e \bar{v} &= \hat{x}_i \varepsilon_{ijk} \frac{\partial (f_e v_k)}{\partial v_j} = \hat{x}_i \varepsilon_{ijk} \left(\frac{\partial f_e}{\partial v_j} v_k + f_e \frac{\partial v_k}{\partial v_j} \right) = \hat{x}_i \varepsilon_{ijk} \left(\frac{\partial f_e}{\partial v_j} v_k + f_e \delta_{jk} \right) \\ &= \hat{x}_i \varepsilon_{ijk} \frac{\partial f_e}{\partial v_j} v_k = -\bar{v} \times \bar{\nabla}_{\bar{v}} f_e . \end{aligned} \tag{1.36}$$

The result is the electron density continuity equation:

$$\frac{\partial n_e(\bar{x},t)}{\partial t} + \bar{\nabla} \cdot n_e(\bar{x},t) \bar{u}_e(\bar{x},t) = \left(\frac{\partial n_e(\bar{x},t)}{\partial t} \right)_c . \tag{1.37}$$

When the equation is multiplied by the electron mass, the equation expresses conservation of electron mass. Unless there is collisional generation or capture of free electrons, the rate of change of electron density due to collisions is essentially zero.

Momentum conservation equation

By multiplying the transport equation by $d\bar{v}\,\bar{v}$ and subsequently integrating it over velocity space, one will arrive at the electron flux equation. Then multiplication by the electron mass yields the electron momentum equation. Terms in the integration are as follows:

$$\frac{\partial}{\partial t} \int_{all\,\bar{v}} d\bar{v}\, \bar{v} f_e \equiv \frac{\partial}{\partial t} \left(n_e(\bar{x},t) \bar{u}_e(\bar{x},t) \right), \tag{1.38}$$

$$\int_{all\,\bar{v}} d\bar{v}\, \bar{v}\bar{v} \cdot \bar{\nabla}_{\bar{x}} f_e = \bar{\nabla}_{\bar{x}} \cdot \int_{all\,\bar{v}} d\bar{v}\, f_e \bar{v}\bar{v} \equiv \bar{\nabla}_{\bar{x}} \cdot n_e \langle \bar{v}\bar{v} \rangle, \tag{1.39}$$

$$\int_{all\,\bar{v}} d\bar{v}\,\bar{v}\bar{E}\cdot\bar{\nabla}_{\bar{v}}f_e = \int_{all\,\bar{v}} d\bar{v}\left(\bar{\nabla}_{\bar{v}}\cdot\left(f_e\bar{E}\bar{v}\right) - f_e\bar{E}\right) = \oint_{v=\infty} d\bar{s}_{\bar{v}}\left(f_e\bar{E}\,\bar{v}\right)\cdot\hat{n} - \int_{all\,\bar{v}} d\bar{v}\,f_e\bar{E} = -n_e\bar{E}, \quad (1.40)$$

$$\begin{aligned}\int_{all\,\bar{v}} d\bar{v}\,\bar{v}\left(\bar{v}\times\bar{B}\cdot\bar{\nabla}_{\bar{v}}f_e\right) &= \int_{all\,\bar{v}} d\bar{v}\left(\bar{\nabla}_{\bar{v}}\cdot\left(f_e\bar{v}\times\bar{B}\bar{v}\right) - f_e\bar{v}\times\bar{B}\right)\\ &= \oint_{v=\infty} d\bar{s}_{\bar{v}}\left(f_e\bar{v}\times\bar{B}\,\bar{v}\right)\cdot\hat{n} - \int_{all\,\bar{v}} d\bar{v}\,f_e\bar{v}\times\bar{B} = -n_e\bar{u}_e\times\bar{B}.\end{aligned} \quad (1.41)$$

Again, the surface integrals are zero because the particle velocities are bounded. The following less commonly known identities are useful:

$$\bar{\nabla}_{\bar{x}}\cdot\left(f_e\bar{v}\,\bar{v}\right) = \hat{x}_j\frac{\partial}{\partial x_i}\left(f_e v_i v_j\right) = \hat{x}_j\left(\frac{\partial f_e}{\partial v_i}v_i v_j + f_e\frac{\partial v_i}{\partial x_i}v_j + f_e v_i\frac{\partial v_j}{\partial x_i}\right) = \bar{v}\bar{v}\cdot\bar{\nabla}_{\bar{x}}f_e, \quad (1.42)$$

$$\bar{\nabla}_{\bar{v}}\cdot\left(f_e\bar{E}\,\bar{v}\right) = \hat{x}_j\frac{\partial}{\partial v_i}\left(f_e E_i v_j\right) = \hat{x}_j\left(\frac{\partial f_e}{\partial v_i}E_i v_j + f_e\frac{\partial E_i}{\partial v_i}v_j + f_e E_i\delta_{ij}\right) = \bar{v}\bar{E}\cdot\bar{\nabla}_{\bar{v}}f_e + f_e\bar{E}, \quad (1.43)$$

$$\begin{aligned}\bar{\nabla}_{\bar{v}}\cdot\left(f_e\bar{v}\times\bar{B}\,\bar{v}\right) &= \hat{x}_l\frac{\partial}{\partial v_i}\left(f_e\varepsilon_{ijk}v_j B_k v_l\right)\\ &= \hat{x}_l\varepsilon_{ijk}\left(\frac{\partial f_e}{\partial v_i}v_j B_k v_l + f_e\delta_{ij}B_k v_l + f_e v_j\frac{\partial B_k}{\partial v_i}v_l + f_e v_j B_k\delta_{il}\right) = \bar{v}\bar{v}\times\bar{B}\cdot\bar{\nabla}_{\bar{v}}f_e + f_e\bar{v}\times\bar{B}.\end{aligned} \quad (1.44)$$

The electron flux conservation equation becomes:

$$\frac{\partial}{\partial t}\left(n_e\bar{u}_e\right) + \bar{\nabla}_{\bar{x}}\cdot n_e\langle\bar{v}\bar{v}\rangle + \frac{en_e}{m_e}\left(\bar{E}+\frac{1}{c}\bar{u}_e\times\bar{B}\right) = \int_{all\,\bar{v}} d\bar{v}\,\bar{v}\left(\frac{\partial f_e}{\partial t}\right)_c. \quad (1.45)$$

When the distribution function is nearly Maxwellian, it is useful to express the electron velocity as $\bar{v} = \bar{u}_e + \delta\bar{v}$, where $\langle\delta\bar{v}\rangle = 0$. Then $\langle\bar{v}\bar{v}\rangle = \bar{u}_e\bar{u}_e + \langle\delta\bar{v}\delta\bar{v}\rangle$. Neglecting free electron capture and generation due to collisions, the first two terms of the electron flux equation become Eq. 1.46.

$$
\begin{aligned}
\frac{\partial}{\partial t}\left(n_e\bar{u}_e\right) + \bar{\nabla}_{\bar{x}}\cdot n_e\langle\bar{v}\bar{v}\rangle &= \frac{\partial n_e}{\partial t}\bar{u}_e + n_e\frac{\partial\bar{u}_e}{\partial t} + \bar{\nabla}_{\bar{x}}\cdot n_e\left(\bar{u}_e\bar{u}_e\right) + \bar{\nabla}_{\bar{x}}\cdot n_e\langle\delta\bar{v}\delta\bar{v}\rangle \\
&= \left(\frac{\partial n_e}{\partial t}+\bar{\nabla}_{\bar{x}}\cdot n_e\bar{u}_e\right)\bar{u}_e + n_e\left(\frac{\partial\bar{u}_e}{\partial t}+\bar{u}_e\cdot\bar{\nabla}_{\bar{x}}\bar{u}_e\right) + \bar{\nabla}_{\bar{x}}\cdot n_e\langle\delta\bar{v}\delta\bar{v}\rangle \qquad (1.46) \\
&= n_e\left(\frac{\partial\bar{u}_e}{\partial t}+\bar{u}_e\cdot\bar{\nabla}_{\bar{x}}\bar{u}_e\right) + \hat{x}_i\frac{\partial}{\partial x_i}\left(n_e\delta_{ij}\left\langle\delta v_j^2\right\rangle\right).
\end{aligned}
$$

The following identities are useful:

$$
\begin{aligned}
\bar{\nabla}_{\bar{x}}\cdot n_e\left(\bar{u}_e\bar{u}_e\right) &= \hat{x}_j\frac{\partial}{\partial x_i}\left(n_e u_{ei}u_{ej}\right) = \hat{x}_j\left(\frac{\partial n_e}{\partial x_i}u_{ei}u_{ej}+n_e\left(\frac{\partial u_{ei}}{\partial x_i}u_{ej}+u_{ei}\frac{\partial u_{ej}}{\partial x_i}\right)\right) \qquad (1.47)\\
&= \bar{u}_e\bar{u}_e\cdot\bar{\nabla}_{\bar{x}}n_e + n_e\bar{u}_e\bar{\nabla}_{\bar{x}}\cdot\bar{u}_e + n_e\bar{u}_e\cdot\bar{\nabla}_{\bar{x}}\bar{u}_e = \bar{u}_e\bar{\nabla}_{\bar{x}}\cdot n_e\bar{u}_e + n_e\bar{u}_e\cdot\bar{\nabla}_{\bar{x}}\bar{u}_e\,,
\end{aligned}
$$

$$
\bar{\nabla}_{\bar{x}}\cdot n_e\langle\delta\bar{v}\delta\bar{v}\rangle = \hat{x}_j\frac{\partial}{\partial x_i}\left(n_e\left\langle\delta v_i\delta v_j\right\rangle\right) = \hat{x}_i\frac{\partial}{\partial x_i}\left(n_e\delta_{ij}\left\langle\delta v_j^2\right\rangle\right). \qquad (1.48)
$$

No summation is implied by δv_j^2. It is convenient to define a diagonal temperature tensor: $\mathrm{T}_{eij} = m_e\delta_{ij}\left\langle\delta v_j^2\right\rangle$. If the temperature is isotropic, then $\bar{\bar{\mathrm{T}}}_e = \frac{1}{3}m_e\left\langle\delta v^2\right\rangle\bar{\bar{I}} = T_e\bar{\bar{I}}$. For this isotropic case only:

$$
\bar{\nabla}_{\bar{x}}\cdot n_e\langle\delta\bar{v}\delta\bar{v}\rangle = \hat{x}_i\frac{\partial}{\partial x_i}\left(\frac{n_eT_e\delta_{ij}}{m_e}\right) = \hat{x}_i\frac{1}{m_e}\left(\frac{\partial n_e}{\partial x_i}T_e+n_e\frac{\partial T_e}{\partial x_i}\right)\delta_{ij} = \frac{1}{m_e}\bar{\nabla}_{\bar{x}}\left(n_eT_e\right). \qquad (1.49)
$$

The electron velocity equation becomes

$$
n_e\left(\frac{\partial\bar{u}_e}{\partial t}+\bar{u}_e\cdot\bar{\nabla}_{\bar{x}}\bar{u}_e\right)+\frac{\bar{\nabla}_{\bar{x}}\left(n_eT_e\right)}{m_e}+\frac{en_e}{m_e}\left(\bar{E}+\frac{1}{c}\bar{u}_e\times\bar{B}\right) = \int_{\mathrm{all}\,\bar{v}} d\bar{v}\,\bar{v}\left(\frac{\partial f_e}{\partial t}\right)_c. \qquad (1.50)
$$

It is tedious to express the collision integral on the right in terms of the magnetic field, density, electron drift velocity and electron temperature. When the product of electron-cyclotron radian frequency, $\omega_{ce} = eB/m_ec$, and mean electron-ion collision time, τ_e, is much greater than unity, the collision integral is reasonably approximated by the sum of two terms, a frictional force term and a thermal force term [Braginskii 1958, 1965].

$$\int_{all\,\vec{v}} d\vec{v}\,\vec{v}\left(\frac{\partial f_e}{\partial t}\right)_c = \frac{1}{m_e}\left(\vec{R}_{eu}+\vec{R}_{eT}\right), \text{ where}$$

$$\vec{R}_{eu} \equiv en_e\left(\eta_\parallel \vec{J}_{col\,\parallel} + \eta_\perp \vec{J}_{col\,\perp}\right), \quad \vec{J}_{col} \equiv en_e(\vec{u}_i - \vec{u}_e), \quad \eta_\perp = 1.96\,\eta_\parallel = \frac{m_e}{e^2 n_e \tau_e} \text{ for } Z=1,$$

$$\vec{R}_{eT} \equiv -0.71 n_e \vec{\nabla}_\parallel T_e - \frac{3n_e}{2\omega_{ce}\tau_e}\hat{b}\times\vec{\nabla}T_e \text{ for } Z=1.$$

The quantity $\vec{R}_{eu}$ is the *frictional force density*, while the quantity $\vec{R}_{eT}$ is the *thermal force density*. The unit vector $\hat{b}$ is parallel to $\vec{B}$. The subscripts $\parallel$ and $\perp$ refer to directions parallel and perpendicular to $\vec{B}$, respectively.

Thermal conductivity of the electrons along lines of magnetic flux tends to be high, so that the first term in $\vec{R}_{eT}$ can be relatively small. Thermal conductivity perpendicular to magnetic field lines is not quite so high, but the assumption that $\omega_{ce}\tau_e >> 1$ can make the second term in $\vec{R}_{eT}$ small as well. It is for these reasons that the thermal force is often neglected. The remaining frictional force can be expressed in tensor notation as $\vec{R}_{eu} = en_e\bar{\bar{\eta}}\cdot\vec{J}_{col}$. Multiplying Eq. 1.50 by $m_e/n_e e$ and taking $\rho \equiv n_e = n_i$ gives Eq. 1.51.

$$\vec{E} = -\frac{\vec{\nabla}(\rho T_e)}{e\rho} - \frac{m_e\left(\vec{u}_e\cdot\vec{\nabla}\right)\vec{u}_e}{e} - \frac{\vec{u}_e\times\vec{B}}{c} + \bar{\bar{\eta}}\cdot\vec{J}_{col}. \tag{1.51}$$

Note that $\vec{J}_{col}$ is not equal to the electrodynamic $\vec{J} = e\left(\rho_i\vec{u}_i - \rho_e\vec{u}_e\right)$ in non-neutral or relativistic situations. When the electron inertia is ignored, this equation reduces to Eq. 1.1 that is used in the pinch simulation:

$$\vec{E} = -\frac{\vec{\nabla}(\rho T_e)}{e\rho} - \frac{\vec{u}_e\times\vec{B}}{c} + \bar{\bar{\eta}}\cdot\vec{J}_{col}. \tag{1.52}$$

Energy conservation equations

Multiplying the kinetic equation by $d\bar{v}\ \bar{v}\bar{v}$ and subsequently integrating over all of velocity space gives the second velocity moment equation, which when multiplied by half of the electron mass is just the generalized electron energy tensor equation. The terms in the equation integrate as follows:

$$\frac{\partial}{\partial t}\int_{all\,\bar{v}} d\bar{v}\ \bar{v}\bar{v}f_e \equiv \frac{\partial}{\partial t}\big(n_e(\bar{x},t)\langle\bar{v}\bar{v}\rangle(\bar{x},t)\big), \tag{1.53}$$

$$\int_{all\,\bar{v}} d\bar{v}\ \bar{v}\bar{v}\bar{v}\cdot\bar{\nabla}_{\bar{x}}f_e = \bar{\nabla}_{\bar{x}}\cdot\int_{all\,\bar{v}} d\bar{v}\, f_e\bar{v}\bar{v}\bar{v} \equiv \bar{\nabla}_{\bar{x}}\cdot n_e\langle\bar{v}\bar{v}\bar{v}\rangle, \tag{1.54}$$

$$\begin{aligned}\int_{all\,\bar{v}} d\bar{v}\ \bar{v}\bar{v}\bar{E}\cdot\bar{\nabla}_{\bar{v}}f_e &= \int_{all\,\bar{v}} d\bar{v}\left(\bar{\nabla}_{\bar{v}}\cdot\left(f_e\bar{E}\bar{v}\bar{v}\right) - f_e\left(\bar{E}\bar{v}+\bar{v}\bar{E}\right)\right)\\ &= \oint_{v=\infty} ds_{\bar{v}}\left(f_e\bar{E}\bar{v}\bar{v}\right)\cdot\hat{n} - \int_{all\,\bar{v}} d\bar{v}\, f_e\left(\bar{E}\bar{v}+\bar{v}\bar{E}\right) = -n_e\left(\bar{E}\bar{u}_e+\bar{u}_e\bar{E}\right),\end{aligned} \tag{1.55}$$

$$\begin{aligned}\int_{all\,\bar{v}} d\bar{v}\ \bar{v}\bar{v}\left(\bar{v}\times\bar{B}\right)\cdot\bar{\nabla}_{\bar{v}}f_e &= \int_{all\,\bar{v}} d\bar{v}\left(\bar{\nabla}_{\bar{v}}\cdot\left(f_e\bar{v}\times\bar{B}\bar{v}\bar{v}\right) - f_e\left(\left(\bar{v}\times\bar{B}\right)\bar{v}+\bar{v}\left(\bar{v}\times\bar{B}\right)\right)\right)\\ &= \oint_{v=\infty} ds_{\bar{v}}\, f_e\left(\bar{v}\times\bar{B}\right)\bar{v}\bar{v}\cdot\hat{n} - \int_{all\,\bar{v}} d\bar{v}\, f_e\left(\left(\bar{v}\times\bar{B}\right)\bar{v}+\bar{v}\left(\bar{v}\times\bar{B}\right)\right)\\ &= -n_e\left\langle\left(\bar{v}\times\bar{B}\right)\bar{v}+\bar{v}\left(\bar{v}\times\bar{B}\right)\right\rangle.\end{aligned} \tag{1.56}$$

The following less commonly known vector identities are useful:

$$\bar{\nabla}_{\bar{x}}\cdot\left(f_e\bar{v}\bar{v}\bar{v}\right) = \hat{x}_j\hat{x}_k\frac{\partial}{\partial x_i}\left(f_e v_i v_j v_k\right) = \hat{x}_j\hat{x}_k\left(\frac{\partial f_e}{\partial x_i}v_i v_j v_k\right) = \bar{v}\bar{v}\bar{v}\cdot\bar{\nabla}_{\bar{x}}f_e, \tag{1.57}$$

$$\begin{aligned}\bar{\nabla}_{\bar{v}}\cdot\left(f_e\bar{E}\bar{v}\bar{v}\right) &= \hat{x}_j\hat{x}_k\frac{\partial}{\partial v_i}\left(f_e E_i v_j v_k\right)\\ &= \hat{x}_j\hat{x}_k\left(\frac{\partial f_e}{\partial v_i}E_i v_j v_k + f_e\frac{\partial E_i}{\partial v_i}v_j v_k + f_e E_i\delta_{ij}v_k + f_e E_i v_j\delta_{ik}\right) = \bar{v}\bar{v}\bar{E}\cdot\bar{\nabla}_{\bar{v}}f_e + f_e\left(\bar{E}\bar{v}+\bar{v}\bar{E}\right),\end{aligned} \tag{1.58}$$

$$\begin{aligned}\bar{\nabla}_{\bar{v}}\cdot\left(f_e\bar{v}\times\bar{B}\bar{v}\bar{v}\right) &= \hat{x}_l\hat{x}_m\frac{\partial}{\partial v_i}\left(f_e\varepsilon_{ijk}v_jB_kv_lv_m\right)\\ &= \hat{x}_l\hat{x}_m\varepsilon_{ijk}\left(\begin{aligned}\frac{\partial f_e}{\partial v_i}v_jB_kv_lv_m + f_e\delta_{ij}B_kv_lv_m + f_ev_j\frac{\partial B_k}{\partial v_i}v_lv_m\\ + f_ev_jB_k\delta_{il}v_m + f_ev_jB_kv_l\delta_{im}\end{aligned}\right)\\ &= \bar{v}\bar{v}\bar{v}\times\bar{B}\cdot\bar{\nabla}_{\bar{v}}f_e + f_e\left(\left(\bar{v}\times\bar{B}\right)\bar{v}+\bar{v}\left(\bar{v}\times\bar{B}\right)\right).\end{aligned}\tag{1.59}$$

The generalized electron second velocity moment tensor equation is then

$$\frac{\partial}{\partial t}\left(n_e\langle\bar{v}\bar{v}\rangle\right)+\bar{\nabla}_{\bar{x}}\cdot n_e\langle\bar{v}\bar{v}\bar{v}\rangle+\frac{en_e}{m_e}\left(\bar{E}\bar{u}_e+\bar{u}_e\bar{E}+\frac{1}{c}\left\langle\left(\bar{v}\times\bar{B}\right)\bar{v}+\bar{v}\left(\bar{v}\times\bar{B}\right)\right\rangle\right) = \int_{all\,\bar{v}} d\bar{v}\,\bar{v}\bar{v}\left(\frac{\partial f_e}{\partial t}\right)_c. \tag{1.60}$$

Of course, one of the most useful consequences of this equation is extracted when it is doubly contracted with the identity tensor $\bar{\bar{I}} \equiv \bar{x}_i\bar{x}_j\delta_{ij}$.

$$\langle\bar{v}\bar{v}\rangle:\bar{\bar{I}} = \left\langle v^2\right\rangle,\quad \bar{\nabla}_{\bar{x}}\cdot n_e\langle\bar{v}\bar{v}\bar{v}\rangle:\bar{\bar{I}} = \bar{\nabla}_{\bar{x}}\cdot n_e\left\langle v^2\bar{v}\right\rangle,\quad \left(\bar{E}\bar{u}_e+\bar{u}_e\bar{E}\right):\bar{\bar{I}} = 2\bar{u}_e\cdot\bar{E}, \tag{1.61a,b,c}$$

$$\left\langle\left(\bar{v}\times\bar{B}\right)\bar{v}+\bar{v}\left(\bar{v}\times\bar{B}\right)\right\rangle:\bar{\bar{I}} = \left\langle\varepsilon_{ijk}v_jB_kv_l+v_i\varepsilon_{lmn}v_mB_n\right\rangle\delta_{il} = \left\langle\varepsilon_{ijk}v_jB_kv_i+v_i\varepsilon_{imn}v_mB_n\right\rangle = 0. \tag{1.62}$$

The quantity $\frac{1}{2}m_en_e\left\langle v^2\right\rangle$ is just the electron kinetic energy density. So, the double contraction yields the more commonly used scalar electron energy equation:

$$\frac{m_e}{2}\frac{\partial}{\partial t}\left(n_e\left\langle v^2\right\rangle\right) + \frac{m_e}{2}\bar{\nabla}_{\bar{x}}\cdot n_e\left\langle v^2\bar{v}\right\rangle + en_e\bar{u}_e\cdot\bar{E} = \frac{m_e}{2}\int_{all\,\bar{v}} d\bar{v}\,v^2\left(\frac{\partial f_e}{\partial t}\right)_c. \tag{1.63}$$

Reexpression of the collision integral and further simplifying assumptions make the energy equation usable for simulation [Braginskii 1958, 1965]:

$$\frac{3n_e}{2}\left(\frac{\partial T_e}{\partial t}+\bar{u}_e\cdot\bar{\nabla}_{\bar{x}}T_e\right) = -n_eT_e\bar{\nabla}\cdot\bar{u}_e - \bar{\nabla}\cdot\bar{q}_e - \bar{\bar{\pi}}_e:\bar{\nabla}\bar{u}_e + Q_e; \tag{1.64}$$

$$T_e \equiv \tfrac{1}{3}m_e\left\langle\delta v_e^2\right\rangle,$$

$\bar{q}_e = \bar{q}_{eu}+\bar{q}_{eT}$, with $\bar{q}_{eu} \equiv 0.71n_eT_e\bar{u}_{e\parallel} + \dfrac{3n_eT_e}{2\omega_{ce}\tau_e}\hat{b}\times\bar{u}_e$ for $Z=1$,

$$\bar{q}_{eT} \equiv -\frac{3.16 n_e T_e \tau_e}{m_e}\bar{\nabla}_{\|}T_e - \frac{4.66 n_e T_e}{m_e \omega_{ce}^2 \tau_e}\bar{\nabla}_{\perp}T_e - \frac{5 c n_e T_e}{2eB}\hat{b}\times\bar{\nabla}T_e \text{ for } Z=1,$$

$$\pi_{eij} \equiv \int_{all\,\bar{v}} d\bar{v}\, v_i v_j f_e - n_e T_e \delta_{ij}, \quad Q_e = \eta_{\|}J_{\|}^2 + \eta_{\perp}J_{\perp}^2 + \frac{\bar{J}\cdot\bar{R}_{eT}}{e n_e} - \frac{3 m_e n_e}{m_i \tau_e}(T_e - T_i).$$

The subscripts $\|$ and $\perp$ refer to directions parallel and perpendicular to $\bar{B}$, respectively.

Appendix 1.A2
Magnetic vector potential and magnetic flux density rate equations in axisymmetric cylindrical coordinates

The electron fluid momentum equation, assuming quasineutrality, ignoring electron-plasma oscillations, and ignoring electron advection (see Appendix 1.A1) is Eq. 1.65.

$$\bar{E} = -\frac{\bar{\nabla}(\rho T_e)}{e\rho} - \frac{\bar{u}_e \times \bar{B}}{c} + \bar{\bar{\eta}} \cdot \bar{J} \tag{1.65}$$

This equation can be used to find the updated electric field, given an updated electron velocity and an updated magnetic field. With the use of $\bar{B} = \bar{\nabla} \times \bar{A}$ and $\bar{\nabla} \cdot \bar{A} = 0$, one also has for the electric field $\bar{E} = -\bar{\nabla}\varphi - \dot{\bar{A}}/c = \bar{E}_{irr} - \dot{\bar{A}}/c$, so that

$$\dot{\bar{A}} = c\bar{E}_{irr} + \frac{c\bar{\nabla}(\rho T_e)}{e\rho} + \bar{u}_e \times \bar{\nabla} \times \bar{A} - \frac{c^2 \bar{\bar{\eta}} \cdot \bar{\nabla} \times \bar{\nabla} \times \bar{A}}{4\pi}. \tag{1.66}$$

From this the various terms in 3-d become

$$\bar{\nabla}(\rho T_e) = \hat{r}\frac{\partial(\rho T_e)}{\partial r} + \hat{\theta}\frac{1}{r}\frac{\partial(\rho T_e)}{\partial \theta} + \hat{z}\frac{\partial(\rho T_e)}{\partial z}, \tag{1.67}$$

$$\bar{B} = \bar{\nabla} \times \bar{A} = \hat{r}\left(\frac{1}{r}\frac{\partial A_z}{\partial \theta} - \frac{\partial A_\theta}{\partial z}\right) + \hat{\theta}\left(\frac{\partial A_r}{\partial z} - \frac{\partial A_z}{\partial r}\right) + \hat{z}\frac{1}{r}\left(\frac{\partial(rA_\theta)}{\partial r} - \frac{\partial A_r}{\partial \theta}\right), \tag{1.68}$$

$$\begin{aligned} \bar{u}_e \times \bar{\nabla} \times \bar{A} = {} & \hat{r}\left(\frac{u_{e\theta}}{r}\left(\frac{\partial(rA_\theta)}{\partial r} - \frac{\partial A_r}{\partial \theta}\right) - u_{ez}\left(\frac{\partial A_r}{\partial z} - \frac{\partial A_z}{\partial r}\right)\right) \\ & + \hat{\theta}\left(u_{ez}\left(\frac{1}{r}\frac{\partial A_z}{\partial \theta} - \frac{\partial A_\theta}{\partial z}\right) - \frac{u_{er}}{r}\left(\frac{\partial(rA_\theta)}{\partial r} - \frac{\partial A_r}{\partial \theta}\right)\right) \\ & + \hat{z}\left(u_{er}\left(\frac{\partial A_r}{\partial z} - \frac{\partial A_z}{\partial r}\right) - u_{e\theta}\left(\frac{1}{r}\frac{\partial A_z}{\partial \theta} - \frac{\partial A_\theta}{\partial z}\right)\right). \end{aligned} \tag{1.69}$$

Choosing $\bar{\bar{\eta}}$ to be diagonal, $\bar{\bar{\eta}} = \hat{r}\hat{r}\eta_r + \hat{\theta}\hat{\theta}\eta_\theta + \hat{z}\hat{z}\eta_z$, the last term becomes

$$\begin{aligned}\bar{\bar{\eta}}\cdot\bar{\nabla}\times\bar{\nabla}\times\bar{A} = \hat{r}\,\eta_r & \left(\frac{1}{r^2}\frac{\partial}{\partial\theta}\left(\frac{\partial(rA_\theta)}{\partial r}-\frac{\partial A_r}{\partial\theta}\right) - \frac{\partial}{\partial z}\left(\frac{\partial A_r}{\partial z}-\frac{\partial A_z}{\partial r}\right)\right) \\ + \hat{\theta}\,\eta_\theta & \left(\frac{\partial}{\partial z}\left(\frac{1}{r}\frac{\partial A_z}{\partial\theta}-\frac{\partial A_\theta}{\partial z}\right) - \frac{\partial}{\partial r}\left(\frac{1}{r}\left(\frac{\partial(rA_\theta)}{\partial r}-\frac{\partial A_r}{\partial\theta}\right)\right)\right) \\ + \hat{z}\,\eta_z\frac{1}{r} & \left(\frac{\partial}{\partial r}\left(r\left(\frac{\partial A_r}{\partial z}-\frac{\partial A_z}{\partial r}\right)\right) - \frac{\partial}{\partial\theta}\left(\frac{1}{r}\frac{\partial A_z}{\partial\theta}-\frac{\partial A_\theta}{\partial z}\right)\right).\end{aligned} \tag{1.70}$$

The rate equations for each component of $\bar{A}$ are then

$$\begin{aligned}\dot{A}_r = {} & cE_{irr,\,r} + \frac{c}{e\rho}\frac{\partial(\rho T_e)}{\partial r} + \frac{u_{e\theta}}{r}\left(\frac{\partial(rA_\theta)}{\partial r}-\frac{\partial A_r}{\partial\theta}\right) - u_{ez}\left(\frac{\partial A_r}{\partial z}-\frac{\partial A_z}{\partial r}\right) \\ & - \frac{c^2\eta_r}{4\pi}\left(\frac{1}{r^2}\frac{\partial}{\partial\theta}\left(\frac{\partial(rA_\theta)}{\partial r}-\frac{\partial A_r}{\partial\theta}\right) - \frac{\partial}{\partial z}\left(\frac{\partial A_r}{\partial z}-\frac{\partial A_z}{\partial r}\right)\right),\end{aligned} \tag{1.71}$$

$$\begin{aligned}\dot{A}_\theta = {} & cE_{irr,\theta} + \frac{c}{e\rho r}\frac{\partial(\rho T_e)}{\partial\theta} + u_{ez}\left(\frac{1}{r}\frac{\partial A_z}{\partial\theta}-\frac{\partial A_\theta}{\partial z}\right) - \frac{u_{er}}{r}\left(\frac{\partial(rA_\theta)}{\partial r}-\frac{\partial A_r}{\partial\theta}\right) \\ & + \frac{c^2\eta_\theta}{4\pi}\left(\frac{\partial}{\partial z}\left(\frac{1}{r}\frac{\partial A_z}{\partial\theta}-\frac{\partial A_\theta}{\partial z}\right) - \frac{\partial}{\partial r}\left(\frac{1}{r}\left(\frac{\partial(rA_\theta)}{\partial r}-\frac{\partial A_r}{\partial\theta}\right)\right)\right),\end{aligned} \tag{1.72}$$

$$\begin{aligned}\dot{A}_z = {} & cE_{irr,\,z} + \frac{c}{e\rho}\frac{\partial(\rho T_e)}{\partial z} + u_{er}\left(\frac{\partial A_r}{\partial z}-\frac{\partial A_z}{\partial r}\right) - u_{e\theta}\left(\frac{1}{r}\frac{\partial A_z}{\partial\theta}-\frac{\partial A_\theta}{\partial z}\right) \\ & + \frac{c^2\eta_z}{4\pi}\frac{1}{r}\left(\frac{\partial}{\partial r}\left(r\left(\frac{\partial A_r}{\partial z}-\frac{\partial A_z}{\partial r}\right)\right) - \frac{\partial}{\partial\theta}\left(\frac{1}{r}\frac{\partial A_z}{\partial\theta}-\frac{\partial A_\theta}{\partial z}\right)\right).\end{aligned} \tag{1.73}$$

Taking $\dot{\bar{B}} = -c\bar{\nabla}\times\bar{E}$, we get

$$\dot{\bar{B}} = \frac{-c\bar{\nabla}\rho\times\bar{\nabla}(\rho T_e)}{e\rho^2} + \bar{\nabla}\times(\bar{u}_e\times\bar{B}) - \frac{c^2\bar{\nabla}\times(\bar{\bar{\eta}}\cdot\bar{\nabla}\times\bar{B})}{4\pi}. \tag{1.74}$$

In 3-d cylindrical coordinates, the terms on the right become

$$\begin{aligned}\bar{\nabla}\rho\times\bar{\nabla}(\rho T_e) = {} & \hat{r}\,\frac{\rho}{r}\left(\frac{\partial\rho}{\partial\theta}\frac{\partial T_e}{\partial z}-\frac{\partial\rho}{\partial z}\frac{\partial T_e}{\partial\theta}\right) + \hat{\theta}\,\rho\left(\frac{\partial\rho}{\partial z}\frac{\partial T_e}{\partial r}-\frac{\partial\rho}{\partial r}\frac{\partial T_e}{\partial z}\right) \\ & + \hat{z}\,\frac{\rho}{r}\left(\frac{\partial\rho}{\partial r}\frac{\partial T_e}{\partial\theta}-\frac{\partial\rho}{\partial\theta}\frac{\partial T_e}{\partial r}\right),\end{aligned} \tag{1.75}$$

$$\begin{aligned}\bar{\nabla}\times\left(\bar{u}_e\times\bar{B}\right) &= \hat{r}\left(\frac{1}{r}\frac{\partial\left(u_{er}B_\theta-u_{e\theta}B_r\right)}{\partial\theta}-\frac{\partial\left(u_{ez}B_r-u_{er}B_z\right)}{\partial z}\right)\\ &+\hat{\theta}\left(\frac{\partial\left(u_{e\theta}B_z-u_{ez}B_\theta\right)}{\partial z}-\frac{\partial\left(u_{er}B_\theta-u_{e\theta}B_r\right)}{\partial r}\right)\\ &+\hat{z}\left(\frac{\partial\left(ru_{ez}B_r-ru_{er}B_z\right)}{\partial r}-\frac{\partial\left(u_{e\theta}B_z-u_{ez}B_\theta\right)}{\partial\theta}\right),\end{aligned}\tag{1.76}$$

$$\bar{\eta}\cdot\bar{\nabla}\times\bar{B} = \hat{r}\,\eta_r\left(\frac{1}{r}\frac{\partial B_z}{\partial\theta}-\frac{\partial B_\theta}{\partial z}\right)+\hat{\theta}\,\eta_\theta\left(\frac{\partial B_r}{\partial z}-\frac{\partial B_z}{\partial r}\right)+\hat{z}\,\frac{\eta_z}{r}\left(\frac{\partial(rB_\theta)}{\partial r}-\frac{\partial B_r}{\partial\theta}\right),\tag{1.77}$$

$$\left(\bar{\nabla}\times\left(\bar{\bar{\eta}}\cdot\bar{\nabla}\times\bar{B}\right)\right)_r = \frac{1}{r}\frac{\partial}{\partial\theta}\left(\frac{\eta_z}{r}\left(\frac{\partial(rB_\theta)}{\partial r}-\frac{\partial B_r}{\partial\theta}\right)\right)-\frac{\partial}{\partial z}\left(\eta_\theta\left(\frac{\partial B_r}{\partial z}-\frac{\partial B_z}{\partial r}\right)\right),\tag{1.78}$$

$$\left(\bar{\nabla}\times\left(\bar{\bar{\eta}}\cdot\bar{\nabla}\times\bar{B}\right)\right)_\theta = \frac{\partial}{\partial z}\left(\eta_r\left(\frac{1}{r}\frac{\partial B_z}{\partial\theta}-\frac{\partial B_\theta}{\partial z}\right)\right)-\frac{\partial}{\partial r}\left(\frac{\eta_z}{r}\left(\frac{\partial(rB_\theta)}{\partial r}-\frac{\partial B_r}{\partial\theta}\right)\right),\tag{1.79}$$

$$\left(\bar{\nabla}\times\left(\bar{\bar{\eta}}\cdot\bar{\nabla}\times\bar{B}\right)\right)_z = \frac{1}{r}\frac{\partial}{\partial r}\left(\eta_\theta\left(\frac{\partial B_r}{\partial z}-\frac{\partial B_z}{\partial r}\right)\right)-\frac{1}{r}\frac{\partial}{\partial\theta}\left(\eta_r\left(\frac{1}{r}\frac{\partial B_z}{\partial\theta}-\frac{\partial B_\theta}{\partial z}\right)\right).\tag{1.80}$$

The rate equations for the components of $\bar{B}$ are then

$$\begin{aligned}\dot{B}_r &= -\frac{c}{e\rho r}\left(\frac{\partial\rho}{\partial\theta}\frac{\partial T_e}{\partial z}-\frac{\partial\rho}{\partial z}\frac{\partial T_e}{\partial\theta}\right)+\frac{1}{r}\frac{\partial\left(u_{er}B_\theta-u_{e\theta}B_r\right)}{\partial\theta}-\frac{\partial\left(u_{ez}B_r-u_{er}B_z\right)}{\partial z}\\ &-\frac{c^2}{4\pi}\left(\frac{1}{r}\frac{\partial}{\partial\theta}\left(\frac{\eta_z}{r}\left(\frac{\partial(rB_\theta)}{\partial r}-\frac{\partial B_r}{\partial\theta}\right)\right)-\frac{\partial}{\partial z}\left(\eta_\theta\left(\frac{\partial B_r}{\partial z}-\frac{\partial B_z}{\partial r}\right)\right)\right),\end{aligned}\tag{1.81}$$

$$\begin{aligned}\dot{B}_\theta &= -\frac{c}{e\rho}\left(\frac{\partial\rho}{\partial z}\frac{\partial T_e}{\partial r}-\frac{\partial\rho}{\partial r}\frac{\partial T_e}{\partial z}\right)+\frac{\partial\left(u_{e\theta}B_z-u_{ez}B_\theta\right)}{\partial z}-\frac{\partial\left(u_{er}B_\theta-u_{e\theta}B_r\right)}{\partial r}\\ &-\frac{c^2}{4\pi}\left(\frac{\partial}{\partial z}\left(\eta_r\left(\frac{1}{r}\frac{\partial B_z}{\partial\theta}-\frac{\partial B_\theta}{\partial z}\right)\right)-\frac{\partial}{\partial r}\left(\frac{\eta_z}{r}\left(\frac{\partial(rB_\theta)}{\partial r}-\frac{\partial B_r}{\partial\theta}\right)\right)\right),\end{aligned}\tag{1.82}$$

$$\begin{aligned}\dot{B}_z &= -\frac{c}{e\rho r}\left(\frac{\partial\rho}{\partial r}\frac{\partial T_e}{\partial\theta}-\frac{\partial\rho}{\partial\theta}\frac{\partial T_e}{\partial r}\right)+\frac{\partial\left(ru_{ez}B_r-ru_{er}B_z\right)}{\partial r}-\frac{\partial\left(u_{e\theta}B_z-u_{ez}B_\theta\right)}{\partial\theta}\\ &-\frac{c^2}{4\pi r}\left(\frac{\partial}{\partial r}\left(\eta_\theta\left(\frac{\partial B_r}{\partial z}-\frac{\partial B_z}{\partial r}\right)\right)-\frac{\partial}{\partial\theta}\left(\eta_r\left(\frac{1}{r}\frac{\partial B_z}{\partial\theta}-\frac{\partial B_\theta}{\partial z}\right)\right)\right).\end{aligned}\tag{1.83}$$

Assuming axisymmetry for 2-D r-z problems, and looking at the equation for the E-field:

$$\bar{E}_{irr} = \hat{r}\frac{\partial\varphi}{\partial r} + \hat{\theta}\frac{1}{r}\frac{\partial\varphi}{\partial\theta} + \hat{z}\frac{\partial\varphi}{\partial z} \Rightarrow E_{irr,\theta} = 0. \qquad (1.84)$$

Each of the terms in the equations for the magnetic vector potential and the magnetic flux density simplify, giving

$$\dot{A}_\theta = \frac{c^2\eta_\theta}{4\pi}\left(\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r}(rA_\theta)\right)+\frac{\partial^2 A_\theta}{\partial z^2}\right) - u_{ez}\frac{\partial A_\theta}{\partial z} - \frac{u_{er}}{r}\frac{\partial}{\partial r}(rA_\theta), \qquad (1.85)$$

$$\begin{aligned}\dot{B}_\theta &= \left[\left(\frac{c^2}{4\pi}\frac{\partial}{\partial r}\left(\frac{\eta_z}{r}\frac{\partial}{\partial r}r\right)-\frac{\partial}{\partial r}u_{er}\right) + \left(\frac{c^2}{4\pi}\frac{\partial}{\partial z}\left(\eta_r\frac{\partial}{\partial z}\right)-\frac{\partial}{\partial z}u_{ez}\right)\right]B_\theta \\ &+ \frac{\partial}{\partial z}(u_{e\theta}B_z) + \frac{\partial}{\partial r}(u_{e\theta}B_r) + \frac{c}{e\rho}\left(\frac{\partial\rho}{\partial r}\frac{\partial T_e}{\partial z}-\frac{\partial\rho}{\partial z}\frac{\partial T_e}{\partial r}\right).\end{aligned} \qquad (1.86)$$

Multiplying A_θ by r gives

$$r\dot{A}_\theta = \left[\left(\frac{c^2\eta_\theta}{4\pi}r\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r}\right)-u_{er}\frac{\partial}{\partial r}\right) + \left(\frac{c^2\eta_\theta}{4\pi}\frac{\partial^2}{\partial z^2}-u_{ez}\frac{\partial}{\partial z}\right)\right]rA_\theta. \qquad (1.87)$$

Appendix 1.A3
Finite difference equations used for the electron fluid and the electric and magnetic fields

Take ρ_c to be the cut-off plasma density. Then $\rho^{1*} = \rho^1$ when the ion density is greater than ρ_c, while $\rho^{1*} = \rho_c$ when the ion density is less than ρ_c. Points at which $\rho^{1*} < \rho_c$ are called *vacuum points*. All of the other points are called *plasma points*. Typically, ρ_c is less than a few 10^{12} per cubic centimeter. We have, taking $A = rA_\theta$ and $B = B_\theta$, the set of equations Eqs. 1.88a-t. When ADI is used to relax on the equations $(\nabla^2 \vec{A})_\theta = 0$ and $(\nabla^2 \vec{B})_\theta = 0$ at vacuum points, a relaxation parameter $(\Delta t / 2)'$ is used which has little to do with the physical time step of the simulation; it is chosen to relax toward the solutions in a small number of physical time steps.

Update the z-component of the electron velocity:

$$u_r^1 = u_{ir}^1 + \frac{c}{4\pi e \rho^{1*}} \frac{B_{01}^0 - B_{0\text{-}1}^0}{2\Delta z} \tag{1.88a}$$

R-pass on azimuthal magnetic vector potential:

Plasma:

$$\frac{A^{1/2} - A^0}{\Delta t / 2} = \frac{c^2 \eta}{4\pi} \frac{r}{\Delta r^2} \left(\frac{A_{10}^{1/2} - A^{1/2}}{r_{1/2}} - \frac{A^{1/2} - A_{\text{-}10}^{1/2}}{r_{\text{-}1/2}} \right) - u_r^1 \frac{A_{10}^{1/2} - A_{\text{-}10}^{1/2}}{2\Delta r} + \frac{c^2 \eta}{4\pi} \frac{A_{01}^0 - 2A^0 + A_{0\text{-}1}^0}{\Delta z^2} - u_z^0 \frac{A_{01}^0 - A_{0\text{-}1}^0}{2\Delta z} \tag{1.88b.1}$$

Vacuum:

$$\frac{A^{1/2} - A^0}{(\Delta t / 2)'} = \frac{r}{\Delta r^2} \left(\frac{A_{10}^{1/2} - A^{1/2}}{r_{1/2}} - \frac{A^{1/2} - A_{\text{-}10}^{1/2}}{r_{\text{-}1/2}} \right) + \frac{A_{01}^0 - 2A^0 + A_{0\text{-}1}^0}{\Delta z^2} \tag{1.88b.2}$$

Z-pass on azimuthal magnetic vector potential:

Plasma:
$$\frac{\hat{A}-A^{1/2}}{\Delta t/2} = \frac{c^2\eta}{4\pi}\frac{r}{\Delta r^2}\left(\frac{A_{10}^{1/2}-A^{1/2}}{r_{1/2}}-\frac{A^{1/2}-A_{-10}^{1/2}}{r_{-1/2}}\right) - u_r^1\frac{A_{10}^{1/2}-A_{-10}^{1/2}}{2\Delta r} + \frac{c^2\eta}{4\pi}\frac{\hat{A}_{01}-2\hat{A}+\hat{A}_{0-1}}{\Delta z^2} - u_z^0\frac{\hat{A}_{01}-\hat{A}_{0-1}}{2\Delta z} \qquad \text{(c.1)}$$

Vacuum:
$$\frac{\hat{A}-A^{1/2}}{(\Delta t/2)'} = \frac{r}{\Delta r^2}\left(\frac{A_{10}^{1/2}-A^{1/2}}{r_{1/2}}-\frac{A^{1/2}-A_{-10}^{1/2}}{r_{-1/2}}\right) + \frac{\hat{A}_{01}-2\hat{A}+\hat{A}_{0-1}}{\Delta z^2} \qquad \text{(c.2)}$$

Vacuum relaxation of azimuthal magnetic vector potential:

$$\frac{A^{1/2}-A^0}{(\Delta t/2)'} = \frac{r}{\Delta r^2}\left(\frac{A_{10}^{1/2}-A^{1/2}}{r_{1/2}}-\frac{A^{1/2}-A_{-10}^{1/2}}{r_{-1/2}}\right) + \frac{A_{01}^0-2A^0+A_{0-1}^0}{\Delta z^2} \qquad \text{(d.1)}$$

$$\frac{\hat{A}-A^{1/2}}{(\Delta t/2)'} = \frac{r}{\Delta r^2}\left(\frac{A_{10}^{1/2}-A^{1/2}}{r_{1/2}}-\frac{A^{1/2}-A_{-10}^{1/2}}{r_{-1/2}}\right) + \frac{\hat{A}_{01}-2\hat{A}+\hat{A}_{0-1}}{\Delta z^2} \qquad \text{(d.2)}$$

Get r- and z-components of magnetic field density from curl of magnetic vector potential:

$$\hat{B}_r = -\frac{\hat{A}_{01}-\hat{A}_{0-1}}{2r\Delta z}, \qquad \hat{B}_z = \frac{\hat{A}_{10}-\hat{A}_{-10}}{2r\Delta r} \qquad \text{(e.1,2)}$$

Estimate azimuthal electron velocity:

$$\hat{u} = u_i^1 + \frac{c}{4\pi e\rho^{1*}}\left(\frac{1}{\Delta r^2}\left(\frac{\hat{A}_{10}-\hat{A}}{r_{1/2}}-\frac{\hat{A}-\hat{A}_{-10}}{r_{-1/2}}\right) + \frac{1}{r}\frac{\hat{A}_{01}-2\hat{A}+\hat{A}_{0-1}}{\Delta z^2}\right) \qquad \text{(f)}$$

Line integrations for entrained vacuum:

Method 1: In case a void is newly formed at the 0th time level, find a mean rB product over the area of each void:

$$k^0 \equiv \int_S drdz\, rB^0 \Big/ \int_S drdz. \qquad \text{(g.1)}$$

Reset $B^0 = k^0/r$ in each void, and perform line integration of the electric field to update the azimuthal magnetic flux density:

$$\int_S drdz\, \dot{B}^{1/2} = \dot{k}^{1/2}\int_S drdz\, r^{-1} = \oint_{\partial S} d\vec{\ell}\cdot\vec{E}^0. \qquad \text{(g.2)}$$

Use $B^1 = B^0 + \Delta t\,\dot{k}^{1/2}/r$ and $B^{1/2} = \frac{1}{2}\left(B^0+B^1\right)$ for points in the vacuum region.

Method 2: Perform line averaging of the azimuthal magnetic flux density around each void:

$$k^0 = \oint_{\partial S} d\ell \, rB^0 \Big/ \oint_{\partial S} d\ell . \tag{h.1}$$

Reset $B^0 = k^0/r$ before proceeding with combined plasma-vac R-pass on B.

R-pass on azimuthal magnetic flux density:

Donor cell differencing notation with angular braces $\langle\;\rangle$, is explained in Appendix 1.A4.

Method 1: Enforce Dirichlet conditions at all vacuum points, i.e. $B^{1/2} = k^{1/2}/r$.

$$\begin{aligned}\frac{B^{1/2}-B^0}{\Delta t/2} = {} & \frac{c^2}{4\pi\Delta r^2}\left(\frac{\eta_{z\frac{1}{2}0}(r_1 B^{1/2}_{10} - rB^{1/2})}{r_{1/2}} - \frac{\eta_{z-\frac{1}{2}0}(rB^{1/2} - r_{-1}B^{1/2}_{-10})}{r_{-1/2}}\right) - \\ & \frac{u^1_{r\frac{1}{2}0}\langle B\rangle^{1/2}_{\frac{1}{2}0} - u^1_{r-\frac{1}{2}0}\langle B\rangle^{1/2}_{-\frac{1}{2}0}}{\Delta r} + \frac{c^2}{4\pi}\frac{\eta_{r0\frac{1}{2}}B^0_{01} - (\eta_{r0\frac{1}{2}} + \eta_{r0-\frac{1}{2}})B^0 + \eta_{r0-\frac{1}{2}}B^0_{0-1}}{\Delta z^2} - \\ & \frac{u^0_{z0\frac{1}{2}}\langle B\rangle^0_{0\frac{1}{2}} - u^0_{z0-\frac{1}{2}}\langle B\rangle^0_{0-\frac{1}{2}}}{\Delta z} + \frac{\hat{u}_{01}\hat{B}_{z01} - \hat{u}_{0-1}\hat{B}_{z0-1}}{2\Delta z} + \frac{\hat{u}_{10}\hat{B}_{r10} - \hat{u}_{-10}\hat{B}_{r-10}}{2\Delta r} + \\ & \frac{c}{4e\rho^{1/2}}\left(\frac{\rho^{1/2}_{10} - \rho^{1/2}_{-10}}{\Delta r}\left(\frac{T^0_{01} - T^0_{0-1}}{\Delta z}\right) - \frac{\rho^{1/2}_{01} - \rho^{1/2}_{0-1}}{\Delta z}\left(\frac{T^0_{10} - T^0_{-10}}{\Delta r}\right)\right)\end{aligned} \tag{i.1}$$

Method 2: Perform combined plasma-vacuum R-pass on B:

Plasma:

$$\begin{aligned}\frac{B^{1/2}-B^0}{\Delta t/2} = {} & \frac{c^2}{4\pi\Delta r^2}\left(\frac{\eta_{z\frac{1}{2}0}(r_1 B^{1/2}_{10} - rB^{1/2})}{r_{1/2}} - \frac{\eta_{z-\frac{1}{2}0}(rB^{1/2} - r_{-1}B^{1/2}_{-10})}{r_{-1/2}}\right) - \\ & \frac{u^1_{r\frac{1}{2}0}\langle B\rangle^{1/2}_{\frac{1}{2}0} - u^1_{r-\frac{1}{2}0}\langle B\rangle^{1/2}_{-\frac{1}{2}0}}{\Delta r} + \frac{c^2}{4\pi}\frac{\eta_{r0\frac{1}{2}}B^0_{01} - (\eta_{r0\frac{1}{2}} + \eta_{r0-\frac{1}{2}})B^0 + \eta_{r0-\frac{1}{2}}B^0_{0-1}}{\Delta z^2} - \\ & \frac{u^0_{z0\frac{1}{2}}\langle B\rangle^0_{0\frac{1}{2}} - u^0_{z0-\frac{1}{2}}\langle B\rangle^0_{0-\frac{1}{2}}}{\Delta z} + \frac{\hat{u}_{01}\hat{B}_{z01} - \hat{u}_{0-1}\hat{B}_{z0-1}}{2\Delta z} + \frac{\hat{u}_{10}\hat{B}_{r10} - \hat{u}_{-10}\hat{B}_{r-10}}{2\Delta r} + \\ & \frac{c}{4e\rho^{1/2}}\left(\frac{\rho^{1/2}_{10} - \rho^{1/2}_{-10}}{\Delta r}\left(\frac{T^0_{01} - T^0_{0-1}}{\Delta z}\right) - \frac{\rho^{1/2}_{01} - \rho^{1/2}_{0-1}}{\Delta z}\left(\frac{T^0_{10} - T^0_{-10}}{\Delta r}\right)\right)\end{aligned} \tag{i.2}$$

Vacuum:

$$\frac{B^{1/2}-B^0}{(\Delta t/2)'} = \frac{1}{\Delta r^2}\left(\frac{r_1 B^{1/2}_{10} - rB^{1/2}}{r_{1/2}} - \frac{rB^{1/2} - r_{-1}B^{1/2}_{-10}}{r_{-1/2}}\right) + \frac{B^0_{01} - 2B^0 + B^0_{0-1}}{\Delta z^2} \tag{i.3}$$

Update z-component of electron velocity:

$$u_z^1 = u_{iz}^1 - \frac{c}{4\pi e\rho^{1*}}\frac{r_1 B_{10}^{1/2} - r_{-1}B_{-10}^{1/2}}{2r\Delta r} \tag{j}$$

Z-pass on azimuthal magnetic flux density:

Method 1: Enforce Dirichlet conditions at all vacuum points, i.e. $B^1 = k^1/r$.

$$\begin{aligned}\frac{B^1-B^{1/2}}{\Delta t/2} = {} & \frac{c^2}{4\pi\Delta r^2}\left(\frac{\eta_{z\frac12 0}(r_1B_{10}^{1/2}-rB^{1/2})}{r_{1/2}} - \frac{\eta_{z-\frac12 0}(rB^{1/2}-r_{-1}B_{-10}^{1/2})}{r_{-1/2}}\right) - \\ & \frac{u^1_{r\frac12 0}\langle B\rangle^{1/2}_{\frac12 0} - u^1_{r-\frac12 0}\langle B\rangle^{1/2}_{-\frac12 0}}{\Delta r} + \frac{c^2}{4\pi}\frac{\eta_{r0\frac12}B^1_{01} - (\eta_{r0\frac12}+\eta_{r0-\frac12})B^1 + \eta_{r0-\frac12}B^1_{0-1}}{\Delta z^2} - \\ & \frac{u^1_{z0\frac12}\langle B\rangle^1_{0\frac12} - u^1_{z0-\frac12}\langle B\rangle^1_{0-\frac12}}{\Delta z} + \frac{\hat u_{01}\hat B_{z01} - \hat u_{0-1}\hat B_{z0-1}}{2\Delta z} + \frac{\hat u_{10}\hat B_{r10}-\hat u_{-10}\hat B_{r-10}}{2\Delta r} + \\ & \frac{c}{4e\rho^{1/2}}\left(\frac{\rho^{1/2}_{10}-\rho^{1/2}_{-10}}{\Delta r}\left(\frac{T^0_{01}-T^0_{0-1}}{\Delta z}\right) - \frac{\rho^{1/2}_{01}-\rho^{1/2}_{0-1}}{\Delta z}\left(\frac{T^0_{10}-T^0_{-10}}{\Delta r}\right)\right)\end{aligned} \tag{k.1}$$

Method 2: Perform combined plasma-vacuum Z-pass on B:

Plasma:

$$\begin{aligned}\frac{B^1-B^{1/2}}{\Delta t/2} = {} & \frac{c^2}{4\pi\Delta r^2}\left(\frac{\eta_{z\frac12 0}(r_1B_{10}^{1/2}-rB^{1/2})}{r_{1/2}} - \frac{\eta_{z-\frac12 0}(rB^{1/2}-r_{-1}B_{-10}^{1/2})}{r_{-1/2}}\right) - \\ & \frac{u^1_{r\frac12 0}\langle B\rangle^{1/2}_{\frac12 0} - u^1_{r-\frac12 0}\langle B\rangle^{1/2}_{-\frac12 0}}{\Delta r} + \frac{c^2}{4\pi}\frac{\eta_{r0\frac12}B^1_{01} - (\eta_{r0\frac12}+\eta_{r0-\frac12})B^1 + \eta_{r0-\frac12}B^1_{0-1}}{\Delta z^2} - \\ & \frac{u^1_{z0\frac12}\langle B\rangle^1_{0\frac12} - u^1_{z0-\frac12}\langle B\rangle^1_{0-\frac12}}{\Delta z} + \frac{\hat u_{01}\hat B_{z01} - \hat u_{0-1}\hat B_{z0-1}}{2\Delta z} + \frac{\hat u_{10}\hat B_{r10}-\hat u_{-10}\hat B_{r-10}}{2\Delta r} + \\ & \frac{c}{4e\rho^{1/2}}\left(\frac{\rho^{1/2}_{10}-\rho^{1/2}_{-10}}{\Delta r}\left(\frac{T^0_{01}-T^0_{0-1}}{\Delta z}\right) - \frac{\rho^{1/2}_{01}-\rho^{1/2}_{0-1}}{\Delta z}\left(\frac{T^0_{10}-T^0_{-10}}{\Delta r}\right)\right)\end{aligned} \tag{k.2}$$

Vacuum:

$$\frac{B^1-B^{1/2}}{(\Delta t/2)} = \frac{1}{\Delta r^2}\left(\frac{r_1B_{10}^{1/2}-rB^{1/2}}{r_{1/2}} - \frac{rB^{1/2}-r_{-1}B_{-10}^{1/2}}{r_{-1/2}}\right) + \frac{B^1_{01}-2B^1+B^1_{0-1}}{\Delta z^2} \tag{k.3}$$

Z-pass on azimuthal magnetic vector potential:

Plasma: $$\frac{A^1-A^{1/2}}{\Delta t/2} = \frac{c^2\eta}{4\pi}\frac{r}{\Delta r^2}\left(\frac{A^{1/2}_{10}-A^{1/2}}{r_{1/2}}-\frac{A^{1/2}-A^{1/2}_{-10}}{r_{-1/2}}\right) - u^1_r\frac{A^{1/2}_{10}-A^{1/2}_{-10}}{2\Delta r} + \frac{c^2\eta}{4\pi}\frac{A^1_{01}-2A^1+A^1_{0-1}}{\Delta z^2} - u^1_z\frac{A^1_{01}-A^1_{0-1}}{2\Delta z} \qquad \text{(m.1)}$$

Vacuum: $$\frac{A^1-A^{1/2}}{(\Delta t/2)'} = \frac{r}{\Delta r^2}\left(\frac{A^{1/2}_{10}-A^{1/2}}{r_{1/2}}-\frac{A^{1/2}-A^{1/2}_{-10}}{r_{-1/2}}\right) + \frac{A^1_{01}-2A^1+A^1_{0-1}}{\Delta z^2} \qquad \text{(m.2)}$$

Vacuum relaxation on azimuthal magnetic vector potential:

$$\frac{A^{1/2}-A^0}{(\Delta t/2)'} = \frac{r}{\Delta r^2}\left(\frac{A^{1/2}_{10}-A^{1/2}}{r_{1/2}}-\frac{A^{1/2}-A^{1/2}_{-10}}{r_{-1/2}}\right) + \frac{A^0_{01}-2A^0+A^0_{0-1}}{\Delta z^2} \qquad \text{(n.1)}$$

$$\frac{A^1-A^{1/2}}{(\Delta t/2)'} = \frac{r}{\Delta r^2}\left(\frac{A^{1/2}_{10}-A^{1/2}}{r_{1/2}}-\frac{A^{1/2}-A^{1/2}_{-10}}{r_{-1/2}}\right) + \frac{A^1_{01}-2A^1+A^1_{0-1}}{\Delta z^2} \qquad \text{(n.2)}$$

Get updated r- and z-components of magnetic flux density from curl of magnetic vector potential:

$$B^1_r = -\frac{A^1_{01}-A^1_{0-1}}{2r\Delta z}, \qquad B^1_z = \frac{A^1_{10}-A^1_{-10}}{2r\Delta r} \qquad \text{(p.1, 2)}$$

Update azimuthal electron velocity:

$$u^1 = u^1_i + \frac{c}{4\pi e\rho^{1*}}\left(\frac{1}{\Delta r^2}\left(\frac{A^1_{10}-A^1}{r_{1/2}}-\frac{A^1-A^1_{-10}}{r_{-1/2}}\right) + \frac{1}{r}\frac{A^1_{01}-2A^1+A^1_{0-1}}{\Delta z^2}\right) \qquad \text{(q)}$$

Crudely apply special relativity to prevent runaway electron velocity:

Assume $\vec{J} = \rho e\left(\vec{u}_i-\gamma_e^2\vec{u}_e\right)$. Then $\gamma_e^2\vec{u}_e = \vec{u}_i - \frac{c}{4\pi e\rho}\vec{\nabla}\times\vec{B} \equiv \vec{\alpha}$. To get γ_e^2, use

$$\alpha^2 = \gamma_e^4 u_e^2 = \gamma_e^4 c^2\left(1-1/\gamma_e^2\right) = c^2\left(\gamma_e^4-\gamma_e^2\right). \qquad \text{(r.1)}$$

Then $\vec{u}_e = \gamma_e^{-2}\vec{\alpha}$. (r.2)

Update the electric field:

Plasma: $$E^1_r = -\frac{1}{e\rho^1}\frac{\rho^1_{10}T^1_{10}-\rho^1_{-10}T^1_{-10}}{2\Delta r} - \frac{1}{c}\left(u^1B^1_z-u^1_zB^1\right) + \eta_r e\rho^1\left(u^1_{ir}-\left(\gamma_e^2\right)^1u^1_r\right) \qquad \text{(s.1)}$$

$$E^1 = -\frac{1}{c}\left(u^1_zB^1_r-u^1_rB^1_z\right) + \eta e\rho^1\left(u^1_i-\left(\gamma_e^2\right)^1u^1\right) \qquad \text{(s.2)}$$

$$E_z^1 = -\frac{1}{e\rho^1}\frac{\rho_{01}^1 T_{01}^1 - \rho_{0-1}^1 T_{0-1}^1}{2\Delta z} - \frac{1}{c}\left(u_r^1 B^1 - u^1 B_r^1\right) + \eta_z e\rho^1\left(u_{iz}^1 - \left(\gamma_e^2\right)^1 u_z^1\right) \tag{s.3}$$

Vacuum:
$$E_r^1 = -\frac{1}{e\rho_c}\frac{\rho_{10}^{1*} T_{10}^1 - \rho_{-10}^{1*} T_{-10}^1}{2\Delta r} - \frac{1}{c}\left(u^1 B_z^1 - u_z^1 B^1\right) \tag{s.4}$$

$$E^1 = -\frac{1}{c}\left(u_z^1 B_r^1 - u_r^1 B_z^1\right) \tag{s.5}$$

$$E_z^1 = -\frac{1}{e\rho_c}\frac{\rho_{01}^{1*} T_{01}^1 - \rho_{0-1}^{1*} T_{0-1}^1}{2\Delta z} - \frac{1}{c}\left(u_r^1 B^1 - u^1 B_r^1\right) \tag{s.6}$$

<u>Alternative vacuum update for electric field:</u>

For $i \in \{r, \theta\}$:

$$\frac{E_i^{1/2} - E_i^0}{(\Delta t/2)_i} = \frac{1}{\Delta r^2}\left(\frac{r_1 E_{i10}^{1/2} - r E_i^{1/2}}{r_{1/2}} - \frac{r E_i^{1/2} - r_{-1} E_{i-10}^{1/2}}{r_{-1/2}}\right) + \frac{E_{i01}^0 - 2E_i^0 + E_{i0-1}^0}{\Delta z^2} \tag{1.88t.1}$$

$$\frac{E_i^1 - E_i^{1/2}}{(\Delta t/2)_i} = \frac{1}{\Delta r^2}\left(\frac{r_1 E_{i10}^{1/2} - r E_i^{1/2}}{r_{1/2}} - \frac{r E_i^{1/2} - r_{-1} E_{i-10}^{1/2}}{r_{-1/2}}\right) + \frac{E_{i01}^1 - 2E_i^1 + E_{i0-1}^1}{\Delta z^2} \tag{1.88t.2}$$

$$\frac{E_z^{1/2} - E_z^0}{(\Delta t/2)_z} = \frac{1}{r\Delta r^2}\left(r_{1/2}\left(E_{z10}^{1/2} - E_z^{1/2}\right) - r_{-1/2}\left(E_z^{1/2} - E_{z-10}^{1/2}\right)\right) + \frac{E_{z01}^0 - 2E_z^0 + E_{z0-1}^0}{\Delta z^2} \tag{1.88t.3}$$

$$\frac{E_z^1 - E_z^{1/2}}{(\Delta t/2)_z} = \frac{1}{r\Delta r^2}\left(r_{1/2}\left(E_{z10}^{1/2} - E_z^{1/2}\right) - r_{-1/2}\left(E_z^{1/2} - E_{z-10}^{1/2}\right)\right) + \frac{E_{z01}^1 - 2E_z^1 + E_{z0-1}^1}{\Delta z^2} \tag{1.88t.4}$$

Boundary considerations

When Equations 1.88t.1-t.4 are used to find the electric fields in the extrained vacuum region, boundary conditions are needed at the maximum radius, or wall radius, r_w. For boundary conditions on E_r and E_θ, it makes sense to use Neumann boundary conditions $E_{r10} = r_{-1}E_{r\text{-}10}/r_1$ and $E_{\theta 10} = r_{-1}E_{\theta\text{-}10}/r_1$. These boundary conditions are consistent with an inverse radial dependence of E_r and E_θ. An inverse radial dependence of E_r could mock up the case of a cylindrical line charge density completely within the radius r_w, whereas an inverse radial dependence of E_θ could mock up a cylindrically symmetric time-varying axial magnetic flux completely within the radius r_w. For a boundary condition on E_z, it makes sense to use the Neumann boundary condition $E_{z01} = E_{z0\text{-}1}$. Such a boundary condition would be consistent with a uniform axial electric field.

For simulations that allow axial magnetic flux to leave the spatial region, an external Neumann radial boundary condition on A_θ is useful. This Neumann condition amounts to the assignment $A_{10} = A_{-10}$ in the above equations for advancement of A_θ. Remember that $rA_\theta \rightarrow A$ in the above finite-difference equations. It also makes sense to assume $A_{10} = A_{-10}$ for purposes of calculating $u_{e\theta}$.

Appendix 1.A4
Finite difference equations used for the ion fluid

For each advected quantity q, donor cell differencing is specified by the notation in Equations 1.89 and 1.90 [Hirt 1968].

$$\frac{1}{r}\frac{\partial}{\partial r}(ru_r q) \doteq \frac{1}{r\Delta r}\left(r_{1/2}u_{r1/2,0}\langle q\rangle_{1/2,0} - r_{-1/2}u_{r-1/2,0}\langle q\rangle_{-1/2,0}\right), \qquad (1.89)$$

$$\frac{\partial}{\partial z}(u_z q) \doteq \frac{1}{\Delta z}\left(u_{z0,1/2}\langle q\rangle_{0,1/2} - u_{z0,-1/2}\langle q\rangle_{0,-1/2}\right). \qquad (1.90)$$

The quantities in angular brackets depend on the advection velocity components.

$$\langle q\rangle_{i+1/2,0} = \begin{cases} q_{i,0}, & \text{if } u_{r\,i+1/2,0} \geq 0 \\ q_{i+1,0}, & \text{if } u_{r\,i+1/2,0} < 0 \end{cases} \quad \text{for } i = -1, 0\,. \qquad (1.91)$$

$$\langle q\rangle_{0,j+1/2} = \begin{cases} q_{0,j}, & \text{if } u_{z0,j+1/2} \geq 0 \\ q_{0,j+1}, & \text{if } u_{z0,j+1/2} < 0 \end{cases} \quad \text{for } j = -1, 0\,. \qquad (1.92)$$

Quantities with half-integral subscripts are linearly interpolated: for example, : $u_{r\pm1/2,0} \equiv \frac{1}{2}(u_{r\pm1,0} + u_{r0,0})$, $u_{z0,\pm1/2} \equiv \frac{1}{2}(u_{z0,\pm1} + u_{z0,0})$.

Inside the plasma, away from boundaries, the ion fluid advance is performed with Equations 1.93-1.98.

$$\rho^{1/2} = \rho - \frac{\Delta t}{2}\left(\begin{array}{l} \frac{1}{r\Delta r}\left(r_{1/2}u_{r1/2,0}\langle\rho\rangle_{1/2,0} - r_{-1/2}u_{r-1/2,0}\langle\rho\rangle_{-1/2,0}\right) + \\ \frac{1}{\Delta z}\left(u_{z0,1/2}\langle\rho\rangle_{0,1/2} - u_{z0,-1/2}\langle\rho\rangle_{0,-1/2}\right) \end{array}\right), \qquad (1.93)$$

$$f_r^{1/2} = f_r^{-1/2} - \Delta t\left(\begin{array}{l} \frac{1}{r\Delta r}\left(r_{1/2}u_{r1/2,0}\,\langle f_r\rangle_{1/2,0} - r_{-1/2}u_{r-1/2,0}\,\langle f_r\rangle_{-1/2,0}\right) + \\ \frac{1}{\Delta z}\left(u_{r0,1/2}\,\langle f_r\rangle_{0,1/2} - u_{r0,-1/2}\,\langle f_r\rangle_{0,-1/2}\right) \end{array}\right) + \qquad (1.94)$$

$$\Delta t\frac{f_\theta u_\theta}{r} - \frac{\Delta t}{2m\Delta r}\left(\rho_{1,0}T_{1,0} - \rho_{-1,0}T_{-1,0}\right) + \frac{q\Delta t}{m}\left(\rho E_r + \frac{1}{c}\left(f_\theta B_z - f_z B_\theta\right)\right),$$

$$f_\theta^{1/2} = f_\theta^{-1/2} - \Delta t\begin{pmatrix} \frac{1}{r\Delta r}\left(r_{1/2}u_{r1/2,0}\left\langle f_\theta\right\rangle_{1/2,0} - r_{-1/2}u_{r-1/2,0}\left\langle f_\theta\right\rangle_{-1/2,0}\right) + \\ \frac{1}{\Delta z}\left(u_{z0,1/2}\left\langle f_\theta\right\rangle_{0,1/2} - u_{z0,-1/2}\left\langle f_\theta\right\rangle_{0,-1/2}\right) \end{pmatrix} - \Delta t\frac{f_\theta u_\theta}{r} + \frac{q\Delta t}{m}\left(\rho E_\theta + \frac{1}{c}\left(f_z B_r - f_r B_z\right)\right), \tag{1.95}$$

$$f_z^{1/2} = f_z^{-1/2} - \Delta t\begin{pmatrix} \frac{1}{r\Delta r}\left(r_{1/2}u_{r1/2,0}\left\langle f_z\right\rangle_{1/2,0} - r_{-1/2}u_{r-1/2,0}\left\langle f_z\right\rangle_{-1/2,0}\right) + \\ \frac{1}{\Delta z}\left(u_{z0,1/2}\left\langle f_z\right\rangle_{0,1/2} - u_{z0,-1/2}\left\langle f_z\right\rangle_{0,-1/2}\right) \end{pmatrix} - \frac{\Delta t}{2m\Delta z}\left(\rho_{0,1}T_{0,1} - \rho_{0,-1}T_{0,-1}\right) + \frac{q\Delta t}{m}\left(\rho E_z + \frac{1}{c}\left(f_r B_\theta - f_\theta B_r\right)\right), \tag{1.96}$$

$$T^{1/2} = T - \frac{\Delta t}{2}\begin{pmatrix} \frac{1}{r\Delta r}\left(r_{1/2}u_{r1/2,0}\langle T\rangle_{1/2,0} - r_{-1/2}u_{r-1/2,0}\langle T\rangle_{-1/2,0}\right) + \\ \frac{1}{\Delta z}\left(u_{z0,1/2}\langle T\rangle_{0,1/2} - u_{z0,-1/2}\langle T\rangle_{0,-1/2}\right) \end{pmatrix} + \frac{\Delta t T}{3}\left(\frac{1}{r\Delta r}\left(r_{1/2}u_{r1/2,0} - r_{-1/2}u_{r-1/2,0}\right) + \frac{1}{2\Delta z}\left(u_{z0,1/2} - u_{z0,-1/2}\right)\right), \tag{1.97}$$

$$\bar{u}^{1/2} = \bar{f}^{1/2}/\rho^{1/2}. \tag{1.98}$$

In Eqs. 1.93-1.98, ρ is the ion fluid density, $\bar{u}$ is the ion fluid drift velocity, $\bar{f} \equiv \rho\bar{u}$ is the ion fluid flux density, and T is the ion fluid energy per particle. The density advance from the 0 to the 1/2 time level is first-order accurate in time. The flux advance from the -1/2 to the 1/2 time level is second-order accurate in time, disregarding lower order accuracy of the density and velocity at the central 0 time level. Central differencing is used for the pressure contribution, although smoother differencing might be used.

Once the density and flux are determined at the 1/2 time level, the density is advanced to the 1 time level to first-order accuracy in Δt:

$$\rho^1 = \rho + \frac{\Delta t}{2}\begin{pmatrix} \frac{1}{r\Delta r}\left(r_{1/2}u^{1/2}_{r1/2,0}\langle\rho\rangle^{1/2}_{1/2,0} - r_{-1/2}u^{1/2}_{r-1/2,0}\langle\rho\rangle^{1/2}_{-1/2,0}\right) + \\ \frac{1}{\Delta z}\left(u^{1/2}_{z0,1/2}\langle\rho\rangle^{1/2}_{0,1/2} - u^{1/2}_{z0,-1/2}\langle\rho\rangle^{1/2}_{0,-1/2}\right) \end{pmatrix}. \tag{1.99}$$

With minor modifications, equations used to advance $\vec{f}$ from the -1/2 to the 1/2 time level are used to advance $\vec{f}$ from the 1/2 to 1 time level with time step $\Delta t/2$. In the latter case, quantities on the right-hand side are at the 1/2 time level, except for the electric and magnetic fields, which remain at the 0 time level. Finally, $\vec{u}^1 = \vec{f}^1/\rho^1$, and the fluid velocity $\vec{u}^1$ and density ρ^1 are used in the electron-field advance. Both $\vec{u}^1$ and ρ^1 are first order in the time step, so that the new fields $\vec{E}^1$ and $\vec{B}^1$ are first-order accurate in time as well. Of course, the above equations are altered at the boundaries $r=0$ and $r=r_w$, where r_w is the outer wall radius. At $r=0$ there can be no radial or azimuthal flux by symmetry: $f_r = f_\theta = 0$. The remaining flux-conservative equations used on axis are:

$$\rho^{1/2} = \rho - \frac{\Delta t}{2}\left(\frac{4u_{r1/2,0}\langle\rho\rangle_{1/2,0}}{\Delta r} + \frac{u_{z0,1/2}\langle\rho\rangle_{0,1/2} - u_{z0,-1/2}\langle\rho\rangle_{0,-1/2}}{\Delta z} \right), \tag{1.100}$$

$$\begin{aligned} f_z^{1/2} = f_z^{-1/2} - \Delta t\left(\frac{4u_{r1/2,0}\,\langle f_z\rangle_{1/2,0}}{\Delta r} + \frac{u_{z0,1/2}\,\langle f_z\rangle_{0,1/2} - u_{z0,-1/2}\,\langle f_z\rangle_{0,-1/2}}{\Delta z} \right) - \\ \frac{\Delta t}{2m\Delta z}\left(\rho_{0,1}T_{0,1} - \rho_{0,-1}T_{0,-1}\right) + \frac{q\Delta t}{m}\left(\rho E_z + \frac{1}{c}\left(f_r B_\theta - f_\theta B_r\right)\right), \end{aligned} \tag{1.101}$$

$$\begin{aligned} T^{1/2} = T - \frac{\Delta t}{2}\left(\frac{4u_{r1/2,0}\langle T\rangle_{1/2,0}}{\Delta r} + \frac{1}{\Delta z}\left(u_{z0,1/2}\langle T\rangle_{0,1/2} - u_{z0,-1/2}\langle T\rangle_{0,-1/2}\right) \right) + \\ \frac{\Delta t T}{3}\left(\frac{4u_{r1/2,0}}{\Delta r} + \frac{1}{2\Delta z}\left(u_{z0,1/2} - u_{z0,-1/2}\right) \right). \end{aligned} \tag{1.102}$$

At the rigid wall, $r=r_w$, there is no net radial flux: $f_r = 0$. The remaining flux-conservative equations are:

$$\rho^{1/2} = \rho - \frac{\Delta t}{2}\left(\frac{-2r_{-1/2}u_{r\,-1/2,0}\langle\rho\rangle_{-1/2,0}}{r_{-1/4}\Delta r} + \frac{u_{z\,0,1/2}\langle\rho\rangle_{0,1/2} - u_{z\,0,-1/2}\langle\rho\rangle_{0,-1/2}}{\Delta z} \right), \tag{1.103}$$

$$f_\theta^{1/2} = f_\theta^{-1/2} - \Delta t \left(\begin{array}{l} \dfrac{-2r_{-1/2}u_{r-1/2,0}\,\langle f_\theta \rangle_{-1/2,0}}{r_{-1/4}\Delta r} + \\ \dfrac{1}{\Delta z}\left(u_{z0,1/2}\,\langle f_\theta \rangle_{0,1/2} - u_{z0,-1/2}\,\langle f_\theta \rangle_{0,-1/2}\right) \end{array} \right) + \tag{1.104}$$
$$\frac{q\Delta t}{m}\left(\rho E_\theta + \frac{1}{c}\left(f_z B_r - f_r B_z\right)\right) ,$$

$$f_z^{1/2} = f_z^{-1/2} - \Delta t \left(\begin{array}{l} \dfrac{-2r_{-1/2}u_{r-1/2,0}\,\langle f_z \rangle_{-1/2,0}}{r_{-1/4}\Delta r} + \\ \dfrac{1}{\Delta z}\left(u_{z0,1/2}\,\langle f_z \rangle_{0,1/2} - u_{z0,-1/2}\,\langle f_z \rangle_{0,-1/2}\right) \end{array} \right) - \tag{1.105}$$
$$\frac{\Delta t}{2m\Delta z}\left(\rho_{0,1}T_{0,1} - \rho_{0,-1}T_{0,-1}\right) + \frac{q\Delta t}{m}\left(\rho E_z + \frac{1}{c}\left(f_r B_\theta - f_\theta B_r\right)\right) .$$

$$T^{1/2} = T - \frac{\Delta t}{2}\left(\frac{-2r_{-1/2}u_{r-1/2,0}\langle T \rangle_{-1/2,0}}{r_{-1/4}\Delta r} + \frac{1}{\Delta z}\left(u_{z0,1/2}\langle T \rangle_{0,1/2} - u_{z0,-1/2}\langle T \rangle_{0,-1/2}\right) \right) + \tag{1.106}$$
$$\frac{\Delta t T}{3}\left(\frac{-2r_{-1/2}u_{r1/2,0}}{r_{-1/4}\Delta r} + \frac{1}{2\Delta z}\left(u_{z0,1/2} - u_{z0,-1/2}\right) \right) .$$

Appendix 1.A5
Exact evolution of the angular magnetic vector potential over the whole simulation region

As in Figure 1.4, label the points in R_2 that lie along S_1 with index $i \in [1, M]$. Label the points in R_1 that lie along S_1 with index $j \in [1, N]$. Assume that line segments connecting various cell centers define the stairstep boundary S_1. The angular component of the magnetic vector potential, ϕ, (notation is from subsection 1.2.3) is then sought at cell corner points in $R \equiv R_1 \cup R_2$. Any cell corner necessarily "lies along" S_1 if it terminates a cell edge that is cut by a boundary line segment of S_1. Other points should be kept distinct from these and need not be labeled.

Let c_{IJ} be the value at point $i = I$ of the solution to $X_2\phi = 0$ in R_2 with the given boundary conditions along S and with the boundary conditions $\phi_{j=J} = 1$ and $\phi_{j \neq J} = 0$ for points in R_2 along S_1. There will be MN such values. Let c_{I0} be the value at point $i = I$ of the solution to $X_2\phi = 0$ in R_2 with the given boundary conditions along S and with the boundary conditions $\phi_j = 0 \ \forall \ j$. There will be M of these latter values.

It is expected that whatever the time advanced ϕ in region R_2, it is constrained to obey M equations, Equations 1.107.

$$\phi_i = c_{i0} + \sum_{j=1}^{N} c_{ij}\phi_j \quad \forall \ i \in [1, M] \tag{1.107}$$

If there are P points in R_1, discretization of $\dot{\phi} = X_1\phi + f_1$ will yield P linear equations in $P+M$ unknowns, including the P unknowns in R_1 and the unknowns $\phi_i \ \forall \ i \in [1, M]$. However, the latter M unknowns can be quickly eliminated using Eqs.

1.107. Solving the resultant P equations in P unknowns would complete the time advance in region R_1. Values in region R_2 would then be determined by a Dirichlet problem in region R_2, or by a linear combination of R_2 solutions from which the c_{ij} were determined.

The amount of work necessary for this "exact" advancement is considerable. Unfortunately, the c_{ij} would require $M(N+1)$ distinct elliptic solutions, translating to the solution of $M(N+1)$ linear systems all having the same coefficient matrix, but each with a different right-hand side. This might make the method useful from a research perspective, but not from a practical viewpoint -- there are too many arithmetic operations associated with the $M(N+1)$ right-hand sides, especially considering that the problem needs solving over 1000 times in a reasonable simulation.

Chapter 1 references

1. Anderson, D.V., Cooper, A., Gruber, R., and Schwenn, U., Parallel Computation of Three-dimensional MHD Stability of Plasmas, Lawrence Livermore National Laboratory, UCRL-100571-abs (1989)
2. Andrianov, A.M. *et al.*, High Current Pulse Discharges, *Proc. of the Second U.N. Internat. Conf. on the Peaceful Uses of Atomic Energy, Geneva* **31**, 348-364 (1958)
3. Bennett, W.H., Magnetically Self-Focussing Streams, *Phys. Rev.* **45**, 890-897 (1934)
4. Birdsall, C.K., and Langdon, A.B., Plasma Physics via Computer Simulation, (McGraw Hill, New York, NY, 1985)
5. Boris, J.P., Proc. Fourth Conf. Numerical Simulation of Plasmas, Naval Res. Lab., Wash., D.C., 3-67, 2-3 Nov. 1970.
6. Braginskii, S.I., The Behavior of a Completely Ionized Plasma in a Strong Magnetic Field, *Soviet Physics JETP* **6**, 494-501 (1958)
7. Braginskii, S.I., in Reviews of Plasma Physics, ed. by M.A. Leontovich **1**, 205 (Consultants Bureau, New York, 1965)
8. Carruthers, R. and Davenport, P.A., Observations of the Instability of Constricted Gaseous Discharges, *Proc. Phys. Soc.* **70**, 49-50 (1957)
9. Chandrasekhar, S., On the Inhibition of Convection by a Magnetic Field, *Phil. Mag.* (7) **43**, 501 (1952)
10. Chandrasekhar, S., The Stability of Viscous Flow Between Rotating Cylinders in the Presence of a Magnetic Field, *Proc. Roy. Soc. A* **216**, 293 (1953)
11. Chodura, R., A Hybrid Fluid-Particle Model of Ion Heating in High-Mach-Number Shock Waves, *Nucl. Fusion* **15**, 55 (1975)

12. Cochran, F.L., and Robson, A.E., Stability of a Z Pinch With Rising Current, *Phys. Fluids B* **2**, 123-128 (1990)

13. Coppins, M., Bond, D.J., and Haines, M.G., A Study of the Stability of the Z Pinch Under Fusion Conditions Using the Hall Fluid Model, *Phys. Fluids* **27**, 2886-2889 (1984)

14. Curran, S.C., Allen, K.W., Bodin, H.A.B., Fitch, R.A., Peacock, N.J., and Reynolds, J.A., Studies of High-current Gas Discharges at High Rates of Build Up, *Proc. of the Second U.N. Internat. Conf. on the Peaceful Uses of Atomic Energy, Geneva* **31**, 365 (1958)

15. Curzon, F.L., Folkierski, A., Latham, R., and Nation, J.A., Experiments on the Growth Rate of Surface Instabilites in a Linear Pinched Discharge, *Proc. Royal Soc. A* **257**, 388-401 (1960)

16. Forslund, D.W., and Freidberg, J.P., Theory of Laminar Collisionless Shocks, *Phys. Rev. Lett.* **27**, 1189 (1971)

17. Freidberg, J.P. and Hewett, D.W., Eigenmode Analysis of Resistive MHD Stability by Matrix Shooting, *J. Plasma Phys.* **26**, 177-192 (1977)

18. Hain, K., and Lüst, R., Zur Stabilität zylindersymmetrischer Plasmakonfigurationen mit Volumenströmen, *Z. Naturforsch.* **13a**, 936-940 (1958)

19. Hain, K., Hain, G., Roberts, K.V., Roberts, S.J., and Köppendörfer, W., Fully Ionized Pinch Collapse, *Z. Naturforsch.* **15a**, 1039-1050 (1960)

20. Hamasaki, S. and Krall, N.A., Numerical Modeling of the Implosion Heating Experiment, *Phys. Fluids* **20**, 229 (1977)

21. Hamasaki, S., Krall, N.A., Wagner, C.E., and Byrne, R.N., Effect of Turbulence on Theta Pinch Modeling by Hybrid Numerical Models, *Phys. Fluids* **20**, 65 (1977)

22. Hewett, D.W., A Global Method of Solving the Electron-Field Equations in a Zero-Inertia-Electron-Hybrid Plasma Simulation Code, *J. Comput. Phys.* **38**, 378-395 (1980)

23. Hirt, C.W., Heuristic Stability Theory for Finite-Difference Equations, *J. Comput. Phys.* **2**, 339-355 (1968)

24. Kruskal, M. and Schwarzschild, M., Some Instabilities of a Completely Ionized Plasma, *Proc. Roy. Soc. A* **223**, 348-360 (1954)

25. Kurchatov, I.V., Russian Thermonuclear Experiments, *Nucleonics* **14**, 36-43, 123 (1956)

26. Langdon, A.B., Effects of the Spatial Grid in Simulation Plasmas, *J. Comput. Phys.* **6**, 247-267 (1970)

27. Lapidus, L., and Pinder, G.F., Numerical Solution of Partial Differential Equations in Science and Engineering, John Wiley & Sons, New York, NY (1982)

28. Larson, D.J. Electromagnetic Plasma Simulation Using Three-Dimensionsl Unstructured Grids, Lawrence Livermore National Laboratory, UCRL-JC-115331 (1993)

29. Lawson, J.D., *J. Nucl. Energy C* **1**, 31 (1959)

30. Lindemuth, I.R., Two-dimensional Fiber Ablation in the Solid-Deuterium Z Pinch, *Phys. Rev. Lett.* **65**, 179-182 (1990)

31. Lindemuth, I.R., A Boundary Condition for Computational Magnetohydrodynamics, *J. Comput. Phys.* **25**, 104-117 (1977)

32. Lindemuth, I.R., The Alternating-Direction Implicit Numerical Solution of Time-dependent, Two-dimensional, Two-Fluid Magnetohydrodynamic Equations, Ph.D. thesis, Lawrence Livermore Laboratory, UCRL-51103 (1971)

33. Lindemuth, I.R., and Freeman, B.L., Shock Dynamics and Neutron Production in an Explosive Generator Driven Dense Plasma Focus, *Appl. Phys. Lett.* **40**, 462-465 (1982)

34. Lindemuth, I.R., and Killeen, J., Alternating Direction Implicit Techniques for Two-Dimensional Magnetohydrodynamic Calculations, *J. Comput. Physics* **13**, 181-208 (1973)

35. Lindemuth, I.R., McCall, G.H., and Nebel, R.A., Fiber Ablation in the Solid-Deuterium Z Pinch, *Phys. Rev. Lett.* **62**, 264-267 (1989)

36. Lundquist, S., On the Stability of Magneto-Hydrostatic Fluids, *Phys. Rev.* **83**, 307-311 (1951)

37. Millikan, R.A., and Sawyer, R.A., *Phys. Rev.* **12** 2, 167 (1918)

38. Millikan, R.A., and Bowen, I.S., Extreme Ultraviolet Spectra, *Phys. Rev.* **23**, 1-34 (1924)

39. Mirin, A.A., and Killeen, J., Nonlinear Three-dimensional MHD Calculations of Resistive Instabilities in a Reversed Field Pinch, Lawrence Livermore National Laboratory, UCRL-88753-extended abstract (1983)

40. Nielsen, P.D., A Computational Investigation of the Limits to Pease-Braginskii Collapse of a Z-Pinch, Ph.D. thesis, Lawrence Livermore National Laboratory, UCRL-53166 (1981)

41. Okuda, H. Verification of Theory for Plasma of Finite-Size Particles, *Phys. Fluids* **15**, 1268-1274 (1972)

42. Pease, R.S., Equilibrium Characteristics of a Pinched Gas Discharge Cooled by Bremsstrahlung Radiation, *Proc. Phys. Soc. B* **70**, 11-23 (1957)

43. Schnack, D.D., Barnes, D.C., Mikic, Z., Harned, D.S., Caramana, E.J., and Nebel, R.A., Numerical Simulation of Reversed-field Pinch Dynamics, *Comput. Phys. Comm.* **43**, 17-28 (1986)

44. Sgro, A.G. and Nielson, C.W., Hybrid Model Studies of Ion Dynamics and Magnetic Field Diffusion During Pinch Implosions, *Phys. Fluids* **19**, 126-133 (1976)

45. Sheehey, P.T., Magnetohydrodynamic Simulation of Solid-Deuterium-Initiated Z-Pinch Experiments, Ph.D. thesis, Los Alamos National Laboratory, LA-12724-T Thesis (1994)

46. Sheehey, P., Hammel, J.E., Lindemuth, I.R., Scudder, D.W., Schlachter, J.S., Lovberg, R.H., and Riley, R.A., Jr., Two-dimensional Direct Simulation of Deuterium-fiber-initiated Z Pinches With Detailed Comparison to Experiment, *Phys. Fluids B* **4**, 3698-3706 (1992)

47. Shearer, J.W., Contraction of Z pinches actuated by radiation losses, *Phys. Fluids* **19**, 1426-1428 (1976)

48. Struve, R.E., An Experimental Study of Finite Larmor Radius Effects in a Linear Z-pinch, Ph.D. thesis, Lawrence Livermore Laboratory, UCRL-52993 (1980)

49. Tayler, R.J., Hydromagnetic Instabilities of an Ideally Conducting Fluid, *Proc. Phys. Soc. B* **70**, 31 (1957)

50. Tayler, R.J., The Influence of an Axial Magnetic Field on the Stability of a Constricted Gas Discharge, *Proc. Phys. Soc. B* **70**, 1049 (1957)

51. Tuck, J.L., Review of Controlled Thermonuclear Research at Los Alamos for mid 1958, *Proc. of the Second U.N. Internat. Conf. on the Peaceful Uses of Atomic Energy, Geneva* **32**, 3-25 (1958)

52. Vikhrev, V.V., Contraction of a Z-pinch as a Result of Losses to Radiation, *JETP Lett.* **27**, 95 (1978)

53. Vikhrev, V.V. and Korzhavin, M., Effect of Anomalous Conductivity on the Dynamics of the Plasma Focus, *Sov. J. Plasma Phys.* **4**, 411 (1978)

54. Vikhrev, V.V. and Gureev, K.G., Dynamics of a Strongly Radiating Plasma in a Noncylindrical Z-pinch, *Sov. Phys. Tech. Phys.* **23**, 1295 (1978)

55. Wood, R.W., A New Form of Cathode Discharge and the Production of X-rays, Together With Some Notes on Diffraction, *Phys. Rev.* **5** 1, 1 (1897)

56. Woods, L.C., Principles of Magnetoplasma Dynamics (Clarendon Press, Oxford 1987)

Chapter 2
Field Simulation Results: Benchmarks and Observations

The purpose of this chapter is to present results that can be used to guage the accuracy and performance of the screw pinch simulation algorithm. The first section, 2.1, introduces ideal linear MHD theory pertaining to two-dimensional (2-d) stability of one-dimensional (1-d) radial screw pinch equilibria. The MHD theory is background material for the equilibrium studies presented in Section 2.2, in which screw pinch simulations with fluid ions are shown to agree with the theory in a few important limits. The third section, 2.3, presents simulations in which the plasma is far from equilibrium, driven strongly by axial currents and azimuthal electric fields. These results are briefly compared with similar simulations [Hewett 1980; Sgro and Nielson 1976], and are representative of the most significant applications of the algorithm. The last section, 2.4, compares new and old update methods for the angular magnetic field B_θ in entrained vacuum voids;. differences between the updates are shown to significantly affect the initial compression of a dynamic Z-pinch.

To the computational plasma pinch community, more comparisons between computer simulations and other independent analyses are better. Mistakes are caught earlier and better understanding of the plasmas is achieved. So, to avoid syntactical and logical errors in the pinch simulation algorithm, simple comparisons with previous work were sought. It was determined that the pinch simulation should reproduce results of analytic stability theory in the proper limits. One early stability analysis was a candidate for comparison with the axisymmetric pinch simulation [Tayler 1957b]; it involves compressible plasmas with distributed current profiles. In the analysis, the physical variables of the plasma equilibrium have the values

$$\rho = \rho_0[1-(r/r_1)^{2n+2}], \qquad p = p_0[1-(r/r_1)^{2n+2}], \qquad B_\theta = B_{\theta 0}(r/r_0)^{n+1},$$
$$J_z = \frac{(n+2)cB_{\theta 0}}{4\pi r_0}(r/r_0)^n, \qquad \frac{B_{\theta 0}^2}{4\pi}\frac{n+2}{2n+2}(r_1/r_0)^{2n+2} = p_0.$$

The conducting plasma of maximum radius r_0 is surrounded by a medium of density $\rho_1 = \rho_0[1-(r_0/r_1)^{2n+2}]$, pressure $p_1 = p_0[1-(r_0/r_1)^{2n+2}]$, and magnetic field $B_\theta = B_{\theta 0}r_0/r$. Unfortunately, the current density J_z is discontinuous at r_0, and this means that B_θ has a discontinuous derivative at r_0. Such a discontinuity would be a significant source of error in the finite-difference based simulation, making it more difficult to compare the simulation with Tayler's theory. For this reason, Tayler's results were not practical for comparison purposes.

Instead, plasma equilibria with smooth physical variables were sought as a basis of comparison. The natural stability theory for such equilibria is the ideal MHD normal mode analysis of a screw pinch, first presented by Hain and Lüst [1958]. This formulation, which is detailed in Section 2.1, leads to simple numerical analysis of any screw pinch equilibrium, allowing accurate study of diffuse equilibria such as Bennett equilibria and reverse-field pinch equilibria. It can be used to study discontinuous equilibria such at the Tayler equilibria, too, but discontinuities require many more finite-difference points for adequate resolution. Once implemented numerically, the study of a given equilibrium is a matter of specification of the proper input parameters; no tedious algebraic manipulations are necessary.

The comparison of simulations and stability theory for Bennett equilibria demonstrated that neglect of the Hall term and diagmagnetic drift terms in the general Ohm's law was appropriate for a few fastest-growing instability e-folding times. This is a comforting result of Section 2.2. These terms are expected to be ignorable in most stability analyses, even though the magnitudes of the terms can be considerable.

Other benchmarks of the pinch simulation algorithm, such as benchmarks for the compressional stage of a Z-pinch are not as straightforward as the stability comparisons. About the best that can be done is to compare independent simulation algorithms or to compare simulation algorithms to experiment. In Section 2.3, 1-d simulations with the algorithm of this dissertation are compared with previous 1-d simulations of theta- and Z-pinches. Agreement between the present and past algorithms increases confidence in the simulation endeavor. Naturally, agreement between two independent simulations does not guarantee that both are correct, even in a narrow range of expected validity.

2.1 Stability analysis of screw pinch MHD equilibria

2.1.1 Screw pinch equilibrium equation

One-dimensional radial screw pinch equilibria exhibit a force balance between a pressure profile $p(r)$ and a magnetic flux density profile $\vec{B}(r) = \hat{\theta}\, B_\theta(r) + \hat{z}\, B_z(r)$ obeying Eq. 2.1. MKS units are used throughout Section 2.1.

$$\frac{d}{dr}\left(p + \frac{B_\theta^2 + B_z^2}{2\mu_0}\right) + \frac{B_\theta^2}{\mu_0 r} = 0. \tag{2.1}$$

All axial and azimuthal derivatives are zero. There are an infinite number of sets of screw pinch profiles, $\{\, p(r),\ B_\theta(r), \text{ and } B_z(r) \,\}$, satisfying Eq. 2.1; some are ideal MHD stable and others are not. The simulated versus theoretical stability of a few sets of profiles can be used to benchmark the screw pinch algorithm. Fortunately, the behavior of each set of profiles can be predicted by numerical means from linear MHD theory, which is indepent of the screw pinch algorithm. The linear normal mode formulation of ideal MHD theory, presented in Section 2.1.2, is the theory that can predict the behavior.

For comparing ideal linear MHD theory to the time-dependent pinch simulation algorithm the equilibrium profiles chosen for this dissertation are all Bennett profiles discussed in Section 2.2.1. In 1-d simulation checks the Bennett profiles are perturbed by a radially dependent ion drift velocity: theoretically, the plasma exhibits stable oscillation at a predictable frequency. For 2-d simulation checks in this dissertation, sausage instabilities of 1-d Bennett equilibria are seeded with 2-d perturbations in ion drift velocity. The sausage instabilities lead to 2-d r- and z- dependence of all of the field quantities, and are ideally suited to test the full machinery of the algorithm in the linear limit. When the 1- and 2-d simulations were executed, they showed good agreement with ideal linear MHD theory.

2.1.2 Normal mode formulation

When a MHD equilibrium is unstable, certain small-amplitude spatial modes will grow exponentially in time. Before the modes grow too large, the rate of growth is predictable by linear MHD theory. When a MHD equilibrium is stable, all small-amplitude spatial modes added to the equilibrium will oscillate in time at a predictable frequency.

Quantitative theoretical prediction of instability growth rates and stable oscillation frequencies requires numerical solution of the eigenvalue problem of the normal-mode formulation of linearized ideal MHD stability theory. In ideal MHD, the plasma dynamics are dictated by the following single-fluid equations (MKS units):

$$\partial \rho/\partial t = \rho\bar{\nabla}\cdot\bar{v}, \qquad \partial p/\partial t = -\gamma p\bar{\nabla}\cdot\bar{v}, \qquad (2.2a,b)$$

$$\rho(\partial v/\partial t)_{\perp} = \bar{J}\times\bar{B}-\bar{\nabla}_{\perp}p, \qquad \rho(\partial v/\partial t)_{\parallel} = -\hat{b}\cdot\bar{\nabla}p, \qquad (2.2c,d)$$

$$\bar{E}+\bar{v}\times\bar{B} = 0, \qquad \partial B/\partial t = -\bar{\nabla}\times\bar{E}, \qquad (2.2e,f)$$

$$\bar{\nabla}\times\bar{B} = \mu_0\bar{J}, \qquad \bar{\nabla}\cdot\bar{B} = 0. \qquad (2.2g,h)$$

The mass density is ρ, while $\bar{v}$ is the fluid mass drift velocity, p is the fluid pressure, $\bar{J}$ is the electric current density, $\bar{E}$ is the electric field, $\bar{B}$ is the magnetic flux density with unit vector $\hat{b}$, γ is the ratio of specific heats, and μ_0 is the permeability of free space. The operator $\bar{\nabla}_{\perp}$ is the gradient operator in the plane perpendicular to $\bar{B}$ and $\hat{b}$. Because $\rho_e = \rho_i = \rho$, while the electron mass is much smaller than the ion mass, the single-fluid mass drift velocity $\bar{v}$ is essentially the ion mass drift velocity $\bar{u}_i$.

Equation 2.2a expresses mass conservation. Equation 2.2b is a simplified form of energy conservation. Equations 2.2c, 2.2d, and 2.2e are reexpressions of the electron and ion momentum conservation equations. Equations 2.2f and 2.2g are Faraday's law and Ampere's law without displacement current, respectively. Equation 2.2h is implied by Eq. 2.2f, but is explicitly included so as not to be forgotten in equilibrium, when time derivatives are zero.

The type of plasma equilibria used to compare stability theory and simulation are static ideal MHD equilibria. Equilibria are obtained from Eqs. 2.2 by zeroing time derivatives. Static equilibrium implies $\bar{v}_0 \equiv 0$. Relations between equilibrium quantities become $\bar{J}_0\times\bar{B}_0 = \bar{\nabla}p_0$, $\mu_0\bar{J}_0 = \bar{\nabla}\times\bar{B}_0$, and $\bar{\nabla}\cdot\bar{B}_0 = 0$, where the subscripted 0 indicates an equilibrium. The equations $\bar{J}_0\times\bar{B}_0 = \bar{\nabla}p_0$ and $\mu_0\bar{J}_0 = \bar{\nabla}\times\bar{B}_0$ can be combined to yield Eq. 2.1.

The stability theory begins with the assumption that all extrinsic quantities $q(\bar{r},t)$ of the plasma have a small harmonic variation in addition to the equilibrium values:

$$q(\bar{r},t) = q_0(\bar{r}) + q_1(\bar{r})e^{-i\omega t}. \tag{2.3}$$

The quantities $q_1(\bar{r})$ are then phasors. The remainder of this section is a borrowed presentation [Freidberg 1987] of the normal mode theory first laid out in the late 1950s [Hain and Lüst 1958]. In the formulation, the phasor drift velocity is set equal to the derivative of a phasor displacement vector $\bar{\xi}(\bar{r})$:

$$\bar{v}_1 e^{-i\omega t} \equiv \frac{\partial}{\partial t}\left(\bar{\xi}e^{-i\omega t}\right) = -i\omega\bar{\xi}e^{-i\omega t}. \tag{2.4}$$

The problem is then reduced to an eigenvalue equation for the phasor displacement vector, Eq. 2.8. To arrive at the eigenvalue equation, conservation of mass, conservation of energy, and Faraday's Law yield expressions for the phasor mass density ρ_1, phasor pressure p_1, and phasor magnetic flux density $\bar{B}_1$ in terms of $\bar{\xi}(\bar{r})$:

$$\rho_1 = -\bar{\nabla}\cdot(\rho_0\bar{\xi}), \tag{2.5}$$
$$p_1 = -\bar{\xi}\cdot\bar{\nabla}p_0 - \gamma p_0\bar{\nabla}\cdot\bar{\xi}, \tag{2.6}$$
$$\bar{B}_1 = \bar{\nabla}\times(\bar{\xi}\times\bar{B}_0). \tag{2.7}$$

Then conservation of momentum gives the linear eigenvalue equation obeyed by $\bar{\xi}(\bar{r})$:

$$-\omega^2\rho_0\bar{\xi} = \bar{F}(\bar{\xi}), \tag{2.8}$$
$$\bar{F}(\bar{\xi}) \equiv \frac{1}{\mu_0}\left[(\bar{\nabla}\times\bar{B}_1)\times\bar{B}_0 + (\bar{\nabla}\times\bar{B}_0)\times\bar{B}_1\right] + \bar{\nabla}(\bar{\xi}\cdot\bar{\nabla}p_0 + \gamma p_0\bar{\nabla}\cdot\bar{\xi}). \tag{2.9}$$

The quantity $\bar{F}(\bar{\xi})$ is a phasor force operator acting on $\bar{\xi}(\bar{r})$. The goal is to determine the eigenvectors $\bar{\xi}$ and associated eigenvalues $-\omega^2$ of $\bar{F}$. If ω is purely imaginary, then the mode associated with $\bar{\xi}$ is exponentially unstable. If ω is purely real, then the mode associated with $\bar{\xi}$ is exponentially stable, i.e. oscillatory about the equilibrium quantities $q_0(\bar{r})$ with radian frequency ω.

As it stands, Eq. 2.8 can be difficult to solve because the phasor force operator mixes up the components of $\bar{\xi}$. In order to uncouple the components of $\bar{\xi}$ in the general screw pinch analysis, it is convenient to choose orthogonal basis vectors $\hat{r}$, $\hat{\eta}$, and $\hat{b}$. so that

$$\bar{\xi} = \hat{r}\xi_r + \hat{\eta}\xi_\eta + \hat{b}\xi_\parallel, \tag{2.10}$$

$$\bar{B}_1 = \hat{r}\tilde{B}_r + \hat{\eta}\tilde{B}_\eta + \hat{b}\tilde{B}_\parallel, \tag{2.11}$$

$$\hat{b} = (\hat{\theta}B_{0\theta} + \hat{z}B_{0z}) / B_0, \tag{2.12}$$

$$\hat{\eta} = (\hat{\theta}B_{0z} - \hat{z}B_{0\theta}) / B_0. \tag{2.13}$$

Obviously, $\hat{r}$ is the unit radius vector, $\hat{b}$ is the unit vector in the direction of the equilibrium magnetic flux density, and $\hat{\eta}$ is a unit vector perpendicular to both $\hat{r}$ and $\hat{b}$.

For the general screw pinch, equilibrium quantities are a function of radius only, $q_0(\bar{r}) = q_0(r)$, and it is convenient to assume that the phasors have harmonic dependence in the θ- and z-coordinates, $q_1(\bar{r}) \rightarrow q_1(r)e^{i(m\theta+kz)}$, with k real and m an integer. In this case, the magnetic flux density and pressure phasors can be written in terms of the components of the phasor displacement. This is achieved by substituting Eqs. 2.10 - 2.13 into $\bar{B}_1 = \bar{\nabla}\times(\bar{\xi}\times\bar{B}_0)$.

$$\tilde{B}_r = iF\xi_r, \tag{2.14}$$

$$\tilde{B}_\eta = iF\xi_\eta + \xi_r\left[\frac{B_{0\theta}}{B_0}\frac{\partial B_{0z}}{\partial r} - \frac{rB_{0z}}{B_0}\frac{\partial}{\partial r}\left(\frac{B_{0\theta}}{r}\right)\right], \tag{2.15}$$

$$\tilde{B}_\parallel = -iG\xi_\eta - \frac{B_0}{r}\frac{\partial}{\partial r}(r\xi_r) + \frac{\xi_r}{B_0}\left(\mu_0\frac{\partial p_0}{\partial r} + \frac{2B_{0\theta}^2}{r}\right), \tag{2.16}$$

$$p_1 = -\xi_r \frac{\partial p_0}{\partial r} - \gamma p_0 \left[\frac{1}{r} \frac{\partial}{\partial r}(r\xi_r) + \frac{i}{B_0}\left(G\xi_\eta + F\xi_\parallel \right) \right]. \tag{2.17}$$

$$F = kB_{0z} + mB_{0\theta}/r, \tag{2.18}$$

$$G = mB_{0z}/r - kB_{0\theta}. \tag{2.19}$$

The quantity γ is the ratio of specific heats, and the constant i is the square root of -1. Everything is determined once Eq. 2.20 for the phasor displacement is solved.

$$\frac{\partial}{\partial r}\left[A \frac{\partial}{\partial r}(r\xi_r) \right] - Cr\xi_r = 0. \tag{2.20}$$

Equation 2.20 is a homogeneous Sturm-Liouville equation. The coefficients are

$$A(r{:}\omega) = \frac{\rho_0(V_s^2 + V_a^2)}{r} \frac{(\omega - \omega_a^2)(\omega - \omega_h^2)}{(\omega - \omega_f^2)(\omega - \omega_s^2)}, \tag{2.21}$$

$$\begin{aligned} C(r{:}\omega) = & -\frac{\rho_0(\omega^2 - \omega_a^2)}{r} + \frac{4k^2V_a^2B_{0\theta}^2(\omega^2 - \omega_g^2)}{\mu_0 r^3(\omega^2 - \omega_f^2)(\omega^2 - \omega_s^2)} \\ & + \frac{\partial}{\partial r}\left[\frac{B_{0\theta}}{\mu_0 r^2}\left(B_{0\theta} - \frac{2kG(V_a^2 - V_s^2)(\omega^2 - \omega_h^2)}{(\omega^2 - \omega_f^2)(\omega^2 - \omega_s^2)} \right) \right]. \end{aligned} \tag{2.22}$$

When the problem is couched in terms of the homogeneous Sturm-Liouville equation, the characteristic frequency ω enters as an adjustible parameter in a self-adjoint operator; therefore Equation 2.20 is not itself a standard eigevalue problem. For each choice of ω, the coefficients A and C are determined by the equilibrium profiles, so that numerical solution of the equation under the constraint of boundary conditions can proceed by shooting techniques. The values of ω for which Eq. 2.20 is solvable are then the desired eigenvalues of Eq. 2.8. The newly defined quantities in Eqs. 2.21 and 2.22 are all simple functions of the equilibrium profiles:

$$\omega_a^2 = \frac{F^2}{\mu_0\rho_0}, \quad \omega_h^2 = \frac{V_s^2\omega_a^2}{V_s^2+V_a^2}, \quad \omega_g^2 = \frac{V_s^2\omega_a^2}{V_a^2},$$

$$\omega_{f,s}^2 = \frac{V_s^2+V_a^2}{2}\left(k^2+\frac{m^2}{r^2}\right)\left(1\pm(1-\alpha^2)^{1/2}\right), \text{ with } \alpha^2 = \frac{4V_s^2\omega_a^2}{\left(k^2+m^2/r\right)(V_s^2+V_a^2)^2},$$

$$V_s^2 = \gamma p_0/\rho_0, \quad V_a^2 = \frac{B_0^2}{\mu_0\rho_0}.$$

For ω_f^2 use the sum, and for ω_s^2 use the difference in the corresponding $\omega_{f,s}^2$ formula. In the $\omega_{f,s}^2$ formula, α must be in the range $0 \le \alpha \le 1$. Once the radial displacement phasor is found, the other components of the displacement phasor are given by

$$\xi_\parallel = \frac{-i\gamma p_0 F}{\rho_0 B_0 r(\omega-\omega_f^2)(\omega-\omega_s^2)}\left[(\omega-\omega_a^2)\frac{\partial}{\partial r}(r\xi_r) + \frac{2kGB_{0\theta}\xi_r}{\mu_0\rho_0}\right], \tag{2.23}$$

$$\xi_\eta = \frac{-i}{\mu_0\rho_0 B_0 r(\omega^2-\omega_f^2)(\omega^2-\omega_s^2)}\left[\begin{array}{c} G(\omega^2-\omega_h^2)(\gamma\mu_0 p_0+B_0^2)\frac{\partial}{\partial r}(r\xi_r) \\ +\ 2kB_0^2B_{0\theta}(\omega^2-\omega_g^2)\xi_r \end{array}\right]. \tag{2.24}$$

2.1.3 Numerical solution of the normal mode eigenequation

The axial boundary condition for solution of Eq. 2.20 is $\xi_r(r=0)=0$. For purposes of benchmarking the screw pinch algorithm, the boundary condition $\xi_r(r_w)\equiv 0$ at the wall radius r_w is convenient. This corresponds to a perfectly reflecting immovable wall and is simple to implement in the pinch simulation algorithm.

To solve the eigenmode Eq. 2.20, a simple and robust implicit matrix shooting technique can be used. In shooting techniques, the equation is first finite-differenced by $N+1$ points $r_i = i\Delta r$, $0 \le i \le N+1$, evenly distributed over the interval $0 \le r \le r_w$, so that $\Delta r = r_w/N$. Three-point central differencing is used for evaluation of the second derivative, i.e.

$$\frac{\partial}{\partial r}\left[A\frac{\partial q}{\partial r}\right]_i \cong \frac{A_{i+1/2}(q_{i+1}-q_i) - A_{i\text{-}1/2}(q_i-q_{i-1})}{\Delta r^2},\ 1\le i\le N, \tag{2.25}$$

where $A_{1/2}=2A_1$, and $A_{i\pm1/2}=(A_i+A_{i\pm1})/2$ for $2\le i\le N$. Note that the derivative is left unexpanded for purposes of finite differencing. The resultant difference formula in Eq. 2.25 then has a conservative form that is important to use in the simulation [Lindemuth 1975]. The result is a tridiagonal system of equations $c_i\xi_{ri-1}+a_i\xi_{ri}+b_i\xi_{ri+1} = r_i$, $1\le i\le N$. This set of equations needs two additional independent and consistent constraints to have a solution, and the constraints must be consistent with ω.

Only when coefficients A and C are of the same sign everywhere is the problem an elliptic boundary value problem with a solution meeting the boundary conditions for arbitrary ω. This does not give discrete eigenmodes amenable to benchmarking. Discrete eigenmodes only arise when A and C are of opposite sign over a sufficiently large radial interval. In this case the problem is not elliptic, and a shooting technique is necessary to find the eigenvalue ω that yields one of the desired boundary conditions, $\xi_r(r_w)=0$, assuming the other, $\xi_r(r=0)=0$.

Dirichlet conditions on the first two radial points are independent constraints on the tridiagonal system that are always consistent with a choice of ω. These conditions, $\xi_{r0}\equiv\xi_r(r=0)=0$ and $\xi_{r1}\equiv\xi_r(r=\Delta r)=\xi_1$, reduce the problem to an initial value problem amenable to shooting. The following two subsections discuss ways in which the shooting technique can be implemented.

Forward shooting techniques

There are many methods that can be used in an attempt to solve the eigenequation by forward shooting. By forward shooting, it is meant that an unknown is determined by radial grid coefficients up to the point at which the unknown is being updated. One simple forward shooting technique marches forward with the assignment

$$\xi_{ri+1} = (r_i - a_i\xi_{ri} - c_i\xi_{ri-1})/b_i,\ i = 2, ..., N+1. \tag{2.26}$$

Often, methods of higher order accuracy in grid spacing are used, such as Runge-Kutta methods. Unfortunately, the Runge-Kutta methods often fail to solve the problem. Regions where $AC > 0$ cause problems because the forward techniques pick up from numerical error a growing solution that causes computational overflow. Probably any commonly used forward solver based on successive solution of $f(\xi_{r0}, \xi_{r1}, ..., \xi_{ri+1}) = 0$ for ξ_{ri+1} can fail by such computational overflow. Note that the above scheme can be written

$$f(\xi_{r0}, \xi_{r1}, ..., \xi_{ri+1}) = r_i - b_i\xi_{ri+1} - a_i\xi_{ri} - c_i\xi_{ri-1} = 0. \tag{2.27}$$

Such schemes do not take full advantage of all information in the coefficients of the tridiagonal system. They do not "look ahead" far enough.

Implicit matrix shooting technique

As suggested by a previous eigenmode analysis [Freidberg and Hewett 1981], a more "forward-looking" shooting technique was successfully attempted. The method first gathers information from coefficients "ahead" on the grid; this is naturally achieved with a backward elimination :

$$c'_N = c_N/a_N, \qquad b'_N = b_N/a_N, \qquad r'_N = r_N/a_N, \tag{2.28a,b,c}$$

$$b'_i = \frac{-b_i b'_{i+1}}{a_i - b_i c'_{i+1}},\ c'_i = \frac{c_i}{a_i - b_i c'_{i+1}},\ r'_i = \frac{r_i - b_i r'_{i+1}}{a_i - b_i c'_{i+1}},\ i = N-1, ..., 1. \tag{2.28d,e,f}$$

Up to this point, none of the unknowns are determined, but forward information has been incorporated in the primed coefficients. With a simple forward elimination using the primed coefficients, the solution is completed:

$$\xi_{N+1} = (r_1' - \xi_1)/b_1', \tag{2.29a}$$

$$\xi_i = r_i' - c_i'\xi_{i-1} - b_i'\xi_{N+1} \text{ for } i = 2, \ldots, N. \tag{2.29b}$$

Solution by this technique gives a nontrivial answer for any ω^2 with less overflow problems. Furthermore, when shooting for $\xi_{N+1} = 0$ while adjusting ω^2 no special choice for the value of ξ_1 is necessary; the soul effect of ξ_1 is to scale the eigenfunction without affecting the eigenvalue. When shooting for a fixed $\xi_{N+1} \neq 0$, the eigenvalue should depend on ξ_1: the eigenfunction scales as ξ_1 only if ξ_{N+1}/ξ_1 is somehow fixed.

In this technique, there is not a single unknown that is determined without incorporation of information from all of the coefficients and boundary constraints. Each unknown ξ_{ri+1} is ultimately determined by an equation of the form

$$f(\xi_{r1}, \ldots, \xi_{ri+1}; \xi_{ri+2}, \ldots, \xi_{rN+1}) = 0. \tag{2.30}$$

The semicolon indicates a weak implicit dependence of the ξ_{ri+1} on $\xi_{ri+2}, \ldots, \xi_{rN+1}$ through the primed coefficients from the backward elimination. This additional dependence stabilizes the technique for Eq. 2.20.

2.2 Equilibrium stability test cases

Agreement between a pinch simulation with zero resistivity $\bar{\bar{\eta}} = 0$ and ideal linear MHD theory might at first be unexpected. This is because the pinch simulation is based on a nonideal MHD fluid theory for the electrons, governed by an Ohm's law that retains the *Hall term,* $\bar{J}\times\bar{B}/ce\rho$, and the electron pressure term or *diamagnetic drift term,* $\bar{\nabla}(\rho T_e)/e\rho$. Rewriting Eq. 1.1, Ohm's law for the pinch simulation is

$$\bar{E} + \bar{u}_i\times\bar{B}/c = \bar{\bar{\eta}}\cdot\bar{J} + \left(\bar{J}\times\bar{B}/c - \bar{\nabla}(\rho T_e)\right)/e\rho \quad \text{(Gaussian units).}$$

In pinch equilibra with $\bar{u}_i \equiv \bar{v} = 0$, it is easy to imagine cases for which $\bar{E}\neq 0$. The nonzero electric field is necessary to balance the ion pressure gradient force. Therefore, the Hall term and electron pressure terms are quite significant. On the other hand, Ohm's law for ideal MHD is $\bar{E}+\bar{u}_i\times\bar{B}/c = 0$, which totally neglects the Hall and electron pressure contributions! So how can ideal MHD theory and the pinch simulation agree?

Agreement can be attained between ideal MHD stability theory and an $\bar{\bar{\eta}}=0$ simulation as long as $(\bar{J}\times\bar{B}/c - \bar{\nabla}(\rho T_e))/e\rho \propto \bar{\nabla}g$ for some scalar function g. When Ohm's law is substituted into Faraday's law, the Hall and diagmagnetic drift terms can then be neglected because $\bar{\nabla}\times\bar{\nabla}g \equiv 0$. This is indeed what happens for $T_e = T_i$ and $\bar{u}_i = 0$ [Freidberg 1987]:

$$\frac{\bar{J}\times\bar{B}-\bar{\nabla}p_e}{e\rho} \equiv \frac{1}{2e}\bar{\nabla}\left(\frac{\gamma}{\gamma-1}\frac{p}{\rho}\right) \quad \text{(MKS units).}$$

2.2.1. One-dimensional simulations initialized with perturbed stable Bennett profiles

The first simulations to be compared with ideal linear MHD theory were 1-d. The simulations incorporate all three components of drift velocity (1d3v) and are used to check the radially dependent part of the pinch algorithm. In each simulation, a small radially dependent velocity perturbation is added to an equilibrium Bennett profile. The perturbation was determined from the normal mode eigenequation, Eq. 2.20. Perturbations are proportional to a single stable eigenmode of the profile with infinite wavelength in the axial direction, $(k=0)$. For the $k=0$ case, only stable radial modes exist in the linear ideal MHD approximation. These simulations exhibit indisputable agreement with linear MHD theory. The Bennett profiles are of the form of Eqs. 2.31-2.33.

$$B_\theta = \frac{\mu_0 I_0}{2\pi}\frac{r}{r^2+r_0^2}, \tag{2.31}$$

$$J_z = \frac{I_0}{\pi}\frac{r_0^2}{(r^2+r_0^2)^2}, \tag{2.32}$$

$$p = \frac{\mu_0 I_0^2}{8\pi^2}\frac{r_0^2}{(r^2+r_0^2)^2}+p_\infty. \tag{2.33}$$

When a uniform B_z is added to the problem, these profiles of B_θ, J_z, and p maintain equilibrium, but the characteristic frequencies of Eq. 2.8 are altered. From the profiles in Eqs. 2.31 - 2.33, four test cases were devised by variation of the ion and electron temperature and the magnitude of a uniform B_z.

For all four cases, the pressure profile is factored into a density profile and a temperature profile, $p=\rho k_B T$. Ignoring the constant pressure contribution $p_\infty \equiv \rho_\infty k_B T_0$, the density and temperature have the same spatial shape, as in Eq. 2.34. In all of the simulations presented in this section $\rho_\infty = 0$.

$$\rho = \frac{\mu_0 I_0^2}{8\pi^2 k_B T_0}\left(\frac{1}{r^2+r_0^2}\right)+\frac{\rho_\infty T_0}{T} \quad \text{and} \quad T = \frac{T_0 r_0^2}{r^2+r_0^2}. \tag{2.34}$$

The static Bennett equilibria initialized in the pinch simulations are not free of an electric field unless the ion temperature is zero, $T_i=0$. To see this, observe that the initial electric field from Eq. 1.1 is $\bar{E}_0 = (-\bar{\nabla}p_{e0}+\bar{J}_0\times\bar{B}_0)/e\rho_0$. The mass drift velocity is initially zero, $\bar{v}_0 \equiv 0$, so that $\bar{J}_0 = \bar{J}_e$.

parameter	value
r_w	0.5 cm
r_0	$r_w/3$
I_0	200 kA
k	0
radial grid pts.	121

Table 2.1. Equilibrium Bennett profile parameters used in initialization of the one-dimensional simulations.

From the equilibrium equation $\bar{J}_0 \times \bar{B}_0 = \bar{\nabla} p_0$ it follows that the initial electric field is zero if all of the plasma pressure is carried by the electron fluid so that $\bar{\nabla} p_{e0} = \bar{\nabla} p_0$, implying $T_i = 0$. The analysis leading to the eigenequation, Eq. 2.20, is not restricted to the electric field-free case, and is valid as long as the pressure p is the total pressure of electrons and ions. Since quasineutral plasmas are considered here, this means that the temperature T used in the eigenmode analysis must be the sum of the electron fluid temperature T_e and ion fluid temperature T_i.

Simulation results of the four test cases are placed in Table 2.2. The oscillation frequencies in the simulations are read directly from the plots of magnetic field energy such as in Figures 2.3 and 2.4. That the magnetic field energy is the appropriate diagnostic for the oscillation frequency is shown by Eq. 2.35 below. The distribution of temperature among the ions and electrons does not effect the oscillation frequency in the pinch simulations. This is predicted by linear MHD theory (Compare the first two cases in Table 2.2). For all four cases in Table 2.2 simulation and theory agree within 5%.

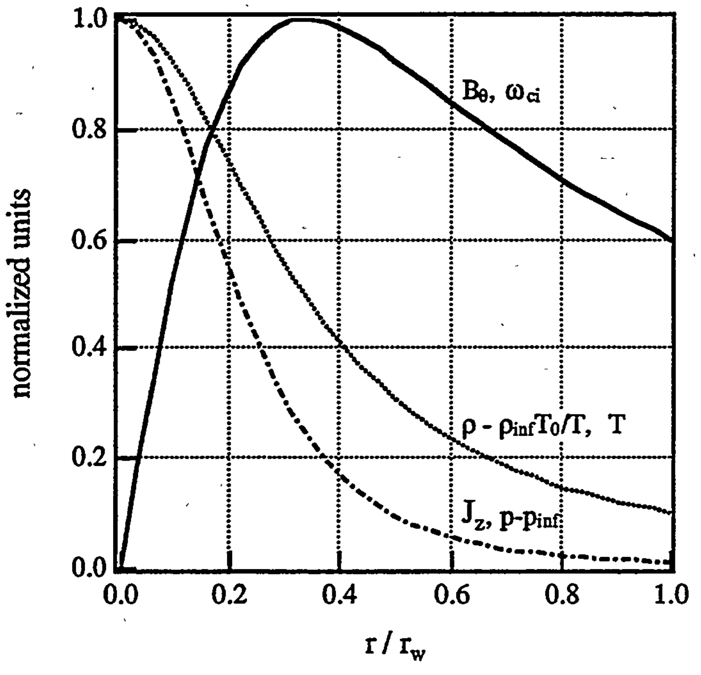

Figure 2.1. Bennett profile for stability comparisons.

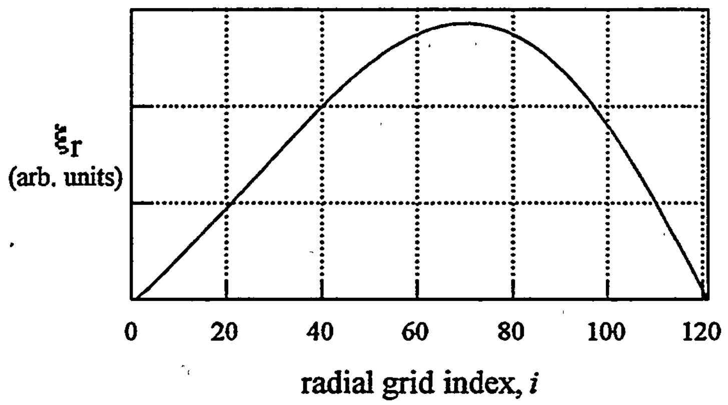

Figure 2.2. Radial displacement phasor ξ_r used to perturb the unmagnetized Bennett profile ($B_z = 0$). As a perturbation, the fluid ions are given a small outward drift velocity proportional to ξ_r. The shape of ξ_r is independent of temperature T_0 and the distribution of T_0 between ions and electrons.

temperatures	B_z (Tesla)	ω_{th} (rad/s)	ω_{sim} (rad/s)
$T_0 = 2T_{e0} = 2T_{i0} = 50eV$	0	6.59e7	6.3e7
$T_0 = T_{i0} = 50eV,\ T_{e0} = 0$	0	6.59e7	6.3e7
$T_0 = 2T_{e0} = 2T_{i0} = 100eV$	0	9.32e7	9.0e7
$T_0 = 2T_{e0} = 2T_{i0} = 50eV$	10.0	8.90e7	9.0e7

Table 2.2. Perturbed Bennett profile oscillation frequencies: ω_{th} from linear ideal MHD theory and ω_{sim} from the one-dimensional screw pinch simulation.

In Figures 2.3 and 2.4 below, the oscillation frequency of the electic field energy is twice that of the magnetic field energy. This is not surprising after analysis of the field energy contributions. Assuming small effects from nonlinearities, the magnetic field energy should have the form in Eq. 2.35.

$$\begin{aligned} U_B &\doteq \tfrac{1}{4}\int_R dzdr\, r\left(B_{\theta 0} + B_{\theta 1}\sin(\omega t)\right)^2 \\ &\doteq \tfrac{1}{4}\int_R dzdr\, r\left(B_{\theta 0}^2 + 2B_{\theta 0}B_{\theta 1}\sin(\omega t) + B_{\theta 1}^2\sin^2(\omega t)\right) . \end{aligned} \tag{2.35}$$

Since by construction the equilibrium magnetic field is much larger than the phasor perturbation, $B_{\theta 0}(r) >> B_{\theta 1}(r)$, the phasor component of the magnetic field energy should have a dominant contribution at the eigenmode frequency ω, which is thought to be the case in Figures 2.3 and 2.4.

On the other hand, the frequency of the dominant contribution in the electric field energy is difficult to discern because there are more terms through which the phasor perturbations enter. To investigate this, the electric field can be expressed as the sum of electrostatic and Hall contributions, $\bar{E} = \bar{E}_{es} + \bar{E}_{Hall}$, where $\bar{E}_{es} = -\bar{\nabla}(\rho T_e)/e\rho$ and

$\bar{E}_{Hall} = -\bar{u}_e \times \bar{B}/c$. Then the electric field energy U_E can be written $U_E = U_{E,es} + U_{E,cross} + U_{E,Hall}$, where

$$\begin{aligned} U_{E,es} &\equiv \int_R dzdr\, rE_{es}^2 \doteq \frac{1}{4e^2\rho_0^2}\int_R dzdr\, r\bar{\nabla}^2 p_e \\ &\doteq \frac{1}{4e^2\rho_0^2}\int_R dzdr\, r\left(\bar{\nabla}^2 p_0 + \bar{\nabla}p_0\cdot\bar{\nabla}p_1 \sin(\omega t) + \bar{\nabla}^2 p_1 \sin^2(\omega t)\right) , \end{aligned} \tag{2.36}$$

$$\begin{aligned} U_{E,cross} &\equiv \frac{1}{2}\int_R dzdr\, r\bar{E}_{es}\cdot\bar{E}_{Hall} \equiv \frac{1}{2ce\rho}\int_R dzdr\, r\bar{\nabla}p_e\cdot\bar{u}_e\times\bar{B} \\ &= \frac{1}{2ce\rho}\int_R dzdr\, r\bar{\nabla}p_e\cdot\left(\hat{r}u_{er} + \hat{z}u_{ez}\right)\times\hat{\theta}B_\theta \propto \frac{1}{2ce\rho}\int_R dzdr\, r\partial_r p_e u_{ez} B_\theta \\ &\propto \frac{1}{2ce\rho}\int_R dzdr\, r\partial_r\left(p_{e0} + p_{e1}\sin(\omega t)\right)\left(u_{ez0} + u_{ez1}\sin(\omega t)\right)\left(B_{\theta 0} + B_{\theta 1}\sin(\omega t)\right) , \end{aligned} \tag{2.37}$$

$$\begin{aligned} U_{E,Hall} &\equiv \int_R dzdr\, rE_{Hall}^2 \\ &\doteq \frac{1}{4}\int_R dzdr\, r\left(u_{er1}^2\sin^2(\omega t) + \left(u_{ez0} + u_{ez1}\sin(\omega t)\right)^2\right)\left(B_{\theta 0} + B_{\theta 1}\sin(\omega t)\right)^2 . \end{aligned} \tag{2.38}$$

In these electric field energy contributions, no account has been made of the phases of the various contributions. Clearly, the electric field energy contains phasor contributions at frequencies ω, 2ω, 3ω, and 4ω. The relative magnitudes of the contributions are less clear. But the 1-d simulations show that the contributions at frequency 2ω can be significant.

What type of oscillation are the 1-d plasmas exhibiting? It appears to be oscillation of a standing electromagnetic type consisting of extraordinary plasma waves counterpropagating through the inhomogeneous plasma.

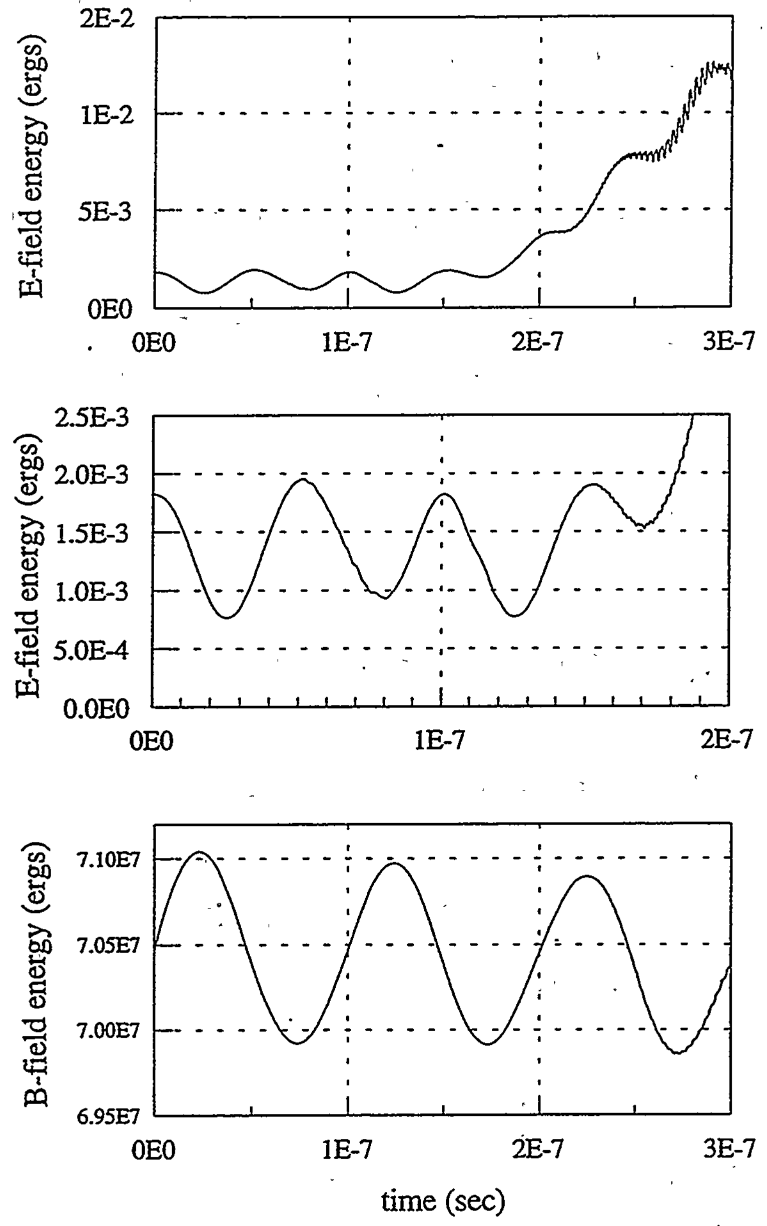

Figure 2.3. Electric field and magnetic field energy versus time for the perturbed unmagnetized Bennett profile ($B_z = 0$). Initial peak ion and electron temperatures are 25eV ($T_0 = 2T_{e0} = 2T_{i0} = 50eV$). Other equilibrium Bennett profile parameters are in Table 2.1.

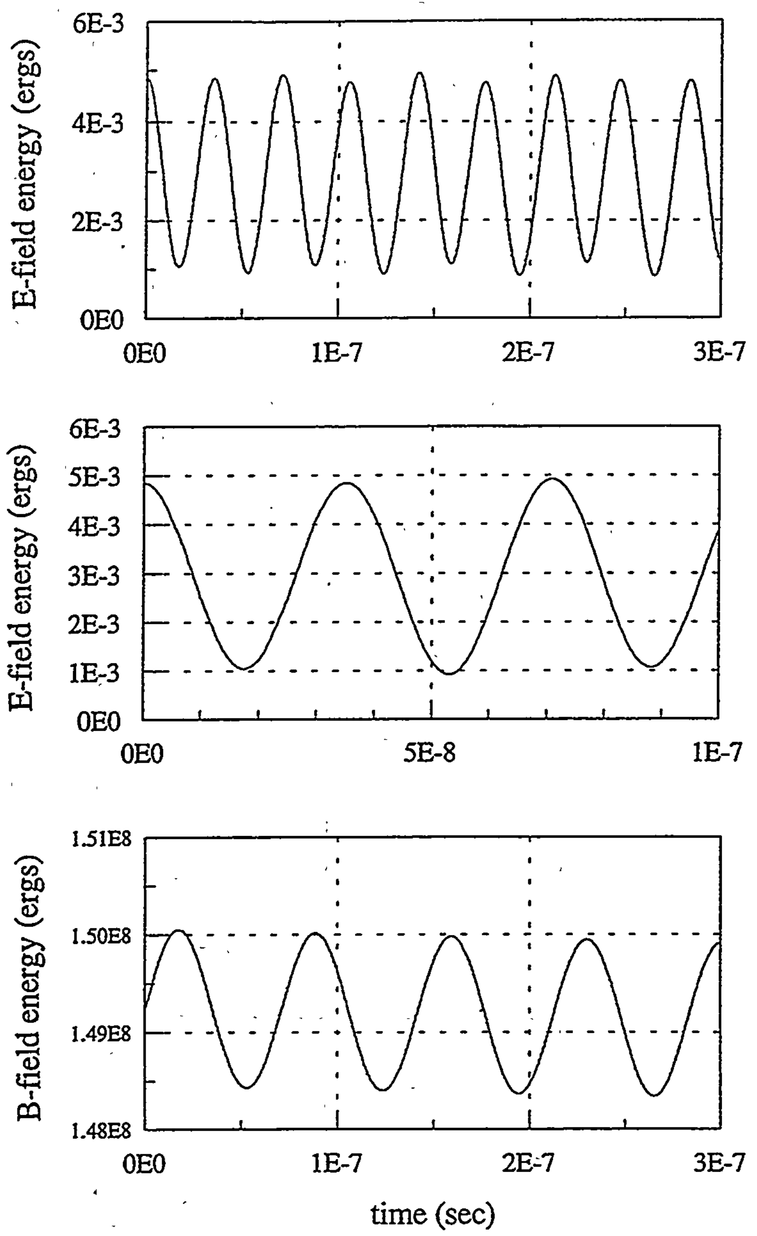

Figure 2.4. Electric field and magnetic field energy versus time for the perturbed magnetized Bennett profile ($B_z = 10T$). Initial peak ion and electron temperatures are 25eV ($T_0 = 2T_{e0} = 2T_{i0} = 50eV$). The solenoidal electric field at r_w is zeroed throughout the simulation. Other equilibrium Bennett profile parameters are in Table 2.1.

2.2.2. Two-dimensional simulations initialized with Bennett profiles that have sausage instabilities

In 2-d rz-coordinates, pinch simulations were tested against stability theory using the same initial Bennett profiles as in the 1-d simulations. In each simulation, the ion drift velocity was initially perturbed by a displacement phasor $\bar{\xi}_0$ of an unstable eigenmode. Characteristic frequencies of the sausage modes ($k>0$, $m=0$) were obtained by numerical solution of the full eigenmode equation, Eq. 2.20, just as for the $k=0$ modes. For the equilibria, an on-axis temperature of $T_0=100\text{eV}$ was chosen in order to make the e-folding times and the oscillation periods less than a few thousand simulation time steps. In the simulations, periodic boundary conditions were applied in the axial direction.

parameter	value
r_w	0.5 cm
r_0	$r_w/3$
I_0	200 kA
$T_0=2T_i=2T_e$	100eV
k	8π cm^{-1}
grid size, $N_r \times N_z$	33×17

Table 2.3. Simulation parameters common to Bennett profile test cases 1-4.

Again, uniform axial magnetic fields of varying strength were applied, affecting only the characteristic frequencies of the equilibria. To investigate a predicted transition to stability, axial magnetic fields of 0T, 0.5T, 5T, and 7T were each initialized with the

Bennett profile. For axial fields of 0T and 0.5T, the longest axial mode of the simulation grid has several different stable and unstable radial modes according to theory. At 5T, only one radial mode is unstable, while at 7T there are no unstable radial modes. Some characteristic frequencies for the longest axial mode of the simulation grid ($\lambda = r_w/2$) are given in Table 2.4. The l-numbers are equal to the number of nodes in ξ_r between the

test case		1	2	3	4
B_z		0	0.5 Tesla	5.0 Tesla	7.0 Tesla
$\omega_{ci,\ \mathrm{max}}$		1.15e9 s^{-1}	1.15e9 s^{-1}	1.25e9 s^{-1}	1.33e9 s^{-1}
$\omega_{th,gr}$	$l=0$	2.21e7 s^{-1}	2.88e7 s^{-1}	3.55e7 s^{-1}	none
	$l=1$	1.64e7 s^{-1}	2.31e7 s^{-1}	none	none
	$l=2$	1.31e7 s^{-1}	1.96e7 s^{-1}	none	none
$\omega_{th,osc}$	$l=0$	2.96e7 ras/s	3.32e7 rad/s	none	3.70e7 rad/s
		3.52e8 rad/s	3.52e8 rad/s	3.81e8 rad/s	3.93e8 rad/s
	$l=1$	1.23e7 rad/s	2.51e7 rad/s	2.92e7 rad/s	6.49e7 rad/s
		3.83e8 rad/s	3.84e8 rad/s	3.83e8 rad/s	4.02e8 rad/s
$\omega_{sim,gr}$		2.0e7 s^{-1}	2.7e7 s^{-1}	3.5e7 s^{-1}	chaotic
$\omega_{sim,osc}$		3.6e8 rad/s	3.7e8 rad/s	3.8e8 rad/s	chaotic

Table 2.4. Comparison of instability growth rates and oscillation frequencies from a linear MHD eigenmode analysis ($\omega_{th,gr}$ and $\omega_{th,osc}$) and from pinch simulations with fluid ions ($\omega_{sim,gr}$ and $\omega_{sim,osc}$). Quantities $\omega_{sim,gr}$ and $\omega_{sim,osc}$ are characteristic frequencies from the inner product $\langle \bar{\xi}_0, \bar{u}_{ir} \rangle$ defined in Eq. 2.41. Integer numbers l are radial mode indices.

axis and the wall. For each equilibrium and each axial wavenumber k, there is the possibility of multiple stable and unstable radial modes with different mode number l and characteristic frequency ω. As shown by Table 2.4, there can be multiple stable modes with the same number of nodes between the axis and the wall, at least for $l=0$ and $l=1$.

It is interesting to note that the Bennett equilibria have a mixture of stable and unstable eigenmodes with comparable magnitudes of characteristic frequency. This leads one to suspect that the stable and unstable modes have a significant effect on the evolution of each other -- after all, the governing equations are actually nonlinear, and the stable modes are not orthogonal to the unstable modes.

To extract the characteristic frequencies ω from the simulation, the electric field energy was initially used as a diagnostic. But after realizing that the electric field energy might contain significant contributions that grow at the rates ω, 2ω, 3ω, and 4ω, it was determined that a better growth rate diagnostic was necessary. A better diagnostic came from the orthonormality relation for the $\bar{\xi}$-modes, which for normalized $\bar{\xi}_m$ and $\bar{\xi}_n$ is given by Eq. 2.40.

$$2\pi\int_{rz} dzdr\, r\rho_0 \bar{\xi}_m \cdot \bar{\xi}_n = \delta_{mn} \tag{2.40}$$

Here, ρ_0 is the equilibrium mass density and δ_{mn} is the Kronecker delta. Since the ion drift velocity $\bar{u}_i$ can be written as a linear combination of the $\bar{\xi}$-modes, the inner product of $\bar{\xi}_0$ and $\bar{u}_i$ (induced by Eq. 2.40) should exhibit the growth rate of the $\bar{\xi}_0$-mode. This inner product is defined to be:

$$\left\langle \bar{\xi}_0, \bar{u}_{ir} \right\rangle \equiv 2\pi\int_{rz} dzdr\, r\rho_0 \bar{\xi}_0 \cdot \bar{u}_{ir} \tag{2.41}$$

In Table 2.4, $\omega_{sim,gr}$ and $\omega_{sim,osc}$ are growth rates and oscillation frequencies observed in the inner product of $\bar{\xi}_0$ and $\bar{u}_i$, defined in Eq. 2.41.

In each simulation of Table 2.4, except for the 7T case, an attempt was made to perturb the plasma with the exponentially unstable $l = 0$ mode. In the 7T case the plasma was perturbed with the unstable $l = 0$ mode from MHD theory for the 5T case. For the

0T, 0.5T and 5T cases these perturbations have theoretical growth rates $\omega_{th,gr}$ of 2.21e7, 2.88e7 and 3.55e7 sec^{-1}, respectively.

Theoretical growth rates and growth rates from the simulation agree within 10%. This narrows the range of mistakes that could have been made in the computer code of the pinch simulation. It also shows the worth of comparison to ideal MHD stability theory, and it computationally verifies that stability is independent of the magnitude of the Hall term and diamagnetic drift term in Ohm's law.

Without a more careful analysis of the simulations, it seems plausible that the simulated growth rates for the 0T, 0.5T, and 5T cases are $5-10\%$ low because of mode coupling. This is suggested by the spatial behavior of the radial ion drift velocity u_{ir}, in the 0T and 0.5 T cases, which quickly changes shape on time scales shorter than the theoretical growth rates. In the 5T case, spatial plots of u_{ir} suggest that mainly the seeded mode grows until the first peak in Figure 2.9, which occurs at roughly 130ns. For B_{z0}=7T, the profile was perturbed with an unstable mode of the 5T case. The plots for B_{z0}=7T suggest relative stability, because the instabilities that eventually grow are temporally delayed, and have shorter spatial wavelengths than the seed.

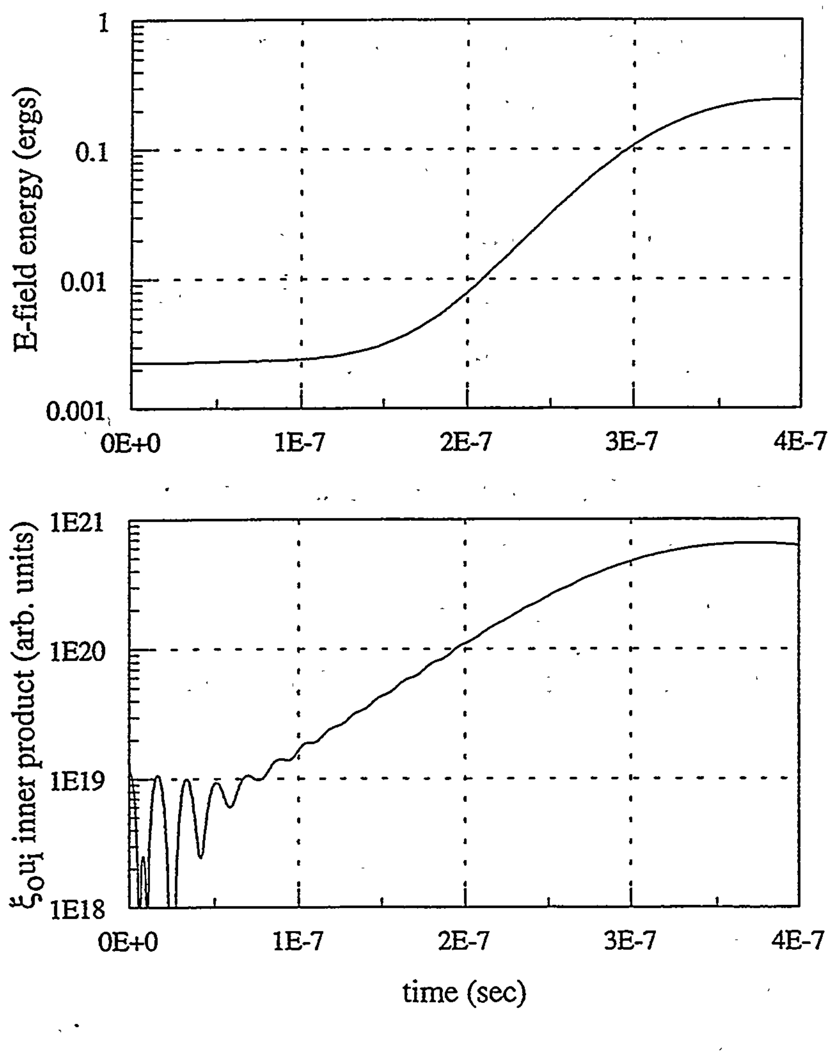

Figure 2.5. Electric field energy and $\left\langle \bar{\xi}_0, \bar{u}_i \right\rangle$ inner product as a function of time for a two-dimensional simulation with an initial axial magnetic field of 0T.

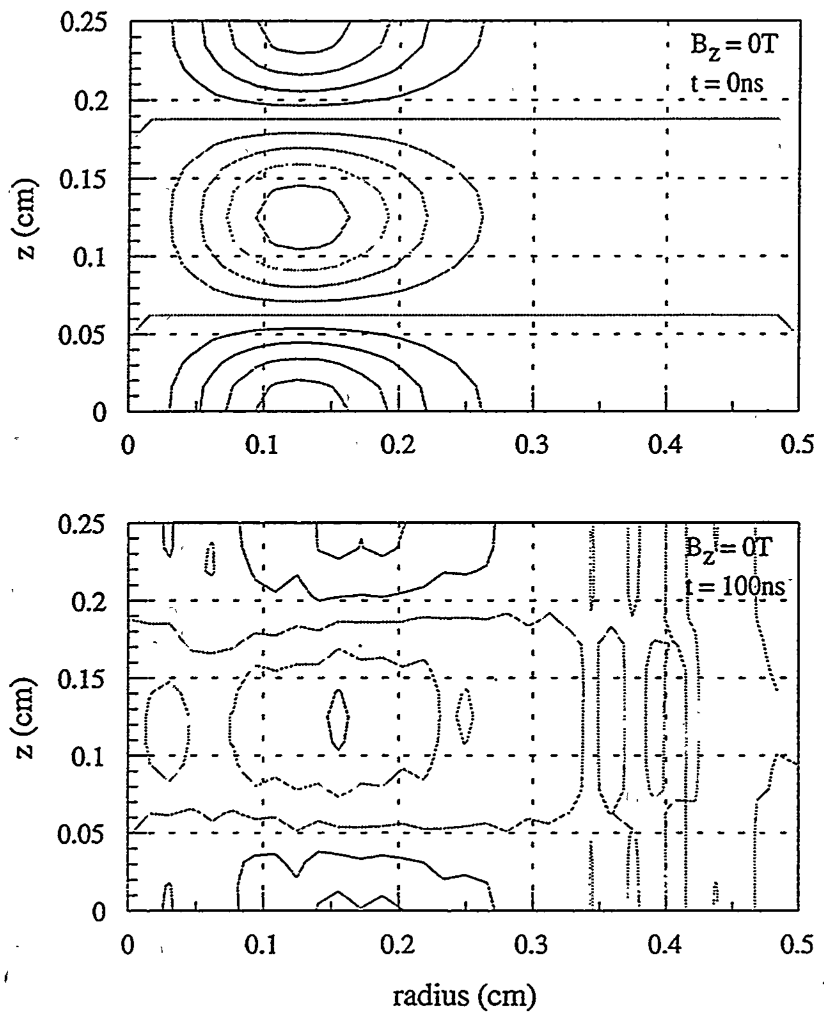

Figure 2.6. Spatial contour plots of ion radial drift velocity u_{ir} at times $t = 0\text{ns}$ and $t = 100\text{ns}$ for a two-dimensional simulation with an initial axial magnetic field of 0T.

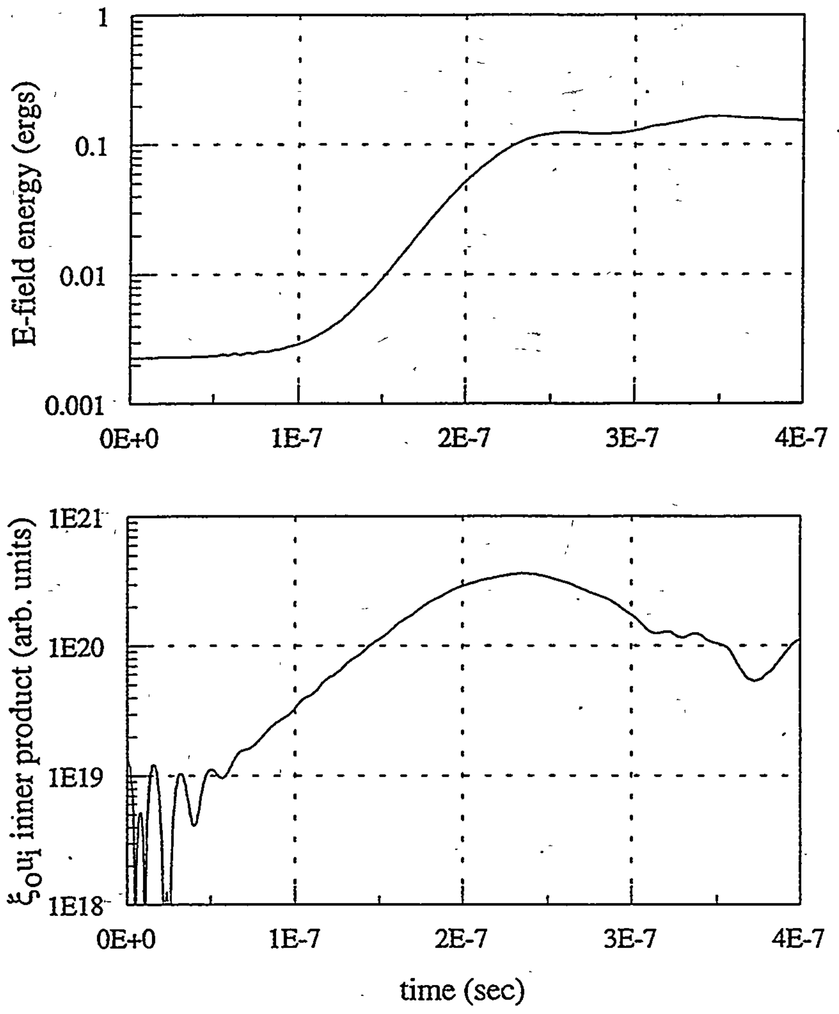

Figure 2.7. Electric field energy and $\langle \bar{\xi}_0, \bar{u}_i \rangle$ inner product as a function of time for a two-dimensional simulation with an initial axial magnetic field of 0.5T.

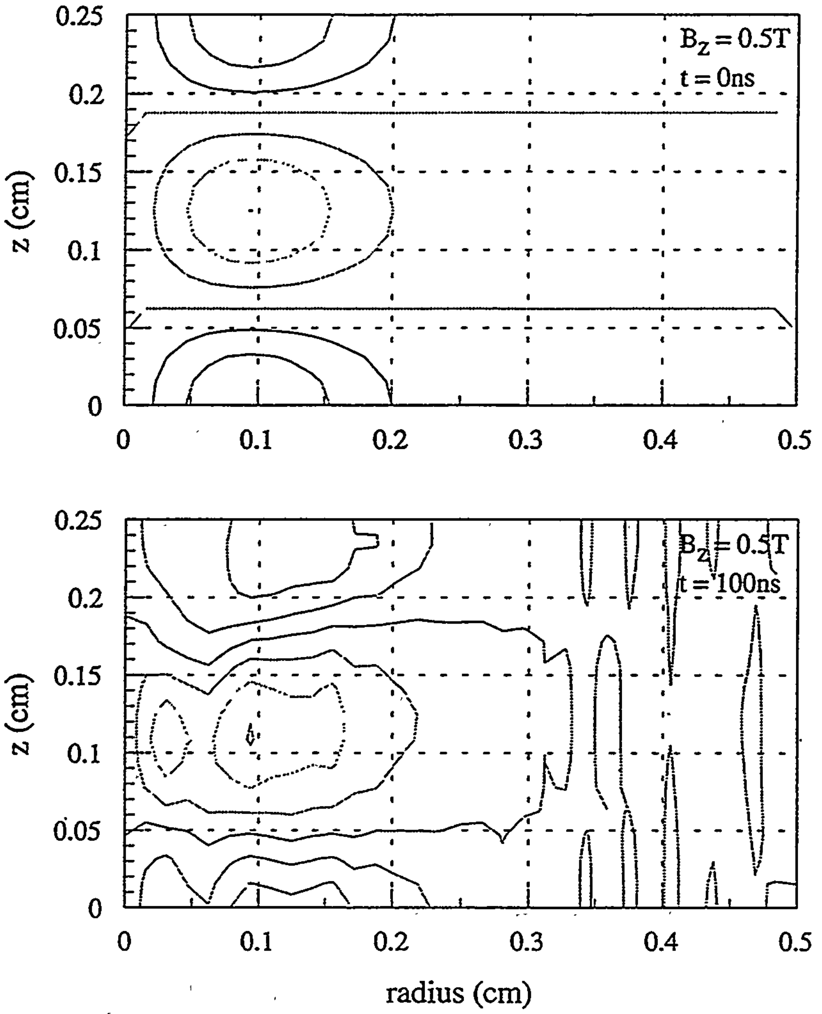

Figure 2.8. Spatial contour plots of ion radial drift velocity u_{ir} at times $t = 0$ns and $t = 100$ns for a two-dimensional simulation with an initial axial magnetic field of 0.5T.

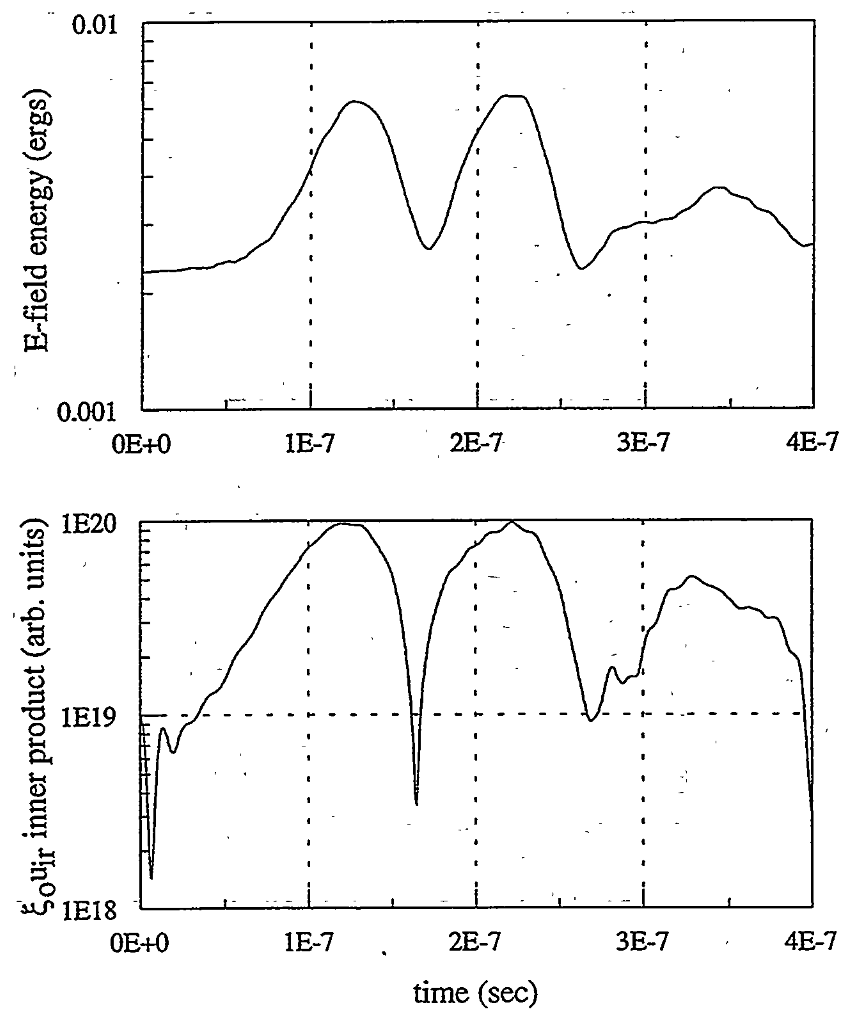

Figure 2.9. Electric field energy and $\left\langle \bar{\xi}_0, \bar{u}_i \right\rangle$ inner product as a function of time for a two-dimensional simulation with an initial axial magnetic field of 5T.

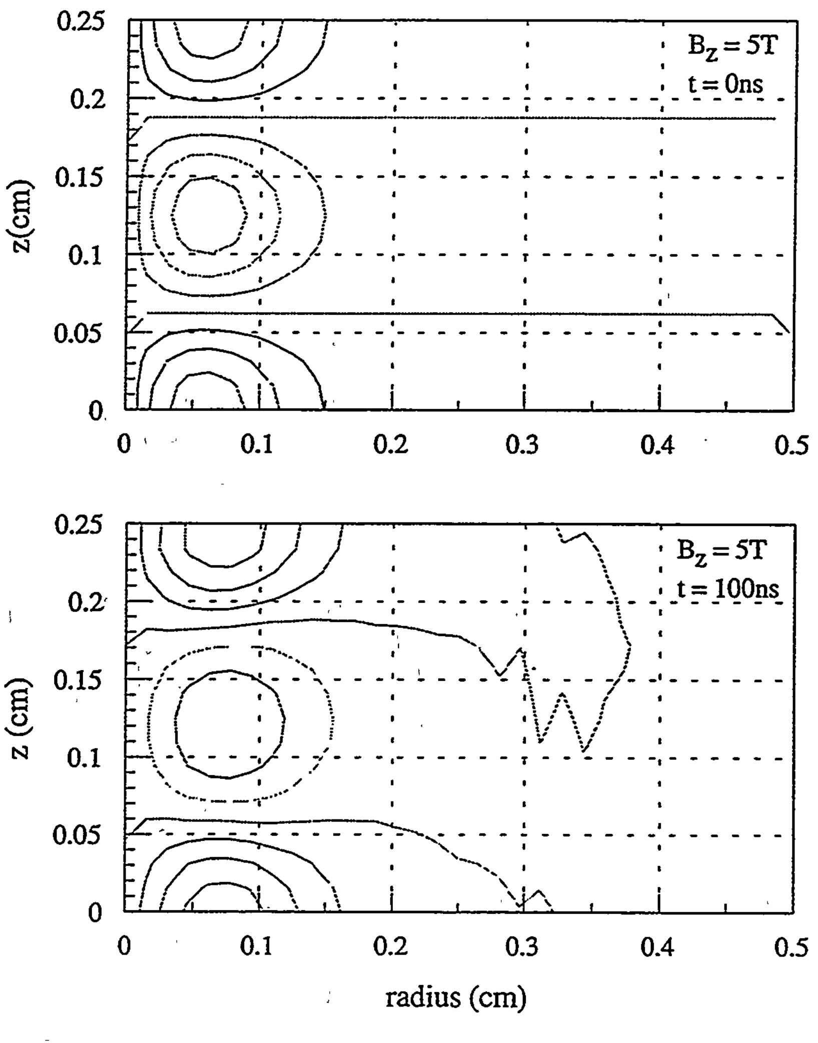

Figure 2.10. Spatial contour plots of ion radial drift velocity u_{ir} at times $t = 0\text{ns}$ and $t = 100\text{ns}$ for a two-dimensional simulation with an initial axial magnetic field of 5T.

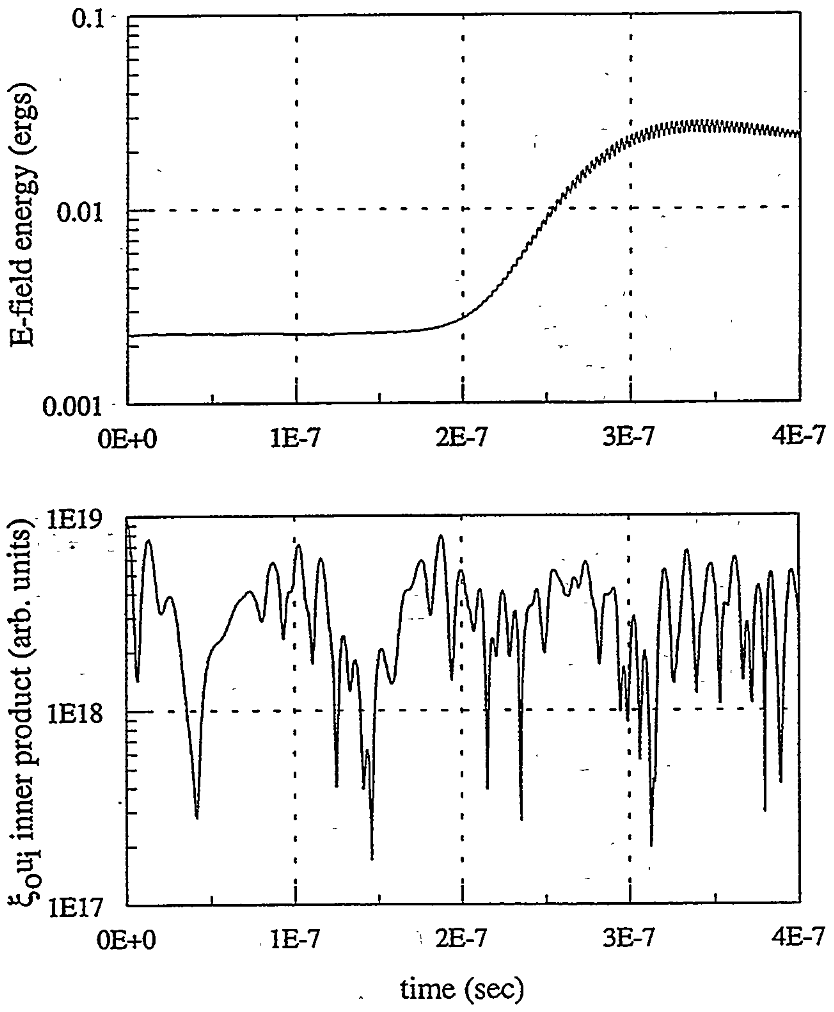

Figure 2.11. Electric field energy and $\left\langle \bar{\xi}_0, \bar{u}_i \right\rangle$ inner product as a function of time for a two-dimensional simulation with an initial axial magnetic field of 7T.

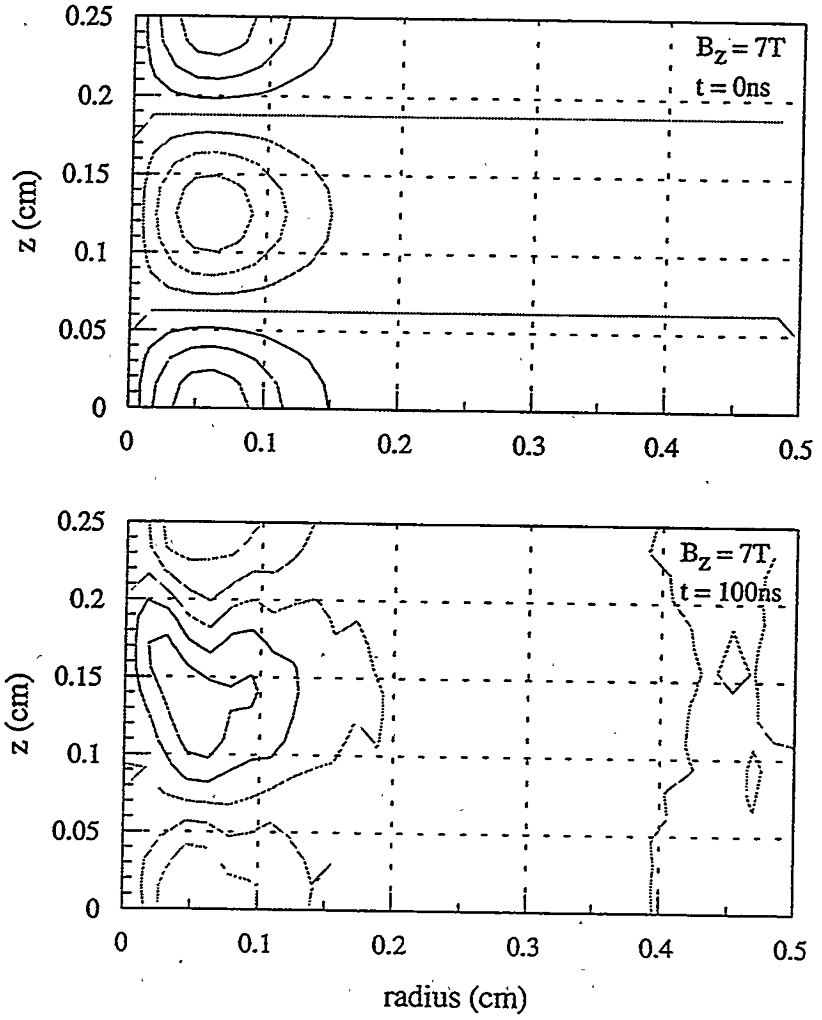

Figure 2.12. Spatial contour plots of ion radial drift velocity u_{ir} at times $t = 0$ns and $t = 100$ns for a two-dimensional simulation with an initial axial magnetic field of 7T.

2.3 Strongly driven test cases

Pinch simulations are certainly not restricted to MHD equilibrium studies. Indeed, simulations are largely motivated by zeta- and theta-pinch experiments involving fast compression of plasmas. High compressions have been the goal for applications in both nuclear fusion research and X-radiation sources. Plasmas in such experiments are far from equilibrium.

2.3.1 One-dimensional simulation of a zeta-pinch

In order to compare the simulation of a zeta-pinch with earlier computational work, a simulation with parameters in Table 2.5 was implemented. The simulation was initialized with a uniform density of 100% ionized deuteron plasma (ion mass m_i=3.346e-24 kg), and at time t=0 an axially directed plasma current I_z=250kA was applied to produce a magnetic flux density B_θ=0.5T at the wall radius r_w=10cm. First, the plasma separates from the wall, leaving a vacuum region between the wall and the plasma, and then the plasma compresses. In this particle-in-cell (PIC) simulation, particles reacting to the radial electric field at the plasma-vacuum interface are sent streaming radially inward at the implosion velocity. The pinch simulation of this dissertation and earlier simulations [Hewett 1980] are shown to yield essentially the same results.

Just as described in Chapter 1, simulaton regions with ion density below a cutoff density ρ_{cutoff} are declared vacuum regions. Then for purposes of advancing the electron fluid and fields, the density is assumed to be ρ_{cutoff} in the vacuum regions, thus avoiding division by extremely small densities and the associated numerical instabilities.

parameter	value	parameter	value
r_w	10 cm	η_{vac}	1 e-12 s
$T_e = T_{i0}$	1eV	η_{pl}	1 e-17 s
I_z	250 kA	m_i	3.346 e-24 g
ρ_0	3 e14 cm^{-3}	N_i	10,000
ρ_{cutoff}	3 e12 cm^{-3}	radial grid pts.	41

Table 2.5. Parameters used in the one-dimensional zeta-pinch simulation.

In Table 2.5 ρ_0 is the particle density of the initially uniform plasma. The resistivity used in the dense plasma regions is η_{pl}, and the resistivity used for all vacuum points is η_{vac}. The number of PIC ions used in the simulation is denoted N_i.

There is a significant difference between the present simulation and past simulations in regard to resistivity and smoothing at plasma-vacuum interfaces. A past hybrid simulation used 5-point spatial smoothing of density in the numerator of update formulas at every ion time step [Hewett 1980]. The smoothing was used to diminish effects of the discrete grid at plasma-vacuum interfaces. It was applied after the points on the grid were determined to be inside or outside the plasma. Typically, the inverse density $(1/\rho)_{i,j}$ was replaced by an inverse of an averaged density over nearby grid points:

$$8/\left(4\rho_{i,j}+\rho_{i-1,j}+\rho_{i+1,j}+\rho_{i,j-1}+\rho_{i,j+1}\right) \to (1/\rho)_{i,j}.$$

This inverse averaged density replaced the inverse density in the update formulas, such as Eqs. 1.8, and tended to diminish the electron fluid current at low-density plasma points. The result is a decrease in the the Hall and diamagnetic drift contributions (Eq. 1.1) that spatially smooths the interfacial electric field as the plasma advances across the grid. Unfortunately, the smoothing is nonphysical.

The new simulation does not use such spatial smoothing. Instead, it uses an interpolated resistivity that helps diminish grid effects in a more physical manner. The prescription for the resistivity is simple. For plasma densities above $0.2\rho_0$, the resistivity is set to η_{pl}, and for regions with densities below ρ_{cutoff}, the resistivity is set to η_{vac}. Between ρ_{cutoff} and $0.2\rho_0$ the resistivity is linear in the inverse of the plasma density; a reasonable approximation to the increased collisionality in denser regions. The linear patch is forced to be continuous. The spatially dependent resistivity effects the B_θ rate equation in a way that forces more current through the denser plasma regions. The lower electron fluid drift velocities at low-density plasma points allow the plasma to move more smoothly across the finite-difference grid.

The velocity components as a function of radius (Figs. 2.14 and 2.18) from the new simulation are in close agreement with earlier simulations with the same parameters (Figs. 2.15 and 2.19). The main difference is the small "barb" that appears near the inward edge of the inward streaming particles. This barb is a consequence of initial plasma "blowoff" from the outer wall, and is not a surprising result, since treatment of the plasma-wall interaction is tricky. No special effort was made to ensure that the simulations give the same treatment of the plasma-wall interaction.

Note the high-frequency transients in the electric field energy of Figure 2.16. These are nonphysical transients resulting from advancement of the plasma-vacuum interface across the discrete finite-difference grid, and the interpolated resistivity mentioned above was an attempt to diminish the transient amplitude. Future effort will be spent on reduction of such transients because they yield a nonphysical velocity spread that might have a serious effect on the peak particle density calculated on axis.

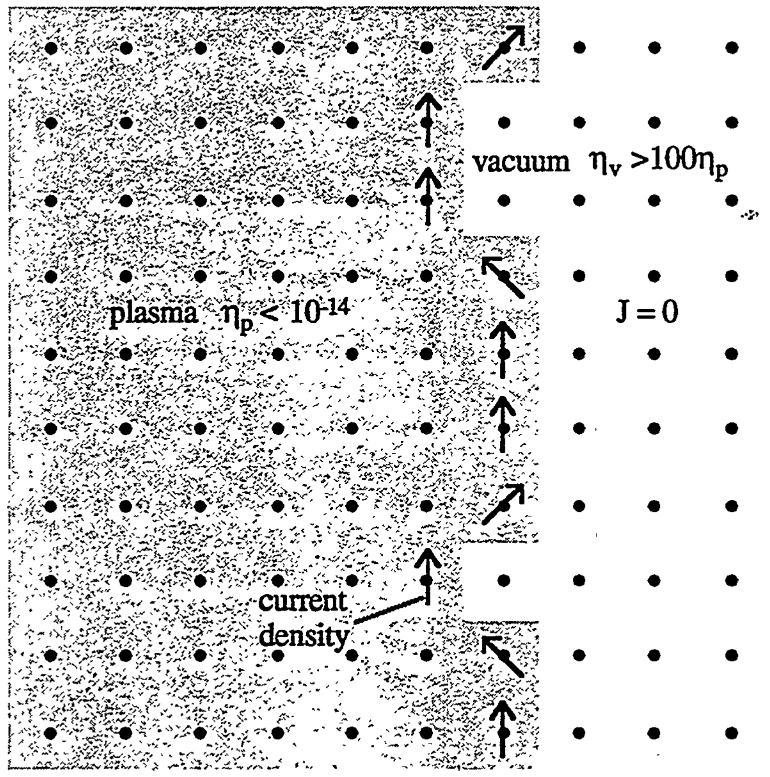

Figure 2.13. Spatially varying resistivity across a plasma-vacuum interface confines current in the plasma. Plasma points such as those through which arrows are drawn can have densities approaching cutoff and are amenable to resistivity smoothing techniques.

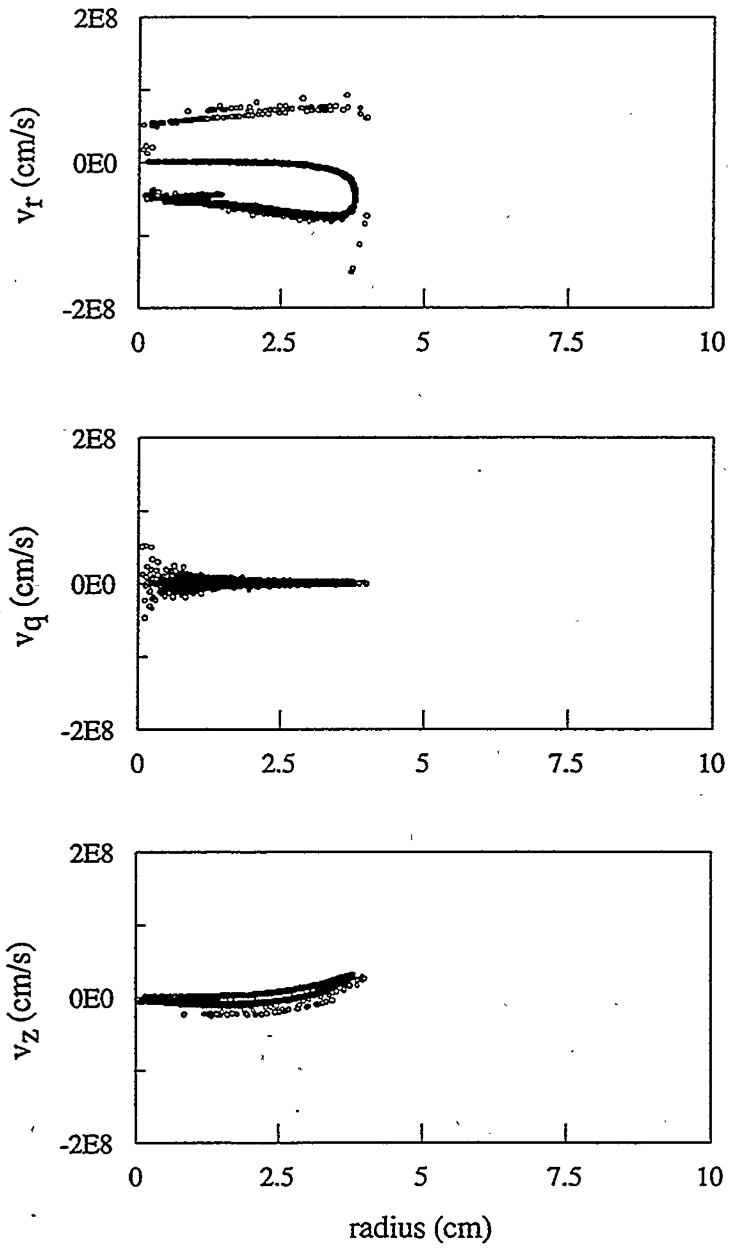

Figure 2.14. Radial, azimuthal, and axial particle velocities versus radius for the Z-pinch with parameters in Table 2.5.

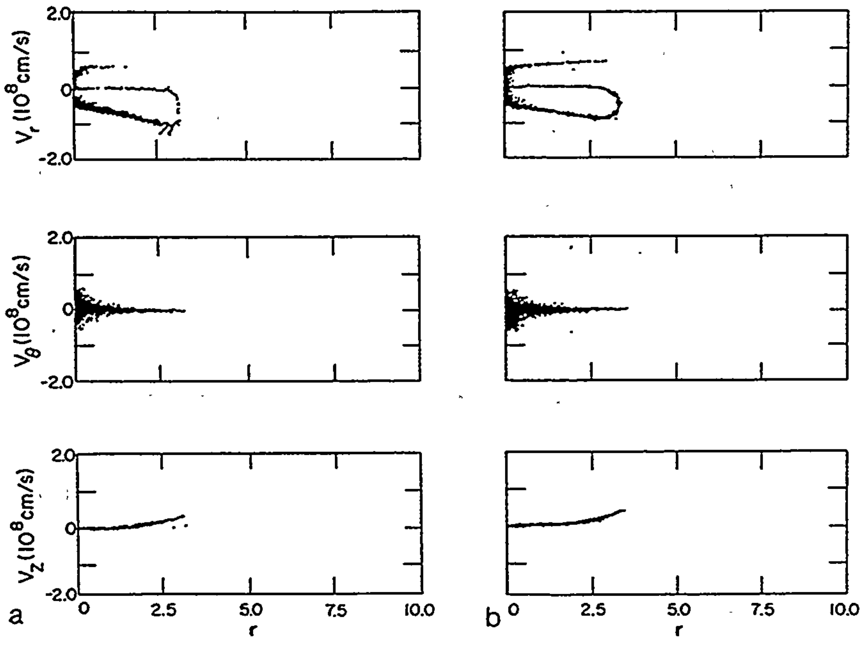

FIG. 2. Result of an implosion driven by a 5-kG external B_θ field with other conditions remaining similar to those described in Fig. 1. (a) Result obtained using the one-dimensional Sgro-Nielson algorithm (courtesy of A. G. Sgro). (b) Result obtained using the two-dimensional algorithm described in the text.

Figure 2.15. Radial, azimuthal, and axial particle velocities versus radius from earlier simulations of a Z-pinch with parameters in Table 2.5. Courtesy of D.W. Hewett [1980].

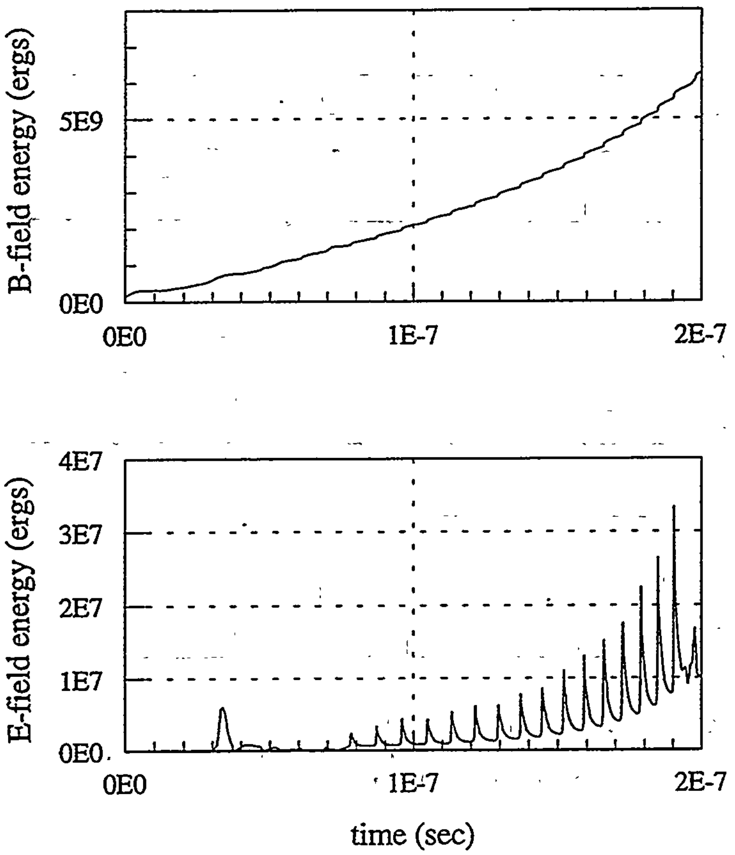

Figure 2.16. Magnetic and electric field energies versus time for the Z-pinch simulation with parameters in Table 2.5.

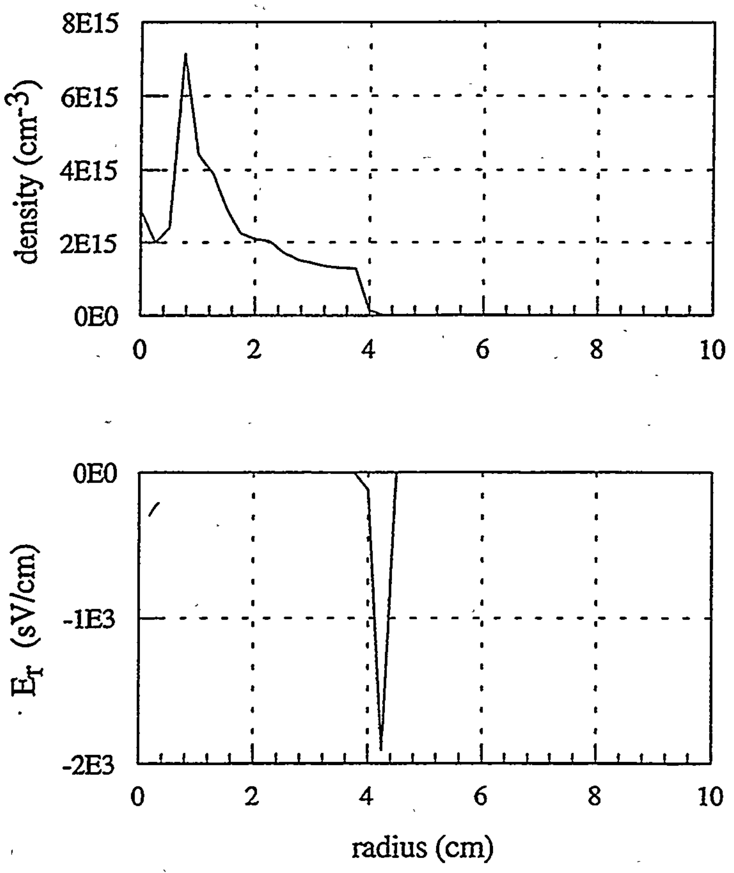

Figure 2.17. Density and radial electric field versus radius at time t=200ns for the Z-pinch simulation with parameters in Table 2.5.

2.3.2 One-dimensional theta-pinch simulation

For comparison with past simulations, a theta-pinch simulation with parameters in Table 2.6 was implemented. Again, the simulation was initialized with a uniform density of 100% ionized deuteron plasma (ion mass m_i=3.346e-24 kg). At time $t=0$ an azimuthal electric field $E_{\theta,w}$ was applied at the wall to produce a vacuum axial magnetic field B_z of roughly 0.5T between the plasma and the wall at r_w=10cm. Again, the plasma separates from the wall and particles reacting to the radial electric field at the plasma-vacuum interface are sent streaming radially inward at the implosion velocity.

parameter	value	parameter	value
r_w	10 cm	η_{vac}	1 e-12 s
$T_e = T_{i0}$	1eV	η_{pl}	1 e-17 s
$E_{\theta,w}$	2 sV/cm	m_i	3.346 e-24 g
ρ_0	3 e14 cm^{-3}	N_i	10,000
ρ_{cutoff}	3 e12 cm^{-3}	radial grid pts.	41

Table 2.6. Parameters used in the one-dimensional theta-pinch simulation.

While the axial magnetic field in the vacuum was not *fixed* at 5kG throughout the simulation, a value of $E_{\theta,w}$ was chosen to keep the field within 10% of 0.5T throughout most of the simulation. Then the results can be compared with earlier simulations [Hewett 1980]. Other parameters in Table 2.6 are defined in Section 2.3.1.

The implosion velocity in Figure 2.18 is within 10% of that of the earlier simulations shown in Figure 2.19 [Hewett 1980]. These also agree with results of Sgro and Nielson [1976]. Note that the velocity spread in Figure 2.18 due to field transients is

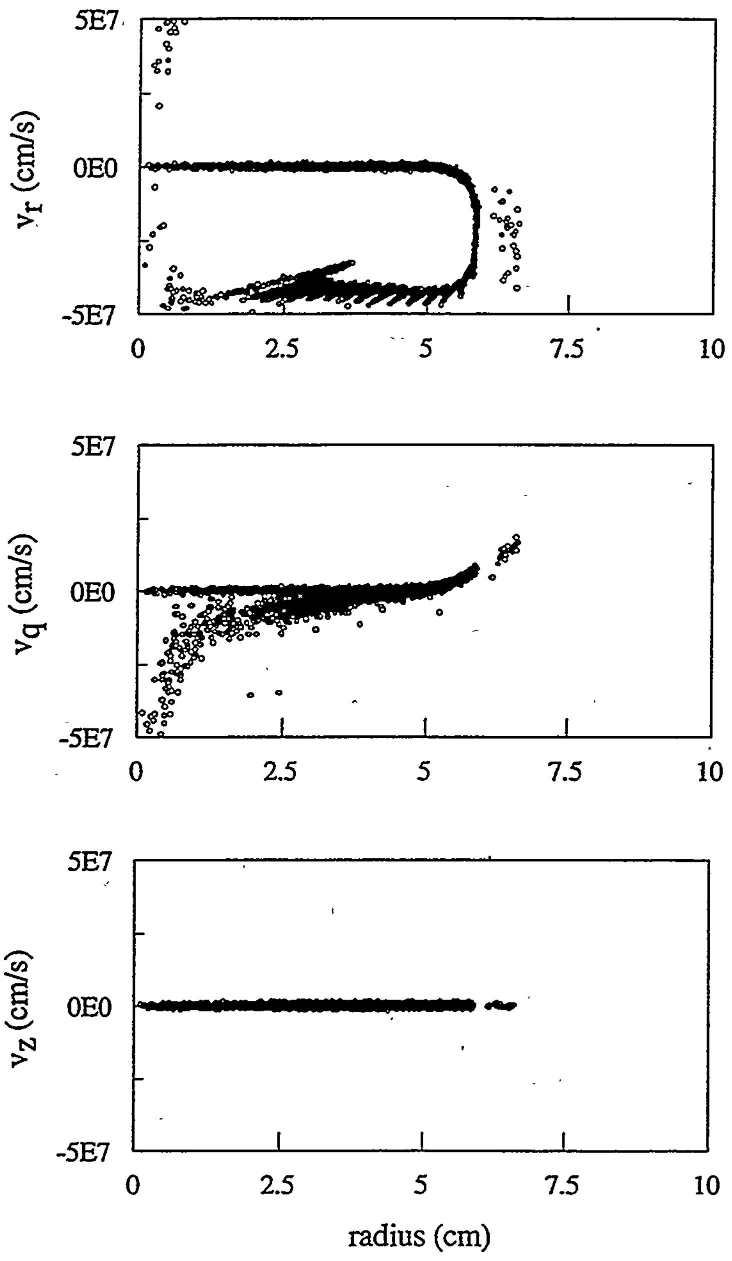

Figure 2.18. Radial, azimuthal, and axial particle velocities versus radius for the theta-pinch simulation with parameters in Table 2.6.

relatively greater than in the Z-pinch case. The "barb" on the inward-moving ion beam is also more apparent. Also note in Figure 2.18 the outer annulus of particles that have separated from the bulk plasma. It is likely that the axial magnetic field diffused through

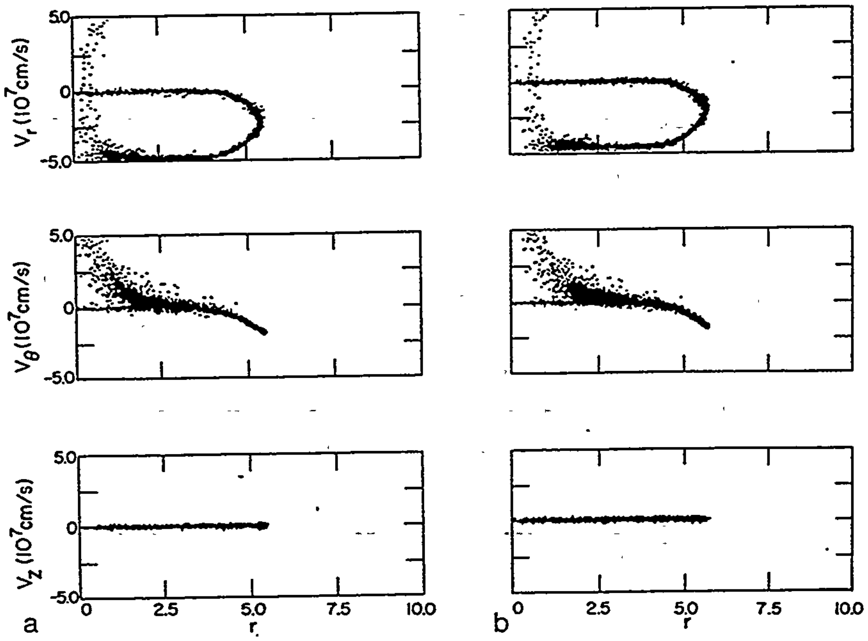

FIG. 1. The ion phase space that results after 200 ηsec. The $t=0$ configuration consisted of a homogeneous 1-eV deuterium plasma with a density of 3×10^{14} which is subjected to an external B_z implosion field of 5 kG. (a) Result obtained using the one-dimensional algorithm of Sgro and Nielson (courtesy of A. G. Sgro). (b) Result obtained using the two-dimensional algorithm described in the text.

Figure 2.19. Radial, azimuthal, and axial particle velocities versus radius from earlier simulations of a theta-pinch with parameters in Table 2.6. Courtesy of D.W. Hewett [1980].

these outer particles which became "trapped" on the field lines. This is more likely to happen when the resistivity is interpolated in the plasma in order to smooth discrete effects of the grid. In Figure 2.20, transients in the electric field energy are relatively smaller than for the zeta-pinch case. This is primarily due to the presence of a substantial

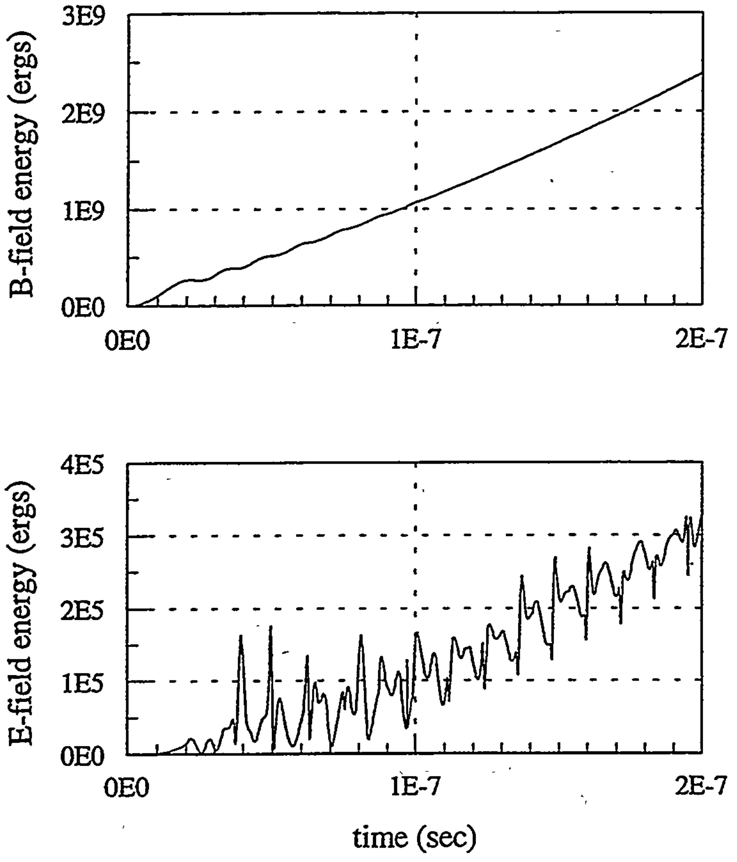

Figure 2.20. Magnetic and electric field energies versus time for the theta-pinch simulation with parameters in Table 2.6.

E_r in the extrained vacuum region because $u_{e\theta} = u_{i\theta}$ is not enforced in the vacuum of this simulation. Otherwise this would seem surprising since the transient velocity spread indicates significant transient electric fields.

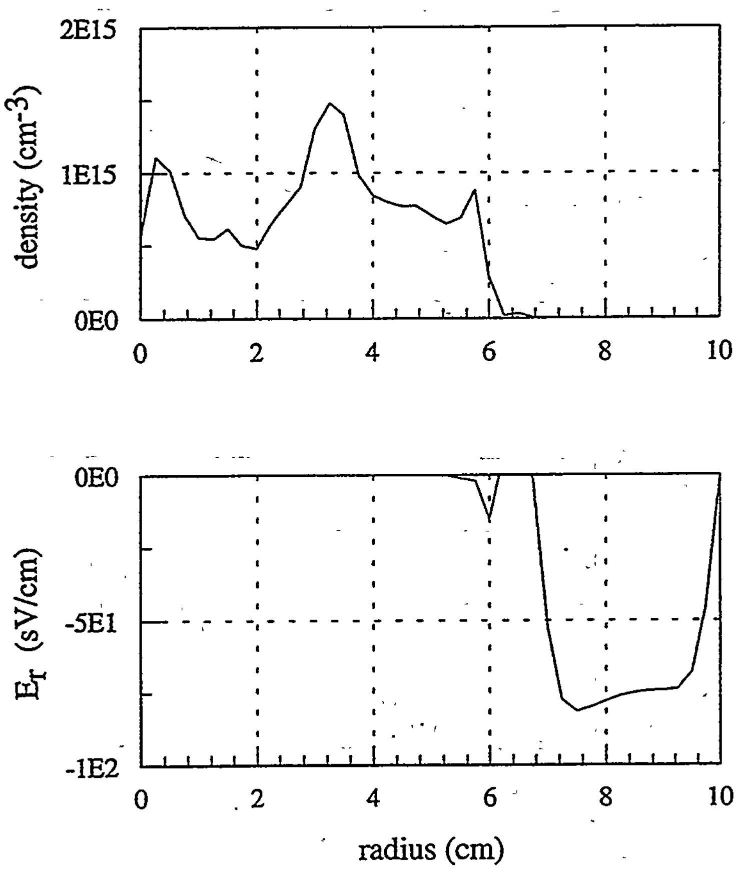

Figure 2.21. Density and radial electric field versus radius at time t=200ns for the theta-pinch with parameters in Table 2.6.

2.4 Entrained vacuum void treatment of B_θ

One of the most significant contributions of this dissertation is a new azimuthal magnetic field update method for entrained vacuum void regions. Previous hybrid algorithms used an ADI method to relax toward the solution of $(\nabla^2\bar{B})_\theta = 0$ in each entrained vacuum void [Hewett 1980]. For details on this previous method, refer to Method 2 of the B_θ finite difference equations in Appendix 1.A3. Such methods drove the curl of the current density to zero in the plasma surrounding the vacuum void, which is nonphysical. In general $\bar{\nabla}\times\bar{J}\neq 0$ except in vacuum regions away from the plasma, where $\bar{J}=0$ to a good approximation (see Figure 1.3 to visualize this).

The new update method, Method 1, is based on the integral form of Faraday's law, as described in Chapter 1. At each simulation time step the line integral of the poloidal electric field is calculated around the boundary of each and every entrained vacuum void region (see Figure 1.2 to visualize this). The line integrals yield good estimates of the time derivative of the angular magnetic field B_θ in the voids, and hence can be used to update B_θ in each void. While the line integrals require more computation, the curl of the current density is not driven to zero in the plasma surrounding the void, potentially making the method more accurate. For details, refer to Method 1 of the B_θ finite difference equations in Appendix 1.A3.

The purpose of this subsection is to compare simulations using different vacuum void treatments for B_θ, Method 1 and Method 2. Each method is used for simulation of a dynamic Z-pinch with an initially uniform plasma density and without axial magnetic fields. Early in the simulation, entrained voids are not present, so that the simulations evolve in exactly the same way. Later in the simulation voids appear near the plasma-vacuum interface due to instabilities. These instabilities are a combination of equilibrium sausage instabilities and magneto-Rayleigh-Taylor (MRT) instabilities from acceleration of the plasma-vacuum interface. Because the plasma behavior is nonlinear and turbulent, small differences in evolution of the voids leads to dynamic states that are significantly different later in the simulation; qualitatively, this is what is observed.

The parameters defining the simulation are in Table 2.7. The number of particles used in the simulation fits in the 32Mbyte memory of a Pentium PC (32 bit single precision coordinates and velocity components). Storage for field quantities on the grid is negligible compared to the storage used for the particle locations and velocities; typical for hybrid simulations with particle-in-cell (PIC) ions. Except for the added axial dimension, the simulation parameters are the same as for the 1-d Z-pinch simulation of Section 2.3.1 with PIC ions.

parameter	value	parameter	value
space size, $r_w \times L_z$	10×2.5 cm	η_{vac}	1 e-12 s
$T_e = T_{i0}$	1eV	η_{pl}	1 e-16 s
I_z	250 kA	m_i	3.346 e-24 g
ρ_0	3 e14 cm^{-3}	N_i	400,000
ρ_{cutoff}	3 e12 cm^{-3}	*grid size,* $N_r \times N_z$	65×17

Table 2.7. Parameters used in the two-dimensional Z-pinch simulations for comparing entrained vacuum void treatments of B_θ.

The pinch was simulated for an elapsed time of 200ns. Up to 140ns, the field energies of two simulations track closely. By t=150ns, the evolution of the plasma from case to case begins to differ; entrained vacuum voids near the plasma-vacuum interface lead to slightly different spatial density plots.

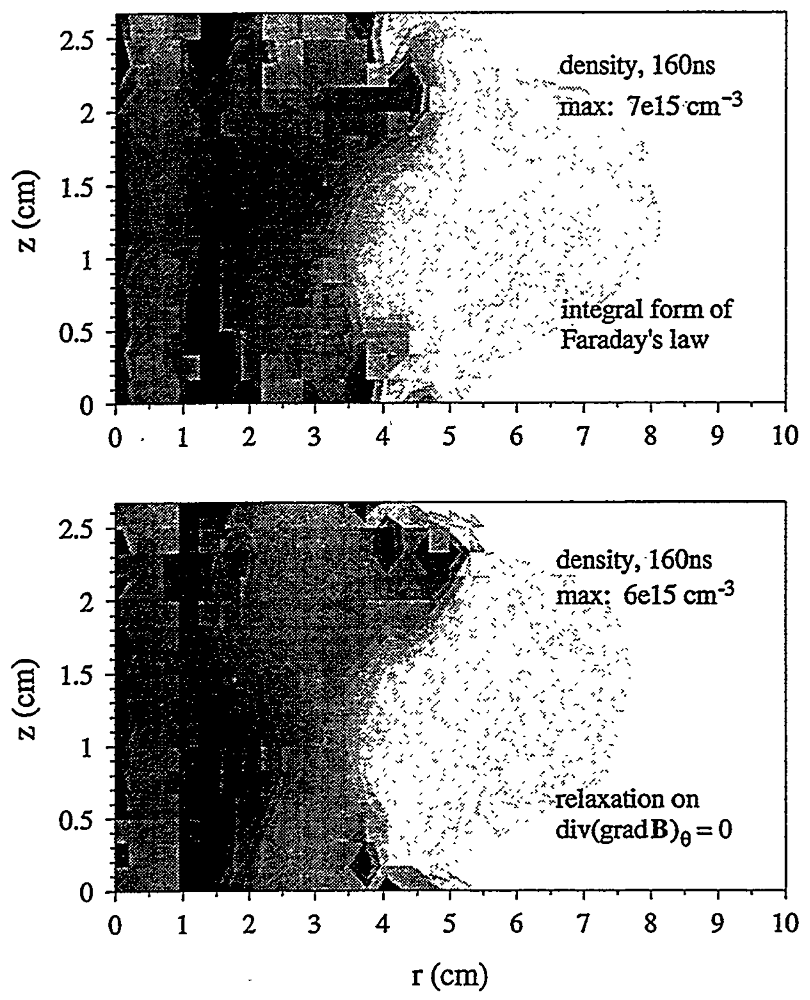

Figure 2.22. Plots of number density at time t=160ns for the Z-pinch. B_θ is updated by two different methods. Peak ion number densities (darkest) are roughly given.

At t=150ns, Method 1 yields a sudden increase in field energies. The field energy is in turbulent azimuthal magnetic and poloidal electric fields in the low density plasma of the plasma-vacuum interface. Current vortices associated with dilute plasma and entrained vacuum regions "spin up", creating the large magnetic fields, as in Figure 2.23.

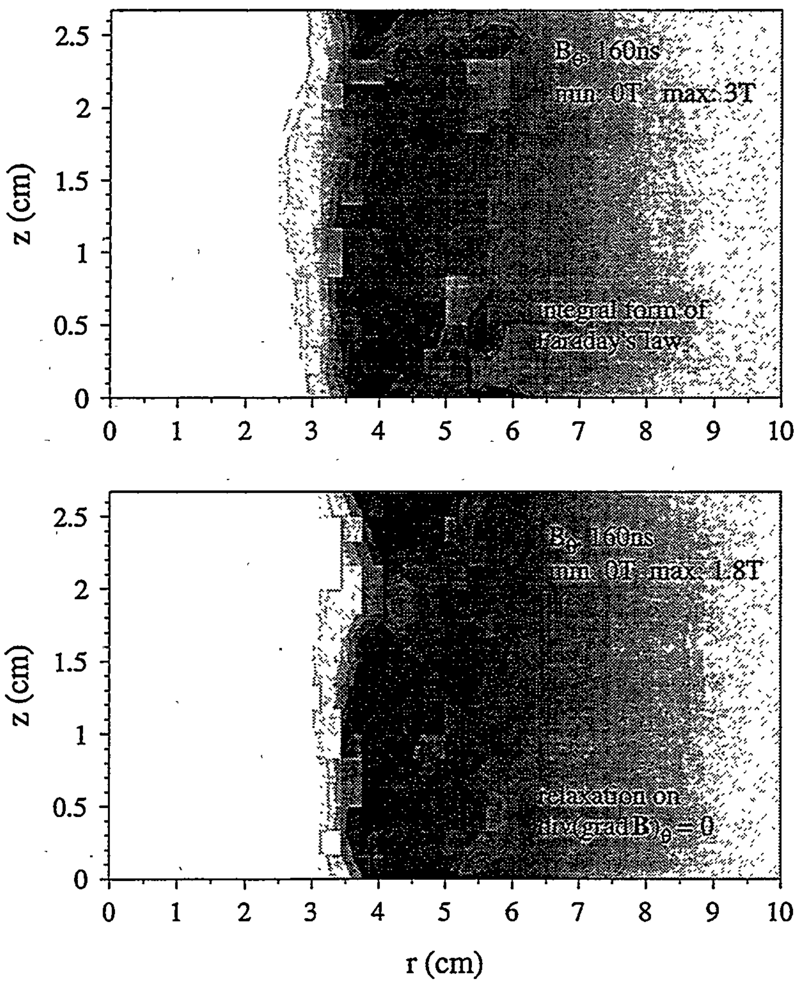

Figure 2.23. Plots of B_θ at time t=160ns for the Z-pinch. B_θ in entrained vacuum voids is updated by two different methods. Maximum fields (darkest) are roughly given.

The mixture of vacuum voids, dilute plasma, and rapidly swirling electron fluid then generates high poloidal electric fields, as exemplified by Figures 2.24 and 2.26. Density plots at t=160ns suggest that the vortices cause the plasma to neck off earlier for Method 1. Provided that B_θ is largely augmented in the voids, this makes sense.

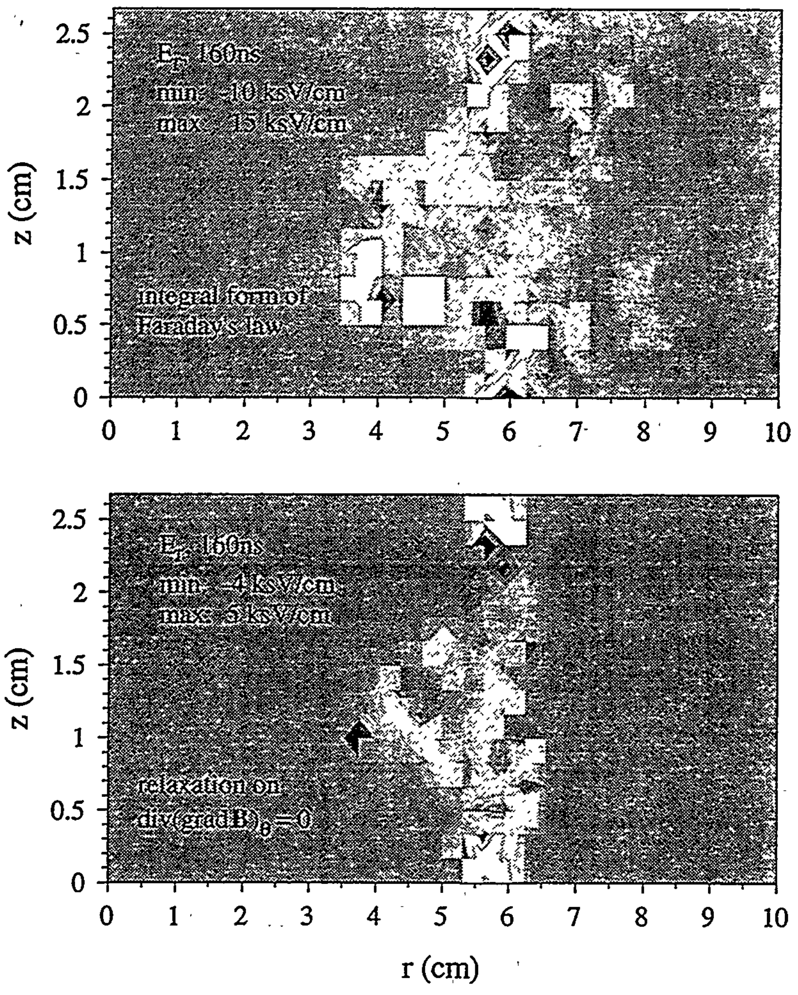

Figure 2.24. Plots of E_r at time t=160ns for the Z-pinch. B_θ in entrained vacuum voids is updated by two different methods. Maximum fields (darkest) are roughly given.

Larger magnetic field pressures inside the voids could drive the nearby bulk plasma toward the axis with greater force. At $t \cong 180$ns, the turbulence dissipates for Method 1; this lends credence to numerical stability and is physically reasonable. The plasma appears to begin stagnating; there would then be less acceleration to drive MRT instability, allowing dissipation due to resistivity and other thermalization mechanisms.

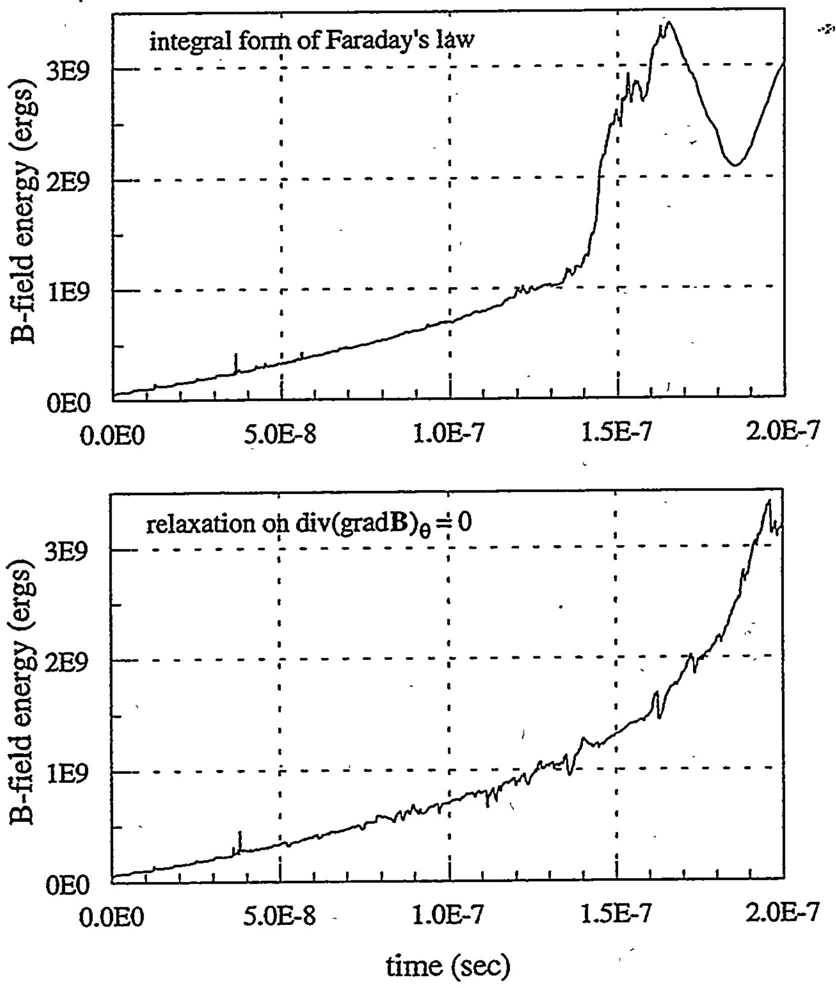

Figure 2.25. Magnetic field energy as a function of time for the two-dimensional Z-pinch.

Comparison of Methods 1 and 2 demonstrates that turbulence must be carefully treated for realistic calculation of pinch times and peak densities, and that the turbulence works to degrade compression of the screw pinch. Furthermore, the magnitude of the turbulent fields tends to decrease confidence in the general utility of phenomenological

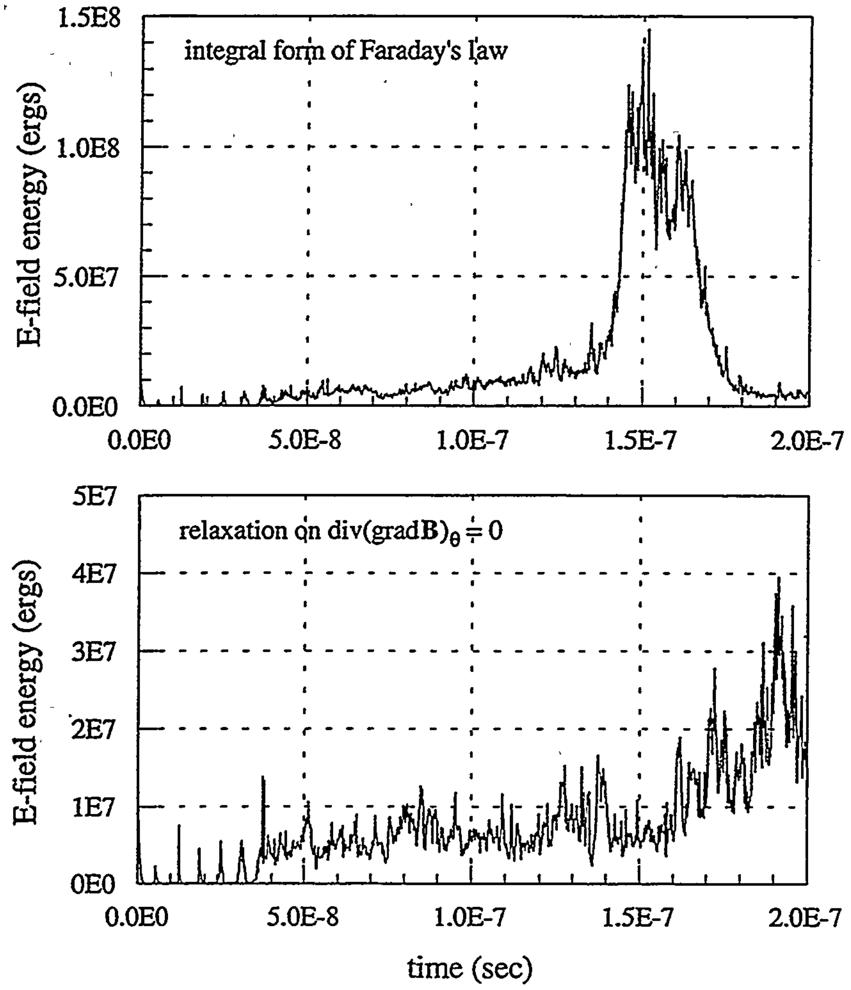

Figure 2.26. Electric field energy as a function of time for the two-dimensional Z-pinch.

treatments of turbulence in 1-d simulations [Thornhill *et al.* 1994]. The amount of turbulence suggested by Method 1 might also account for large discrepancies between Z-pinch experiments and recent 2-d simulations [Hammer *et al.* 1996].

In comparisons of Methods 1 and 2, a few more cases should be investigated. It would be most interesting to increase the spatial resolution and to decrease the grid spacing and the cutoff density. In such a parameter regime for the simulation, the methods should maintain agreement for longer simulation times. Then, raising the cutoff density would immediately illustrate any undesirable effects that Method 1 or 2 might have on simulation accuracy. Of course, these sort of algorithmic checks would fall into the class of heroic computer runs, since they would require large amounts of computer memory and execution time.

Chapter 2 references

1. Freidberg, J.P., Ideal Magnetohydrodynamics (Plenum Press, New York 1987)
2. Freidberg, J.P. and Hewett, D.W., Eigenmode Analysis of Resistive MHD Stability by Matrix Shooting, *J. Plasma Physics* **26**, 177-192 (1981)
3. Hain, K., and Lüst, R., Zur Stabilität zylindersymmetrischer Plasmakonfigurationen mit Volumenströmen, *Z. Naturforsch.* **13a**, 936-940 (1958)
4. Hammer, J.H., Eddleman, J.L., Springer, P.T., Tabak, M., *et al.*, Two-dimensional radiation-magnetohydrodynamic simulations of SATURN imploding Z pinches, *Phys. Plasmas* **3**, 2063-2069 (1996)
5. Hewett, D.W., A Global Method of Solving the Electron-Field Equations in a Zero-Inertia-Electron-Hybrid Plasma Simulation Code, *J. Comput. Phys.* **38**, 378-395 (1980)
6. Kruskal, M. and Schwarzschild, M., Some Instabilities of a Completely Ionized Plasma, *Proc. of the Royal Society A* **223**, 348-360 (1954)
7. Lindemuth, I.R., Conservation Form in Computational Magnetohydrodynamics, *J. Comput. Phys.* **18**, 119-131 (1975)
8. Sgro, A.G. and C.W. Nielson, Hybrid Model Studies of Ion Dynamics and Magnetic Field Diffusion During Pinch Implosions, *Phys. Fluids* **19**, 126-133 (1976)
9. Tayler, R.J., Hydromagnetic Instabilities of an Ideally Conducting Fluid, *Proc. Phys. Soc. B* **70**, 31 (1957)
10. Tayler, R.J., The Influence of an Axial Magnetic Field on the Stability of a Constricted Gas Discharge, *Proc. Phys. Soc. B* **70**, 1049 (1957)
11. Thornhill, J.W., and Whitney, K.G., Phenomenological modeling of turbulence in Z-pinch implosions, *Phys. Plasmas* **1**, 321-330 (1994)

Chapter 3
Details of the ADI Method in Two Spatial Dimensions

This chapter outlines the alternating-direction-implicit (ADI) methods chosen to advance the plasma simulation in time. ADI methods are among the best numerical workhorses for enforcing the governing equations of the plasma simulation. Section 3.1 begins with the motivation for using ADI instead of Crank-Nicholson methods or direct implicit methods. It then introduces the two-dimensional Peaceman-Rachford (PR) ADI method. Use of preconditioning and dynamic relaxation parameter adjustment for accelerating PR ADI are mentioned, followed by a proof of convergence of preconditioned PR ADI for a fixed time step. Section 3.2 is concluded with an outline of the dynamic ADI (DADI) method based on the Doss-Miller prescription for parameter adjustment. In Section 3.3, methods of spatial domain decomposition (SDD) and interprocessor communication are discussed for implementation of ADI on a parallel computer. Finally, in Section 3.4, a parallel DADI algorithm is given in the form of a flowchart, and a few results regarding parallel and algorithmic scaling performance are shown.

3.1 Diffusive equations in the simulation

The algorithm underlying the screw pinch simulation is based on the evolution of time-dependent *diffusive* equations obeyed by A_θ and B_θ:

$$r\dot{A}_\theta = \left[\left(\frac{c^2\eta_\theta}{4\pi}\frac{\partial^2}{\partial z^2} - u_{ez}\frac{\partial}{\partial z}\right) + \left(\frac{c^2\eta_\theta}{4\pi} r\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r}\right) - u_{er}\frac{\partial}{\partial r}\right)\right] rA_\theta, \qquad (3.1)$$

$$\dot{B}_\theta = \left[\left(\frac{c^2}{4\pi}\frac{\partial}{\partial z}\left(\eta_r\frac{\partial}{\partial z}\right)-\frac{\partial}{\partial z}u_{ez}\right)+\left(\frac{c^2}{4\pi}\frac{\partial}{\partial r}\left(\frac{\eta_z}{r}\frac{\partial}{\partial r}r\right)-\frac{\partial}{\partial r}u_{er}\right)\right]B_\theta$$
$$+\ \frac{\partial}{\partial z}\left(u_{e\theta}B_z\right)+\frac{\partial}{\partial r}\left(u_{e\theta}B_r\right)+\frac{c}{e\rho}\left(\frac{\partial\rho}{\partial r}\frac{\partial T_e}{\partial z}-\frac{\partial\rho}{\partial z}\frac{\partial T_e}{\partial r}\right). \tag{3.2}$$

The nomenclature *diffusive* is used because Eqs. 3.1 and 3.2 are not classic diffusion equations like Eq. 3.3 below, yet they exhibit diffusive behavior when the resistivity $\bar{\bar{\eta}}$ is sufficiently large.

$$\partial_t u = \bar{\nabla}\cdot\varepsilon\bar{\nabla}u - \mu u - \rho;\ \ \varepsilon,\mu>0,\ \bar{r}\in R,\ \text{BCs on } \partial R. \tag{3.3}$$

In the classic case of Eq. 3.3, $\varepsilon(\bar{r})$ is the diffusivity, $\mu(\bar{r})$ is a real positive function of space, and $\rho(\bar{r})$ is a source. Equations 3.1 and 3.2 are much more complicated than Eq. 3.3 because the quantities analogous to ε, μ, and ρ depend on the unknowns A_θ and B_θ, in addition to spatial coordinates. This makes Eqs. 3.1 and 3.2 coupled, nonlinear, advective, and diffusive. However, when small time steps are taken for numerical evolution of A_θ and B_θ, the coupling terms, nonlinear terms, and advective terms can be eliminated or linearized by time-lagging appropriate factors. Furthermore, if the resistivity is sufficiently large then the time evolution of A_θ and B_θ is indeed dominated by the diffusion. It is only when the resistivity is small that the rate equations are dominated by advective and nonlinear terms. This is the case in Eqs. 3.4 and 3.5 where $\bar{\bar{\eta}}=0$.

$$r\dot{A}_\theta = -\left(u_{ez}\frac{\partial}{\partial z}+u_{er}\frac{\partial}{\partial r}\right)rA_\theta, \tag{3.4}$$

$$\dot{B}_\theta = -\left(\frac{\partial}{\partial z}u_{ez}+\frac{\partial}{\partial r}u_{er}\right)B_\theta+\frac{\partial}{\partial z}\left(u_{e\theta}B_z\right)+\frac{\partial}{\partial r}\left(u_{e\theta}B_r\right)+$$
$$\frac{c}{e\rho}\left(\frac{\partial\rho}{\partial r}\frac{\partial T_e}{\partial z}-\frac{\partial\rho}{\partial z}\frac{\partial T_e}{\partial r}\right). \tag{3.5}$$

Equations 3.4 and 3.5 are nondiffusive transport equations. Fortunately, this does not adversely affect the ADI method when it is used to advance the equations. In addition to Eqs. 3.1 and 3.2, other equation needed to model the plasma pinches are Eqs. 3.6 and 3.7.

$$\left(\nabla^2 \bar{A}\right)_\theta = \left(\frac{1}{r}\frac{\partial^2}{\partial z^2} + \frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial}{\partial r}\right)\right)(rA_\theta) = 0, \qquad \nabla^2 \bar{E} = 0. \qquad (3.6, 3.7)$$

Equation 3.7 was used by Hewett [1980] in vacuum regions, but in pinch simulations of this dissertation, a new method of updating $\bar{E}$ in the vacuum is used. With $u \equiv u(\bar{r},t)$ playing the role of rA_θ. E_r, or E_θ, Equations 3.6 and 3.7 for these quantities can be written in the form of Eq. 3.8.

$$\nabla^2 u = \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) + \frac{\partial^2 u}{\partial z^2} = \rho, \qquad u \equiv u(\bar{r},t). \qquad (3.8)$$

Note that the solution to the elliptic equation, Eq. 3.8, is the same as the time asymptotic $(t \to \infty)$ solution to Eq. 3.3 for the special case $\mu(\bar{r}) \equiv 0$ and $\varepsilon(\bar{r}) \equiv 1$.

To understand why ADI methods are preferrable to other common methods for evolution of Eq. 3.3 and solution of Eq. 3.8, one can write down the finite-difference form corresponding to each method and then analyze the stability and accuracy of the method. For this discussion, all of the interesting finite-difference methods use second-order accurate central differencing formulas to approximate the second derivatives. In axisymmetric cylindrical rz-coordinates, derivatives in the Laplacian are approximated according to Eqs. 3.9 and 3.10.

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) \doteq \frac{r_{1/2}(u_{1,0} - u) - r_{-1/2}(u - u_{-1,0})}{r\Delta r^2}, \qquad (3.9)$$

$$\frac{\partial^2 u}{\partial z^2} \doteq \frac{u_{0,1} - 2u + u_{0,-1}}{\Delta z^2}. \qquad (3.10)$$

These equations are based on the notation $r_i = i\Delta r$, $z_j = j\Delta z$, and $u_{i,j} \equiv u(r_i, z_j)$, where the indices are suppressed so that $u_{i,j} \rightarrow u$, $u_{i\pm 1,j} \rightarrow u_{\pm 1,0}$, and $u_{i,j\pm 1} \rightarrow u_{0,\pm 1}$. The quantities Δr and Δz are the uniform grid spacings in the r- and z-directions. The indices i and j take on integer and half-integer values. When time levels are introduced in the equations, similar notation is used with a superscripted time index, e.g. $u^n_{i,j} \equiv u(r_i, z_j, t_n)$.

Reasonable methods for comparison to ADI in the evolution of Eq. 3.3 include the classical explicit method and the Crank-Nicholson method. On the other hand, a reasonable method to compare with ADI for solution of Eq. 3.8 is a plain direct solution method. All four methods, including ADI, are second-order accurate in grid spacing, since Eqs. 3.9 and 3.10 are used for the Laplacian operator. While the classical explicit method is easiest to implement for Eq. 3.3, it is only first-order accurate in time and is conditionally stable--instability occurs for larger time steps. On the other hand, the Crank-Nicholson method is second-order accurate in time and is unconditionally stable, a considerable improvement over the explicit method in this sense. However, the ADI method for Eq. 3.3 is second-order accurate in time and unconditionally stable, but requires much less work than the Crank-Nicholson method!

The finite-difference forms in cylindrical coordinates, for the evolution of Eq. 3.3 and the solution of Eq. 3.8 are given here:

Classical explicit advancement of $\partial_t u = \nabla^2 u - \rho$:

$$\frac{u^1 - u}{\Delta t} = \frac{r_{1/2}(u_{1,0} - u) - r_{-1/2}(u - u_{-1,0})}{r\Delta r^2} + \frac{u_{0,1} - 2u + u_{-0,1}}{\Delta z^2} - \rho .$$

Crank-Nicholson advancement of $\partial_t u = \nabla^2 u - \rho$:

$$\frac{u^1 - u}{\Delta t} = \frac{1}{2}\left(\frac{r_{1/2}(u_{1,0} - u) - r_{-1/2}(u - u_{-1,0})}{r\Delta r^2} + \frac{u_{0,1} - 2u + u_{0,-1}}{\Delta z^2}\right) +$$

$$\frac{1}{2}\left(\frac{r_{1/2}(u^1_{1,0} - u^1) - r_{-1/2}(u^1 - u^1_{-1,0})}{r\Delta r^2} + \frac{u^1_{0,1} - 2u^1 + u^1_{0,-1}}{\Delta z^2}\right) - \rho^{1/2}$$

ADI advancement of $\partial_t u = \nabla^2 u - \rho$:

$$\frac{u^{1/2} - u}{\Delta t / 2} = \left(\frac{r_{1/2}(u^{1/2}_{1,0} - u^{1/2}) - r_{-1/2}(u^{1/2} - u^{1/2}_{-1,0})}{r\Delta r^2} + \frac{u_{0,1} - 2u + u_{0,-1}}{\Delta z^2}\right) - \rho^{1/2},$$

$$\frac{u^1 - u^{1/2}}{\Delta t / 2} = \left(\frac{r_{1/2}(u^{1/2}_{1,0} - u^{1/2}) - r_{-1/2}(u^{1/2} - u^{1/2}_{-1,0})}{r\Delta r^2} + \frac{u^1_{0,1} - 2u^1 + u^1_{0,-1}}{\Delta z^2}\right) - \rho^{1/2}.$$

Direct method for solution of $\nabla^2 u = \rho$:

$$\frac{r_{1/2}(u_{1,0} - u) - r_{-1/2}(u - u_{-1,0})}{r\Delta r^2} + \frac{u_{0,1} - 2u + u_{0,-1}}{\Delta z^2} = \rho.$$

The explicit advance is not useful for the pinch simulation because it has limited accuracy and stability, it requires too much computer time to simulate a reasonably long physical time interval. For this reason, the Crank-Nicholson and direct methods become the focus of further comparisons with ADI here.

When the Crank-Nicholson method for Eq. 3.3 and the direct method for Eq. 3.8 are written as a system of equations for unknowns on a 6×6 spatial grid, the matrix representation has the form in Figure 3.1 (nonzero coefficients are denoted by an x, but are otherwise undistinguished). The system of Figure 3.1 would arise from a 6×6

spatial grid, $i,j \in [0,5]$, around which Dirichlet conditions are applied at points $i = 0$ *or* 5 and $j = 0$ *or* 5.. The coefficient matrix in Figure 3.1 has a standard 5-stripe structure.

$$\left(\begin{array}{cccc|cccc|cccc|cccc}
x&x&&&x&&&&&&&&&&& \\
x&x&x&&&x&&&&&&&&&& \\
&x&x&x&&&x&&&&&&&&& \\
&&x&x&&&&x&&&&&&&& \\
\hline
x&&&&x&x&&&x&&&&&&& \\
&x&&&x&x&x&&&x&&&&&& \\
&&x&&&x&x&x&&&x&&&&& \\
&&&x&&&x&x&&&&x&&&& \\
\hline
&&&&x&&&&x&x&&&x&&& \\
&&&&&x&&&x&x&x&&&x&& \\
&&&&&&x&&&x&x&x&&&x& \\
&&&&&&&x&&&x&x&&&&x \\
\hline
&&&&&&&&x&&&&x&x&& \\
&&&&&&&&&x&&&x&x&x& \\
&&&&&&&&&&x&&&x&x&x \\
&&&&&&&&&&&x&&&x&x
\end{array}\right)
\begin{pmatrix} x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x \end{pmatrix}
=
\begin{pmatrix} x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x \end{pmatrix}.$$

Figure 3.1. A tridiagonal-block tridiagonal system of equations for a direct solution of $(\bar{\nabla} \cdot \varepsilon \bar{\nabla} - \mu)u = \rho$ or a Crank-Nicholson advance of $\partial_t u = (\bar{\nabla} \cdot \varepsilon \bar{\nabla} - \mu)u - \rho$. A 6×6 grid with Dirichlet boundary conditions on the perimeter is assumed. Dashed lines merely delineate blocks of coefficients.

It is a matrix that is blockwise tridiagonal, each block being either diagonal or tridiagonal; this makes *tridiagonal-block tridiagonal matrix* a reasonable term for it.

If the grid has size $(M+2)\times(N+2)$ and Dirichlet boundary conditions are applied, the matrix like that in Figure 3.1 has either M rows of $N \times N$ blocks, or N rows of $M \times M$ blocks, with structure depending on the ordering of the unknowns. It can be shown that the number of arithmetic operations (AOs) necessary for direct solution of such a linear system scales to leading order like MN^3 or M^3N. See Appendix 3.A1.

While the MN^3 scaling of AOs is much better than the $\frac{2}{3}M^3N^3$ scaling achievable with Gaussian elimination on a dense $MN \times MN$ matrix, it simply does not compare to the more favorable MN scaling exhibited by ADI for advancement of Eq. 3.3 and solution of Eq. 3.8. This is the reason ADI is used instead of the Crank-Nicholson method and the direct method. Actually, when ADI is used for solution of Eq. 3.8, it essentially finds the $t \to \infty$ solution of Eq. 3.3, and this should be kept in mind during the development of the ADI method below. The reason ADI solves Equations 3.3 and 3.8 faster is that it relies on straightforward solution of tridiagonal systems of equations.

3.2 Peaceman-Rachford ADI method

3.2.1. Implementation

In two dimensions, a very useful ADI method is that of Peacman and Rachford (1955). The method has been studied extensively, and is detailed here in cartesian coordinates. Suppose u in Eq. 3.3 is given at time t^n over the whole spatial domain. Peaceman-Rachford ADI advances the iterate $u^n_{i,j} \equiv u(i\Delta x_1, j\Delta x_2, t^n)$ a time step Δt^n to time level $t^{n+1} = t^n + \Delta t$ according to Equations 3.11 and 3.12.

$$\begin{aligned}\frac{u'_{i,j} - u^n_{i,j}}{\Delta t^n / 2} = & \frac{(\varepsilon_{i+1,j} + \varepsilon_{i,j})u'_{i+1,j} - (\varepsilon_{i+1,j} + 2\varepsilon_{i,j} + \varepsilon_{i-1,j})u'_{i,j} + (\varepsilon_{i,j} + \varepsilon_{i-1,j})u'_{i-1,j}}{2\Delta x_1^2} \\ & + \frac{(\varepsilon_{i,j+1} + \varepsilon_{i,j})u^n_{i,j+1} - (\varepsilon_{i,j+1} + 2\varepsilon_{i,j} + \varepsilon_{i,j-1})u^n_{i,j} + (\varepsilon_{i,j} + \varepsilon_{i,j-1})u^n_{i,j-1}}{2\Delta x_2^2} \\ & - \frac{\mu_{i,j}}{2}\left(u'_{i,j} + u^n_{i,j}\right) - \rho_{i,j} \ , \end{aligned} \qquad (3.11)$$

$$\frac{u_{i,j}^{n+1} - u'_{i,j}}{\Delta t^n / 2} = \frac{(\varepsilon_{i+1,j} + \varepsilon_{i,j})u'_{i+1,j} - (\varepsilon_{i+1,j} + 2\varepsilon_{i,j} + \varepsilon_{i-1,j})u'_{i,j} + (\varepsilon_{i,j} + \varepsilon_{i-1,j})u'_{i-1,j}}{2\Delta x_1^2}$$
$$+ \frac{(\varepsilon_{i,j+1} + \varepsilon_{i,j})u_{i,j+1}^{n+1} - (\varepsilon_{i,j+1} + 2\varepsilon_{i,j} + \varepsilon_{i,j-1})u_{i,j}^{n+1} + (\varepsilon_{i,j} + \varepsilon_{i,j-1})u_{i,j-1}^{n+1}}{2\Delta x_2^2} \tag{3.12}$$
$$- \frac{\mu_{i,j}}{2}\left(u_{i,j}^{n+1} + u'_{i,j}\right) - \rho_{i,j} .$$

Here the grid indices have been retained. The intermediate iterate u' is implicity determined by Eq. 3.11 via a tridiagonal system solution. Once u' is determined, it can be substituted into Eq 3.12 through which u^{n+1} is determined by another tridiagonal system solution. The quantities $\varepsilon_{i,j}$, $\mu_{i,j}$, and $\rho_{i,j}$ are known functions of the grid. The step Δt^n reflects a physical time step if it is sufficiently small that the iterates follow the evolution of the Eq. 3.3. In order to condense the notation of the method, it is convenient to suppress the grid indices and define the following quantities in Eqs. 3.13 and 3.14.

$$\omega^n \equiv 2 / \Delta t^n, \;\; \varepsilon_{\pm 1/2,0} \equiv \tfrac{1}{2}(\varepsilon_{\pm 1,0} + \varepsilon), \;\; \varepsilon_{0,1/2} \equiv \tfrac{1}{2}(\varepsilon_{0,\pm 1} + \varepsilon), \tag{3.13a,b,c}$$

$$\delta_1^2 u^n = \frac{\varepsilon_{1/2,0}u_{1,0}^n - (\varepsilon_{1/2,0} + \varepsilon_{-1/2,0})u^n + \varepsilon_{-1/2,0}u_{-1,0}^n}{\Delta x_1^2}, \tag{3.14a}$$

$$\delta_2^2 u^n = \frac{\varepsilon_{0,1/2}u_{0,1}^n - (\varepsilon_{0,1/2} + \varepsilon_{0,-1/2})u^n + \varepsilon_{0,-1/2}u_{0,-1}^n}{\Delta x_2^2}. \tag{3.14b}$$

With this notation the Peaceman-Rachford prescription can be expressed in the compact form of Eqs 3.15.

$$\left(-\omega^n + \delta_1^2 - \mu / 2\right)u' = -\left(\omega^n + \delta_2^2 - \mu / 2\right)u^n + \rho, \tag{3.15a}$$
$$\left(-\omega^n + \delta_2^2 - \mu / 2\right)u^{n+1} = -\left(\omega^n + \delta_1^2 - \mu / 2\right)u' + \rho. \tag{3.15b}$$

Taking $H = \delta_1^2 - \mu/2$ and $V = \delta_2^2 - \mu/2$, the equations become

$$\left(-\omega^n + H\right)u' = -\left(\omega^n + V\right)u^n + \rho, \tag{3.16a}$$

$$\left(-\omega^n + V\right)u^{n+1} = -\left(\omega^n + H\right)u^{*} + \rho. \tag{3.16b}$$

On the boundary of the problem in Eq. 3.3, ∂R, either Robbins or periodic boundary conditions are enforced. Robbins boundary conditions take the form $au + b\hat{n}\cdot\vec{\nabla}u = c$. The Robbins boundary condition includes Dirichlet $(b=0)$ and Neumann $(a=0)$ boundary conditions as special cases. Quantities a, b, and c can vary from one point to the other on the boundaries. When the equations and boundary conditions are used to solve Eq. 3.3, boundary conditions on the intermediate unknown u^{*} affect the accuracy significantly. As pointed out by the literature, one should use Eq. 3.17 for the case of Dirichlet points $u(\bar{r},t) = g(\bar{r},t)$ on ∂R, [Fairweather and Mitchell 1967].

$$g^{*}_{i,j} = \tfrac{1}{2}(\omega^n + V)g^n_{i,j} + \tfrac{1}{2}(\omega^n - V)g^{n+1}_{i,j}. \tag{3.17}$$

Equation 3.17 is not the "obvious" choice $g^{*}_{i,j} = g^{n+1/2}_{i,j}$ for the boundary points.

Directly discretizing Eq. 3.3 in Cartesian coordinates yields $\partial_t u = (H+V)u - \rho$ using the second order central differencing of Eqs. 3.11 and 3.12. To solve $(H+V)u = \rho$, one could use a direct method or one could use ADI, stepping toward infinite times. If a direct method were used, the coefficient matrix representing $H+V$ in the linear system of equations would have the form in Figure 3.2, provided the unknowns were in either the H-ordering or the V-ordering (for a 4×4 grid with the possibility of doubly periodic boundary conditions). When the unknowns are in the then H-ordering, it means that the grid values are numbered so as to increment the leftmost grid index i first, and the rightmost grid index j second. For the 4×4 case, the column vector u in the H-ordering, u_H, is given by $u_H = \left(u_{1,1}, u_{2,1}, u_{3,1}, u_{4,1}, u_{1,2}, u_{2,2}, \ldots, u_{1,4}, u_{2,4}, u_{3,4}, u_{4,4}\right)^t$, where the superscript t indicates transposition from a row vector to a column vector, or vice versa. In the V-ordering, u is $u_V = \left(u_{1,1}, u_{1,2}, u_{1,3}, u_{1,4}, u_{2,1}, u_{2,2}, \ldots, u_{4,1}, u_{4,2}, u_{4,3}, u_{4,4}\right)^t$.

$$pop_H(H+V) = \left(\begin{array}{cccc|cccc|cccc|cccc}
x & x & & x & x & & & & & & & & x & & & \\
x & x & x & & & x & & & & & & & & x & & \\
& x & x & x & & & x & & & & & & & & x & \\
x & & x & x & & & & x & & & & & & & & x \\
\hline
x & & & & x & x & & x & x & & & & & & & \\
& x & & & x & x & x & & & x & & & & & & \\
& & x & & & x & x & x & & & x & & & & & \\
& & & x & x & & x & x & & & & x & & & & \\
\hline
& & & & x & & & & x & x & & x & x & & & \\
& & & & & x & & & x & x & x & & & x & & \\
& & & & & & x & & & x & x & x & & & x & \\
& & & & & & & x & x & & x & x & & & & x \\
\hline
x & & & & & & & & x & & & & x & x & & x \\
& x & & & & & & & & x & & & x & x & x & \\
& & x & & & & & & & & x & & & x & x & x \\
& & & x & & & & & & & & x & x & & x & x
\end{array}\right).$$

Figure 3.2. Matrix showing nonzero coefficients of the matrix operator $H+V=\delta_1^2+\delta_2^2-\mu$ in the H-ordering, allowing periodic boundary conditions in the directions of H and V. Elements that can be nonzero are marked with an x.

While the direct method with coefficient matrix in Figure 3.2 can be used to solve $(H+V)u=\rho$, it is computationally cheaper to iteratively apply Eqs. 3.11 and 3.12 of PR ADI. The coefficient matrices H and $-\omega^n+H$ have populations in the H-ordering that are shown in Figure 3.3, while the coefficient matrices V and $-\omega^n+V$ have populations in the H-ordering that are shown in Figure 3.4. Naturally, $pop_V(H)=pop_H(V)$ and $pop_V(V)=pop_H(H)$. Matrices H and V depicted in Figures 3.3 and 3.4. are periodic generalizations of tridiagonal matrices. The work to solve a system of equations with a coefficient matrix of Figure 3.3 or 3.4, $O(MN)$, is much less than the work to solve a system of equations with a coefficient matrix of Figure 3.2, $O(M^3N)$, where there are N blocks of size $M \times M$ in the coefficient matrices.

$$pop_H(H) = \left(\begin{array}{cccc|cccc|cccc|cccc}
x & x & & x & & & & & & & & & & & & \\
x & x & x & & & & & & & & & & & & & \\
 & x & x & x & & & & & & & & & & & & \\
x & & x & x & & & & & & & & & & & & \\
\hline
 & & & & x & x & & x & & & & & & & & \\
 & & & & x & x & x & & & & & & & & & \\
 & & & & & x & x & x & & & & & & & & \\
 & & & & x & & x & x & & & & & & & & \\
\hline
 & & & & & & & & x & x & & x & & & & \\
 & & & & & & & & x & x & x & & & & & \\
 & & & & & & & & & x & x & x & & & & \\
 & & & & & & & & x & & x & x & & & & \\
\hline
 & & & & & & & & & & & & x & x & & x \\
 & & & & & & & & & & & & x & x & x & \\
 & & & & & & & & & & & & & x & x & x \\
 & & & & & & & & & & & & x & & x & x
\end{array}\right),$$

Figure 3.3. Matrix showing nonzero coefficients of the matrix operator $H = \delta_1^2 - \mu/2$ in the H-ordering, allowing periodic boundary conditions in the direction of H.

$$pop_H(V) = \left(\begin{array}{cccc|cccc|cccc|cccc}
x & & & & x & & & & & & & & x & & & \\
 & x & & & & x & & & & & & & & x & & \\
 & & x & & & & x & & & & & & & & x & \\
 & & & x & & & & x & & & & & & & & x \\
\hline
x & & & & x & & & & x & & & & & & & \\
 & x & & & & x & & & & x & & & & & & \\
 & & x & & & & x & & & & x & & & & & \\
 & & & x & & & & x & & & & x & & & & \\
\hline
 & & & & x & & & & x & & & & x & & & \\
 & & & & & x & & & & x & & & & x & & \\
 & & & & & & x & & & & x & & & & x & \\
 & & & & & & & x & & & & x & & & & x \\
\hline
x & & & & & & & & x & & & & x & & & \\
 & x & & & & & & & & x & & & & x & & \\
 & & x & & & & & & & & x & & & & x & \\
 & & & x & & & & & & & & x & & & & x
\end{array}\right).$$

Figure 3.4. Matrix showing nonzero coefficients of the matrix operator $V = \delta_2^2 - \mu/2$ in the H-ordering, allowing periodic boundary conditions in the direction of V.

As long as the time step in the ADI method is greater that zero, the tridiagonal coefficient matrices $-\omega^n + H$ and $-\omega^n + V$ are diagonally dominant. Diagonal dominance of an $N \times N$ matrix $A = \left(a_{i,j}\right)$ means

$$\sum_{j=1,\, j\neq i}^{N} \left|a_{i,j}\right| < \left|a_{i,i}\right| \quad \forall \quad i \in [1,N]. \tag{3.18}$$

Diagonal dominance implies that the system with coefficient matrix A has a unique solution that can be determined by standard Gaussian elimination without pivoting. This allows quick solution of Eqs. 3.16, because Gaussian elimination in a tridiagonal system proceeds in a very simple manner.

3.2.2 Convergence properties

In this section, basic convergence properties of the 2-D Peaceman-Rachford (PR) ADI for solution of $(\bar{\nabla} \cdot \varepsilon \bar{\nabla} - \mu)u = \rho$ are investigated. Remember that such a solution is the same as the time asymptotic solution to Eq. 3.3. The condensed finite-difference Eqs. 3.16 are reproduced here:

$$\left(-\omega^n + H\right)u' = -\left(\omega^n + V\right)u^n + \rho \tag{3.19a}$$

$$\left(-\omega^n + V\right)u^{n+1} = -\left(\omega^n + H\right)u' + \rho \tag{3.19b}$$

When the intermediate unknown u' is eliminated from Eqs. 3.19, one obtains an expression for u^{n+1} in terms of u^n and the other given quantities:

$$u^{n+1} = \left(-\omega^n + V\right)^{-1}\left\{\left(-\omega^n - H\right)\left(-\omega^n + H\right)^{-1}\left[\left(-\omega^n - V\right)u^n + \rho\right] + \rho\right\} \tag{3.20}$$

It is convenient to define the error vector $e^n \equiv u^n - u$, where u is the exact discrete solution to $(H+V)u=\rho$, Eq. 3.3. Subtracting rearranged versions of the equation $(-\omega^n+H)u=-(\omega^n+V)u+\rho$ from Eqs. 3.19 yields the following:

$$\begin{array}{rcl} (-\omega+H)u' & = & (-\omega-V)u^n + \rho \\ -\ [(-\omega+H)u & = & (-\omega-V)u\ +\rho] \\ \hline (-\omega+H)\,e' & = & (-\omega-V)e^n \end{array}\text{, and } \begin{array}{rcl} (-\omega+V)u^{n+1} & = & (-\omega-H)u'\ +\rho \\ -\ [(-\omega+V)u & = & (-\omega-H)u\ +\rho] \\ \hline (-\omega+V)e^{n+1} & = & (-\omega-H)e' \end{array}.$$

Therefore, $e^{n+1} = (-\omega+V)^{-1}(-\omega-H)(-\omega+H)^{-1}(-\omega-V)e^n$, just like Eq. 3.20 without the source ρ. There is no mechanism by which ρ can affect the rate of convergence to the solution of $(H+V)u=\rho$; the sole effect of the source term ρ is to change the initial value of the error vector e^0.

In the past it was shown that the PR ADI method is convergent for $(H+V)u=\rho$ with any fixed time step. Furthermore, it was shown to remain convergent when preconditioned by a symmetric matrix [Wachspress and Habetler 1960]. The preconditioned ADI method is implemented by application of ADI to the new system $(\hat{H}+\hat{V})u=\hat{\rho}$, where $\hat{H}=B^{-1}H$, $\hat{V}=B^{-1}V$, and $\hat{\rho}=B^{-1}\rho$. The new system has the same solution as $(H+V)u=\rho$ for well-behaved preconditioners B^{-1}. The idea is to choose B^{-1} so that $\hat{H}$ and $\hat{V}$ have smaller spectral radii than H and V, respectively; then each relaxation step diminishes more components of the error. In this dissertation, one of the version of the PR ADI that is used is *diagonally scaled*, for which $B^{-1}=diag^{-1}(H+V)$.

A proof of convergence in the symmetrically preconditioned case for a fixed time step is reproduced below [Lambert *et al.* 1996]. A longer, but more general proof of convergence for a fixed time step was published earlier [Wachspress and Habetler 1960]. Convergence has been proven for an arbitrary sequence of finite positive time steps only when H and V are symmetric, negative definite, and commutative [Varga 1962]; along these lines, optimum sequences of time steps have been found for the model problem of the Laplace equation on the unit square. It is not known by the author whether the symmetrically preconditioned method must converge for an arbitrary sequence of finite positive time steps. Empirical evidence in this dissertation suggests that convergence is accelerated by varying the time step in many preconditioned and noncommutative cases. Indeed, the time step variation makes the method one of the fastest of all methods.

Theorem 3.1

The preconditioned Peaceman-Rachford ADI scheme of the form

$$(B-\tfrac{\Delta t^n}{2}H)u' = (B+\tfrac{\Delta t^n}{2}V)u^n - \tfrac{\Delta t^n}{2}\rho, \tag{3.21a}$$

$$(B-\tfrac{\Delta t^n}{2}V)u^{n+1} = (B+\tfrac{\Delta t^n}{2}H)u' - \tfrac{\Delta t^n}{2}\rho, \tag{3.21b}$$

with H, V, and $-B$ symmetric and negative definite, converges to the solution of $(H+V)u = \rho$ for any fixed finite positive time step $\Delta t^n = \Delta t$.

Proof

The method can be expressed as $u^{n+1} = C(\Delta t)u^n + [I - C(\Delta t)]A^{-1}\rho$. To see this, combine Eqs. 3.25 to get u^{n+1} in terms of u^n and ρ:

$$\begin{aligned} u^{n+1} &= (B-\tfrac{\Delta t}{2}V)^{-1}\Big[(B+\tfrac{\Delta t}{2}H)(B-\tfrac{\Delta t}{2}H)^{-1}\big((B+\tfrac{\Delta t}{2}V)u^n - \tfrac{\Delta t}{2}\rho\big) - \tfrac{\Delta t}{2}\rho\Big] \\ &= C(\Delta t)u^n - \tfrac{\Delta t}{2}(B-\tfrac{\Delta t}{2}V)^{-1}\big((B+\tfrac{\Delta t}{2}H)(B-\tfrac{\Delta t}{2}H)^{-1}+I\big)\rho . \end{aligned} \tag{3.23}$$

The definition $C(\Delta t) \equiv (B-\tfrac{\Delta t}{2}V)^{-1}(B+\tfrac{\Delta t}{2}H)(B-\tfrac{\Delta t}{2}H)^{-1}(B+\tfrac{\Delta t}{2}V)$ is used. Then

$$\begin{aligned} I - C(\Delta t) &= (I-\hat{V})^{-1}\left[(I-\hat{V}) - (I+\hat{H})(I-\hat{H})^{-1}(I+\hat{V})\right] \\ &= (I-\hat{V})^{-1}(I-\hat{H})^{-1}\left[(I-\hat{H})(I-\hat{V}) - (I+\hat{H})(I+\hat{V})\right] \\ &= -(I-\hat{V})^{-1}(I-\hat{H})^{-1}(I+\hat{H}+I-\hat{H})\hat{A} = -(I-\hat{V})^{-1}\left((I+\hat{H})(I-\hat{H})^{-1}+I\right)\hat{A} \\ &= -\tfrac{\Delta t}{2}(B-\tfrac{\Delta t}{2}V)^{-1}\left((B+\tfrac{\Delta t}{2}H)(B-\tfrac{\Delta t}{2}H)^{-1}+I\right)A \end{aligned}$$

where $\hat{H} = \frac{\Delta t}{2}B^{-1}H$, and similarly for $\hat{V}$ and $\hat{A}$. The fact that $I+\hat{H}$ commutes with $(I-\hat{H})^{-1}$ has been used. Now, $C(\Delta t)$ is similar to $Q(\Delta t)$:

$$C(\Delta t) = (B+\tfrac{\Delta t}{2}V)^{-1}Q(\Delta t)(B+\tfrac{\Delta t}{2}V), \text{ where} \tag{3.24}$$

$$Q(\Delta t) = (B+\tfrac{\Delta t}{2}V)(B-\tfrac{\Delta t}{2}V)^{-1}(B+\tfrac{\Delta t}{2}H)(B-\tfrac{\Delta t}{2}tH)^{-1}. \tag{3.25}$$

For fixed time step, the error vector $e^n \equiv u^n - u$ advances according to $e^{n+1} = C^n(\Delta t)e^0$. Then as long as the largest eigenvalue of $C(\Delta t)$ has magnitude less than 1, the method with fixed time-step will converge.

Since B is symmetric and positive definite, it has a Cholesky decomposition of the form $B = LL^t$. Then $Q(\Delta t) = L(I+\tilde{V})(I-\tilde{V})^{-1}(I+\tilde{H})(I-\tilde{H})^{-1}L^{-1}$, where $\tilde{V} \equiv \frac{\Delta t}{2}L^{-1}VL^{-t}$ and $\tilde{H} \equiv \frac{\Delta t}{2}L^{-1}HL^{-t}$. Note that $\tilde{H}$ and $\tilde{V}$ are symmetric and negative definite by congruence with H and V. We take the vector induced B^{-1}-norm of Q defined by

$$\|Q\|_{B^{-1}} = \sup_{x\neq 0}\frac{\|Qx\|_{B^{-1}}}{\|x\|_{B^{-1}}}, \text{ where } \|x\|_{B^{-1}}^2 = x^t B^{-1} x.$$

The acronym *sup* is short for *supremum*, or the least upper bound. Note right away that

$$\|x\|_{B^{-1}}^2 = x^t L^{-t} L^{-1} x = \left\|L^{-1}x\right\|_2^2, \qquad \|Qx\|_{B^{-1}}^2 = x^t Q^t L^{-t} L^{-1} Q x = \left\|L^{-1}Qx\right\|_2^2.$$

Then $\|Qx\|_{B^{-1}}^2 = \left\|(I+\tilde{V})(I-\tilde{V})^{-1}(I+\tilde{H})(I-\tilde{H})^{-1}L^{-1}x\right\|_2^2$, and

$$\|Q\|_{B^{-1}}^2 = \sup_{x\neq 0}\frac{x^tQ^tL^{-t}L^{-1}Qx}{x^tL^{-t}L^{-1}x} \geq \frac{u^tq^tL^{-t}L^{-1}qu}{u^tL^{-t}L^{-1}u} = \frac{|q|^2u^tL^{-t}L^{-1}u}{u^tL^{-t}L^{-1}u} = \rho(Q)^2.$$

where q is the largest eigenvalue of Q with corresponding eigenvector u, and $\rho(Q)$ is the magnitude of q, or the spectral radius of Q. Obviously, with the definition $y = L^{-1}x$,

$$\begin{aligned}\|Q\|_{B^{-1}}^2 &= \sup_{y\neq 0}\frac{\left\|(I+\tilde{V})(I-\tilde{V})^{-1}(I+\tilde{H})(I-\tilde{H})^{-1}y\right\|_2^2}{\|y\|_2^2}\\ &= \left\|(I+\tilde{V})(I-\tilde{V})^{-1}(I+\tilde{H})(I-\tilde{H})^{-1}\right\|_2^2\end{aligned}$$

From the submultiplicative propterty of p-norms,

$$\|Q\|_{B^{-1}}^2 \leq \left\|(I+\tilde{V})(I-\tilde{V})^{-1}\right\|_2^2\left\|(I+\tilde{H})(I-\tilde{H})^{-1}\right\|_2^2. \qquad (3.26)$$

It is readily verified that the matrices $(I+\tilde{H})(I-\tilde{H})^{-1}$ and $(I+\tilde{V})(I-\tilde{V})^{-1}$ are symmetric, given that H and V are symmetric, and it is well-known that the L2-norm of a real symmetric matrix is just the largest eigenvalue of the matrix. Therefore,

$$\|Q\|_{B^{-1}}^2 \leq \rho\left((I+\tilde{V})(I-\tilde{V})^{-1}\right)^2\rho\left((I+\tilde{H})(I-\tilde{H})^{-1}\right)^2. \qquad (3.27)$$

Let $\tilde{\eta}$ and $\tilde{\upsilon}$ be the eigenvalues of least magnitude of $\tilde{H}$ and $\tilde{V}$, respectively. Clearly, the spectral radii introduced above are less than unity since $\tilde{\eta}$ and $\tilde{\upsilon}$ are negative:

$$\rho\left((I+\tilde{V})(I-\tilde{V})^{-1}\right) = \frac{1+\tilde{\upsilon}}{1-\tilde{\upsilon}} < 1.$$

Therefore, $\rho(C(\Delta t)) = \rho(Q(\Delta t)) \leq \|Q(\Delta t)\|_{B^{-1}} < 1$. This is sufficient to prove that $C^n(\Delta t) \rightarrow 0$ for any positive finite Δt, so that the theorem is proved.

3.2.3 Dynamic alternating-direction-implicit (DADI) method

Relaxation parameter adjustment for accelerated convergence to the $t \rightarrow \infty$ solution of Eq. 3.3 had been studied by many authors [Douglas 1963; Peaceman and Rachford 1955; Varga 1962; Wachspress 1962]. Most of the early parameter adjustment methods were based on arduous pre-runtime analyses of the prototypical model problem. The model problem has a simple diffusivity and simple boundary conditions ($\varepsilon = 1$, Dirichlet boundaries on a unit square). Because the early methods were derived for such a simple case, they did not perform well for inhomogeneous ε and complicated boundary conditions.

In 1979, an adaptive method was published that works well for general ε and general boundary conditions [Doss and Miller 1979]. This *Doss-Miller prescription* is adaptive because the parameter sequence is determined at run time, and is not determined by a tedious *a priori* analysis of convergence. The general prescription is as follows:

1. With relaxation parameter $\omega^n \equiv 2/\Delta t^n$ perform the ADI double-pass of Eq. 3.19 to advance form u^n to u^{n+1}. With the same parameter, perform another ADI double-pass to advance from u^{n+1} to u^{n+2}.
2. With a different relaxation parameter, $\tilde{\omega} \equiv \omega^n/2$, perform one ADI double-pass to advance from u^n to $\tilde{u}^{n+2}$.
3. Compute the ratio $r = \left\|\tilde{u}^{n+2} - u^{n+2}\right\|_2 / \left\|u^{n+2} - u^n\right\|_2$.
4. Set $\omega^{n+2} = c\omega^n$ according to a table such as Table 3.1, knowing r.

unscaled		diagonally scaled	
r	c	r	c
$0 < r < 0.05$	0.2	$0 < r < 0.01$	0.25
$0.05 \leq r < 0.15$	0.5	$0.01 \leq r < 0.05$	0.5
$0.15 \leq r < 0.3$	0.9	$0.05 \leq r < 0.1$	0.9
$0.3 \leq r < 0.4$	2.	$0.1 \leq r < 0.2$	2.
$0.4 \leq r < 0.6$	4.	$0.2 \leq r < 0.3$	4.
$0.6 \leq r$	16. (reject)	$0.3 \leq r$	16. (reject)

Table 3.1. Tables used for relaxation parameter adjustment for two-dimensional DADI and DSDADI. The parenthesized *reject* implies that u^{n+2} should be replaced by u^n.

When the prescription is used with nonpreconditioned ADI, the method is called *dynamic* ADI (DADI) in this dissertation. When the prescription is used with diagonally scaled ADI (see above), the method is called *diagonally scaled dynamic* ADI (DSDADI). Analysis of the two-dimensional model problem determines that the optimum value of the ratio r is between 0.1 and 0.3 for unscaled DADI [Doss and Miller 1979]. In this dissertation, all of the runtime results from the DADI and DSDADI methods are based on Table 3.1.

It is a property of DADI and DSDADI that as iterations are performed the magnitude of the relaxation parameter oscillates, helping to drive down the various eigencomponents in the error vector e^n. On top of the oscillation is a gradual decrease in parameter magnitude. During the relaxation, the iterates fail to accurately follow in an absolute sense the evolution of the corresponding time-dependent diffusion equation. But absolute accuracy is not necessary; it is only necessary that $\|e^n\| \to 0$.

3.3 Spatial domain decomposition for parallel ADI

3.3.1 The decomposition

Parallelization of the electron-field advance can be achieved by spatial domain decomposition (SDD) in which the uniform two-dimensional orthogonal finite-difference grid is sliced into regular subdomains by coordinate lines. Examples of SDD are given in Figures 3.5 and 3.6, in which the grid is sliced by lines of constant x and z.

The SDD shown in Figure 3.5 is the one chosen for parallel implementation of ADI. Since the ADI electron-field advance is based on the solution of tridiagonal systems of equations with unknowns along coordinate lines, the unknowns of such systems in Figure 3.5 become distributed over multiple processors. In this way, distributed tridiagonal system solvers become the core of the parallel electron-field algorithm for the pinch simulation. Whereas parallel solution of each distributed tridiagonal system takes over twice the arithmetic operations (AOs) necessary for serial solution when the unknowns are all on one processor, the SDD technique in Figure 3.5 has better scalability than the SDD technique in Figure 3.6 that uses the standard serial solution.

The parallel performance of the SDD in Figure 3.6 suffers because it requires too much communication of grid data. If the spatial region is evenly divided between P processors, with N unknowns per processor, then roughly $(P-1)N/P$ equations have to be communicated from each processor to the $N-1$ other processors between ADI passes. Each messages containing N/P equations can slow execution because of relatively long transmission times. Also, the communication latency associated with the $N-1$ messages further subtracts from performance.

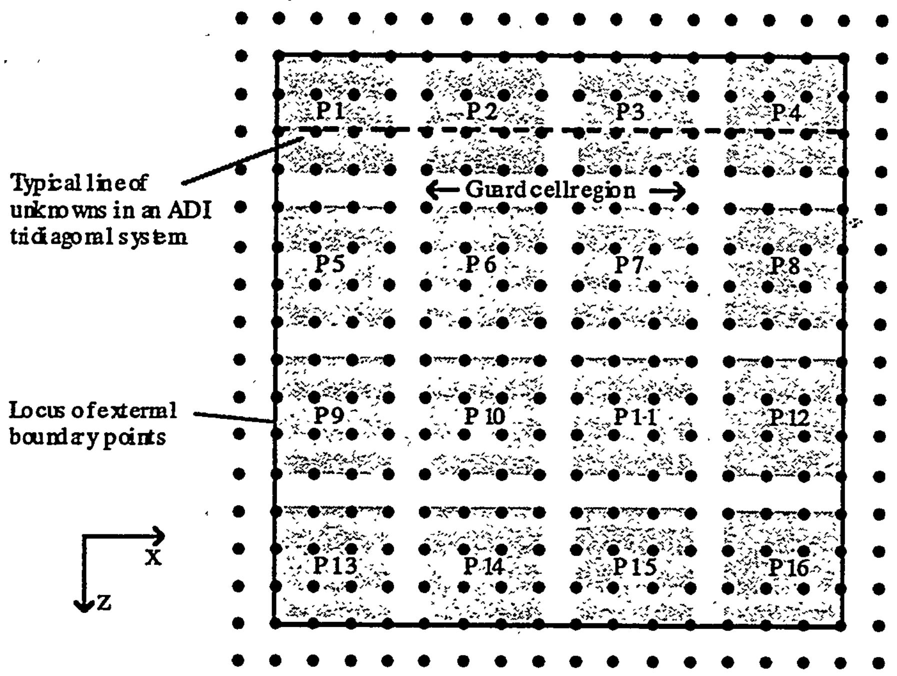
P1
P2
P3
P4
Typical line of unknowns in an ADI tridiagonal system
Guard cell region
P5
P6
P7
P8
Locus of external boundary points
P9
P10
P11
P12
X
Z
P13
P14
P15
P16

Figure 3.5. Typical parallel 2-d ADI spatial domain decomposition (SDD). This is a 4×4 processor spatial domain decomposition SDD of an 18×18 grid. Each of 16 subdomains is associated with a processor P1-P16. Each processor allocates memory for 16 grid points (4×4 grey region) in the interior of a subdomain, along with 20 more points immediately adjacent to the 16 across the designated "guard cell region". External boundary conditions are applied at points connected by the solid line. A tridiagonal system of equations associated with the delineated line of unknowns is distributed evenly over processors P1-P4.

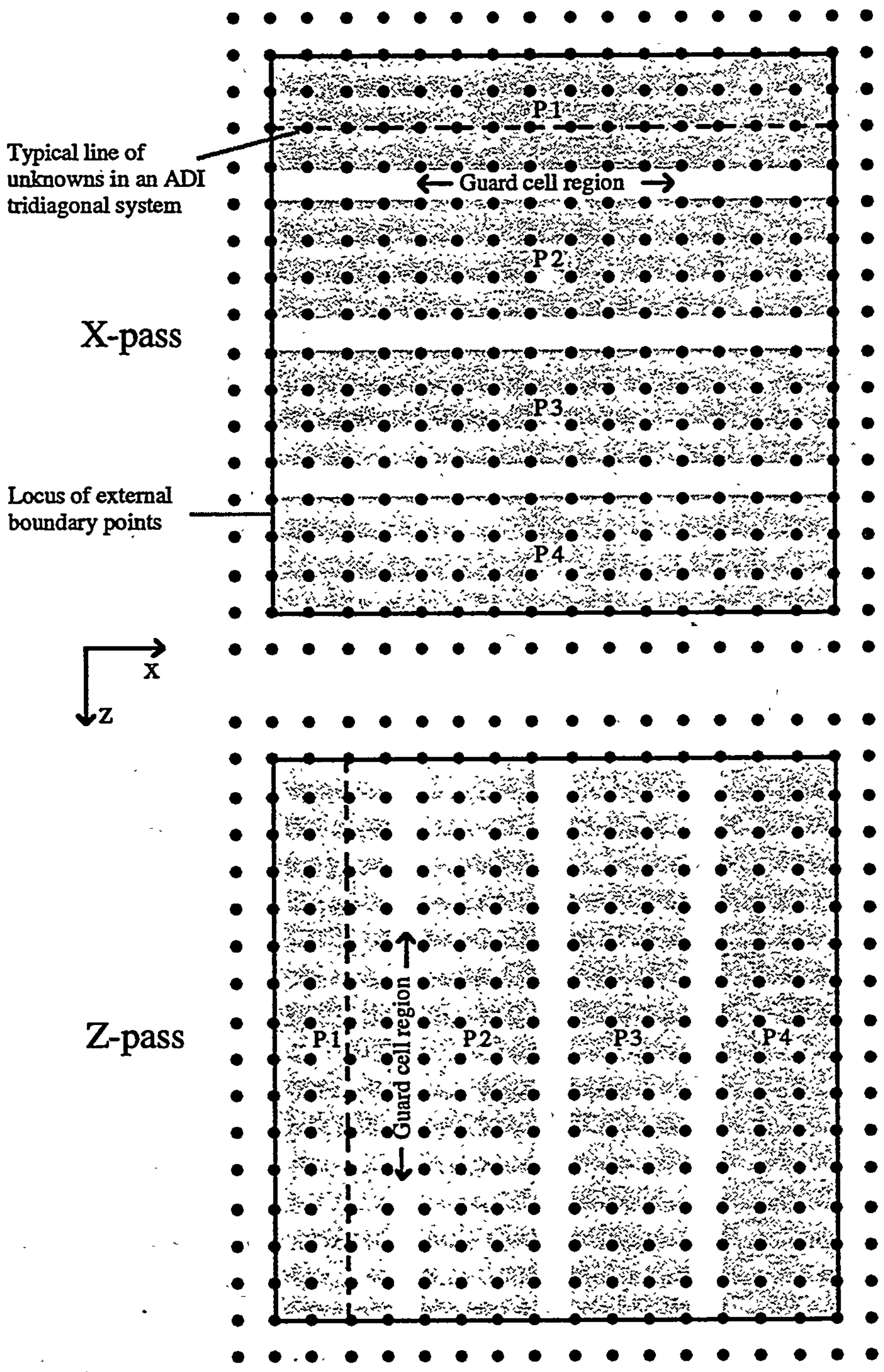
Typical line of unknowns in an ADI tridiagonal system
P1
← Guard cell region →
P2
X-pass
P3
Locus of external boundary points
P4
x
z
Z-pass
P1
← Guard cell region →
P2
P3
P4

Figure 3.6. Four-processor example of the data-transposition method of parallelization. Domains are transposed between passes so that tridiagonal systems can be solved with single-processor code.

On the other hand, the SDD of Figure 3.5 allows straightforward compile-time load balancing and minimizes interprocessor communication. Even division of the spatial region into subdomains is all that is needed for good load balancing. The communication latency is minimized by bunching information for all of the tridiagonal systems crossing coordinate lines before the information is sent to adjacent processors. The number of bytes in interprocessor messages are proportional to the number of unknowns adjacent to the coordinate lines; this makes the communication time per unknown linear in the ratio of the subdomain perimeter to subdomain area for two-dimensional grids. The message lengths are so small for square subdomains that the message transmission time is relatively negligible. Each processor in the decompostion has full information about one particular subdomain throughout the field advance. Unknowns have static memory indexing on each processor. Processors do not directly share the values of grid unknowns in the subdomain interiors, but they have to communicate in order to update the unknowns.

3.3.2 Tagged message passing on parallel computers

The type of interprocessor communication used to implement parallel ADI is called *tagged message passing*. Tagged message passing is implemented by paired subroutine calls; a *send* subroutine call on one processor and a *receive* subroutine call on another processor. Processor A sends a message to processor B when A executes a call to a send subroutine. B receives the message by executiuon of a receive subroutine. At a conceptual minimum, arguments of the send subroutine determine which processor is to receive the message, the message array to be sent, the length of the message to be sent (usually the number of bytes), a message tag that identifies the content of the message, and other parameters that affect communication. In a Fortran implementation, the send call might have the following syntax:

```
call send(procto,message,messagelength,tag,options).
```

Processor B receives the message from processor A and stores it in memory. At a minimum, arguments of the receiving subroutine determine the processor that sent the message, the array space to which the incoming message is to be written, the length of the message that was sent, and a message tag that identifies the content of the incoming message. The following syntax would be typical for a Fortran implementation:

```
call receive(procfrom,message,messagelength,tag,options).
```

Although the syntax of message-passing libraries varies, they all implement the send and receive subroutines in the same conceptual manner.

A useful symbolism for illustration of interprocessor communication uses boxes for the processors and arrows for the messages sent between the processors. The arrow corresponding to each message has a tail at the sending processor and a head at the receiving processor, as in Figure 3.7.

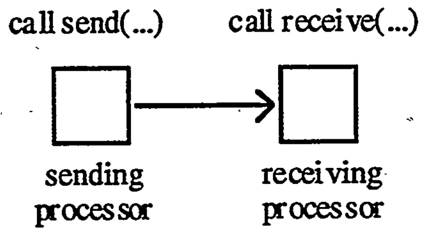

Figure 3.7. Illustration of a send-receive pair (SR pair).

For lack of better terminology, two processors communicating as in Figure 3.7 will be referred to as a *send-receive* pair (SR pair). On many parallel computers, each processor can receive only one message at a time, although it might be waiting for more than one. When the receiving processor waits for a particular message before executing any other machine instructions, then the receive is called a *blocking receive*. In the ADI code blocking receives are used to synchronize the processors, in which case each processor waits for only one message at a time and ceases to perform useful calculations until the message is received.

Usually, each processor on a parallel machine can be waiting to receive a message simultaneously with all other processors; so for P processors, as many as P SR pairs can be executing simultaneously A *volley* is a set of SR pairs executing simultaneously for which no processor waits to receive a message from more than one processor. The volleys used for solving the tridiagonal systems of ADI for the SDD in Figure 3.5 are illustrated in Figure 3.8.

When the residual is calculated after the Z-pass in the dynamic ADI (DADI) method, it becomes necessary to communicate between processors the iterates at the x-boundaries of the subdomains. This is because the residual at a point requires the values from the surrounding four points. This communication of the x-boundary information is called a *nearest-neighbor communication in the x-direction*. It consists of SR pairs between each processor to the neighboring processor in the +x direction followed by SR pairs from each processor to the neighboring processor in the -x-direction. Such communication volleys in the x- and z-directions are illustrated in Figure 3.9.

In the DADI method, global norms must be calculated in order to determine the next time step/relaxation parameter for Eq. 3.8. An all-to-all broadcast is the most efficient manner of communicating the necessary norm contributions. A way to implement an all-to-all broadcast with the least number of volleys was presented in an earlier work [Mattor *et al.* 1995]. In the nth volley of the all-to-all broadcast between P processors, $n = 1,\ldots,\mathrm{int}(\log_2(P-1))+1$, the pth processor sends a message to the $(p+2^{n-1})$th processor, for all $p \in [1,P]$. Whatever the nature of the information that

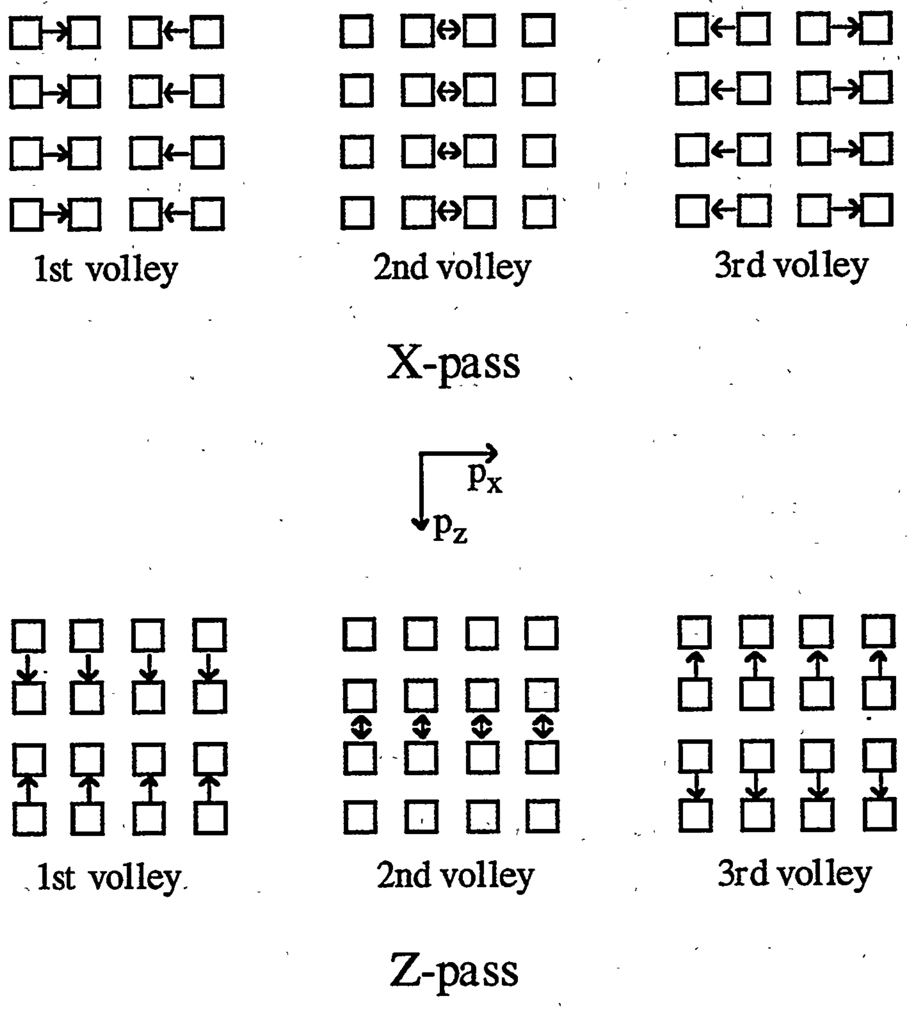

Figure 3.8. Volleys used for solving ADI tridiagonal systems by the two-way skip decoupling method described in Chapter 4. This is based on the SDD of Figure 3.5. The p_x and p_z arethe processor indices in the x- and z-directions.

each processor has before the first volley, that information can be known by all processors after completion of such a broadcast.

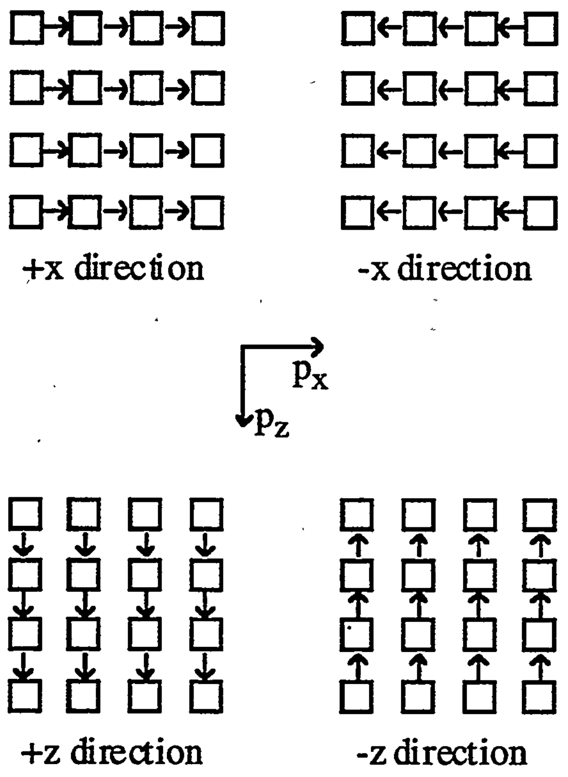

Figure 3.9. Illustration of nearest-neighbor (NN) communication in the x-direction (top) and in the z-direction (bottom) for the 4×4 SDD of Figure 3.5.

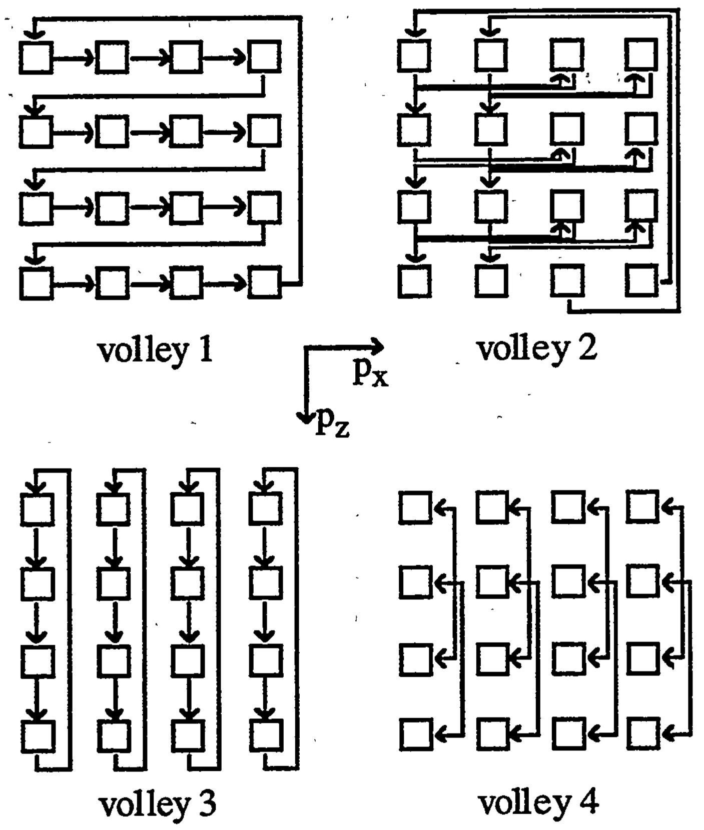

Figure 3.10. Illustration of an efficient all-to-all broadcast between the 16 processors of the SDD in Figure 3.5. This is a 16-processor all-to-all broadcast method used in earlier parallel codes [Mattor *et al.* 1995].

3.4 Parallel two-dimensional DADI

The first step toward writing a parallel computer program for the pinch simulation is to write a program for the ADI method. Once a simple ADI method is made parallel, the A_θ and B_θ updates become achievable with some more work, including parallel versions of line integration for entrained vacuum voids. Other aspects of the pinch simulation, such as evaluation of the electron fluid velocity, the poloidal magnetic flux density, and the electric field, require nearest-neighbor communication in addition to the ADI communication. Indeed, parallelization of the whole pinch simulation is a task that is too large for this dissertation. For this reason, the simplest tractible ADI method was parallelized for solution of $\bar{\nabla}\cdot\varepsilon\bar{\nabla}u-\mu u = \rho$. Convergence to the solution of $\bar{\nabla}\cdot\varepsilon\bar{\nabla}u-\mu u = \rho$ guarantees that the parallel ADI method is free of programming bugs, making convergence a natural milestone in development of parallel ADI for the whole pinch simulation.

3.4.1 DADI algorithm

The parallel ADI algorithm is based on the spatial domain decomposition (SDD) illustrated in Figure 3.5 of the previous section. The SDD technique is utilized because it yields favorable scaling properties. As far as parallel scaling, it is desirable that the execution time varies inversely with the number of processors in the SDD for a fixed number of grid unknowns. Furthermore, one wants the inverse variation to hold for small or large numbers of grid unknowns. It is expected that the SDD of Figure 3.5 is nearly optimal for such parallel scaling. Another reason for chosing an SDD-based parallel implementation is the utility of SDD in parallelization of particle-in-cell (PIC) algorithms that are necessary for modeling kinetics. While it is plausible that the SDD approach is nealy optimal, such has not been proven since a large number of comparisons of the parallel implementations are not a practical goal for this dissertation.

The parallel DADI program based on SDD is written so that the dimensions of the subdomains can be specified along with the number of subdomains for each orthogonal direction of the problem. These are compile-time specifications that evenly divide the spatial domain. When a single domain is specified for the whole problem, the program uses a one-processor serial algorithm. If domains span only one dimension of the whole space, then a serial algorithm is used for the direction that is spanned. Parallel DADI is written so that Robbins boundary conditions can be applied at the external boundary and so that Dirichlet boundary conditions can be applied anywhere in the problem space. Parallel tridiagonal system solvers of both a direct type and an approximate type are incorporated in the program (see Chapter 4). Two different criteria can be used to determine convergence: one based on a residual norm (RN) that is robust for all boundary conditions and another based on an *estimated maximum pointwise relative error* (EMPRE) that works only for nonuniform boundary conditions. The residual norm is given by Eq. 3.29, in which the residual (abbreviated res) is implicitly defined. The operator in the residual is the discrete matrix operator $\bar{\nabla} \cdot \varepsilon \bar{\nabla} - \mu \equiv H + V$.

$$RN \equiv \sqrt{\sum_{i,j} res_{i,j}^2} \equiv \sqrt{\sum_{i,j} \left[\left(\bar{\nabla} \cdot \varepsilon \bar{\nabla} u - \mu u \right)_{i,j} - \rho_{i,j} \right]^2} \tag{3.29}$$

The EMPRE is defined in Eq. 3.30, where the quantity $\bar{u}$ is just the mean value of the unknown $\bar{u}$ over the grid.

$$EMPRE \equiv \frac{\max\limits_{i,j} \left[\left(\bar{\nabla} \cdot \varepsilon \bar{\nabla} - \mu_{i,j} \right) u_{i,j} - \rho_{i,j} \right] \max\limits_{k} \Delta x_k^2}{\min \left[\max\limits_{i,j} u_{i,j} - \bar{u},\ \bar{u} - \min\limits_{i,j} u_{i,j} \right]} \tag{3.30}$$

For an $M \times N$ grid, the mean value of the unknown is given by

$$\bar{u} = \frac{1}{MN} \sum_{i,j} u_{i,j} . \tag{3.31}$$

The EMPRE is a ratio of the worst pointwise error due to curvature and the minimum absolute deviation of the unknown from its average. For solutions that are a constant over the whole spatial grid, the denominator of Eq. 3.30 approaches zero, making the EMPRE useless. The residual norm itself can then be used for a convergence criterion.

In order to test the DADI and DSDADI methods on problems with inhomogeneous diffusivity, a *mosaic* diffusivity $\varepsilon_{i,j} = mosaic_{i,j}$ was generated, where

$$mosaic_{i,j} \equiv 1 + 999 ran(\text{int}(i/3), \text{int}(j/3)), \quad 1 \le i \le M,\ 1 \le j \le N\ ,$$
$$ran(i,j) \equiv (i,j)\text{th random number}, \quad 0 \le ran(i,j) \le 1.$$

Similarly, a mosaic source ρ and mosaic μ were programmed. The form of the diffusivity, source, and μ can be selected through the input file.

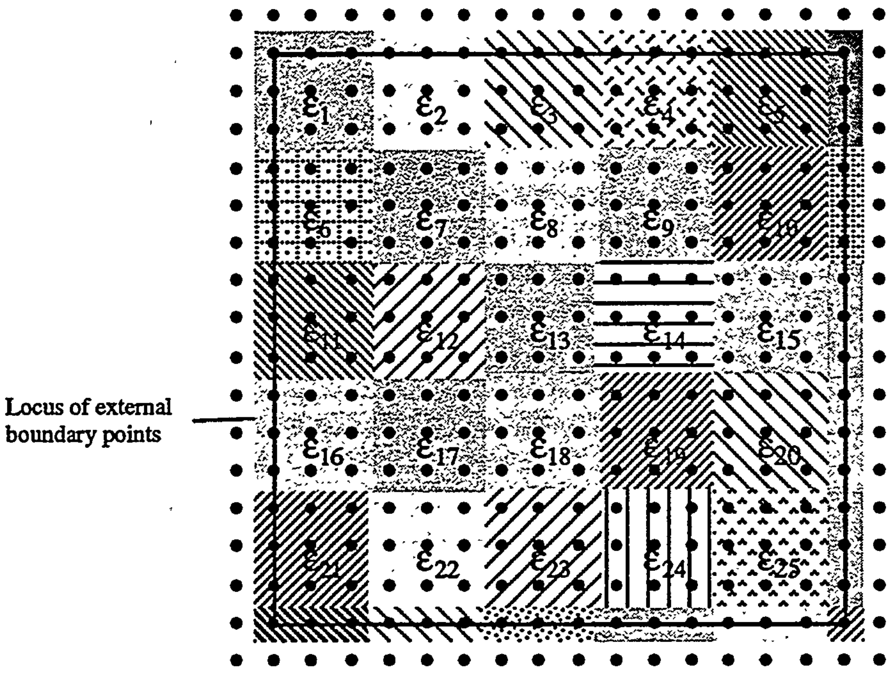

Figure 3.11. Illustration of a 3×3 piecewise constant mosaic permittivity on an 18×18 grid. An inhomogeneous permittivity of this form is used to test DADI and DSCG in an extreme case with ε_i randomly chosen between 1 and 1000.

A concise overview of the program is given by the flowchart of Figure 3.12. Besides the DADI and DSDADI methods , the method of diagonally scaled conjugate conjugate gradients (DSCG) was also programmed with the same SDD, diffusivity, boundary conditions, convergence criteria, and so forth. The DSCG method has an efficient parallel implementation and is good for comparison with the DADI method. Conjugate gradient methods have been studied elsewhere [Golub and Van Loan 1989; Greenbaum *et al.* 1989]. Except for the core of the iteration, the flowchart for the DSCG method is quite similar to that for the DADI method. For brevity, the actual source codes of the parallel programs are not included in this dissertation.

3.4.2 Performance

Dirichlet boundary conditions chosen for testing DADI are such that the discrete solution for $\varepsilon = 1,\ \mu = \rho = 0$ is the same as the exact continuous solution. With i and j the grid indices, these boundary conditions are:

$$\begin{aligned} u_{1,j} &= 1 - j^2, \quad u_{M,j} = M^2 - j^2, \quad j \in [1, N], \\ u_{i,1} &= i^2 - 1, \quad u_{i,N} = i^2 - N^2, \quad i \in [1, M]. \end{aligned}$$

The exact continuous solution to this problem $(u(i,j) = i^2 - j^2)$ has no nonzero spatial derivatives higher than second order. Because the methods are based on second-order central differences, ADI and DSCG should converge to the exact solution $u_{i,j} = i^2 - j^2$ (integer i and j) over the whole grid regardless of grid spacing as long as it is uniform. This convergence is easily verified.

Figure 3.13 shows results of parallel solution of the Laplace equation $\nabla^2 u = 0$ on the Meiko CS-2 (1994). The parallel DADI methods, unscaled and diagonally scaled, are compared with a parallel method of diagonally scaled conjugate gradients (DSCG). For this fixed problem size and convergence criterion, the DADI methods are clearly superior to the DSCG method as the number of processors is increased. The DADI methods

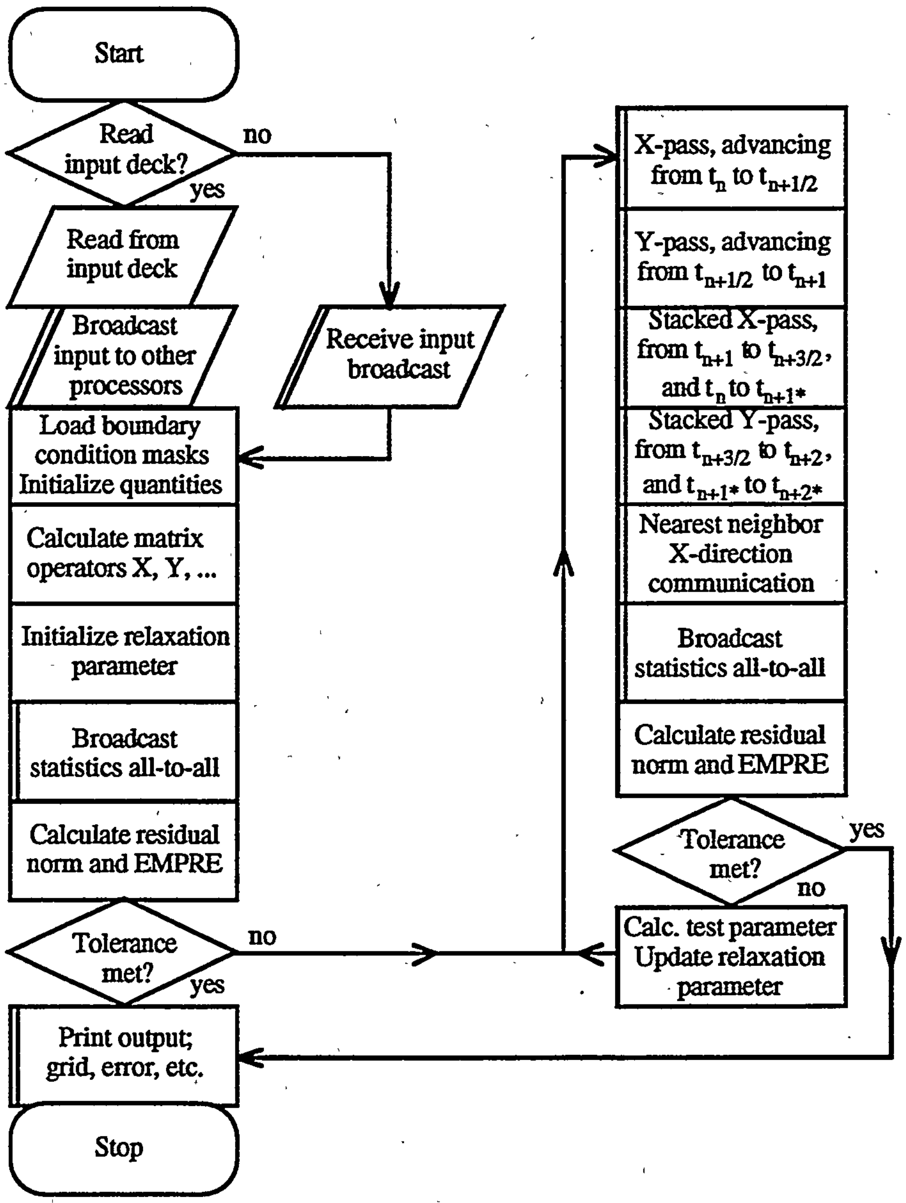
Start
Read input deck?
no
yes
Read from input deck
Broadcast input to other processors
Receive input broadcast
Load boundary condition masks Initialize quantities
Calculate matrix operators X, Y, ...
Initialize relaxation parameter
Broadcast statistics all-to-all
Calculate residual norm and EMPRE
Tolerance met?
no
yes
Print output; grid, error, etc.
Stop
X-pass, advancing from t_n to $t_{n+1/2}$
Y-pass, advancing from $t_{n+1/2}$ to t_{n+1}
Stacked X-pass, from t_{n+1} to $t_{n+3/2}$, and t_n to t_{n+1*}
Stacked Y-pass, from $t_{n+3/2}$ to t_{n+2}, and t_{n+1*} to t_{n+2*}
Nearest neighbor X-direction communication
Broadcast statistics all-to-all
Calculate residual norm and EMPRE
Tolerance met?
yes
no
Calc. test parameter Update relaxation parameter

Figure 3.12. Condensed two-dimensional DADI algorithmic flowchart. Depicted are the most important general steps. Double lines in flowchart symbols indicate that interprocessor communication is necessary.

achieve peak speedup with over 128 processors, whereas the DSCG method has a peak speedup with 80 processors. The DSCG method requires significantly more interprocessor communication to achieve convergence, so that parallel efficiency is greatly diminished for more than 50 processors. Here, the DADI method is capable of executing 10 times faster than the DSCG method. Diagonally scaled DADI (DSDADI) takes a bit longer to execute because diagonal scaling adds to the number of floating point divisions without enhancing convergence; it merely divides diagonal coefficients of $H+V$ by a decimal 1.

Figure 3.14 shows results of parallel solution of $\bar{\nabla}\cdot\varepsilon\bar{\nabla}u=0$ on the Meiko CS-2 (1994). In this case the diffusivity ε has the mosaic form introduced above. Again, DADI and DSDADI are compared with DSCG. In this case, the virtue of DSDADI is apparent; it outperforms DADI and DSCG by roughly a factor of 2 with 128 processors. Of course, both DADI and DSDADI scale better than DSCG to greater numbers of processors, just as for the Laplace equation. However, the mosaic permittivity diminishes the performance of the DADI methods more than the performance of the DSCG method. This suggests that DSCG might be the algorithm of choice for cases of extremely discontinuous permittivity.

Now, for solution of $\bar{\nabla}\cdot\varepsilon\bar{\nabla}u-\mu u=\rho$, the scaling of the DADI methods with increasing number of processors is the same as would be achieved with parallel ADI methods for evolution of the parabolic equation $\partial_t u=\bar{\nabla}\cdot\varepsilon\bar{\nabla}u-\mu u-\rho$. Furthermore, parallelization of DADI for $\bar{\nabla}\cdot\varepsilon\bar{\nabla}u-\mu u = \rho$ simultaneously achieves parallelization of noniterative ADI for evolution of $\partial_t u=\bar{\nabla}\cdot\varepsilon\bar{\nabla}u-\mu u-\rho$. The same cannot be claimed for DSCG or multigrid methods; they do not evolve parabolic equations as efficiently.

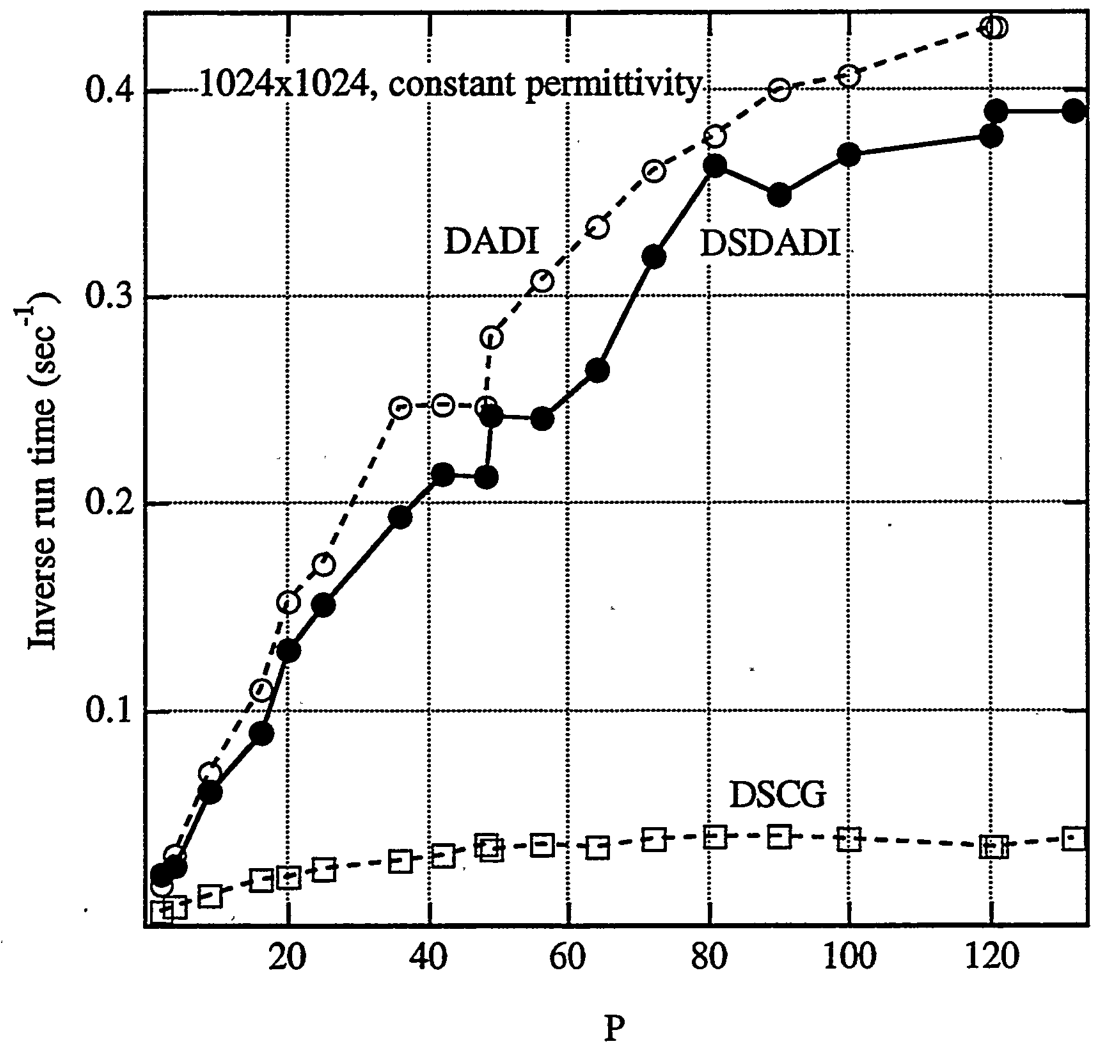

Figure 3.13. Inverse run time versus number of processors P for the solution of the Laplace equation on a 1024×1024 grid by the DSDADI, DADI, and DSCG methods. Boundary conditions are $u_{i,j} = i^2 - j^2$ on ∂R, grid spacings are $\Delta x = \Delta z = 1$, initial conditions are $u_{i,j} = 0$ inside R, and the convergence criterion is $EMPRE < 0.00001$.

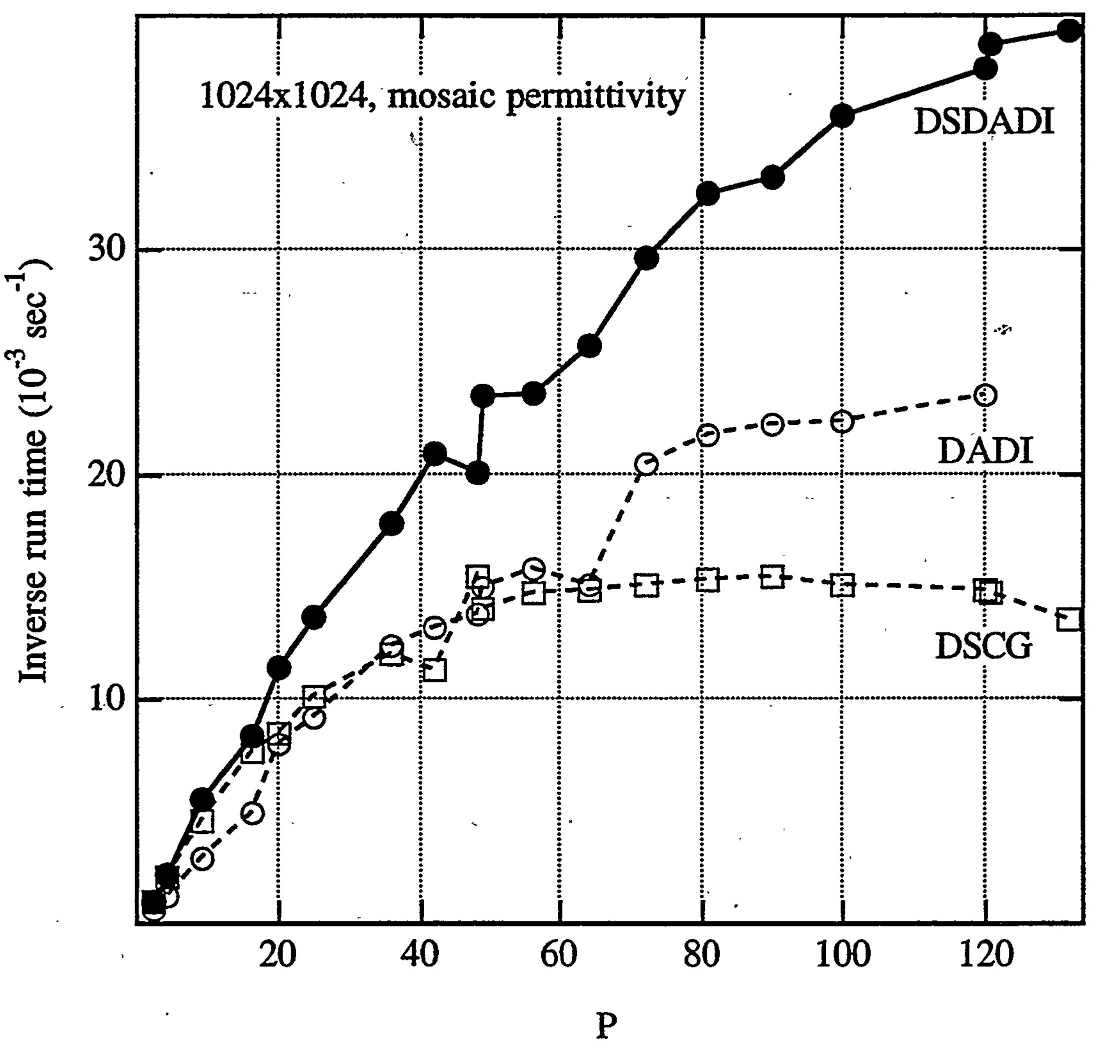

Figure 3.14. Inverse run time versus number of processors P for the solution of $\bar{\nabla} \cdot \varepsilon \bar{\nabla} u = 0$ with mosaic diffusivity on a 1024×1024 grid by the DSDADI, DADI, and DSCG methods. Boundary conditions are $u_{i,j} = i^2 - j^2$ on ∂R, grid spacings are $\Delta x = \Delta z = 1$, initial conditions are $u_{i,j} = 0$ inside R, and the convergence criterion is $EMPRE < 0.00001$.

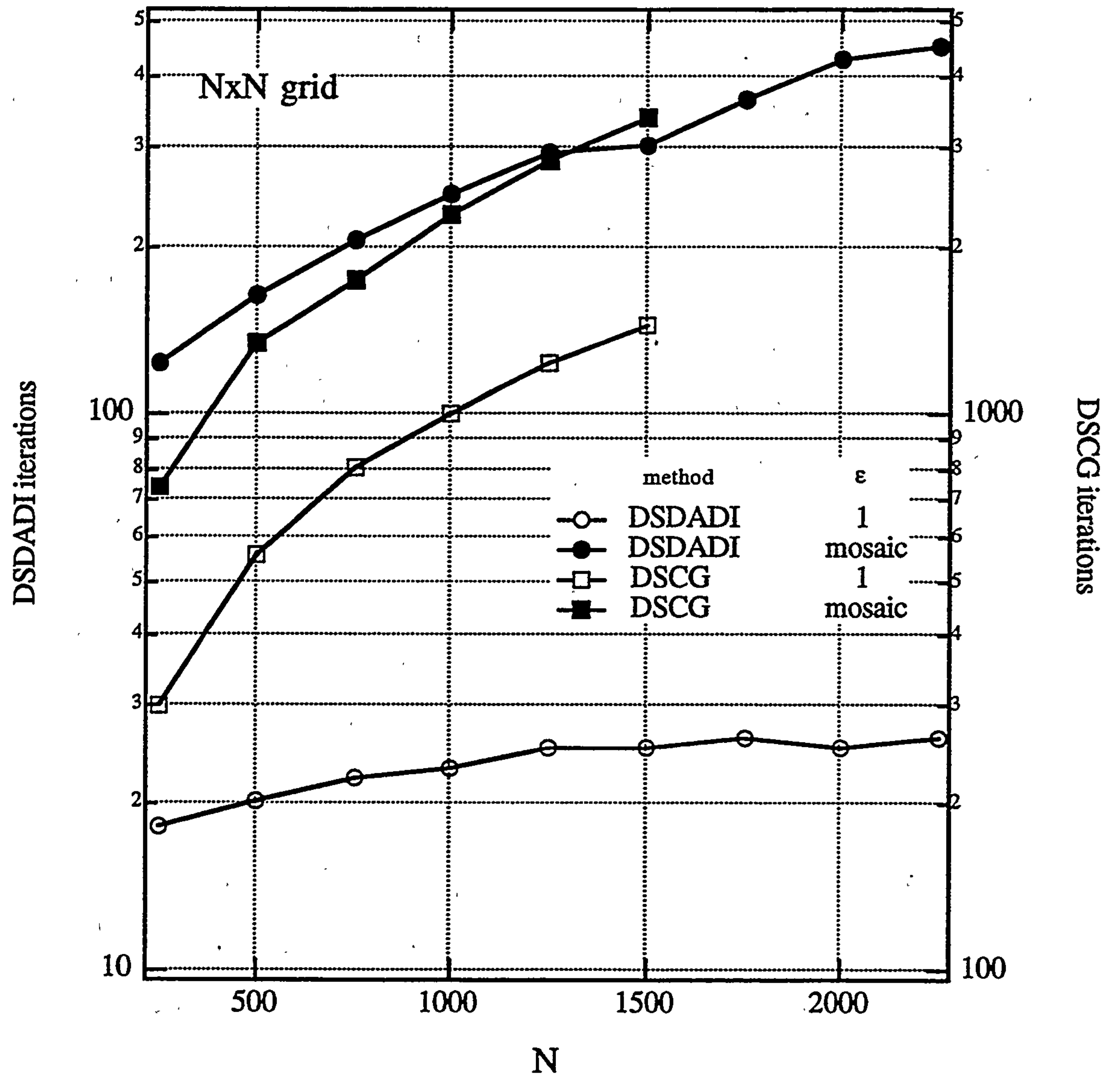

Figure 3.15. Iterations versus grid size, $N \times N$, for solution of $\bar{\nabla} \cdot \varepsilon \bar{\nabla} u = 0$ by the DSDADI and DSCG methods. Boundary conditions are $u_{i,j} = i^2 - j^2$ on ∂R, grid spacings are $\Delta x = \Delta z = 1$, initial conditions are $u_{i,j} = 0$ inside R, and the convergence criterion is $EMPRE < 0.00001$.

Figure 3.15 shows relative performance of the DSDADI method and the DSCG method on the Meiko CS-2 when the problem size is increased. Although this figure is not related to parallel scalability, it should be noted that such large problem sizes were made possible by a parallel computer, which had sufficient memory space for the problem. Performance of the methods is measured in terms of the number of iterations necessary for convergence. For the problems chosen, DSCG requires over ten times the number of iterations (note the seperate scales). The relative rate of increase of iteration counts with increasing N is also higher for DSCG. It is clear that DADI scales better to larger problem sizes, especially for the Laplace equation.

The results presented in this section indicate that ADI methods are practical for parallel computations. This is extremely important for the future of the ADI-based pinch simulation algorithm. On parallel computers, pinch simulation is sure to achieve higher resolution and faster computational speeds so that more detailed comparison of theory and experiment will be possible.

3.5 Design of a parallel screw pinch simulation

Outlined in this section is a parallel algorithm for a complete screw pinch simulation. The problem is laid out in terms of crude but useful pseudocode. In the outline, special attention is given to ADI double-passes and various message-passing schemes necessary for update of all of the field quantities. The ADI double-passes and communication schemes have already been implemented in the parallel DADI algorithm presented above. The work that remains to be done for a complete parallel pinch simulation involves the evaluation of all of the coefficients for the ADI passes, as well as the parallel evaluation of all of the auxiliary field quantities.

1) Assume all t^0 quantities are known, as well as ρ^1, and u_{ir}^1 are known over whole subdomains (including guard points). The density ρ^1 and ion drift velocity u_{ir}^1 are determined by a time advance of PIC ions or fluid ions.

2) Allow calculation of $u_{er}^1 = f\left(\rho^1, u_{ir}^1, B_\theta^0\right)$. The first option is to use $u_{er}^1 = \rho^0(u_{er}^0 - u_{ir}^0)/\rho^1 + u_{ir}^1$ to get u_{er}^1 over whole subdomains, without communication. The second option is to use $u_{er}^1 = u_{ir}^1 + \frac{c}{8\pi e\rho^1\Delta z}\left(B_{\theta 01}^0 - B_{\theta 0-1}^0\right)$, delaying nearest-neighbor communication of u_{er}^1 in the Z-direction until a later time. Calculation of u_{er}^1 at this point follows the method of Hewett.

3) Execute a combined plasma-vacuum R-pass on the A_θ rate equation to find $A_\theta^{1/2}$ This proceeds just as for a DADI R-pass.

4) Allow calculation of a temporary $\tilde{B}_r$, $\tilde{B}_z$ and $\tilde{u}_{e\theta}$ for later insertion in the plasma-vacuum R-pass on B_θ. This requires communication of $A_\theta^{1/2}$ in the R-direction between nearest neighbors. Calculate them over the interior of each subdomain according to the equations (θ subscripts dropped):

$$\tilde{B}_r = -\frac{1}{2r\Delta z}\left(A_{01}^{1/2} - A_{0-1}^{1/2}\right), \qquad \tilde{B}_z = \frac{1}{2r\Delta r}\left(A_{10}^{1/2} - A_{-10}^{1/2}\right),$$

$$\tilde{u}_e = u_i^1 + \frac{c}{4\pi e\rho^1}\left(\frac{A_{01}^{1/2} - 2A^{1/2} + A_{0-1}^{1/2}}{r\Delta z^2} + \frac{1}{\Delta r^2}\left(\frac{A_{10}^{1/2} - A^{1/2}}{r_{1/2}} - \frac{A^{1/2} - A_{-10}^{1/2}}{r_{-1/2}}\right)\right).$$

Communicate $\tilde{B}_r$ and $\tilde{u}_{e\theta}$ in the R-direction between nearest neighbors to update R-guard points. Communicate $\tilde{B}_z$, $\tilde{u}_{e\theta}$, and u_{er}^1 between nearest neighbors in the

Z-direction to update Z-guard points. Evaluation of $\tilde{B}_r$, $\tilde{B}_z$ and $\tilde{u}_{e\theta}$ at this time might lead to more accurate rate equation coupling.

5) Execute a combined plasma-vacuum Z-pass on the A_θ rate equation. This proceeds just as for a DADI Z-pass. Communicate $\hat{A}_\theta$ in the R-direction between nearest neighbors to update R-guard points.

6) Allow $\hat{A}_\theta$ relaxation in the vacuum regions. This is coined the "vacuum cleanup" by Hewett. Use ADI with a trivial parameter variation (not necessarily Doss-Miller DADI).

7) Allow calculation of a temporary $\hat{B}_r$, $\hat{B}_z$ and $\hat{u}_{e\theta}$ for alternative use in the plasma-vacuum R-pass on B_θ. Calculate them over the interior of each subdomain according to the equations (θ subscripts dropped):

$$\hat{B}_r = -\frac{1}{2r\Delta z}\left(\hat{A}_{01} - \hat{A}_{0-1}\right),\ \hat{B}_z = \frac{1}{2r\Delta r}\left(\hat{A}_{10} - \hat{A}_{-10}\right),$$
$$\hat{u}_e = u_i^1 + \frac{c}{4\pi e\rho^1}\left(\frac{\hat{A}_{01} - 2\hat{A} + \hat{A}_{0-1}}{r\Delta z^2} + \frac{1}{\Delta r^2}\left(\frac{\hat{A}_{10} - \hat{A}}{r_{1/2}} - \frac{\hat{A} - \hat{A}_{-10}}{r_{-1/2}}\right)\right).$$

Communicate $\hat{u}_{e\theta}$ in the R-direction between nearest neighbors. Communicate $\hat{u}_{e\theta}$, and u_{er}^1 in the Z-direction between nearest neighbors to update Z-guard points. Calculation of $\hat{B}_r$, $\hat{B}_z$ and $\hat{u}_{e\theta}$ at this point follows the original method of Hewett [1980].

8) Allow calculation of $\hat{E}_{r,z} = f\left(\rho^1, T_e^1, \hat{B}_{r,z}, u_{er}^1, \hat{u}_{e\theta}, u_{ez}^0\right)$, giving $\hat{E}_r$ in the R-interior and $\hat{E}_z$ in the Z-interior. Zero the Ohmic contribution in vacuum regions.

Communicate the result in the R-direction bertween nearest neighbors to update R-guard points. Communicate the result in the Z-direction between nearest neighbors to update Z-guard points:

9) Allow calculation of $E_{r,z}^{1/2} = \frac{1}{2}\left(E_{r,z}^{0} + \hat{E}_{r,z}\right)$ over whole subdomains.

10) Allow multiprocessor vacuum void processing:

a) Intraprocessor void labeling

b) Intraprocessor integration -- null contribution along processor boundaries, either for $\overline{rB_\theta}$ or for $\oint_{\partial R} \vec{E} \cdot d\vec{\ell}$.

c) Interprocessor label association and transfer of line integral contributions.

d) Repeated nearest-neighbor communication to transfer updated k^{n+1}.

11) Perform an ADI R-pass on the B_θ equation, just as for a parallel DADI R-pass.

12) Make it an option to calculate $u_{ez}^{1} = f\left(\rho^{n+1}, u_{iz}^{1}, B_\theta^{1/2}\right)$. These u_{ez}^{1} values are easily calulated at R-interior points of each subdomain, and there are the only u_{ez}^{1} values needed for subsequent Z-passes on B_θ and A_θ. Delay communication of u_{ez}^{1} in the R-direction. Calculation of u_{ez}^{1} in this way follows the method of Hewett [1980].

13) Perform a parallel ADI Z-pass on the B_θ equation, just as for a parallel DADI Z-pass. Delay communication of B_θ^{1} in the R-direction.

14). Make it an option to perform a parallel Z-pass on A_θ, proceeding as for a parallel DADI Z-pass. Communicate A_θ^1, u_{ez}^1 and B_θ^1 in the R-direction between nearest neighbors.

15) Make it an option to relax on the A_θ^1 in the vacuum regions. Use ADI with a trivial parameter variation. Communicate the last A_θ^1 between nearest neighbors in the R-direction.

16) Calculate B_r^1, B_z^1 and $u_{e\theta}^1$. These are easily calculated over the interior of each subdomain according to the equations (θ subscripts dropped):

$$B_r^1 = -\frac{1}{2r\Delta z}\left(A_{01}^1 - A_{0-1}^1\right), \qquad B_z^1 = \frac{1}{2r\Delta r}\left(A_{10}^1 - A_{-10}^1\right),$$

$$u_e^1 = u_i^1 + \frac{c}{4\pi e\rho^1}\left(\frac{A_{01}^1 - 2A^1 + A_{0-1}^1}{r\Delta z^2} + \frac{1}{\Delta r^2}\left(\frac{A_{10}^1 - A^1}{r_{1/2}} - \frac{A^1 - A_{-10}^1}{r_{-1/2}}\right)\right).$$

Communicate $u_{e\theta}^1$ in the R-direction between nearest neighbors. Communicate $u_{e\theta}^1$ and u_{er}^1 in the Z-direction between nearest neighbors.

17) Components of $\vec{E}^1$ can be calculated over whole subdomains without additional interprocessor communication.

From the pseudocode, it is obvious that a parallel pinch simulation requires considerably more work than parallel DADI. Implementing parallel DADI is like playing one instrument in an orchestra, while implementing the complete pinch simulation is like conducting the whole orchestra.

Appendix 3.A1
A serial (tridiagonal-block)^n tridiagonal system solution

Perhaps the most important type of linear system arising from a finite-difference approximation to a PDE is a *(tridiagonal-block)^n tridiagonal* system for $n+1$ spatial dimensions. These arise from 3-point second-order central differences for approximation of second order spatial derivatives. In two dimensions the systems are *tridiagonal-block tridiagonal*, whereas in three dimensions the systems are *tridiagonal-block tridiagonal-block tridiagonal* or *(tridiagonal-block)^2 tridiagonal.*

$$
A = \left(\begin{array}{cccc|cccc|cccc|cccc}
x & x & & & x & & & & & & & & & & & \\
x & x & x & & & x & & & & & & & & & & \\
& x & x & x & & & x & & & & & & & & & \\
& & x & x & & & & x & & & & & & & & \\
\hline
x & & & & x & x & & & x & & & & & & & \\
& x & & & x & x & x & & & x & & & & & & \\
& & x & & & x & x & x & & & x & & & & & \\
& & & x & & & x & x & & & & x & & & & \\
\hline
& & & & x & & & & x & x & & & x & & & \\
& & & & & x & & & x & x & x & & & x & & \\
& & & & & & x & & & x & x & x & & & x & \\
& & & & & & & x & & & x & x & & & & x \\
\hline
& & & & & & & & x & & & & x & x & & \\
& & & & & & & & & x & & & x & x & x & \\
& & & & & & & & & & x & & & x & x & x \\
& & & & & & & & & & & x & & & x & x
\end{array}\right)
= \left(\begin{array}{c|c|c|c}
\mathbf{a}_{1,1} & \mathbf{a}_{1,2} & & \\
\hline
\mathbf{a}_{2,1} & \mathbf{a}_{2,2} & \mathbf{a}_{2,3} & \\
\hline
& \mathbf{a}_{3,2} & \mathbf{a}_{3,3} & \mathbf{a}_{3,4} \\
\hline
& & \mathbf{a}_{4,3} & \mathbf{a}_{4,4}
\end{array}\right)
$$

Figure 3.16. Tridiagonal-block tridiagonal coefficient matrix. Vertical lines are visual aides for discerning matrix structure. Generally nonzero matrix elements are denoted by xs. Here, each $\mathbf{a}_{i,j}$ is a block matrix.

The matrix operator A corresponding to the second order central-differencing of $\nabla \cdot \varepsilon \nabla - \mu$ on an $(M+2) \times (N+2)$ uniform rectangular grid with Dirichlet boundary conditions around the outside is a tridiagonal-block tridiagonal operator. The matrix operator can be written as a matrix with M rows of tridiagonal blocks of size $N \times N$. Alternatively, with a reordering of unknowns, it can be written as a matrix with N rows of tridiagonal blocks of size $M \times M$. This is a matrix operator of interest for two dimensional plasma simulations. The operator for a 6×6 grid would have the following structure. The number of arithmetic operations (AOs) necessary to solve the system $Au = b$ with the A of the form in Figure 3.16 is $O(MN^3)$ or $O(M^3N)$. If the A were a general $MN \times MN$ matrix, then $O(M^3N^3)$ AOs would be necessary for solution.

Solution of the tridiagonal-block tridiagonal system for the $M = N = 4$ problem above could proceed as follows:

Reduce $\mathbf{a_{1,1}}$ to the identity

$$\left(\begin{array}{c:c:c:c} \mathbf{1} & \mathbf{a} & & \\ \hdashline \mathbf{a} & \mathbf{a} & \mathbf{a} & \\ \hdashline & \mathbf{a} & \mathbf{a} & \mathbf{a} \\ \hdashline & & \mathbf{a} & \mathbf{a} \end{array}\right) \Rightarrow \left(\begin{array}{cccc:cccc:cccc:cccc}
1 & & & & x & x & x & x & & & & & & & & \\
 & 1 & & & x & x & x & x & & & & & & & & \\
 & & 1 & & x & x & x & x & & & & & & & & \\
 & & & 1 & x & x & x & x & & & & & & & & \\ \hdashline
x & & & & x & x & & & x & & & & & & & \\
 & x & & & x & x & x & & & x & & & & & & \\
 & & x & & & x & x & x & & & x & & & & & \\
 & & & x & & & x & x & & & & x & & & & \\ \hdashline
 & & & & x & & & & x & x & & & x & & & \\
 & & & & & x & & & x & x & x & & & x & & \\
 & & & & & & x & & & x & x & x & & & x & \\
 & & & & & & & x & & & x & x & & & & x \\ \hdashline
 & & & & & & & & x & & & & x & x & & \\
 & & & & & & & & & x & & & x & x & x & \\
 & & & & & & & & & & x & & & x & x & x \\
 & & & & & & & & & & & x & & & x & x
\end{array}\right)$$

Eliminate $\mathbf{a}_{2,1}$

$$\left(\begin{array}{c:c:c:c}
1 & a & & \\ \hdashline
 & a & a & \\ \hdashline
 & a & a & a \\ \hdashline
 & & a & a
\end{array}\right)
\Rightarrow
\left(\begin{array}{cccc:cccc:cccc:cccc}
1 & & & & x & x & x & x & & & & & & & & \\
 & 1 & & & x & x & x & x & & & & & & & & \\
 & & 1 & & x & x & x & x & & & & & & & & \\
 & & & 1 & x & x & x & x & & & & & & & & \\ \hdashline
 & & & & x & x & x & x & x & & & & & & & \\
 & & & & x & x & x & x & & x & & & & & & \\
 & & & & x & x & x & x & & & x & & & & & \\
 & & & & x & x & x & x & & & & x & & & & \\ \hdashline
 & & & & x & & & & x & x & & & x & & & \\
 & & & & & x & & & x & x & x & & & x & & \\
 & & & & & & x & & & x & x & x & & & x & \\
 & & & & & & & x & & & x & x & & & & x \\ \hdashline
 & & & & & & & & x & & & & x & x & & \\
 & & & & & & & & & x & & & x & x & x & \\
 & & & & & & & & & & x & & & x & x & x \\
 & & & & & & & & & & & x & & & x & x
\end{array}\right)$$

Reduce $\mathbf{a}_{2,2}$ to the identity

$$\left(\begin{array}{c:c:c:c}
1 & a & & \\ \hdashline
 & 1 & a & \\ \hdashline
 & a & a & a \\ \hdashline
 & & a & a
\end{array}\right)$$

Finish forward elimination

$$\left(\begin{array}{c:c:c:c}
1 & a & & \\ \hdashline
 & 1 & a & \\ \hdashline
 & & 1 & a \\ \hdashline
 & & & 1
\end{array}\right)$$

Back-substutute

$$\left(\begin{array}{c:c:c:c}
1 & & & \\ \hdashline
 & 1 & & \\ \hdashline
 & & 1 & \\ \hdashline
 & & & 1
\end{array}\right)$$

At the start, each block matrix $\mathbf{a}$ is either diagonal or tridiagonal, but a diagonal block $\mathbf{a}_{i,i}$ quickly fills as $\mathbf{a}_{i,i-1}$ is eliminated. To reduce the new filled $\mathbf{a}_{i,i}$ to a block identity matrix requires a number of AOs that scales as the cube of the number of equations in the block -- it is just as expensive as a Gaussian elimination on a dense matrix with the same number of equations. If there are M rows of $N \times N$ blocks, the total number of arithmetic operations scales as $O(MN^3)$, as claimed above.

Chapter 3 references

1. Doss, S. and Miller, K., Dynamic ADI Methods for Elliptic Equations, *SIAM J. Numer. Anal.* **16**, 837-856 (1979)

2. Fairweather, G. and Mitchell, A.R. A New Computational Procedure for A.D.I. Methods, *SIAM J. Numer. Anal.* **4**, 163-170 (1967)

3. Golub, G.H. and VanLoan, C.F., Matrix Computations, Johns Hopkins University Press, Baltimore, MD (1989)

4. Greenbaum, A., Li, C., and Chao, H.Z., Parallelizing Preconditioned Conjugate Gradient Algorithms, *Comput. Phys. Comm.* **53**, 295-309 (1989)

5. Lambert, M.A., Rodrigue, G.H., and Hewett, D.W., Parallel DADI Methods for Solution of the Steady State Diffusion Equation, submitted to *Parallel Computing* (April 1996)

6. Lapidus, L. and Pinder, G.F., Numerical Solution of Partial Differential Equations in Science and Engineering, John Wiley & Sons, New York, NY (1982)

7. Peaceman, D.W., and Rachford Jr., H.H., The Numerical Solution of Parabolic and Elliptic Differential Eqautions, *SIAM J.* **3**, 28-41 (1955)

8. Varga, R.S., Matrix Iterative Analysis, Prentice-Hall, Englewood Cliffs, NJ (1962)

9. Wachspress, E.L., Optimum Alternating-Direction-Implicit Iteration Parameters for a Model Problem, *SIAM, J.* **10**, 339-351 (1962)

10. Wachspress, E.L., and Habetler, G.J., An Alternating-Direction-Implicit Iteration Technique, *SIAM J.* **8**, 403-424 (1960)

Chapter 4 Parallel Solution of Distributed Tridiagonal Systems of Equations From Spatial Domain Decomposition

This chapter is devoted to algorithms used in parallel solution of distributed tridiagonal systems of equations for ADI. The first section discusses general tridiagonal systems that can arise from the plasma simulation. The second section discusses a new parallel direct method used to solve the general tridiagonal systems. The final section discusses an approximate parallel tridiagonal solver that finds a solution more quickly than the parallel direct method.

4.1 General Tridiagonal Systems

In the simulation, tridiagonal systems arise from application of alternating-direction-implicit (ADI) methods to equations of the form $\partial_t u = (\nabla \cdot \varepsilon \nabla - \mu)u - \rho$. ADI methods work well for time advance of the equation. For the time asymptotic solution, ADI avoids direct solution of the discrete (tridiagonal-block)^n tridiagonal system of equations via iterative solution of tridiagonal systems. The tridiagonal systems arise from the second order derivatives in orthogonal grid directions, and have the following form for a line of N grid unknowns subject to general physical boundary conditions:

$$a_{1,1}u_1 + a_{1,2}u_2 + a_{1,N}u_N = r_1 \tag{4.1a}$$

$$a_{i,i-1}u_{i-1} + a_{i,i}u_i + a_{i,i+1}u_{i+1} = r_i \quad \forall\ i \in [2, N-1] \tag{4.1b}$$

$$a_{N,1}u_1 + a_{N,N-1}u_{N-1} + a_{N,N}u_N = r_N \tag{4.1c}$$

For an ADI pass on $\nabla^2 u = \rho$ in the r-direction in cylindrical coordinates, the coefficients in Eq. 4.1b would come from Eq. 3.9,

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) \doteq \frac{r_{1/2}(u_{1,0}-u)-r_{-1/2}(u-u_{-1,0})}{r\Delta r^2}. \tag{3.9}$$

The coefficients for this case are:

$$a_{i,i-1} = \frac{r_{i-1/2}}{r\Delta r^2},\ a_{i,i} = -\frac{2}{\Delta r^2}, \text{ and } a_{i,i+1} = \frac{r_{i+1/2}}{r\Delta r^2}. \tag{4.2}$$

Physical boundary conditions lead to constraints affecting Eqs. 4.1a and 4.1c. Periodic boundary conditions are equivalent to the assumption that $u_0 \equiv u_N$ and $u_{N+1} \equiv u_1$, combined with the use of Eq. 4.2 for the coefficients in Eqs. 4.1a and 4.1c.

Robbins boundary conditions are slightly different. When a linear combination of a field quantity and its normal derivative is specified at a boundary point, the quantity is said to be subject to a Robbins boundary condition. For a field quantity u, such a boundary condition is expressible by the mathematical statement $au + b\hat{n}\cdot\bar{\nabla}u = c$. Here $\hat{n}$ is the unit normal to the boundary. Special cases of the Robbins boundary condition are Dirichlet $(b=0)$ and Neumann $(a=0)$ conditions. A Cauchy condition at a boundary point of an open region is the same as independent Dirichlet and Neumann boundary conditions at the same point, and hence is actually two Robbins boundary conditions at the same point. Non-Cauchy Robbins boundary conditions arising from second-order central differencing around the 1st and Nth points of Eqs. 4.1 lead to Eqs. 4.1a and 4.1c in which $a_{1,N}$ and $a_{N,1}$ are zero.

Regardless of boundary conditions, the system of N linearly independent equations in N unknowns has a unique solution. With periodic boundary conditions and $N = 8$ the equations would have the following general structure in matrix form:

$$\begin{pmatrix} x & x & & & & & & x \\ x & x & x & & & & & \\ & x & x & x & & & & \\ & & x & x & x & & & \\ & & & x & x & x & & \\ & & & & x & x & x & \\ & & & & & x & x & x \\ x & & & & & & x & x \end{pmatrix} \begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix} = \begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix}$$

Figure 4.1. General structure of a system of eight equations of the form of Eqs. 4.1, allowing for periodic boundary conditions. Nonzero matrix and vector elements are lettered nondistinctly.

When the boundary conditions are Robbins for the same grid, the first and last unknowns are no longer coupled through the isolated coefficients in the lower left and upper right corners of the matrix; see Figure 4.2. If Dirichlet conditions are applied to the first and last points, the number of unknowns effectively decreases by two, and the off-diagonal coefficients in the first and last equations become zero, as in Figure 4.3.

$$\begin{pmatrix} x & x & & & & & & \\ x & x & x & & & & & \\ & x & x & x & & & & \\ & & x & x & x & & & \\ & & & x & x & x & & \\ & & & & x & x & x & \\ & & & & & x & x & x \\ & & & & & & x & x \end{pmatrix} \begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix} = \begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix}$$

Figure 4.2. Structure of a system of eight equations of the form of Eqs. 4.1 to which general Robbins boundary conditions are applied at the first and last points.

$$\begin{pmatrix} 1 & & & & & & & \\ x & x & x & & & & & \\ & x & x & x & & & & \\ & & x & x & x & & & \\ & & & x & x & x & & \\ & & & & x & x & x & \\ & & & & & x & x & x \\ & & & & & & & 1 \end{pmatrix} \begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix} = \begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix}$$

Figure 4.3. Structure of a system of eight equations of the form of Eqs. 4.1 to which Dirichlet boundary conditions are applied at the first and last points.

If the system arises from periodic boundary conditions, the N-equation system can be efficiently solved on most single-processor computers with $13N-14$ arithmetic operations (AOs). If the system arises from Robbins boundary conditions, the N-equation system can be serially solved with $8N-7$ AOs.

The number of AOs necessary for an efficient and scalable parallel solution is larger since the simplest serial solution technique is inherently recursive. In fact, parallel direct tridiagonal solvers useful for spatial domain decomposition (SDD) over N processors, where N>>2, require a bit more than twice the AOs of direct tridiagonal solvers on a single processor. Such parallel tridiagonal solvers are the backbone of parallel implementations of ADI using SDD and message passing on a multiple-instruction/multiple data (MIMD) computer.

To help introduce the parallel tridiagonal solvers for our SDD ADI method, Figure 4.4 depicts a general (periodic) tridiagonal system of equations in 16 unknowns evenly distributed over four processors. Such an arrangement of the equations naturally arises from the spatial domain decompositions mentioned in Chapter 3.

$$\begin{array}{c} \\ \textbf{P1} \\ \\ \\ \\ \textbf{P2} \\ \\ \\ \\ \textbf{P3} \\ \\ \\ \\ \textbf{P4} \\ \\ \\ \end{array}
\left(\begin{array}{cccc|cccc|cccc|cccc}
x & x &   &   &   &   &   &   &   &   &   &   &   &   &   & x \\
x & x & x &   &   &   &   &   &   &   &   &   &   &   &   &   \\
  & x & x & x &   &   &   &   &   &   &   &   &   &   &   &   \\
  &   & x & x & x &   &   &   &   &   &   &   &   &   &   &   \\ \hline
  &   &   & x & x & x &   &   &   &   &   &   &   &   &   &   \\
  &   &   &   & x & x & x &   &   &   &   &   &   &   &   &   \\
  &   &   &   &   & x & x & x &   &   &   &   &   &   &   &   \\
  &   &   &   &   &   & x & x & x &   &   &   &   &   &   &   \\ \hline
  &   &   &   &   &   &   & x & x & x &   &   &   &   &   &   \\
  &   &   &   &   &   &   &   & x & x & x &   &   &   &   &   \\
  &   &   &   &   &   &   &   &   & x & x & x &   &   &   &   \\
  &   &   &   &   &   &   &   &   &   & x & x & x &   &   &   \\ \hline
  &   &   &   &   &   &   &   &   &   &   & x & x & x &   &   \\
  &   &   &   &   &   &   &   &   &   &   &   & x & x & x &   \\
  &   &   &   &   &   &   &   &   &   &   &   &   & x & x & x \\
x &   &   &   &   &   &   &   &   &   &   &   &   &   & x & x
\end{array}\right)
\left(\begin{array}{c} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{array}\right)
=
\left(\begin{array}{c} x \\ x \\ x \\ \underline{x} \\ x \\ x \\ x \\ \underline{x} \\ x \\ x \\ x \\ \underline{x} \\ x \\ x \\ x \\ x \end{array}\right)$$

Figure 4.4. System of 16 equations of a periodic tridiagonal system evenly distributed over four processors. Horizontal solid lines seperate groups of 4 equations on each processor. Vertical dashed lines are visual aides for alignment of coefficients.

In reference to Figure 4.4, before the parallel SDD method proceeds, each processor P1 - P4 has knowledge of four contiguous equations of the distributed system. Terminal processors, namely P1 and P4, each have equations involving 6 unknowns for the periodic case, and 5 unknowns for the nonperiodic case. The interior processors P2 and P3 each have equations involving 6 unknowns in either case. Each of the susbsystems of four contiguous equations is coupled to neighboring subsystems. Generally, for a system distributed over P processors, the *terminal processors* are the 1th and *P*th processors, the *interior processors* are the remaining processors, and each processor initially has knowledge of a subsystem of equations that is a contiguous subset of the original tridiagonal set.

There are various parallel tridiagonal solvers in the literature that use such an SDD (same as partitioning and/or divide-and-conquer (DAC) approach. Most of them can be distilled into the three simple steps depicted in Figure 4.5.

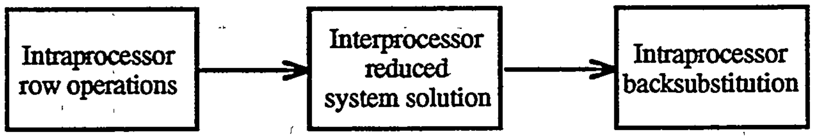
Intraprocessor row operations
Interprocessor reduced system solution
Intraprocessor backsubstitution

Figure 4.5. Three steps in a typical parallel tridiagonal solver based on spatial domain decomposition.

The parallel tridiagonal solvers outlined in this thesis also incorporate these steps, which are used to outline the description of the solvers.

The first step places the subsystem on each processor in a form that allows subsystem decoupling with a minimum amount of redundant work and interprocessor communication. It involves simple row operations between equations on the same processor, which is the origin of the term *intraprocessor row operations*. The operations are performed in a manner that is numerically stable, and proceed with complete parallelism as long as each processor has the same number of equations.

The second step achieves the interprocessor decoupling. It involves the solution of a small multiprocessor group of the equations, sometimes called the *reduced system*, from the first step. The unknowns in the small group of equations are the only unknowns that couple the processors. Determination of the coupling unknowns for a given processor is all that is necessary for the processor to complete direct solution over the associated subdomain. This reduced system solution requires interprocessor communication. Hence, the term *interprocessor reduced system solution* is used to describe this step.

Of the three steps, the third is the most straightforward. Again, it is a completely parallel step as long as the processors begin with the same number of equations. It can be implemented as a nonrecursive backsubstitution of values from the reduced system solution, requiring no interprocessor communication. This is the origin of the term *interprocessor backsubstitution*..

4.2 Intraprocessor row operations: placing tridiagonal submatrices in N-form

4.2.1 Simple formulas for the operations

For efficient parallel solution of the distributed tridiagonal system of Figure 4.4, it is necessary to keep the processors busy throughout the solution without performing too much work beyond the standard serial algorithm. Contrary to this, the standard algorithm for tridiagonal system solution does not yield scalable performance on a parallel computer when SDD is used. Only one processor in the domain decomposition can perform useful work at a given time. This is because the standard algorithm is completely recursive, and would proceed from one equation to the next through each subsystem, then to the next processor, when required, to the next subset of equations, and so on.

In order for multiple processors to work on the problem simultaneously and in a manner that scales, the standard algorithm must be modified to diminish the effect of the recursiveness. Recursiveness is reduced by using intraprocessor row operations which isolate the coupling of the processors to just two equations on each processor. Processors perform these row operations simultaneously. The coupling of the processors then remains in a much smaller subset of the equations that requires recursive solution in parallel. Such an approach is adopted even though it increases the total number of arithmetic operations by a factor of two over the serial algorithm. The exact approach tested in ADI manipulates the subsystems until they are in *N-form*: The term *N-form* comes from the image that the nonzero coefficients form when they are arranged in the standard way of Figure 4.6.

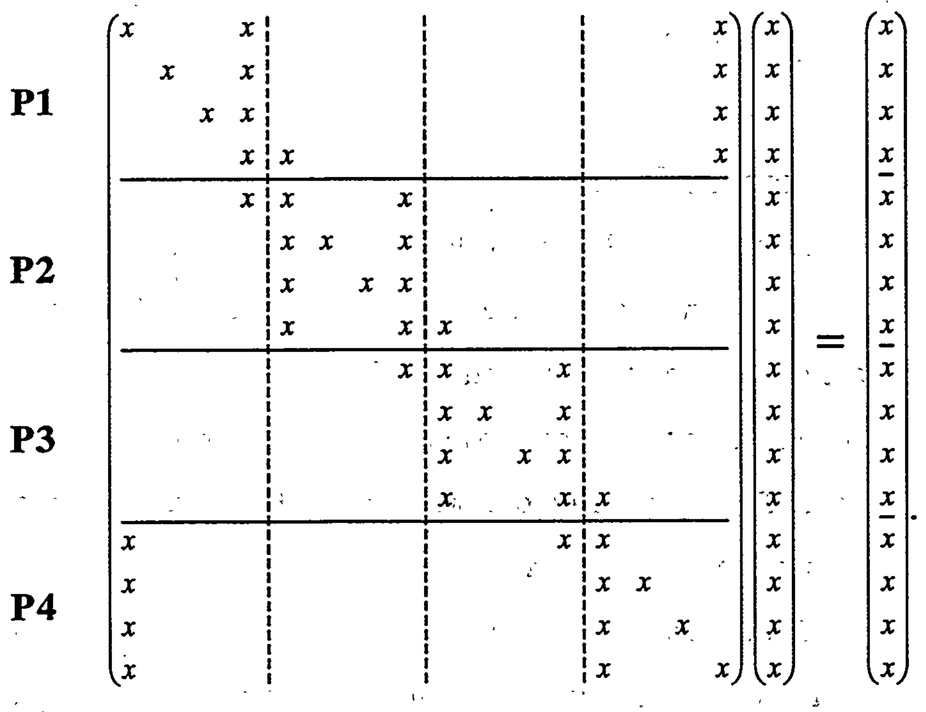

Figure 4.6. System of equations of Figure 4.4 after intraprocessor row operations have been used to place subgroups of equations in N-form. Horizontal solid lines seperate groups of 4 equations on each processor. Vertical dashed lines are visual aides for alignment of coefficients.

To derive the arithmetic of the N-form reduction, it is convenient to start with a representative subsystem of equations on one processor, and to evaluate the elements of the subsystem as the equations are manipulated. The original subsystem for an interior processor with six equations is as follows:

$$\begin{array}{|cccccccc|}\hline c_1 & -a_1 & b_1 & & & & & \\ & c_2 & -a_2 & b_2 & & & & \\ & & c_3 & -a_3 & b_3 & & & \\ & & & c_4 & -a_4 & b_4 & & \\ & & & & c_5 & -a_5 & b_5 & \\ & & & & & c_6 & -a_6 & b_6 \\ \hline \end{array} \begin{pmatrix} \vdots \\ x \\ x \\ x \\ x \\ \vdots \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{pmatrix}$$

Here the coefficients have been distinguished in order to give details of the reduction process. After a forward elimination in the subsystem, the coefficient matrix has the following form:

$$\begin{array}{|cccccccc|}\hline c_1' & 1 & b_1' & & & & & \\ & c_2' & 1 & b_2' & & & & \\ & c_3' & & 1 & b_3' & & & \\ & c_4' & & & 1 & b_4' & & \\ & c_5' & & & & 1 & b_5' & \\ & c_6' & & & & & 1 & b_6' \\ \hline \end{array} \begin{pmatrix} \vdots \\ x \\ x \\ x \\ x \\ \vdots \end{pmatrix} = \begin{pmatrix} r_1' \\ r_2' \\ r_3' \\ r_4' \\ r_5' \\ r_6' \end{pmatrix}$$

The c_i', b_i' and r_i' are given simply by:

$$c_i' = -\frac{c_i}{a_i}, \quad b_i' = -\frac{b_i}{a_i}, \quad r_i' = -\frac{r_i}{a_i}, \qquad i = 1,2; \tag{4.3a,b,c}$$

$$c_i' = \frac{c_i c_{i-1}'}{a_i + c_i b_{i-1}'}, \quad b_i' = \frac{-b_i}{a_i + c_i b_{i-1}'}, \quad r_i' = -\frac{r_i - c_i r_{i-1}'}{a_i + c_i b_{i-1}'}, \quad i = 3, \ldots, N. \tag{4.3d,e,f}$$

No pivoting is necessary in this forward elimination step when the system is diagonally dominant. After another elimination step similar to the forward elimination, but in the backward direction, the subsystem is in N-form:

$$\begin{array}{cccccccc} c_1'' & 1 & & & & & b_1'' & \\ & c_2'' & 1 & & & & b_2'' & \\ & c_3'' & & 1 & & & b_3'' & \\ & c_4'' & & & 1 & & b_4'' & \\ & c_5'' & & & & 1 & b_5'' & \\ & c_6'' & & & & & 1 & b_6'' \end{array} \begin{pmatrix} \vdots \\ x \\ x \\ x \\ x \\ \vdots \end{pmatrix} = \begin{pmatrix} r_1'' \\ r_2'' \\ r_3'' \\ r_4'' \\ r_5'' \\ r_6'' \end{pmatrix}$$

Note that this coefficient matrix "looks like" the letter N. Elements are formulated as follows:

$$c_i'' = c_i', \quad b_i'' = b_i', \quad r_i'' = r_i', \quad i = N-1, N; \tag{4.4a,b,c}$$

$$c_i'' = c_i' - b_i' c_{i+1}'', \quad b_i'' = -b_i' b_{i+1}'', \quad r_i'' = r_i' - b_i' r_{i+1}'', \quad i = N-2, \ldots, 2; \tag{4.4d,e,f}$$

$$c_1'' = \frac{c_1'}{1 - b_1' c_2''}, \quad b_1'' = \frac{-b_1' b_2''}{1 - b_1' c_2''}, \quad r_1'' = \frac{r_1' - b_1' r_2''}{1 - b_1' c_2''}. \tag{4.4g,h,i}$$

The diagonally dominant case does not require pivoting for the backward elimination either. This is beneficial, because pivoting would decrease the execution speed of the algorithm.

After the N-form reduction is complete, the solution to a subsystem is readily found after the 1th and Nth unknowns have been determined (N equations per processor). In this sense the N-form reduction is equivalent to solution of a Dirichlet problem over the subdomain, with the boundary values, namely the 1th and Nth unknowns, yet to be determined. Indeed, if the original tridiagonal system of equations is from a one-dimensional discretization of Poisson's equation, then this reduction to N-form is equivalent to solution of an inhomogeneous Dirichlet problem over each subdomain, with the 1th and Nth unknowns playing the role of the Dirichlet boundary values. For this reason, it is natural to use the term *processor boundary unknown* to describe the 1th and Nth unknowns of each subdomain.

For a terminal processor, the forward elimination of the N-form reduction can proceed in the same manner. The subsystem begins with a coefficient matrix like the following six-equation example:

$$\left[\begin{array}{cc:c:cc:cccccc}
 & \cdots & \cdots & \cdots & c_1 & -a_1 & b_1 & & & & \\
 & \cdots & \cdots & \cdots & & c_2 & -a_2 & b_2 & & & \\
 & \cdots & \cdots & \cdots & & & c_3 & -a_3 & b_3 & & \\
 & \cdots & \cdots & \cdots & & & & c_4 & -a_4 & b_4 & \\
 & \cdots & \cdots & \cdots & & & & & c_5 & -a_5 & b_5 \\
b_6 & \cdots & \cdots & \cdots & & & & & & c_6 & -a_6
\end{array}\right]
\begin{pmatrix} \vdots \\ x \\ x \\ x \\ x \\ x \end{pmatrix}
=
\begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{pmatrix}$$

Forward elimination gives the following coefficient matrix:

$$\left[\begin{array}{cc:c:cc:cccccc}
 & \cdots & \cdots & \cdots & c_1' & 1 & b_1' & & & & \\
 & \cdots & \cdots & \cdots & & c_2' & 1 & b_2' & & & \\
 & \cdots & \cdots & \cdots & & c_3' & & 1 & b_3' & & \\
 & \cdots & \cdots & \cdots & & c_4' & & & 1 & b_4' & \\
 & \cdots & \cdots & \cdots & & c_5' & & & & 1 & b_5' \\
b_6' & \cdots & \cdots & \cdots & & c_6' & & & & & 1
\end{array}\right]
\begin{pmatrix} \vdots \\ x \\ x \\ x \\ x \\ x \end{pmatrix}
=
\begin{pmatrix} r_1' \\ r_2' \\ r_3' \\ r_4' \\ r_5' \\ r_6' \end{pmatrix}$$

The c_i' , b_i' and r_i' are the same as before. For the nonperiodic case, $b_N = b_N' = 0$. On backward elimination the reduction could be just like the interior case:

$$\begin{array}{cc:c:cc:cccccc}
\hline
 & \cdots & \cdots & \cdots & c_1'' & 1 & & & & & b_1'' \\
 & \cdots & \cdots & \cdots & & c_2'' & 1 & & & & b_2'' \\
 & \cdots & \cdots & \cdots & & c_3'' & & 1 & & & b_3'' \\
 & \cdots & \cdots & \cdots & & c_4'' & & & 1 & & b_4'' \\
 & \cdots & \cdots & \cdots & & c_5'' & & & & 1 & b_5'' \\
b_6'' & \cdots & \cdots & \cdots & & c_6'' & & & & & 1
\end{array}
\begin{pmatrix} \vdots \\ x \\ x \\ x \\ x \\ x \end{pmatrix} = \begin{pmatrix} r_1'' \\ r_2'' \\ r_3'' \\ r_4'' \\ r_5'' \\ r_6'' \end{pmatrix}$$

Then the c_i'' , b_i'' and r_i'' would also be the same as for an interior processor, and the processors would be coupled by 2P boundary unknowns. But in the nonperiodic case this would not take advantage of the equation $b_N = b_N' = 0$, which can be used to reduce the number of coupling unknowns from 2P to 2P-2. Instead, the following backward elimination can be used to make the nonperiodic implementation more efficient:

$$\begin{array}{cc:c:cc:cccccc}
\hline
b_1'' & \cdots & \cdots & \cdots & c_1'' & 1 & & & & & \\
b_2'' & \cdots & \cdots & \cdots & & c_2'' & 1 & & & & \\
b_3'' & \cdots & \cdots & \cdots & & c_3'' & & 1 & & & \\
b_4'' & \cdots & \cdots & \cdots & & c_4'' & & & 1 & & \\
b_5'' & \cdots & \cdots & \cdots & & c_5'' & & & & 1 & \\
b_6'' & \cdots & \cdots & \cdots & & c_6'' & & & & & 1
\end{array}
\begin{pmatrix} \vdots \\ x \\ x \\ x \\ x \\ x \end{pmatrix} = \begin{pmatrix} r_1'' \\ r_2'' \\ r_3'' \\ r_4'' \\ r_5'' \\ r_6'' \end{pmatrix}$$

The equation $b_N = b_N' = 0$ would then obviate evaluation of any b_i''. These coefficients are given by:

$$c_N'' = c_N', \qquad b_N'' = b_N', \qquad r_N'' = r_N'; \qquad \text{(4.5a,b,c)}$$

$$c_i'' = c_i' - b_i' c_{i+1}'', \quad b_i'' = -b_i' b_{i+1}'', \quad r_i'' = r_i' - b_i' r_{i+1}'', \quad i = N-1, \ldots, 2; \qquad \text{(4.5d,e,f)}$$

$$c_1'' = \frac{c_1'}{1 - b_1' c_2''}, \quad b_1'' = \frac{-b_1' b_2''}{1 - b_1' c_2''}, \quad r_1'' = \frac{r_1' - b_1' r_2''}{1 - b_1' c_2''}. \qquad \text{(4.5g,h,i)}$$

Reduction to N-form can be achieved with $13N$ arithmetic operations for interior processors. The terminal processors perform about $13N$ arithmetic operations when periodic external boundary conditions are applied, and about $11N$ operations when Robbins external boundary conditions are applied.

Note that the 1th, 4th, 5th, 8th, 9th, 12th, 13th, and 16th equations depicted in Figure 4.6 form a system of equations, depicted in Figure 4.7, that determine coupling unknowns independently of the other equations:

$$\left(\begin{array}{cc:cc:cc:cc}
1 & x & & & & & & x \\
 & 1 & x & & & & & x \\
\hline
 & x & 1 & x & & & & \\
 & & x & 1 & x & & & \\
\hline
 & & & x & 1 & x & & \\
 & & & & x & 1 & x & \\
\hline
x & & & & & x & 1 & \\
x & & & & & & x & 1
\end{array}\right)
\begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix}
=
\begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix}.$$

Figure 4.7 The reduced matrix from the N-form reduction of Figure 4.6. Horizontal solid lines seperate pairs of coupling equations on each processor. Vertical dashed lines are visual aides for alignment of coefficients.

These equations form the reduced system corresponding to Figure 4.6. Details of the solution of the reduced system are covered in Section 4.3.

4.2.2. Useful N-form matrix element relationships

The primary goal of this section is to derive an expression that determines whether coefficients c_N'' and b_1'' are sufficiently small for solution of the distributed tridiagonal system by an approximation technique that reduces interprocessor communication. To see that small coefficients c_N'' and b_1'' can lead to a useful aproximation, observe the following equations that couple the Nth and $(N+1)$th processors:

$$c''_{N,1}x_1 + x_N + b''_{N,N+1}x_{N+1} = r''_N, \tag{4.6a}$$

$$c''_{N+1,N}x_N + x_{N+1} + b''_{N+1,2N}x_{2N} = r''_{N+1}. \tag{4.6b}$$

Here, the numbering of unknowns is such that first equation on the Nth processor has the central coefficient unknown that is numbered the first. Note the correspondencies: c''_N of the Nth processor becomes $c''_{N,1}$; b''_1 on the $(N+1)$th processor becomes $b''_{N+1,2N}$. The equations can be rewritten to solve for the central unknowns:

$$x_N = r''_N - c''_{N,1}x_1 - b''_{N,N+1}x_{N+1}, \tag{4.7a}$$

$$x_{N+1} = r''_{N+1} - c''_{N+1,N}x_N - b''_{N+1,2N}x_{2N}. \tag{4.7b}$$

When the original tridiagonal system is diagonally dominant, the magnitudes of the coefficients $c''_{N,1}$ and $b''_{N+1,2N}$ tend to approach zero with increasing N. On the other hand, the magnitude of the coefficients $b''_{N,N+1}$ and $c''_{N+1,N}$ approach unity with increasing N. As long as the solution has values x_1, x_N, x_{N+1}, and x_{2N} which are not widely varying in magnitude, then smaller coefficients $c''_{N,1}$ and $b''_{N+1,2N}$ make the following approximations become more accurate:

$$x_N \doteq r''_N - b''_{N,N+1}x_{N+1}, \tag{4.8a}$$

$$x_{N+1} \doteq r''_{N+1} - c''_{N+1,N}x_N. \tag{4.8b}$$

Quite often the solution is relatively smooth so that these approximations are good for $N > 10$ or so, which is small enough for practical problems. If these approximations are used for parallel solution of the tridiagonal solution, then nearest neighbor processors need to communicate only one message to each other in order to complete the solution.

This is less communication than necessary when accounting for the coefficients $c''_{N,1}$ and $b''_{N+1,2N}$.

A secondary goal of this section was to find alternative expressions of the N-form reduction in an attempt to find a parallel algorithm requiring fewer FLOPs. Unfortunately, fewer FLOPs were not achieved in this manner. All of the expressions that were discovered are developed by constant allusion to a tridiagonal submatrix with six rows. Also, the assumption of a symmetric diagonally dominant tridiagonal system is made. The number of rows is just sufficient to find patterns in the formulas for solution. In this section, only the interior processors are considered. There should be a straightforward but tedious extension to terminal processors in the general periodic case.

Original subset for an interior processor:

$$\begin{array}{cccccccc}
\hline
c_1 & -a_1 & c_2 & & & & & \\
 & c_2 & -a_2 & c_3 & & & & \\
 & & c_3 & -a_3 & c_4 & & & \\
 & & & c_4 & -a_4 & c_5 & & \\
 & & & & c_5 & -a_5 & c_6 & \\
 & & & & & c_6 & -a_6 & c_7 \\
\hline
\end{array}$$

Subset after forward elimination:

$$\begin{array}{cccccccc}
\hline
c''_1 & 1 & c'_2 & & & & & \\
 & c''_2 & 1 & c'_3 & & & & \\
 & c''_3 & & 1 & c'_4 & & & \\
 & c''_4 & & & 1 & c'_5 & & \\
 & c''_5 & & & & 1 & c'_6 & \\
 & c''_6 & & & & & 1 & c'_7 \\
\hline
\end{array}$$

The $c_i^{'}$ and $c_i^{''}$ can be expressed as (see Appendix Ch4.A3):

$$c_1^{''} = -\frac{c_1}{a_1}, \qquad c_2^{'} = -\frac{c_2}{a_1}, \qquad r_1^{'} = -\frac{r_1}{a_1}, \tag{4.9a}$$

$$c_i^{''} = \frac{-1}{U_{i-1}} \prod_{j=2}^{i} c_j \quad \forall\ i \in [2, N], \tag{4.9b}$$

$$c_i^{'} = -\frac{c_i U_{i-3}}{U_{i-2}} \quad \forall\ i \in [3, N+1], \tag{4.9c}$$

$$r_i^{'} = -\frac{R_{i-1}}{U_{i-1}} \quad \forall\ i \in [2, N]. \tag{4.9d}$$

The quantities U_i and R_i are convieniently defined in the following way

$$U_{-1} = 0,\ U_0 = 1,\ U_i = a_{i+1} U_{i-1} - c_{i+1}^2 U_{i-2},\quad i = 1, \ldots, N-1, \tag{4.10}$$

$$R_0 = 0,\ R_i = r_{i+1} U_{i-1} + c_{i+1} R_{i-1},\quad i = 1, \ldots, N-1, \tag{4.11}$$

Here is the subset after reduction to "N-form":

$$\begin{array}{cccccccc}
\hline
d_1^{''} & 1 & & & & & d_2^{'} & \\
 & d_2^{''} & 1 & & & & d_3^{'} & \\
 & d_3^{''} & & 1 & & & d_4^{'} & \\
 & d_4^{''} & & & 1 & & d_5^{'} & \\
 & d_5^{''} & & & & 1 & d_6^{'} & \\
 & d_6^{''} & & & & & 1 & d_7^{'} \\
\hline
\end{array}$$

The $d_i^{'}$, $d_i^{''}$ and right-hand side can be expressed as follows (see Appendix 4.A3):

$$d_1^{''} = \frac{-c_1 U_{N-2}}{V_2} = \frac{-c_1 V_3}{V_2}, \tag{4.12a}$$

$$d_i^{''} = \frac{-V_{i+2}}{U_{N-2}} \prod_{j=2}^{i} c_j = \frac{-V_{i+2}}{V_3} \prod_{j=2}^{i} c_j \quad \forall\ i \in [2, N-1], \tag{4.12b}$$

$$d''_N = c''_N = \frac{-1}{U_{N-1}}\prod_{j=2}^{N} c_j, \quad d'_2 = \frac{-1}{V_2}\prod_{j=2}^{N} c_j, \tag{4.12c,d}$$

$$d'_i = -\frac{U_{i-3}}{U_{N-2}}\prod_{j=i}^{N} c_j = -\frac{U_{i-3}}{V_3}\prod_{j=i}^{N} c_j \;\; \forall\; i \in [3,N], \tag{4.12e}$$

$$d'_{N+1} = c'_{N+1} = \frac{-c_{N+1}U_{N-2}}{U_{N-1}} = \frac{-c_{N+1}V_3}{U_{N-1}}, \tag{4.12f}$$

$$r''_i = -\frac{V_{i+2}R_{i-1} + c_{i+1}S_{i+2}U_{i-2}}{U_{N-2}}, \quad i = N, \ldots, 3 \tag{4.13a}$$

$$r''_i = -\frac{S_{i+1}}{V_{i+1}}, \quad i = 1, 2. \tag{4.13b}$$

The U_i are as before, and the V_i, R_i, and S_i obey the following:

$$V_{N+2} = 0, \; V_{N+1} = 1, \; V_i = a_{i-1}V_{i+1} - c_i^2 V_{i+2}, \quad i = N, \ldots, 2, \tag{4.14}$$

$$S_{N+1} = 0, \; S_i = r_{i-1}V_{i+1} + c_i S_{i+1}, \quad i = N, \ldots, 2, \tag{4.15}$$

Theformula for the V_i is actually more like the formula for the U_i than it looks. If the c_i are labeled $c_{i-1/2}$, the formulas prove to be "mirror image" relations:

$$U_i = a_{i+1}U_{i-1} - c_{i+1/2}^2 U_{i-2}, \qquad V_i = a_{i-1}V_{i+1} - c_{i-1/2}^2 V_{i+2}. \tag{4.16}$$

Note that U_{N-2} is equal to V_3, and that this was used in the above equations for the d'_i and the d''_i.

Laplacian case

Consider the case in which $a_i = c_i + c_{i+1} + \mu_i + \omega$, as would arise from an ADI method on $(\vec{\nabla} \cdot \varepsilon \vec{\nabla} - \mu)\phi = \rho$. For ADI, $\omega \equiv 2/\Delta t > 0$, and for the Laplacian, $c_i = 1$ and $\mu_i = 0$. Then for ADI in the Laplacian case it is easy to show that the coefficients d'_2 and d''_N are exponentially decreasing functions of the number of equations N in the subset. Furthermore, with increasing ω, d'_2 and d''_N decrease as inverse integral powers of ω.

To arrive at these conclusions, note that the transformation $x = 1 + \omega/2 = 1 + 1/\Delta t$ makes the U_i Chebyshev Type II polynomials in x [Arfken 1985]:

$$U_{-1}(x) = 0,\ U_0(x) = 1,\ U_{i+1}(x) = 2xU_i(x) - U_{i-1}(x),\ i = 1, \ldots, N-1. \tag{4.17}$$

The following formulas for these Chebyshev U_i are convenient:

$$U_i(x) = \frac{\sin((i+1)\arccos x)}{\sin(\arccos x)} = \frac{\left(x+\sqrt{x^2-1}\right)^{2i+2} - 1}{\left(x+\sqrt{x^2-1}\right)^{i}\left[\left(x+\sqrt{x^2-1}\right)^{2} - 1\right]}. \tag{4.18}$$

The trigonometric expression is most useful for $|x| < 1$. It implies that the i roots of each U_i are real and within the interval $|x| < 1$ Also, it implies $U_i(1) = i + 1$ and $U_i(-x) = (-1)^i U_i(x)$. These points imply that for the interval $x > 1$ each polynomial U_i is dominated by $(2x)^i$ by two orders in x, and hence increases as such for fixed i and increasing x.

The other expression is most useful for $x > 1$, and can be used to quantify behavior of the U_i with increasing i and fixed x. To discover the behavior, one only needs to take a derivative with respect to i, assuming for the moment that i is continuous:

$$\begin{aligned}\frac{\partial U_i}{\partial i} &= \left[\left(x+\sqrt{x^2-1}\right)^2 - 1\right]^{-1} \frac{\partial}{\partial i}\left[\left(x+\sqrt{x^2-1}\right)^{i+2} - \left(x+\sqrt{x^2-1}\right)^{-i}\right] \\ &= \left[\left(x+\sqrt{x^2-1}\right)^2 - 1\right]^{-1} \ln\left(x+\sqrt{x^2-1}\right)\left[\left(x+\sqrt{x^2-1}\right)^{i+2} + \left(x+\sqrt{x^2-1}\right)^{-i}\right]\end{aligned}. \tag{4.19}$$

For a constant $x > 1$, $\partial_i U_i$ is bounded below by an exponentially increasing function of the index i. This implies that the U_i are at least exponentially increasing functions of i on that interval.

When larger subdomains are used for tridiagonal system solution, Chebyshev polynomials of increasing index are needed to evaluate and d_N''. In particular,

$$d_N'' = \frac{-1}{U_{N-1}} \prod_{j=2}^{N} c_j . \tag{4.20}$$

Then for the Laplacian case, d_N'' is an exponentially decaying function of the number of equations N in the subdomain. By symmetry and uniqueness of the N-form reduction process, the same observation can be made for d_2'. As explained above the smallness of these coefficients can be used to implement a faster parallel tridiagonal solver to be described in detail in the next section.

4.3 Interprocessor reduced matrix solution

Once the reduced system of Figure 4.7 is obtained, there are two general approaches to solving it. One class of methods uses an all-to-all broadcast to assemble on each processor a copy of the complete reduced system to be solved by the standard tridiagonal algorithm. For lack of better terminology, this first class of methods will be referred to as *all-to-all broadcast methods*. In this first class are the methods of [Kowalik and Kumar 1985] and [Mattor *et al.* 1995], summarized in Appendices 4.A1 and 4.A2. The second class of methods solves the same subset of equations without first broadcasting them. In this class is the direct *two-way skip-decoupling method* tested in parallel DADI for later use in the pinch simulations.

4.3.1 Direct solution by two-way skip decoupling

Parallel direct solution of the reduced system by the two-way skip decoupling method yields the full solution to machine precision. This part of the solution is the only part that requires communication between the processors in the domain decomposition. After reduction to N-form the processors are coupled by the following reduced system (an example with $P = 4$) for the general periodic case:

$$\left(\begin{array}{cccccccc}
1 & b_1 & & & & & & c_1 \\
 & 1 & b_2 & & & & & c_2 \\
\hline
 & c_3 & 1 & b_3 & & & & \\
 & & c_4 & 1 & b_4 & & & \\
\hline
 & & & c_5 & 1 & b_5 & & \\
 & & & & c_6 & 1 & b_6 & \\
\hline
b_7 & & & & & c_7 & 1 & \\
b_8 & & & & & & c_8 & 1
\end{array}\right)
\begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix}
=
\left(\begin{array}{c} r_1 \\ r_2 \\ \hline r_3 \\ r_4 \\ \hline r_5 \\ r_6 \\ \hline r_7 \\ r_8 \end{array}\right)$$

Note that if the system is nonperiodic, the coefficients c_1, c_2, b_7, and b_8 are zero, and the reduced system can be simplified to:

$$\left(\begin{array}{cccccc} 1 & b_2 & & & & \\ \hline c_3 & 1 & b_3 & & & \\ & c_4 & 1 & b_4 & & \\ \hline & & c_5 & 1 & b_5 & \\ & & & c_6 & 1 & b_6 \\ \hline & & & & c_7 & 1 \end{array}\right)\begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \end{pmatrix} = \left(\begin{array}{c} r_2 \\ \hline r_3 \\ r_4 \\ \hline r_5 \\ r_6 \\ \hline r_7 \end{array}\right)$$

Solution of the reduced system in the general case can begin with a forward elimination that yields the following new system:

$$\left(\begin{array}{cccccccc} 1 & b_1' & & & & & & c_1' \\ & 1 & b_2' & & & & & c_2' \\ \hline & & 1 & b_3' & & & & c_3' \\ & & & 1 & b_4' & & & c_4' \\ \hline & & & & 1 & b_5' & & c_5' \\ & & & & & 1 & b_6' & c_6' \\ \hline c_7' & & & & & & 1 & b_7' \\ c_8' & & & & & & & 1 \end{array}\right)\begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix} = \left(\begin{array}{c} r_1' \\ r_2' \\ \hline r_3' \\ r_4' \\ \hline r_5' \\ r_6' \\ \hline r_7' \\ r_8' \end{array}\right)$$

The formulas for the coefficients and the right-hand side after such a forward elimination on P processors are as follows:

$$c_i' = c_i, \quad b_i' = b_i, \quad r_i' = r_i, \quad i = 1,2, \qquad \text{(4.21a,b,c)}$$

$$c_i' = \frac{-c_i c_{i-1}'}{1 - c_i b_{i-1}'}, \quad b_i' = \frac{b_i}{1 - c_i b_{i-1}'}, \quad r_i' = \frac{r_i - c_i r_{i-1}'}{1 - c_i b_{i-1}'}, \quad i = 3, \ldots, 2P-2, \qquad \text{(4.21d,e,f)}$$

$$c_{2P\text{-}1}' = \frac{b_{2P\text{-}1}}{1 - c_{2P\text{-}1} b_{2P\text{-}2}'}, \; b_{2P\text{-}1}' = \frac{b_{2P\text{-}1} - c_{2P\text{-}1} c_{2P\text{-}2}'}{1 - c_{2P\text{-}1} b_{2P\text{-}2}'}, \; r_{2P\text{-}1}' = \frac{r_{2P\text{-}1} - c_{2P\text{-}1} r_{2P\text{-}2}'}{1 - c_{2P\text{-}1} b_{2P\text{-}2}'}, \qquad \text{(4.21g,h,i)}$$

$$c_{2P}' = \frac{c_{2P} - c_{2P} c_{2P-1}'}{1 - c_{2P} b_{2P-1}'}, \; r_{2P}' = \frac{r_{2P} - c_{2P} r_{2P-1}'}{1 - c_{2P} d_{2P-1}'}. \qquad \text{(4.21j,k)}$$

The system can be further reduced by a sequence of backward eliminations, placing it in the following form:

$$\left(\begin{array}{cccccccc}
1 & & & & & & & b_1'' \\
c_2'' & 1 & & & & & & b_2'' \\ \hline
c_3'' & & 1 & & & & & b_3'' \\
c_4'' & & & 1 & & & & b_4'' \\ \hline
c_5'' & & & & 1 & & & b_5'' \\
c_6'' & & & & & 1 & & b_6'' \\ \hline
c_7'' & & & & & & 1 & b_7'' \\
c_8'' & & & & & & & 1
\end{array}\right)
\begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix}
=
\left(\begin{array}{c} r_1'' \\ r_2'' \\ \hline r_3'' \\ r_4'' \\ \hline r_5'' \\ r_6'' \\ \hline r_7'' \\ r_8'' \end{array}\right).$$

The formulas for this general backward elimination are then:

$$c_i'' = c_i', \quad r_i'' = r_i', \quad i = 2P-1, 2P, \quad b_{2P-1}'' = b_{2P-1}', \tag{4.22a,b,c}$$

$$c_i'' = -b_i' c_{i+1}'', \quad b_i'' = c_i' - b_i' b_{i+1}'', \quad r_i'' = r_i' - b_i' r_{i+1}'', \quad i = 2P-2, \ldots, 2, \tag{4.22d,e,f}$$

$$b_1'' = \frac{c_1' - b_1' b_2''}{1 - b_1' c_2''}, \qquad r_1'' = \frac{r_1' - b_1' r_2''}{1 - b_1' c_2''}. \tag{4.22g,h}$$

For the nonperiodic case, the reduced system would be completely solved, because all of the b_i'' and c_i'' would be zero.

For the periodic case, the 1th and $2P$th equations can be used to determine the 1th and $2P$th unknowns. Then a nonrecursive cycle of forward and backward elimination finishes solution, leaving the identity matrix on the left-hand side ($P = 4$):

$$\left(\begin{array}{cccccccc}
1 & & & & & & & \\
 & 1 & & & & & & \\ \hline
 & & 1 & & & & & \\
 & & & 1 & & & & \\ \hline
 & & & & 1 & & & \\
 & & & & & 1 & & \\ \hline
 & & & & & & 1 & \\
 & & & & & & & 1
\end{array}\right)
\begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix}
=
\left(\begin{array}{c} r_1''' \\ r_2''' \\ \hline r_3''' \\ r_4''' \\ \hline r_5''' \\ r_6''' \\ \hline r_7''' \\ r_8''' \end{array}\right).$$

In this case the following formulas are used to obtain the solution:

$$r_1''' = \frac{r_1'' - b_1'' r_{2P}''}{1 - b_1'' c_{2P}''}, \qquad r_{2P}''' = \frac{r_{2P}'' - c_{2P}'' r_1''}{1 - c_{2P}'' b_1''}, \tag{4.23a,b}$$

$$r_i''' = r_i'' - c_i'' r_1''' - b_i'' r_{2P}''' \quad \forall\ i \in [2, 2P-1]. \tag{4.23c}$$

The most significant differences between the all-to-all broadcast methods and the two-way skip decoupling method are in the interprocessor communication. Processors must communicate with messages of different lengths, and they must communicte in different orders. An all-to-all broadcast for the first class of methods requires at least $\log_2 P$ communication volleys between the P processors over which the system is distributed. Here a volley is defined as a group of send-receive pairs that can proceed simultaneously between the processors with no more than one sent message per processor, and without intervening calculations. A send-receive pair is the completion of a message-sending subroutine on a sending processor and a message-receiving subroutine on the receiving processor. In any given volley, more than one pair of processors can be executing a send-receive pair. Also, two isolated processors can be executing two oppositely directed send-receive pairs. Interprocessor communication that achieves an all-to-all broadcast between $P = 8$ processors, using the minimum number of volleys (three), is depicted in Figure 4.9.

There is an alternative to the above all-to-all broadcast method. The reduced matrix can be left distributed over P processors, and communication can proceed as necessary to perform the eliminations of Eqs. 4.21 and 4.22. These equations would then require $2P-2$ communication volleys if the forward and backward eliminations are not done simultaneously, but only $P-1$ volleys if they are done simultaneously in a folded manner. Each of the $P-1$ volleys requires two SR pairs, one between one pair of processors, and one other between another pair of processors. This *two-way skip-decoupling method* for the nonperiodic case is the one studied in this paper. For the periodic case, once the 1th and $2P$th unknowns are determined, Eqs. 4.23c and 4.23d are best implemented with an all-to-all broadcast. Interprocessor communication schemes for the *non*periodic case are illustrated in Figures 4.8 and 4.9.

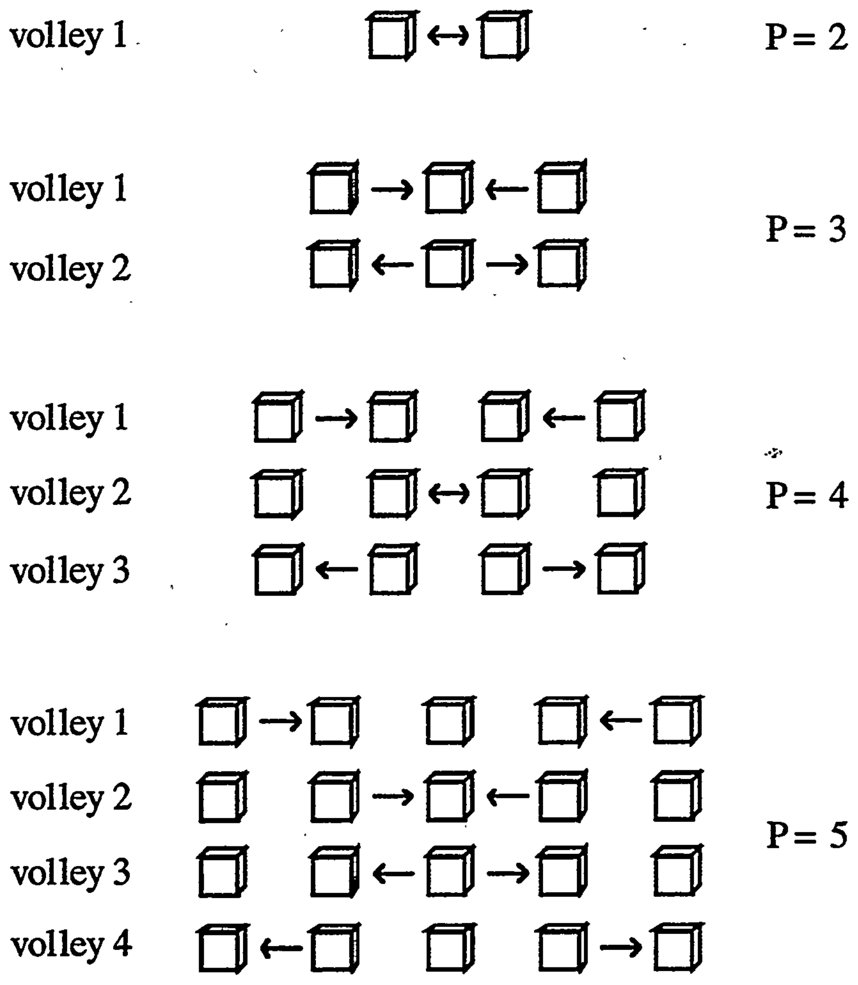

Figure 4.8. Interprocessor communication volleys necessary to solve by the two-way folded skip-decoupling method a nonperiodic tridiagonal system of equations distributed over P processors. Each arrow represents the SR pair of a transferred message. SR pairs in a given volley can be executed simultaneously on most architectures. Values of P from two to five are shown. Extension to $P > 5$ follows the same pattern. For P processors, $P-1$ volleys are always necessary.

4.3.2 Zeroth order approximate solution using only nearest-neighbor communication

Solution 1

As a four-processor example, interprocessor row operations can be performed on the reduced system until the following linear system appears (nonperiodic case):

$$\left(\begin{array}{cccccc} 1 & & b_1' & & & \\ \hline & 1 & b_2' & & & \\ & c_3' & 1 & & b_3' & \\ \hline & c_4' & & 1 & b_4' & \\ & & & c_5' & 1 & \\ \hline & & & c_6' & & 1 \end{array}\right)\begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \end{pmatrix} = \left(\begin{array}{c} r_1' \\ \hline r_2' \\ r_3' \\ \hline r_4' \\ r_5' \\ \hline r_6' \end{array}\right).$$

These coefficients and right-hand side, for P processors, are given by the following formulas:

$$c_{2i-1}' = \frac{c_{2i-1}}{1-b_{2i-1}c_{2i}}, \quad c_{2i}' = \frac{-c_{2i}c_{2i-1}}{1-b_{2i-1}c_{2i}} \quad \forall\ i \in [2, P-1], \tag{4.24a,b}$$

$$b_{2i-1}' = \frac{-b_{2i-1}b_{2i}}{1-b_{2i-1}c_{2i}}, \quad b_{2i}' = \frac{b_{2i}}{1-b_{2i-1}c_{2i}} \quad \forall\ i \in [1, P-2], \tag{4.24c,d}$$

$$r_{2i-1}' = \frac{r_{2i-1} - b_{2i-1}r_{2i}}{1-b_{2i-1}c_{2i}}, \quad r_{2i}' = \frac{r_{2i} - c_{2i}r_{2i-1}}{1-b_{2i-1}c_{2i}} \quad \forall\ i \in [1, P-1]. \tag{4.24e,f}$$

If the b_{2i} and c_{2i-1} are small quantities from the N-form reduction, then the b_i' and c_i' are first order in these small quantities. A zeroth order accurate method is obtained by neglecting these b_i' and c_i', and taking the r_i' as the solution to the reduced system. For this zeroth order method to be implemented with nearest neighbor communication, two communication volleys of $P-1$ send-receive pairs each are necessary. Compare this with the $P-1$ volleys of two send-receive pairs each necessary for the two-way skip decoupling method.

Solution 2

There is another zeroth order accurate method that requires no additional communication over the zeroth order method just described. This method begins with intraprocessor row operations that lead to the following linear system:

$$\left(\begin{array}{cccccc} 1 & b_1' & & & & \\ \hline c_2' & 1 & & b_2' & & \\ c_3' & & 1 & b_3' & & \\ \hline & & c_4' & 1 & & b_4' \\ & & c_5' & & 1 & b_5' \\ \hline & & & & c_6' & 1 \end{array}\right)\begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \end{pmatrix} = \left(\begin{array}{c} r_1' \\ \hline r_2' \\ r_3' \\ \hline r_4' \\ r_5' \\ \hline r_6' \end{array}\right).$$

The coefficients and right-hand side are given by:

$$c_{2i}' = \frac{c_{2i}}{1-b_{2i}c_{2i+1}}, \quad c_{2i+1}' = \frac{-c_{2i+1}c_{2i}}{1-b_{2i}c_{2i+1}} \quad \forall\, i \in [1, P-2], \tag{4.25a,b}$$

$$b_{2i}' = \frac{-b_{2i}b_{2i+1}}{1-b_{2i}c_{2i+1}}, \quad b_{2i+1}' = \frac{b_{2i+1}}{1-b_{2i}c_{2i+1}} \quad \forall\, i \in [1, P-2], \tag{4.25c,d}$$

$$r_{2i}' = \frac{r_{2i} - b_{2i}r_{2i+1}}{1-b_{2i}c_{2i+1}}, \quad r_{2i+1}' = \frac{r_{2i+1} - c_{2i+1}r_{2i}}{1-b_{2i}c_{2i+1}} \quad \forall\, i \in [1, P-2]. \tag{4.25e,f}$$

Now interprocessor row operations on the system can yield the following:

$$\left(\begin{array}{cccccc} 1 & & & b_1'' & & \\ \hline & 1 & & b_2'' & & \\ c_3'' & & 1 & & & b_3'' \\ \hline c_4'' & & & 1 & & b_4'' \\ & & c_5'' & & 1 & \\ \hline & & c_6'' & & & 1 \end{array}\right)\begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \end{pmatrix} = \left(\begin{array}{c} r_1'' \\ \hline r_2'' \\ r_3'' \\ \hline r_4'' \\ r_5'' \\ \hline r_6'' \end{array}\right).$$

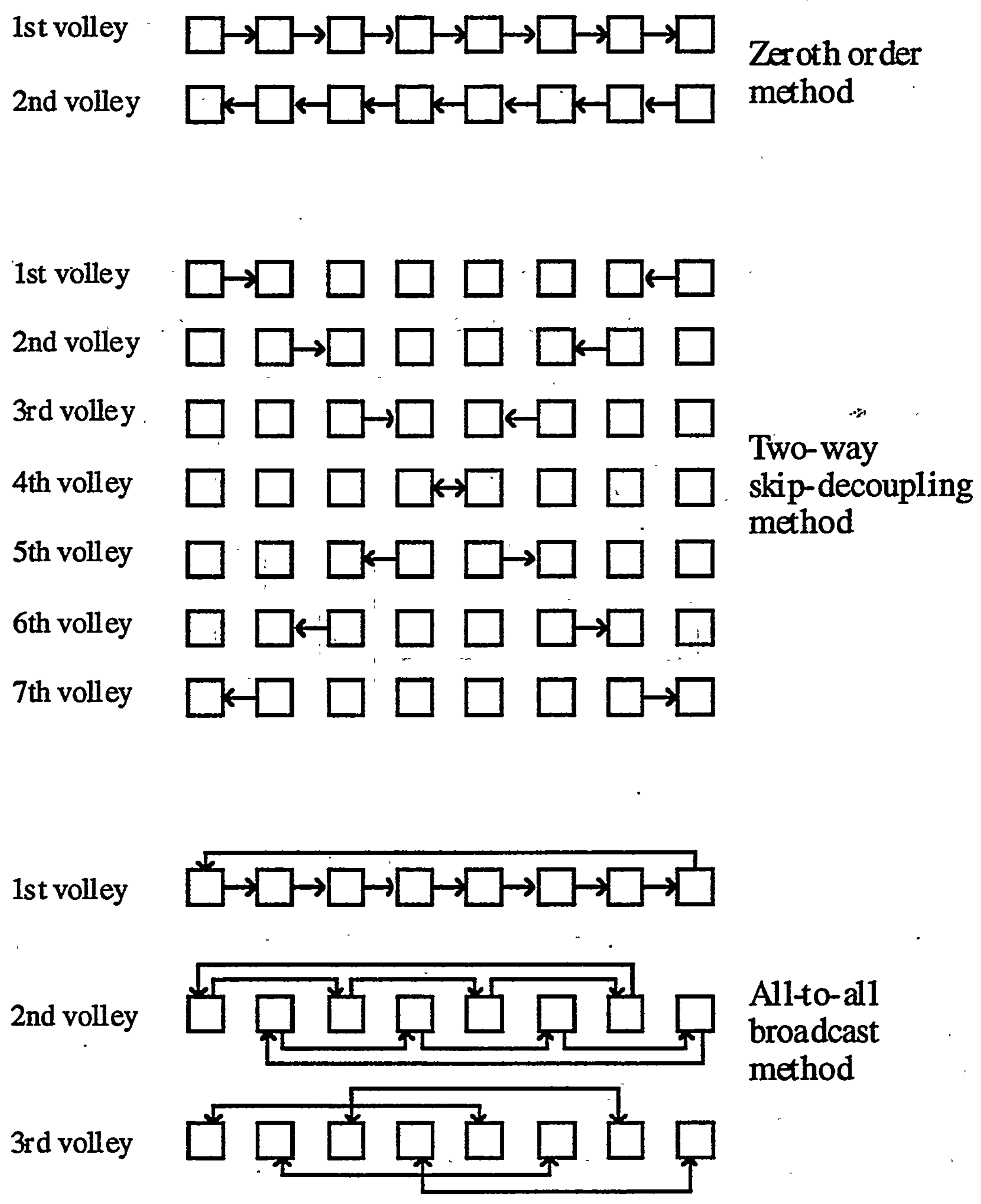

Figure 4.9. Interprocessor communication necessary for solution of an 8-processor reduced matrix by zeroth order methods, the two-way skip-decoupling method, and an all-to-all broadcast method [Mattor *et al.* 1995]. Each processor is represented by a square.

These coefficients and right-hand side are given by:

$$c''_{2i} = \frac{-c'_{2i-1}}{1-b'_{2i-1}c'_{2i}}, \quad c''_{2i-1} = \frac{c'_{2i-1}}{1-b'_{2i-1}c'_{2i}} \quad \forall\, i \in [2, P-1], \tag{4.26a,b}$$

$$b''_{2i} = \frac{b'_{2i}}{1-b'_{2i-1}c'_{2i}}, \quad b''_{2i-1} = \frac{-b'_{2i}}{1-b'_{2i-1}c'_{2i}} \quad \forall\, i \in [1, P-2], \tag{4.26c,d}$$

$$r''_{2i} = \frac{r'_{2i} - c'_{2i}r'_{2i-1}}{1-b'_{2i-1}c'_{2i}}, \quad r''_{2i-1} = \frac{r'_{2i-1} - b'_{2i-1}r'_{2i}}{1-b'_{2i-1}c'_{2i}} \quad \forall\, i \in [1, P-1]. \tag{4.26e,f}$$

All of the b''_i and c''_i are again first order in the b_{2i} and c_{2i-1}. If the r''_i are taken as the solution, then the solution is again zeroth order in b_{2i} and c_{2i-1}.

It is the case that proper manipulation will yield higher order accuracy with an increasing number of nearest-neighbor communication volleys. A first-order accurate method would require at least four volleys of $P-1$ send-receive pairs each. For brevity, an example of this first-order method is not included here. For $P=8$ the interprocessor communication necessary for the various methods is illustrated Figure 4.9.

4.4 Backsubstitution

Once the processor boundary values are determined from solution of the reduced system, the values can be substituted back into equations for the subdomain interior unknowns.

4.4.1 Interior processor

After solution of the reduced system, a typical interior processor is left with the following system of equations (N=6 example):

$$\begin{bmatrix} 1 & & & & & \\ c_2'' & 1 & & & & b_2'' \\ c_3'' & & 1 & & & b_3'' \\ c_4'' & & & 1 & & b_4'' \\ c_5'' & & & & 1 & b_5'' \\ & & & & & 1 \end{bmatrix} \begin{pmatrix} \vdots \\ x \\ x \\ x \\ x \\ \vdots \end{pmatrix} = \begin{pmatrix} r_1''' \\ r_2'' \\ r_3'' \\ r_4'' \\ r_5'' \\ r_6''' \end{pmatrix}$$

To solve for the subdomain interior unknowns, the processor boundary unknowns must be backsubstituted. If the processor boundary values are denoted r_1''' and r_N''', then the interior r_i''' are given by $r_i''' = r_i'' - c_i'' r_1''' - b_i'' r_N'''$ $\forall\, i \in [2, N-1]$. This backsubstitution is trivial because the interior unknowns have already been solved in terms of the processor boundary unknowns. If the processor boundary values are accurate to machine precision, these r_i''' are the machine precision solution to the tridiagonal system. If the processor boundary values are approximate, then these r_i''' are an approximate solution.

4.4.2 Terminal processor

After reduced system solution a terminal processor is left with the following system of equations in the periodic case:

$$\left[\begin{array}{cc|c|cc|cccccc} 1 & \cdots & \cdots & \cdots & \cdots & & & & & & \\ & \cdots & \cdots & \cdots & \cdots & 1 & & & & & \\ b_2'' & \cdots & \cdots & \cdots & \cdots & c_2'' & 1 & & & & \\ b_3'' & \cdots & \cdots & \cdots & \cdots & c_3'' & & 1 & & & \\ b_4'' & \cdots & \cdots & \cdots & \cdots & c_4'' & & & 1 & & \\ b_5'' & \cdots & \cdots & \cdots & \cdots & c_5'' & & & & 1 & \\ & \cdots & \cdots & \cdots & \cdots & & & & & & 1 \end{array}\right] \begin{pmatrix} \vdots \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix} = \begin{pmatrix} r_0''' \\ r_1''' \\ r_2'' \\ r_3'' \\ r_4'' \\ r_5'' \\ r_6''' \end{pmatrix}$$

To solve for the subdomain interior unknowns, the processor boundary unknowns r_0''' and r_1''' must be backsubstituted. For the periodic implementation outlined in this dissertation, r_N''' does not enter this backsubstitution. Instead, only the r_0''' arising from periodicity enters, giving for the interior $r_i''' = r_i'' - b_i'' r_0''' - c_i'' r_1''' \quad \forall\, i \in [2, N-1]$. If the problem is not periodic, then the backsubstitution simplifies. For the ADI implementation of this dissertation the terminal processor is left with the following set of equations:

$$
\begin{array}{cc|c|cc|cccccc}
\hline
\cdots & \cdots & \cdots & \cdots & \cdots & 1 & & & & & \\
\cdots & \cdots & \cdots & \cdots & \cdots & c_2'' & 1 & & & & \\
\cdots & \cdots & \cdots & \cdots & \cdots & c_3'' & & 1 & & & \\
\cdots & \cdots & \cdots & \cdots & \cdots & c_4'' & & & 1 & & \\
\cdots & \cdots & \cdots & \cdots & \cdots & c_5'' & & & & 1 & \\
\cdots & \cdots & \cdots & \cdots & \cdots & c_6'' & & & & & 1
\end{array}
\begin{pmatrix} \vdots \\ x \\ x \\ x \\ x \\ x \end{pmatrix}
=
\begin{pmatrix} r_1''' \\ r_2'' \\ r_3'' \\ r_4'' \\ r_5'' \\ r_6'' \end{pmatrix}
$$

Only backsubstitution of r_1''' is necessary: $r_i''' = r_i'' - c_i'' r_1''' \quad \forall\, i \in [2, N]$. Backsubstitution in the other terminal processor proceeds in an entirely analogous manner in the opposite direction.

After solution of the reduced matrix, solution by back-substitution can be achieved with $4N$ AOs for the interior processors. For terminal processors, $4N$ AOs are needed in the periodic case, but only $2N$ are needed in the Robbins case. The total AOs, not counting those necessary for reduced matrix solution, are $17N$ for interior processors, and either $17N$ or $13N$ for terminal processors.

Appendix 4.A1
The method of Wang and the method of Kowalik and Kumar for parallel solution of a distributed tridiagonal system

In 1981, H.H. Wang wrote of a method for the parallel solution of a nonperiodic tridiagonal system of equations. In 1985, Kowalik and Kumar wrote of a similar method. These methods are briefly described and compared in this appendix. The original distributed system has the form depicted in Figure 4.10.

$$
\begin{array}{c}
\\ \\ \mathbf{P1} \\ \\ \\ \\ \mathbf{P2} \\ \\ \\ \\ \mathbf{P3} \\ \\ \\ \\ \mathbf{P4} \\ \\
\end{array}
\left(\begin{array}{cccc|cccc|cccc|cccc}
a_1^1 & b_1^1 & & & & & & & & & & & & & & \\
c_2^1 & a_2^1 & b_2^1 & & & & & & & & & & & & & \\
 & c_3^1 & a_3^1 & b_3^1 & & & & & & & & & & & & \\
 & & c_4^1 & a_4^1 & b_4^1 & & & & & & & & & & & \\
\hline
 & & & c_1^2 & a_1^2 & b_1^2 & & & & & & & & & & \\
 & & & & c_2^2 & a_2^2 & b_2^2 & & & & & & & & & \\
 & & & & & c_3^2 & a_3^2 & b_3^2 & & & & & & & & \\
 & & & & & & c_4^2 & a_4^2 & b_4^2 & & & & & & & \\
\hline
 & & & & & & & c_1^3 & a_1^3 & b_1^3 & & & & & & \\
 & & & & & & & & c_2^3 & a_2^3 & b_2^3 & & & & & \\
 & & & & & & & & & c_3^3 & a_3^3 & b_3^3 & & & & \\
 & & & & & & & & & & c_4^3 & a_4^3 & b_4^3 & & & \\
\hline
 & & & & & & & & & & & c_1^4 & a_1^4 & b_1^4 & & \\
 & & & & & & & & & & & & c_2^4 & a_2^4 & b_2^4 & \\
 & & & & & & & & & & & & & c_3^4 & a_3^4 & b_3^4 \\
 & & & & & & & & & & & & & & c_4^4 & a_4^4
\end{array}\right)
\left(\begin{array}{c}
x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x
\end{array}\right)
=
\left(\begin{array}{c}
r_1^1 \\ r_2^1 \\ r_3^1 \\ r_4^1 \\ \hline r_1^2 \\ r_2^2 \\ r_3^2 \\ r_4^2 \\ \hline r_1^3 \\ r_2^3 \\ r_3^3 \\ r_4^3 \\ \hline r_1^4 \\ r_2^4 \\ r_3^4 \\ r_4^4
\end{array}\right)
$$

Figure 4.10. Typical distributed nonperiodic tridiagonal system of equations arising from a spatial domain decomposition. In this example, a contiguous groups of four equations is initially known by each of the four processors in the decomposition.

Let P be the number of processors in the decomposition and N be the number of equations in each contiguous group. Then in Figure 4.10, $P=4$ and $N=4$. After intraprocessor row operations and one communication volley of $P-1$ simultaneous send-receive (SR) pairs, one can arrive at the distributed system in Figure 4.11 [Wang; Kowalik and Kumar].

$$\left(\begin{array}{cccc|cccc|cccc|cccc}
a_1 & & & g_1 & & & & & & & & & & & & \\
 & a_2 & & g_2 & & & & & & & & & & & & \\
 & & a_3 & b_3 & & & & & & & & & & & & \\
 & & & a_4 & & & & g_4 & & & & & & & & \\
\hline
 & & & c_5 & a_5 & & & g_5 & & & & & & & & \\
 & & & f_6 & & a_6 & & g_6 & & & & & & & & \\
 & & & f_7 & & & a_7 & g_7 & & & & & & & & \\
 & & & f_8 & & & & a_8 & & & & g_8 & & & & \\
\hline
 & & & & & & & c_9 & a_9 & & & g_9 & & & & \\
 & & & & & & & f_{10} & & a_{10} & & g_{10} & & & & \\
 & & & & & & & f_{11} & & & a_{11} & g_{11} & & & & \\
 & & & & & & & f_{12} & & & & a_{12} & & & & g_{12} \\
\hline
 & & & & & & & & & & & c_{13} & a_{13} & & & g_{13} \\
 & & & & & & & & & & & f_{14} & & a_{14} & & g_{14} \\
 & & & & & & & & & & & f_{15} & & & a_{15} & g_{15} \\
 & & & & & & & & & & & f_{16} & & & & a_{16}
\end{array}\right)
\begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix}
=
\left(\begin{array}{c} r_1 \\ r_2 \\ r_3 \\ r_4 \\ \hline r_5 \\ r_6 \\ r_7 \\ r_8 \\ \hline r_9 \\ r_{10} \\ r_{11} \\ r_{12} \\ \hline r_{13} \\ r_{14} \\ r_{15} \\ r_{16} \end{array}\right)$$

Figure 4.11. Alternative system of equations which can be produced from the original distributed tridiagonal system of Figure 4.10 [Wang 1981; Kowalik and Kumar 1985].

Wang suggested eliminating successively c_5, f_6, f_7, f_8, c_9, f_{10}, f_{11}, ..., f_{16}, with interprocessor communication as necessary. This is a type of forward elimination of subdiagonal elements. The communication would involve $P-1$ sequential volleys of one send-receive pair (SR pair) each for a system distributed over P processors. Wang then completes the solution by eliminating successively g_{15}, g_{14}, ..., g_1, again with interprocessor communication as necessary. This is a type of backward elimination of superdiagonal elements, and would involve $P-1$ sequential volleys of one SR pair each. The undesirable feature of the exact method that Wang outlines is the potentially large number of AOs that must be performed between the last $2P-2$ communication volleys. As these AOs are performed, the processors are not operating in parallel.

To improve parallelism, Kowalik and Kumar begin with the above matrix and identify a small subset of the equations to be solved quickly in order to decouple the processors before completing the solution. In the example above, note that the 4th, 8th, 12th, and 16th equations comprise a tridiagonal system that determines the corresponding unknowns. Determination of these 4th, 8th, 12th, and 16th unknowns would decouple the processors, and hence these equations comprise the "small system" of interest:

$$\left(\begin{array}{c|c|c|c} a_4 & g_4 & & \\ \hline f_8 & a_8 & g_8 & \\ \hline & f_{12} & a_{12} & g_{12} \\ \hline & & f_{16} & a_{16} \end{array}\right)\begin{pmatrix} x \\ x \\ x \\ x \end{pmatrix} = \left(\begin{array}{c} r_4 \\ \hline r_8 \\ \hline r_{12} \\ \hline r_{16} \end{array}\right)$$

Figure 4.12. Reduced matrix for method of Kowalik and Kumar.

An all-to-all broadcast of the small system allows every processor to solve for the 4th, 8th, 12th, and 16th unknowns [Kumar]. After one more communication volley of $2P-2$ SR pairs, these knowns allow intraprocessor backsubstitution to proceed in parallel.

Appendix 4.A2
The method of Mattor, Williams, and Hewett for parallel solution of a distributed tridiagonal system

In 1993, Mattor, Williams and Hewett discovered another method for solving the tridiagonal system depicted in Figure 4.10. Without any interprocessor communication, the distributed system is initially seperated into uncoupled subsystems each of which is given three right-hand sides (except for terminal processors, which get two right-hand sides), as in Figure 4.13. Each coupling coefficient, e.g. b_4^1, in the original coefficient matrix becomes the only nonzero element in one of the new right-hand sides.

$$\begin{array}{c}
\\ \\ \mathbf{P1} \\ \\
\\ \\ \mathbf{P2} \\ \\
\\ \\ \mathbf{P3} \\ \\
\\ \\ \mathbf{P4} \\ \\
\end{array}
\left(\begin{array}{cccc|cccc|cccc|cccc}
a_1^1 & b_1^1 & & & & & & & & & & & & & & \\
c_2^1 & a_2^1 & b_2^1 & & & & & & & & & & & & & \\
 & c_3^1 & a_3^1 & b_3^1 & & & & & & & & & & & & \\
 & & c_4^1 & a_4^1 & & & & & & & & & & & & \\
\hline
 & & & & a_1^2 & b_1^2 & & & & & & & & & & \\
 & & & & c_2^2 & a_2^2 & b_2^2 & & & & & & & & & \\
 & & & & & c_3^2 & a_3^2 & b_3^2 & & & & & & & & \\
 & & & & & & c_4^2 & a_4^2 & & & & & & & & \\
\hline
 & & & & & & & & a_1^3 & b_1^3 & & & & & & \\
 & & & & & & & & c_2^3 & a_2^3 & b_2^3 & & & & & \\
 & & & & & & & & & c_3^3 & a_3^3 & b_3^3 & & & & \\
 & & & & & & & & & & c_4^3 & a_4^3 & & & & \\
\hline
 & & & & & & & & & & & & a_1^4 & b_1^4 & & \\
 & & & & & & & & & & & & c_2^4 & a_2^4 & b_2^4 & \\
 & & & & & & & & & & & & & c_3^4 & a_3^4 & b_3^4 \\
 & & & & & & & & & & & & & & c_4^4 & a_4^4
\end{array}\right)
\begin{pmatrix} x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \\ x \end{pmatrix}
=
\left(\begin{array}{c|c|c}
r_1^1 & & \\
r_2^1 & & \\
r_3^1 & & \\
r_4^1 & & b_4^1 \\
\hline
r_1^2 & c_1^2 & \\
r_2^2 & & \\
r_3^2 & & \\
r_4^2 & & b_4^2 \\
\hline
r_1^3 & c_1^3 & \\
r_2^3 & & \\
r_3^3 & & \\
r_4^3 & & b_4^3 \\
\hline
r_1^4 & c_1^4 & \\
r_2^4 & & \\
r_3^4 & & \\
r_4^4 & &
\end{array}\right)$$

Figure 4.13. Reformulation of the distributed tridiagonal system solution in a way that initially decouples the processors [Mattor, Williams, and Hewett, 1995]. Zero elements are not shown.

Each of the new subsystems is solved for each right-hand side using only intraprocessor row operations:

$$\left(\begin{array}{cccc|cccc|cccc|cccc}
1&&&&&&&&&&&&&&&\\
&1&&&&&&&&&&&&&&\\
&&1&&&&&&&&&&&&&\\
&&&1&&&&&&&&&&&&\\ \hline
&&&&1&&&&&&&&&&&\\
&&&&&1&&&&&&&&&&\\
&&&&&&1&&&&&&&&&\\
&&&&&&&1&&&&&&&&\\ \hline
&&&&&&&&1&&&&&&&\\
&&&&&&&&&1&&&&&&\\
&&&&&&&&&&1&&&&&\\
&&&&&&&&&&&1&&&&\\ \hline
&&&&&&&&&&&&1&&&\\
&&&&&&&&&&&&&1&&\\
&&&&&&&&&&&&&&1&\\
&&&&&&&&&&&&&&&1
\end{array}\right)
\begin{pmatrix} x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x \end{pmatrix}
=
\left(\begin{array}{c|c|c}
r_1 & & t_1\\
r_2 & & t_2\\
r_3 & & t_3\\
r_4 & & t_4\\ \hline
r_5 & s_5 & t_5\\
r_6 & s_6 & t_6\\
r_7 & s_7 & t_7\\
r_8 & s_8 & t_8\\ \hline
r_9 & s_9 & t_9\\
r_{10} & s_{10} & t_{10}\\
r_{11} & s_{11} & t_{11}\\
r_{12} & s_{12} & t_{12}\\ \hline
r_{13} & s_{13} & \\
r_{14} & s_{14} & \\
r_{15} & s_{15} & \\
r_{16} & s_{16} &
\end{array}\right)
\begin{array}{c}
\\ \\ \\ \rightarrow\\ \rightarrow\\ \\ \\ \rightarrow\\ \rightarrow\\ \\ \\ \rightarrow\\ \rightarrow\\ \\ \\ \\
\end{array}$$

The solution of the original distributed system is a linear combination of the subsystem solutions. The coefficients of the linear combination are determined by a reduced system with the structure shown in Figure 4.14. A copy of the whole reduced system is assembled on every processor by an efficient all-to-all broadcast and a pivoting serial tridiagonal solver is used to find the coefficients of the linear combinations.

$$\left(\begin{array}{c|cc|cc|c}
t_4 & -1 & & & & \\ \hline
-1 & s_5 & t_5 & & & \\
 & s_8 & t_8 & -1 & & \\ \hline
 & & -1 & s_9 & t_9 & \\
 & & & s_{12} & t_{12} & -1 \\ \hline
 & & & & -1 & s_{13}
\end{array}\right)
\begin{pmatrix} c_t^1\\ c_s^2\\ c_t^2\\ c_s^3\\ c_t^3\\ c_s^4 \end{pmatrix}
=
\begin{pmatrix} \underline{r_4}\\ r_5\\ \underline{r_8}\\ r_9\\ \underline{r_{12}}\\ r_{13} \end{pmatrix}$$

Figure 4.14. Reduced matrix corresponding to the example in Figure 4.13 [Mattor, Williams and Hewett, 1995].

Pivoting is required for certain instances of internal Dirichlet boundary conditions. Finally, the linear combinations are summed for each unknown:

$$\left(\begin{array}{cccc|cccc|cccc|cccc}
1&&&&&&&&&&&&&&&\\
&1&&&&&&&&&&&&&&\\
&&1&&&&&&&&&&&&&\\
&&&1&&&&&&&&&&&&\\
\hline
&&&&1&&&&&&&&&&&\\
&&&&&1&&&&&&&&&&\\
&&&&&&1&&&&&&&&&\\
&&&&&&&1&&&&&&&&\\
\hline
&&&&&&&&1&&&&&&&\\
&&&&&&&&&1&&&&&&\\
&&&&&&&&&&1&&&&&\\
&&&&&&&&&&&1&&&&\\
\hline
&&&&&&&&&&&&1&&&\\
&&&&&&&&&&&&&1&&\\
&&&&&&&&&&&&&&1&\\
&&&&&&&&&&&&&&&1
\end{array}\right)
\begin{pmatrix} x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x\\x \end{pmatrix}
=
\left(\begin{array}{c}
r_1+c_t^1t_1\\
r_2+c_t^1t_2\\
r_3+c_t^1t_3\\
r_4+c_t^1t_4\\
\hline
r_5+c_s^2s_5+c_t^2t_5\\
r_6+c_s^2s_6+c_t^2t_6\\
r_7+c_s^2s_7+c_t^2t_7\\
r_8+c_s^2s_8+c_t^2t_8\\
\hline
r_9+c_s^3s_9+c_t^3t_9\\
r_{10}+c_s^3s_{10}+c_t^3t_{10}\\
r_{11}+c_s^3s_{11}+c_t^3t_{11}\\
r_{12}+c_s^3s_{12}+c_t^3t_{12}\\
\hline
r_{13}+c_s^4s_{13}\\
r_{14}+c_s^4s_{14}\\
r_{15}+c_s^4s_{15}\\
r_{16}+c_s^4s_{16}
\end{array}\right).$$

Just as for the other parallel tridiagonal methods, it is easy to leave each processor with knowledge of the values at subdomain guard points as well. The guard point values are necessary for an ADI method because the derivatives in alternating directions use three-point templates.

Appendix 4.A3
Matrix element relations in reduction of a tridiagonal subsystem to N-form

Each processor is initially given a subset of the equations of the whole tridiagonal system. To prove that certain approximations are useful, it is convenient to start with a representative subset of the equations, and evaluate the elements of the subset as the equations are reduced to "N-form".

4.A3.1 Interior processor relations

When the tridiagonal system arises from a symmetric differential operator such as $\partial_x(\varepsilon\partial_x)$, the original subset of equations has the following coefficient matrix for an interior processor (elements distinguished):

$$\begin{array}{cccccccc}
\hline
c_1 & -a_1 & c_2 & & & & & \\
 & c_2 & -a_2 & c_3 & & & & \\
 & & c_3 & -a_3 & c_4 & & & \\
 & & & c_4 & -a_4 & c_5 & & \\
 & & & & c_5 & -a_5 & c_6 & \\
 & & & & & c_6 & -a_6 & c_7 \\
\hline
\end{array}$$

After forward elimination, the subset of equations has the following new coefficient matrix:

$$\begin{array}{cccccccc}
\hline
\ddot{c}_1 & 1 & \dot{c}_2 & & & & & \\
 & \ddot{c}_2 & 1 & \dot{c}_3 & & & & \\
 & \ddot{c}_3 & & 1 & \dot{c}_4 & & & \\
 & \ddot{c}_4 & & & 1 & \dot{c}_5 & & \\
 & \ddot{c}_5 & & & & 1 & \dot{c}_6 & \\
 & \ddot{c}_6 & & & & & 1 & \dot{c}_7 \\
\hline
\end{array}$$

The following relationships are seen to exist between the original elements and the elements after forward elimination:

$$c_1'' = -\frac{c_1}{a_1}, \qquad c_2' = -\frac{c_2}{a_1}, \qquad r_1' = -\frac{r_1}{a_1},$$

$$c_2'' = -\frac{c_2}{a_2} = -\frac{c_2}{U_1}, \quad c_3' = -\frac{c_3}{a_2} = -\frac{c_3 U_0}{U_1},$$

$$c_3'' = \frac{-c_3 c_2''}{-a_3 - c_3 c_3'} = \frac{-c_3 c_2}{a_3 a_2 - c_3^2} = \frac{-c_3 c_2}{U_2},$$

$$c_4' = \frac{c_4}{-a_3 - c_3 c_3'} = \frac{-c_4 a_2}{a_3 a_2 - c_3^2} = \frac{-c_4 U_1}{U_2},$$

$$c_4'' = \frac{-c_4 c_3''}{-a_4 - c_4 c_4'} = \frac{-c_4 c_3 c_2}{a_4 U_2 - c_4^2 U_1} = \frac{-c_4 c_3 c_2}{U_3},$$

$$c_5' = \frac{c_5}{-a_4 - c_4 c_4'} = \frac{-c_5 U_2}{a_4 U_2 - c_4^2 U_1} = \frac{-c_5 U_2}{U_3},$$

$$c_5'' = \frac{-c_5 c_4''}{-a_5 - c_5 c_5'} = \frac{-c_5 c_4 c_3 c_2}{a_5 U_3 - c_5^2 U_2} = \frac{-c_5 c_4 c_3 c_2}{U_4},$$

$$c_6' = \frac{c_6}{-a_5 - c_5 c_5'} = \frac{-c_6 U_3}{a_5 U_3 - c_5^2 U_2} = \frac{-c_6 U_3}{U_4},$$

$$c_6'' = \frac{-c_6 c_5''}{-a_6 - c_6 c_6'} = \frac{-c_6 c_5 c_4 c_3 c_2}{U_5}, \qquad c_7' = \frac{c_7}{-a_6 - c_6 c_6'} = \frac{-c_7 U_4}{U_5}.$$

Summarizing,

$$c_1'' = -\frac{c_1}{a_1}, \qquad c_i'' = \frac{-1}{U_{i+1}} \prod_{j=2}^{i} c_j \quad \forall\ i \in [2, N],$$

$$c_2' = -\frac{c_2}{a_1}, \qquad c_i' = -\frac{c_i U_{i-3}}{U_{i-2}} \quad \forall\ i \in [3, N+1].$$

The quantities U_i are recursively defined in the following way:

$$U_{-1} = 0, \; U_0 = 1, \; U_i = a_{i+1} U_{i-1} - c_{i+1}^2 U_{i-2}, \quad i = 1, \ldots, N-1.$$

After complete reduction to N-form after backward elimination the subset of equations has the following coefficient matrix:

$$\begin{array}{cccccccc}
\hline
d_1'' & 1 & & & & & d_2' & \\
 & d_2'' & 1 & & & & d_3' & \\
 & d_3'' & & 1 & & & d_4' & \\
 & d_4'' & & & 1 & & d_5' & \\
 & d_5'' & & & & 1 & d_6' & \\
 & d_6'' & & & & & 1 & d_7' \\
\hline
\end{array}.$$

The d_i'' and d_i' are given by:

$$d_6'' = c_6'' = \frac{-c_6c_5c_4c_3c_2}{U_5}, \qquad d_7' = c_7' = \frac{-c_7U_4}{U_5},$$

$$d_5'' = c_5'' = \frac{-c_5c_4c_3c_2}{U_4}, \qquad d_6' = c_6' = \frac{-c_6U_3}{U_4},$$

$$\begin{aligned}
d_4'' &= c_4'' - c_5'c_5'' = \frac{-c_4c_3c_2U_4 - c_5^2c_4c_3c_2U_2}{U_3U_4} = \frac{-c_4c_3c_2\left(a_5U_3 - c_5^2U_2\right) - c_5^2c_4c_3c_2U_2}{U_3U_4} \\
&= \frac{-c_4c_3c_2a_5}{U_4} = \frac{-c_4c_3c_2V_6}{U_4},
\end{aligned}$$

$$d_5' = -c_5'd_6' = -c_5'c_6' = \frac{-c_5c_6U_2}{U_4},$$

$$\begin{aligned}
d_3'' &= c_3'' - c_4'd_4'' = \frac{-c_3c_2}{U_2} - \frac{c_4U_1}{U_2}\left(\frac{c_4c_3c_2V_6}{U_4}\right) = \frac{-c_3c_2U_4 - c_4^2c_3c_2V_6U_1}{U_2U_4} \\
&= \frac{-c_3c_2\left(a_5\left(a_4U_2 - c_4^2U_1\right) - c_5^2U_2\right) - c_4^2c_3c_2V_6U_1}{U_2U_4} = \frac{-c_3c_2\left(a_5a_4 - c_5^2\right)}{U_4} \\
&= \frac{-c_3c_2V_5}{U_4},
\end{aligned}$$

$$d_4' = -c_4'd_5' = c_4'c_5'c_6' = \frac{-c_4c_5c_6U_1}{U_4},$$

$$\begin{aligned}
d_2'' &= c_2'' - c_3'd_3'' = -\frac{c_2}{U_1} - \frac{c_3}{U_1}\left(\frac{c_3c_2V_5}{U_4}\right) = \frac{-c_2U_4 - c_3^2c_2V_5}{U_1U_4} \\
&= \frac{-c_2\left(a_5\left(a_4\left(a_3U_1 - c_3^2\right) - c_4^2U_1\right) - c_5^2\left(a_3U_1 - c_3^2\right)\right) - c_3^2c_2V_5}{U_1U_4} \\
&= \frac{-c_2\left(a_5\left(a_4a_3 - c_4^2\right) - c_5^2a_3\right)}{U_4} = \frac{-c_2V_4}{U_4},
\end{aligned}$$

$$d_3' = -c_3'd_4' = -c_3'c_4'c_5'c_6' = \frac{-c_3c_4c_5c_6}{U_4},$$

$$d_2' = \frac{-c_2'd_3'}{1 - c_2'd_2''} = \frac{-c_2c_3c_4c_5c_6}{a_1U_4 + a_1c_2'c_2V_4} = \frac{-c_2c_3c_4c_5c_6}{a_1V_3 - c_2^2V_4} = \frac{-c_2c_3c_4c_5c_6}{V_2},$$

$$d_1'' = \frac{c_1''}{1 - c_2'd_2''} = \frac{-c_1U_4}{a_1U_4 + a_1c_2'c_2V_4} = \frac{-c_1U_4}{a_1V_3 - c_2^2V_4} = \frac{-c_1U_4}{V_2} = \frac{-c_1V_3}{V_2}.$$

Look at the following quantities introduced above:

$$\begin{aligned}
V_6 &= a_5, \qquad V_5 = a_5a_4 - c_5^2 = a_4a_5 - c_5^2 = a_4V_6 - c_5^2,\\
V_4 &= a_5\left(a_4a_3 - c_4^2\right) - c_5^2a_3 = a_3\left(a_4a_5 - c_5^2\right) - c_4^2a_5 = a_3V_5 - c_4^2V_6.\\
V_3 &= a_5\left(a_4\left(a_3a_2 - c_3^2\right) - c_4^2a_2\right) - c_5^2\left(a_3a_2 - c_3^2\right) = a_5U_3 - c_5^2U_2 = U_4\\
&= a_2\left(a_5\left(a_3a_4 - c_4^2\right) - c_5^2a_3\right) - c_3^2\left(a_5a_4 - c_5^2\right)\\
&= a_2\left(a_3\left(a_4a_5 - c_5^2\right) - c_4^2a_5\right) - c_3^2\left(a_4a_5 - c_5^2\right) = a_2V_4 - c_3^2V_5,\\
V_2 &= a_1V_3 - c_2^2V_4.
\end{aligned}$$

These V_i can be defined recursively as well:

$$V_{N+2} = 0, \;\; V_{N+1} = 1, \;\; V_i = a_{i-1}V_{i+1} - c_i^2V_{i+2}, \quad i = N, \,\ldots,\, 2.$$

The V_i are "mirror images" of the U_i!

$$\begin{aligned}
&d_1'' = \frac{-c_1U_{N-2}}{V_2} = \frac{-c_1V_3}{V_2}, \qquad d_i'' = \frac{-V_{i+2}}{U_{N-2}}\prod_{j=2}^{i}c_j = \frac{-V_{i+2}}{V_3}\prod_{j=2}^{i}c_j \;\; \forall\, i \in [2, N-1],\\
&d_N'' = \frac{-1}{U_{N-1}}\prod_{j=2}^{N}c_j, \;\; d_2' = \frac{-1}{V_2}\prod_{j=2}^{N}c_j, \;\; d_i' = -\frac{U_{i-3}}{U_{N-2}}\prod_{j=i}^{N}c_j = -\frac{U_{i-3}}{V_3}\prod_{j=i}^{N}c_j \;\; \forall\, i \in [3, N].\\
&d_{N+1}' = c_{N+1}' = \frac{-c_{N+1}U_{N-2}}{U_{N-1}} = \frac{-c_{N+1}V_3}{U_{N-1}}.
\end{aligned}$$

Now, for the right-hand side:

$$\begin{aligned}
r_1' &= -\frac{r_1}{a_1}, \qquad\qquad r_2' = -\frac{r_2}{a_2} = -\frac{R_1}{U_1},\\
r_3' &= \frac{r_3 - c_3r_2'}{-a_3 - c_3c_3'} = -\frac{r_3 + c_3R_1/U_1}{a_3 - c_3c_3/U_1} = -\frac{r_3U_1 + c_3R_1}{a_3U_1 - c_3^2} = -\frac{R_2}{U_2},\\
r_4' &= \frac{r_4 - c_4r_3'}{-a_4 - c_4c_4'} = -\frac{r_4U_2 + c_4R_2}{a_4U_2 - c_4^2U_1} = -\frac{R_3}{U_3},\\
r_5' &= \frac{r_5 - c_5r_4'}{-a_5 - c_5c_5'} = -\frac{r_5U_3 + c_5R_3}{a_5U_3 - c_5^2U_2} = -\frac{R_4}{U_4}, \qquad r_6' = -\frac{R_5}{U_5}.
\end{aligned}$$

Here, the R_i are defined recursively as follows:

$$R_0 = 0, \;\; R_i = r_{i+1}U_{i-1} + c_{i+1}R_{i-1}, \quad i = 1, \,\ldots,\, N-1.$$

Obviously,

$$r_1' = -\frac{r_1}{a_1}, \; r_i' = -\frac{R_{i-1}}{U_{i-1}} \;\; \forall \; i \in [2, N].$$

Then there is a backsubstitution:

$$\begin{aligned}
r_6'' &= r_6' = -\frac{R_6}{U_6}, \qquad\qquad r_5'' = r_5' = -\frac{R_4}{U_4}, \\
r_4'' &= r_4' - c_5' r_5' = -\frac{R_3}{U_3} - \frac{c_5 U_2}{U_3}\frac{R_4}{U_4} = -\frac{R_3 U_4 + c_5 U_2 R_4}{U_3 U_4} \\
&= -\frac{R_3\left(a_5 U_3 - c_5^2 U_2\right) + c_5 U_2\left(r_5 U_3 + c_5 R_3\right)}{U_3 U_4} = -\frac{a_5 R_3 + c_5 r_5 U_2}{U_4} \\
&= -\frac{V_6 R_3 + c_5 r_5 U_2}{U_4} = -\frac{V_6 R_3 + c_5 S_6 U_2}{U_4}, \\
r_3'' &= r_3' - c_4' r_4'' = -\frac{R_2}{U_2} - \frac{c_4 U_1}{U_2}\frac{a_5 R_3 + c_5 r_5 U_2}{U_4} = -\frac{R_2 U_4 + c_4 U_1\left(a_5 R_3 + c_5 r_5 U_2\right)}{U_2 U_4} \\
&= -\frac{R_2\left(a_5\left(a_4 U_2 - c_4^2 U_1\right) - c_5^2 U_2\right) + c_4 U_1\left(a_5\left(r_4 U_2 + c_4 R_2\right) + c_5 r_5 U_2\right)}{U_2 U_4} \\
&= -\frac{\left(a_4 a_5 - c_5^2\right) R_2 + c_4\left(a_5 r_4 + c_5 r_5\right) U_1}{U_4} = -\frac{V_5 R_2 + c_4\left(r_4 V_6 + c_5 r_5 V_7\right) U_1}{U_4} \\
&= -\frac{V_5 R_2 + c_4 S_5 U_1}{U_4}, \\
r_2'' &= r_2' - c_3' r_3'' = -\frac{R_1}{U_1} - \frac{c_3}{U_1}\frac{V_5 R_2 + c_4\left(a_5 r_4 + c_5 r_5\right) U_1}{U_4} = -\frac{R_1 U_4 + c_3\left(V_5 R_1 + c_4\left(a_5 r_4 + c_5 r_5\right) U_1\right)}{U_1 U_4} \\
&= -\frac{R_1\left(-c_4^2 a_5 U_1 + V_5\left(a_3 U_1 - c_3^2\right)\right) + c_3\left(V_5\left(r_3 U_1 + c_3 R_2\right) + c_4\left(a_5 r_4 + c_5 r_5\right) U_1\right)}{U_1 U_4} \\
&= -\frac{R_1\left(a_3 V_5 - c_4^2 a_5\right) + c_3\left(r_3 V_5 + c_4\left(r_4 a_5 + c_5 r_5\right)\right)}{U_4} \\
&= -\frac{V_4 R_1 + c_3\left(r_3 V_5 + c_4\left(r_4 V_6 + c_5 r_5 V_7\right)\right)}{U_4} = -\frac{r_2 V_4 + c_3 S_4 U_0}{U_4} = -\frac{S_3}{U_4} = -\frac{S_3}{V_3}, \\
r_1'' &= \frac{r_1' - c_2' r_2''}{1 - c_2' d_2''} = -\frac{r_1 V_3 + c_2 S_3}{a_1 V_3 - c_2^2 V_4} = -\frac{S_2}{V_2}.
\end{aligned}$$

Here, the S_i are given by

$$S_{N+1} = 0, \; S_i = r_{i-1} V_{i+1} + c_i S_{i+1}, \quad i = N, \ldots, 2.$$

The R_i and S_i also obey recursion relations that are "mirror images".

$$r_i'' = -\frac{V_{i+2}R_{i-1} + c_{i+1}S_{i+2}U_{i-2}}{U_{N-2}}, \quad i=N, \ldots, 3, \qquad r_i'' = -\frac{S_{i+1}}{V_{i+1}}, \quad i=1,\, 2.$$

4.A3.2 Terminal processor relations for the nonperiodic case

The original subset of equations arising from a symmetric operator has the following coefficient matrix for a terminal processor (elements distinguished):

$$\begin{array}{ccccccc}
\hline
c_1 & -a_1 & c_2 & & & & \\
 & c_2 & -a_2 & c_3 & & & \\
 & & c_3 & -a_3 & c_4 & & \\
 & & & c_4 & -a_4 & c_5 & \\
 & & & & c_5 & -a_5 & c_6 \\
 & & & & & c_6 & -a_6
\end{array}$$

After forward elimination, the subset of equations has the following new coefficient matrix:

$$\begin{array}{ccccccc}
\hline
c_1'' & 1 & c_2' & & & & \\
 & c_2'' & 1 & c_3' & & & \\
 & c_3'' & & 1 & c_4' & & \\
 & c_4'' & & & 1 & c_5' & \\
 & c_5'' & & & & 1 & c_6' \\
 & c_6'' & & & & & 1 \\
\hline
\end{array}$$

The c_i' and c_i'' are the same as for an interior processor, except $c_{N+1}' = 0$ for the general case of N equations in the subset. For practical purposes, it might be sufficient leave the subset in this last form. Further reduction of a terminal subset of equations does not buy any speedup in execution time. However, further reduction of the terminal subset is included here for completeness of presentation. After full reduction of the terminal subset is complete, it has the following coefficient matrix:

$$\begin{array}{lllllll}
\hline
e_1'' & 1 & & & & & \\
 & e_2'' & 1 & & & & \\
 & e_3'' & & 1 & & & \\
 & e_4'' & & & 1 & & \\
 & e_5'' & & & & 1 & \\
 & e_6'' & & & & & 1
\end{array}$$

For the case of N equations:

$$e_N'' = c_N'', \quad e_i'' = c_i'' - c_{i+1}' e_{i+1}'' \ \forall \ i = N-1, \ldots, 2, \quad e_1'' = \frac{c_1''}{1 - c_1' e_2''}.$$

Expanding these equations,

$$\begin{aligned}
e_6'' &= c_6'' = \frac{-c_6 c_5 c_4 c_3 c_2}{U_5}, \\
e_5'' &= \frac{-c_5 c_4 c_3 c_2}{U_4} - \frac{c_6 U_3}{U_4} \frac{c_6 c_5 c_4 c_3 c_2}{U_5} = -\left(\frac{U_5 + c_6^2 U_3}{U_5}\right) \frac{c_5 c_4 c_3 c_2}{U_4} = -\frac{c_5 c_4 c_3 c_2 a_6}{U_5} \\
&= -\frac{c_5 c_4 c_3 c_2 W_5}{U_5}, \\
e_4'' &= \frac{-c_4 c_3 c_2}{U_3} - \frac{c_5 U_2}{U_3} \frac{c_5 c_4 c_3 c_2 W_5}{U_5} = -\left(\frac{U_5 + c_5^2 W_5 U_2}{U_5}\right) \frac{c_4 c_3 c_2}{U_3} \\
&= -\left(\frac{a_6\left(a_5 U_3 - c_5^2 U_2\right) - c_6^2 U_3 + c_5^2 W_5 U_2}{U_5}\right) \frac{c_4 c_3 c_2}{U_3} = -\frac{\left(a_5 a_6 - c_6^2\right) c_4 c_3 c_2}{U_5} \\
&= -\frac{c_4 c_3 c_2 W_4}{U_5}, \\
e_3'' &= \frac{-c_3 c_2}{U_2} - \frac{c_4 U_1}{U_2} \frac{c_4 c_3 c_2 W_4}{U_5} = -\left(\frac{U_5 + c_4^2 W_4 U_1}{U_5}\right) \frac{c_3 c_2}{U_2} \\
&= -\left(\frac{a_6\left(a_5\left(a_4 U_2 - c_4^2 U_1\right) - c_5^2 U_2\right) - c_6^2\left(a_4 U_2 - c_4^2 U_1\right) + c_4^2 W_4 U_1}{U_5}\right) \frac{c_3 c_2}{U_2} \\
&= -\frac{\left(a_4 W_4 - c_5^2 W_5\right) c_3 c_2}{U_5} = -\frac{c_3 c_2 W_3}{U_5}, \\
e_2'' &= -\frac{c_2 W_2}{U_5}, \quad e_1'' = \frac{c_1''}{1 - c_2' e_2''} = \frac{-c_1 U_5}{\left(a_1 U_5 - c_2^2 W_2\right)} = -\frac{c_1 U_5}{W_0}.
\end{aligned}$$

Obviously,

$$e_1'' = -\frac{c_1 U_5}{W_0} = -\frac{c_1 W_1}{W_0}, \text{ and } \ e_i'' = \frac{-W_i}{U_5} \prod_{j=2}^{i} c_j \ \forall \ i \in [2, N].$$

Note that in the last equation, $U_5 = W_1$ was used. The U_i are defined as before, and for the case of N equations, the W_i are given by:

$$W_{N+1} = 0, \quad W_N = 1, \quad W_i = a_{i+1} W_{i+1} + c_{i+2}^2 W_{i+2}, \quad i = N-1, \ldots, 0.$$

This is the same recursion relation as for the V_i, but with a different starting point. One can proceed for the right-hand side in a manner similar to that for an internal processor. Doing so would continue to be straightforward but tedious.

Chapter 4 references

1. Arfken G., Mathematical Methods for Physicists, 3rd ed., (Academic Press, 1985)
2. Hofhaus, J. and Van De Velde, E., Alternating-Direction Line-Relaxation Methods on Multicomputers, *SIAM J. Sci. Comp.* **17**, 454-478 (1996)
3. Johnsson, S.L., Saad, Y., and Schultz, M.H., Alternating Direction Methods on Multiprocessors, *Siam J. Sci. Stat. Comput.* **8**, 686-700 (1987)
4. Kowalik, J.S., and Kumar, S.P., "Parallel Algorithms for Recurrence and Tridiagonal Equations", Parallel MIMD Computations: HEP Supercomputer and Its Applications, edited by J.S. Kowalik (MIT Press, 1985)
5. Krechel, A., Plum, H., and Stuben, K., Parallelization and Vectorization Aspects of the Solution of Tridiagonal Linear Systems, *Parallel Computing* **15**, 31-49 (1990)
6. Kumar S.P., Solving Tridiagonal Systems on the Butterfly Parallel Computer, *International Journal of Supercomputer Applications* **3**, 75-81 (1989)
7. Lambert, M.A., Rodrigue, G.H., and Hewett, D.W., Parallel DADI Methods for Solution of the Steady State Diffusion Equation (submitted to *Parallel Computing*, April 1996)
8. Mattor N., Williams, T.J., and Hewett, D.W., Algorithm for Solving Tridiagonal Matrix Problems in Parallel, *Parallel Computing* **21,** 1769-1782 (1995)
9. Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vertterling,W.T., Numerical Recipes (Cambridge University Press, 1986)
10. Wang H.H., A Parallel Method for Tridiagonal Equations, *ACM Transactions on Mathematical Software* **7**, 170-183 (1981)

Conclusion

Simulation results in Chapter 2 demonstrate agreement between ideal MHD stability theory and magnetized and unmagnetized Z-pinch simulations with fluid ions. Characteristic frequencies from ideal stability theory are computationally reproduced and are shown to be independent of the Hall and diagmagnetic drift terms included in the nonideal MHD equations of the pinch simulation. This has led to increased confidence in the electron/field advance used in the simulation.

Compressional pinch simulations in Chapter 2 demonstrate agreement with earlier simulations of comparable complexity, and show that ion transport requires more than a single-fluid ion theory; a particle-in-cell treatment is better for representing the multiply peaked distribution of ions in velocity space during the early compressional stages of the screw pinches.

An interpolated resistivity in the plasma at plasma-vacuum interfaces served to diminish finite-grid effects on the simulation, leading to a smoother time evolution of the interfacial electric field. Smoothing with interpolated resistivity was comparable to that achieved with earlier methods involving nonphysical density averaging. Pinch simulations also demonstrated the utility of explicit updating schemes used along with ADI for calculating B_θ and $\bar{E}$ in vacuum regions of the simulation. The explicit schemes reduced the computational work in the calculation of B_θ and $\bar{E}$, yet they yielded reasonably good simulation results.

Most importantly, the new explicit update of B_θ in entrained vacuum region, which is potentially more accurate than previous ADI methods, predicts a great amount of plasma turbulence in low density plasma regions where entrained vacuum regions might form. Results of the new update technique indicate that the amplitude of turbulence significantly adds to plasma instability, beyond manifestations of the magneto-Rayleigh-

Taylor instability seen in previous ADI simulations. Furthermore, the simulations suggest that vortices in the turbulent regions are metastable, persisting for significant fractions of the time between electrical current initiation and the first compressional stagnation. After more carefully checking the validity of the simulated turbulence, it could be quite interesting to study the vortex dynamics around entrained vacuum voids to see how it might frustrate plasma thermalization before the pinche's completely stagnate.

Attempts to parallelize the ADI method underlying computation of the electron fluid and electromagnetic fields were a great success. As used in DADI solution of the steady-state diffusion equation, the parallel ADI method was demonstrated to have scalability that will allow higher resolution pinch simulations in the future. For a fixed-size finite difference grid, the power of about 256 processors on a parallel computer such as the Cray-T3D or Meiko CS-2 can yield a speedup in execution time in the range of about 10-100, depending on the exact problem. But disregarding parallel scalability, the benefit of greatly expanded computer memory is provided by the parallel computers, making possible computations with higher resolution or more detailed physical models.

Parallelization of the current pinch algorithm is a natural next step, but it still would not yield a realistic pinch simulation; this is because of the gross physical processes left out of the plasma mathematical model. So the pinch simulation of this dissertation is still in a developmental stage. As it stands, the algorithm incorporates a 1970s model of fluid electrons with a 1980s model of PIC ions. The ADI techniques that are the backbone of the electron-field computation have origins in the 1950s, and yet the implementation of ADI on a modern parallel computer is a recent development of the 1990s. The pinch simulation is converging from all of the best technology known by the author, but it has not yet come together is a mature sense, because there are so many physical processes that can take place in plasma screw pinches.

One could start improving the simulation by using a nondiagonal tensor resistivity, instead of a scalar resistivity, to model the effects of enhanced electrical conductivity parallel to the magnetic field lines. Then, for the electron temperature to be calculated accurately, the electron energy equation could be incorporated into the simulation, with a thermal conductivity that is also enhanced along magnetic field lines. These effects have been incorporated in previous simulations, and are necessary for realistic pinch simulation. It might even be necessary to have an anisotropic electron temperature for pinches that are strongly driven. Additions such as these to the pinch simulation are achievable in the time span of a year or so.

Two other general effects that must be incorporated in realistic simulation of compressional screw pinches are radiation effects and collision effects. In the electron energy equation, brehmstrahlung and electron synchrotron radiation are reasonable effects to add, along with effects of collisional atomic/ionic excitation and photoionization. For the ions, photoionization is a large radiation effect, and this has to be incorporated any time the radiation flux is high and the plasma is only partially ionized. Any type of ion collision can directly effect the level of excitation of the ionic species in the plasma. Although excitation or ionization does not have a direct effect on the ion temperature, it can easily change the charge state of the ion, which affects the ion dynamics in the electric and magnetic fields. The collisions that directly affect the ion temperature are ion-ion collisions, both elastic and inelastic. During a radial collapse of ions onto the axis the ions can become very collisional, generating considerable heat that affects the maximally compressed state. Some heat in the ions can be dissipated through inelastic ion-ion collisions that lead to radiation from excited bound ionic states. This is a secondary ion-radiation effect that must be incorporated to properly model radiative collapse of screw pinches. Often, when collision and radiation effects become important, single fluid theory for each species of ions can break down, so that a kinetic description of each species is necessary. Basically, proper inclusion of collision and radiation effects is a difficult task from the outset. It is not clear that all of these effects have been incorporated in any single screw pinch algorithm, but it is clear that they are not easily incorporated in any pinch simulation.

The level to which the pinch simulation is finally developed depends very much on the need to understand in detail the physical phenomena inside screw pinches. Hopefully, if detailed engineering knowledge in necessary, we can finally transcend limitations of past computational methods. It is clear that many competing computer algorithms exist for these computations, and since they involve such large efforts, the political climate tends to have as much effect on development as any stubborn search for truth.