JOURNAL OF COMPUTATIONAL PHYSICS 116, 281–291 (1995)

An Inexact Newton Algorithm for Solving the Tokamak Edge Plasma Fluid Equations on a Multiply-Connected Domain

D. A. KNOLL AND P. R. MCHUGH

Computational Fluid Dynamics Unit, Idaho National Engineering Laboratory, Idaho Falls, Idaho 83415-3895

Received May 21, 1993; revised May 3, 1994

Newton's method is combined with a preconditioned conjugate gradient-like algorithm and finite volume discretization to solve the steady-state two-dimensional tokamak edge plasma fluid equations. A numerical evaluation of the Jacobian is employed. Mesh sequencing, pseudo-transient continuation, and adaptive damping are used to increase the radius of convergence. The computations are performed on a multiply-connected curvilinear geometry in a fully coupled manner. The preconditioned conjugate gradient-like algorithm is shown to have a significant storage advantage over the previously used banded Gaussian elimination, while maintaining the excellent convergence characteristics of the overall algorithm. Simulations of a high recycling divertor and a gaseous divertor on the DIII-D tokamak geometry are used to demonstrate algorithm performance.

1. INTRODUCTION

The large heat flux incident on the divertor plates in the current design of the International Thermonuclear Experimental Reactor (ITER) must be reduced [1]. The plasma outside the last closed magnetic flux surface of the tokamak controls the heat flux to the divertor plates. This region of a tokamak is referred to as the edge plasma. In order to further study, understand, and develop solutions to this problem we need the capability to model the pertinent physics and the proper geometry. Development of a robust and efficient computational tool for the two-dimensional tokamak edge plasma fluid equations that includes: high neutral recycling; fluid drifts; significant concentrations of radiating impurities; and a multiply-connected, curvilinear, non-orthogonal, geometry has proven to be a difficult task. There are many reasons for this difficulty, such as: the multiple time scales which are encountered between electron energy conduction, parallel sound speed, atomic reactions, and cross-field transport; strong nonlinear feedback between radiating impurities and neutrals, and the electron energy equation; the fact that drift flows can dominate over parallel pressure gradient driven flows; and the disparity in spatial scales between magnetic field line connection length and the ionization mean free path of a recycling neutral.

While it is not the intent of this paper to be a review article on edge plasma fluid modeling, it is constructive to describe other solution algorithms which are being applied to this system of equations and contrast them, algorithmically, with inexact Newton's method. The intent is to motivate the choice of inexact Newton's methods. In the past, the two main solution algorithms applied to the tokamak edge plasma equations were the SIMPLE (semi-implicit pressure linked equations) algorithm [2, 3] and ADI (alternating direction implicit) based algorithms [4, 5]. Both of these algorithms are based on splitting, ADI by directions and SIMPLE by conservation equations.

The SIMPLE algorithm combines the continuity and momentum equations to form an implicit pressure equation, which in theory removes the Courant limit on sound waves. The set of conservation equations is then split, and solved/relaxed in a segregated fashion. Since the edge plasma is modeled with an ideal gas equation of state and exhibits strong temperature and density gradients in the recycling region, the energy equation should be very important in determining the implicit pressure. The SIMPLE algorithm does not account for this coupling implicitly. This assumption in the SIMPLE algorithm, along with the segregated solution, is potentially the reason for its poor convergence behavior on low Mach number compressible reacting flow problems with large temperature and density gradients, such as those seen in diffusion flames [6]. Other pressure equation-based semi-implicit algorithms, applied to reacting flow problems, have included information about the energy equation in the pressure equation in order to improve algorithm performance [7].

In contrast to the SIMPLE algorithm the ADI algorithm splits in the grid direction and not by equation. For a scalar, elliptic, 2D equation on a rectangle there are a series of pseudo-time steps that should be cycled through for optimum convergence behavior [4, 8]. These time steps depend on the bounds of the eigenvalues of the two matrices which arise from the splitting [4, 8]. The edge plasma is a problem with widely varying time scales, and the transport along and across the magnetic field have very different propagation speeds. This physics leads to widely varying eigenvalues between and within the two matrices arising from the splitting, and thus it will be difficult to obtain strong convergence behavior out of an ADI algorithm. Another example where this problem arises is in solving the low Mach number compressible Navier–Stokes equations. In

0021-9991/95 $6.00

this case, a large disparity exists between pressure wave propagation and fluid velocity. ADI based algorithms have difficulty with this problem since the eigenvalues vary over a large range and a single pseudo-time step cannot be optimum [9]. It should be mentioned that research has been performed on dynamic ADI (DADI) [10], where a variety of time steps are cycled through adaptively.

The SIMPLE and ADI algorithms make assumptions which were intended to make solutions of large systems of equations tractable, but at the same time which compromised convergence characteristics. Consequently, the removal of splitting, either by equation or direction, will lead to a more robust solution algorithm. For this type of algorithm to be practical, the solution of the resulting large linear system must be made tractable. It is extremely desirable to have Newton's method for linearization in a fully implicit algorithm due to its quadratic convergence behavior. This property also tends to remove ambiguity from the test on convergence.

Because of the multiple time scales and strong nonlinear coupling between equations, fully implicit Newton's methods were first investigated by Knoll *et al.* for this problem [11, 12]. This research used an analytically derived Jacobian with banded Gaussian elimination and mesh sequencing to provide a robust and CPU efficient solution algorithm for a simply-connected Cartesian geometry. This work demonstrated the viability of a fully implicit solution of the edge plasma equations. Several improvements to this original algorithm are described in this paper. The analytic Jacobian evaluation has been replaced by an efficient numerical Jacobian evaluation [13, 14]. The simply-connected Cartesian geometry has been replaced by a more realistic multiply-connected, orthogonal, curvilinear geometry. The multiply-connected geometry poses additional matrix storage concerns and has motivated the switch from a banded Gaussian elimination matrix solution algorithm to a preconditioned conjugate gradient-like iterative algorithm, namely, the transpose-free quasi-minimal residual (TFQMR) algorithm [15]. The degree of required convergence of this inner linear iteration is related to the level of convergence of the outer Newton iteration. This overall algorithm is referred to as an inexact Newton's method [16, 17]. In addition, a pseudo-transient continuation algorithm has been implemented to improve robustness.

It is important to acknowledge other successful applications of Newton's method to the edge plasma fluid equations with more complicated physics. Campbell [18] has applied Newton's method to the one-dimensional along-the-field equations with multifluid radiating impurities. Rognlien *et al.* [19] have applied Newton's method to the two-dimensional equations and included cross-field fluid drifts. Also, there has been recent progress in finite element modeling of the edge plasma using Newton's method [20, 21].

The remainder of the paper is organized as follows: Section 2 gives a brief description of the edge plasma geometry and fluid equations. The origin of the multiply-connected geometry will also be described. The numerical solution algorithm will be described in Section 3, with special attention given to the preconditioned conjugate gradient-like inner iteration. In Section 4 we will present algorithm performance results obtained in simulating a high recycling divertor and a gaseous divertor using a DIII-D tokamak geometry. Conclusions and future research are discussed in Section 5.

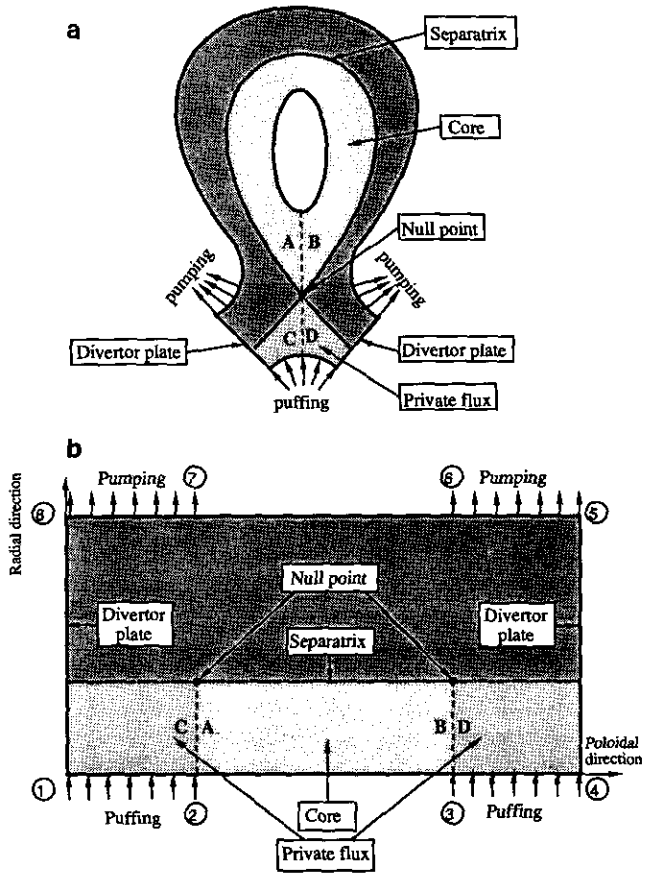

FIG. 1. (a) Physical domain; (b) computational domain.

2. EDGE PLASMA GEOMETRY AND FLUID EQUATIONS

A schematic of the poloidal cross section of a single null divertor tokamak is shown in Fig. 1a and the representative computational geometry is shown in Fig. 1b. The two directions, poloidal and radial, in the physical domain correspond to x and y in the computational domain. The region outside the separatrix is the edge plasma. The region inside the separatrix and above the null point is the core, and the region below the null point is called the private flux region. To accurately model the edge plasma and to minimize the impact of boundary conditions we include the private flux region and part of the core region in our computation. The finite volume grid, which is generated from the equilibrium magnetic field structure, is constructed such that one set of grid lines (x) are parallel to surfaces of

constant magnetic flux in the poloidal plane, and the other set of grid lines (y) are orthogonal to the first set [22, 23]. When this grid is represented in the computational domain using 2D data structures we have a multiply-connected geometry [13]. This means that points which are nearest neighbors in physical space are not always nearest neighbors in computational space. The effects of the multiply-connected geometry on the Jacobian matrix structure will be discussed in Section 3.

The set of edge plasma fluid equations that we will solve in this paper is a simplified form of the Braginskii equations [24]. Ambipolar flow and purely diffusive radial transport are assumed, and the effects of viscous heating have been neglected. In addition, the neutrals that are generated at the divertor plate are assumed to be atoms and are modeled with a single energy diffusion equation [25]. The governing equations, with x representing the poloidal direction and y the radial direction, are

- continuity,

$$\frac{\partial n}{\partial t} + \frac{\partial nu}{\partial x} + \frac{\partial nv}{\partial y} = nn_0\langle\sigma\nu\rangle_{ie} - nn\langle\sigma\nu\rangle_{rec}; \tag{1}$$

- neutral diffusion,

$$\frac{\partial n_0}{\partial t} + \frac{\partial}{\partial x}\left(-Dn_0\frac{\partial n_0}{\partial x}\right) + \frac{\partial}{\partial y}\left(-Dn_0\frac{\partial n_0}{\partial y}\right) = nn\langle\sigma\nu\rangle_{rec} - nn_0\langle\sigma\nu\rangle_{ie}; \tag{2}$$

- parallel momentum,

$$\frac{\partial}{\partial t}(mnu_\parallel) + \frac{\partial}{\partial x}\left(mnuu_\parallel - \eta x\frac{\partial u_\parallel}{\partial x}\right) + \frac{\partial}{\partial y}\left(mnvu_\parallel - \eta_y\frac{\partial u_\parallel}{\partial y}\right) = -\frac{Bx}{B}\left(\frac{\partial P_i}{\partial x} + \frac{\partial P_e}{\partial x}\right); \tag{3}$$

- ion internal energy,

$$\frac{\partial}{\partial t}\left(\frac{3}{2}nT_i\right) + \frac{\partial}{\partial x}\left(\frac{3}{2}nuT_i - \kappa_x^i\frac{\partial T_i}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{3}{2}nvT_i - \kappa_y^i\frac{\partial T_i}{\partial y}\right) = -P_i\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) + \kappa_{eq}(T_e - T_i) + \varepsilon_i nn_0\langle\sigma\nu\rangle_{ie}; \tag{4}$$

- electron internal energy,

$$\frac{\partial}{\partial t}\left(\frac{3}{2}nT_e\right) + \frac{\partial}{\partial x}\left(\frac{3}{2}nuT_e - \kappa_x^e\frac{\partial T_e}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{3}{2}nvT_e - \kappa_y^e\frac{\partial T_e}{\partial y}\right) = -P_e\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) - \kappa_{eq}(T_e - T_i) - \varepsilon_e nn_0\langle\sigma\nu\rangle_{ie}; \tag{5}$$

- equations of state,

$$P_i = nT_i, \quad P_e = nT_e; \tag{6}$$

- radial velocity,

$$v = -\frac{D_\perp}{n}\frac{\partial n}{\partial y}; \tag{7}$$

- poloidal velocity,

$$u = \frac{B_x}{B}u_\parallel. \tag{8}$$

In these equations n is the plasma density, n_0 is the neutral density, and $u_\parallel$ is the velocity along the total magnetic field, B. T_i is the ion temperature, T_e is the electron temperature, and B_x/B is the magnetic field pitch. In this model, the poloidal transport coefficients are classical Braginskii [24] and the radial transport coefficients are anomalous, i.e., empirical. The right-hand side of Eq. (3) represents the combination of the ion pressure gradient and acceleration due to the ambipolar electric field. The electron and ion poloidal thermal conductivities are flux limited in the standard manner [11, 12] to bridge the gap between continuum and kinetic behavior. The neutral diffusion coefficient is developed from diffusion theory assuming thermal equilibrium between neutral atoms and plasma ions [25] and is a function of n, T_i, and T_e.

The source and sink terms due to ionization and recombination depend on the electron impact ionization rate, $\langle\sigma\nu\rangle_{ie}$, and the recombination rate, $\langle\sigma\nu\rangle_{rec}$, which are both table look-up functions of T_e and n. The electron energy loss per ionization, ε_e, is a table look-up function of T_e and n. These rate tables have been produced by ADPAK [26] and include collisional radiative corrections [27]. The closure statements for radial and poloidal velocities do not include the contributions from cross-field drifts. The inclusion of these terms is part of our ongoing research effort.

3. NUMERICAL SOLUTION ALGORITHM

The inexact Newton algorithm with mesh sequencing is characterized by three nested loops. The outer most loop progresses through a series of continually refined grids. The next level is the Newton iteration, and the inner most loop is the preconditioned conjugate gradient-like iteration, which solves the linearized problem.

3.1. *Inexact Newton's Method*

The Newton–Raphson method is a robust technique for solving systems of nonlinear equations of the form

$$\mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x})]^T = 0, \tag{9}$$

where the vector of state variables, $\mathbf{x}$, can be expressed as

$$\mathbf{x} = [x_1, x_2, ..., x_n]^T. \tag{10}$$

Application of the Newton–Raphson method requires the solution of the linear system,

$$\mathbf{J}^n \, \delta\mathbf{x}^n = -\mathbf{F}(\mathbf{x}^n), \tag{11}$$

where the elements of the Jacobian, $\mathbf{J}$, are defined by

$$J_{ij} = \frac{\partial f_i}{\partial x_j} \tag{12}$$

and the new solution approximation is obtained from

$$\mathbf{x}^{n+1} = \mathbf{x}^n + s \, \delta\mathbf{x}^n. \tag{13}$$

This iteration is continued until the norm of $\delta\mathbf{x}$ and/or the norm of $\mathbf{F}(\mathbf{x})$ are below some suitable tolerance level. The scalar, s, is used to damp the update. The damping strategy is designed to prevent the calculation of negative thermodynamic variables and to scale large variable updates when the solution is far from the true solution [11, 12].

Since the use of an iterative technique to solve Eq. (11) does not require the exact solution of the linear system, the resulting algorithm is labeled an "inexact" Newton's method [16, 17]. This feature is advantageous in the sense that the tolerance of the linear equation solve can be relaxed when far from the true solution and tightened as the true solution is approached. In other words, the convergence criterion for the inner linear iteration is related to the current convergence status of the outer Newton iteration. This behavior is controlled using an inner iteration convergence criterion similar to that proposed by Averick and Ortega [16] and Dembo [17]. Specifically, the inner TFQMR iteration is assumed converged when

$$\frac{\|J^n \, \delta\mathbf{x}^n + \mathbf{F}(\mathbf{x}^n)\|_2}{\|\mathbf{F}(\mathbf{x}^n)\|_2} < \gamma_n, \tag{14}$$

The selection of the best value of γ_n is highly empirical. We have found that $\gamma_n = 5 \times 10^{-3}$ works well in practice for the edge plasma equations.

3.2. *Numerical Jacobian*

The elements of the Jacobian in Eq. (12) are evaluated numerically using finite difference approximations,

$$J_{ij} = \frac{f_i(x_1, x_2, ..., x_j + \Delta x_j, ..., x_n) - f_i(x_1, x_2, ..., x_n)}{\Delta x_j} \tag{15}$$

where

$$\Delta x_j = \alpha x_j + \beta \tag{16}$$

and α and β are small perturbation constants.

3.3. *Preconditioned TFQMR Algorithm*

The desire to extend the original solution algorithm to handle multiply-connected domains and two-dimensional multifluid radiating impurities motivated the switch from a banded Gaussian elimination solver to an advanced iterative algorithm for the solution of Eq. (11). The use of a banded Gaussian elimination in solving these types of problems would be very costly, both in terms of CPU time and memory requirements due to the large bandwidth.

An alternative to the use of a direct linear solver is the use of true conjugate gradient methods to solve linear systems of the form $\mathbf{Ax} = \mathbf{b}$. These methods compute approximations to $\mathbf{x}$ in the affine space $\mathbf{x}_0 + \mathbf{K}_m$, where $\mathbf{K}_m$ is the Krylov subspace of dimension m [28]. They are characterized by an optimality condition (minimization of some error norm) and economical or short vector recurrences [29]. Note that for symmetric matrices, short vector recurrences arise naturally, resulting in constant work and storage requirements on each iteration. For nonsymmetric matrices, however, short recurrences do not exist [30] and so the work and storage requirements increase with the iteration number, making the use of true conjugate gradient methods impractical for large problems.

In some instances true conjugate gradient methods may be successfully applied to the normal equations (i.e., $\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{b}$). Disadvantages in this approach, however, are that the condition number of the new system is made much worse, and matrix vector multiplications with $\mathbf{A}^T$ are required. Working with $\mathbf{A}^T$ is undesirable for several reasons: first, the transpose is not always readily available; second, the efficiency of matrix vector multiplications with the transpose may be reduced on vector/parallel computers; and third, working with the transpose eliminates the option of matrix-free implementations of Newton's method [31–33].

The problems associated with the application of true conjugate gradient methods have motivated the development of conjugate gradient-like methods. These methods are derived by relaxing either the optimality condition or giving up short vector recurrences [34]. The unrestarted GMRES algorithm is an example where optimality is obtained by sacrificing economical vector recursions. Conversely, short vector recurrences can be obtained at the expense of optimality by allowing periodic algorithm restarts (i.e., GMRES(k) [35]), by artificially truncating the recursion (i.e., the new direction vector is orthogonal to only the previous s direction vectors), or by using the nonsymmetric Lanczos biorthogonalization procedure (i.e., using three-term recursions to build a pair of biorthogonal bases) [34]. The development and investigation of these conjugate gradient-like algorithms continues to be an active research area.

The transpose-free quasi-minimal residual algorithm

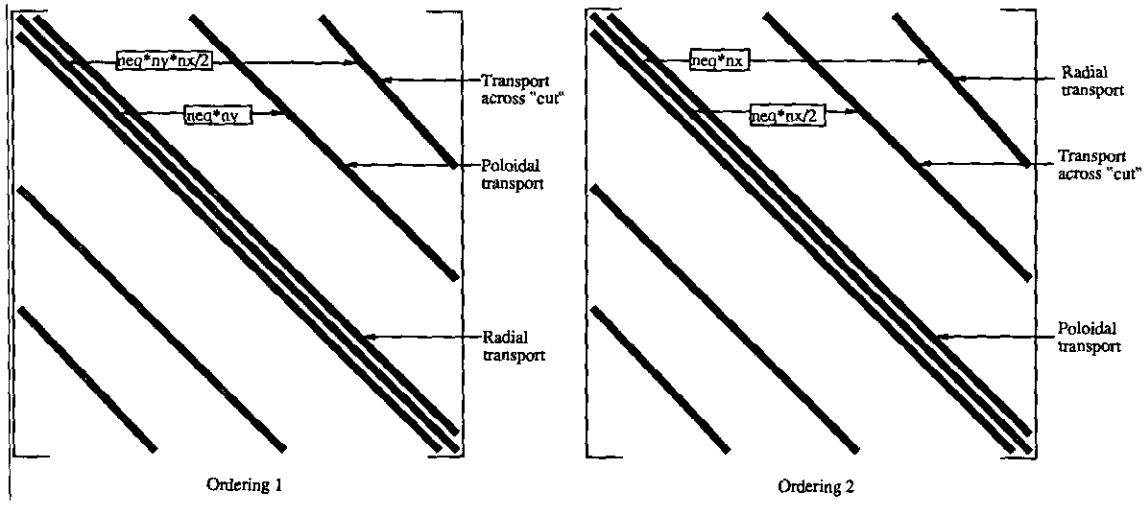

FIG. 2. Jacobian matrix structure from multiply connected domain.

(TFQMR) of Freund [15] is a Lanczos-based conjugate gradient-like algorithm that we have implemented to take advantage of the sparse structure of the Jacobian matrix. In order to accelerate the convergence of the TFQMR algorithm, we currently use incomplete lower–upper (ILU) factorization as a right preconditioner with no fill-in allowed (i.e., **L**, **U** have same sparsity pattern as **J** $\Rightarrow$ ILU(0)] [36, 37]. We have investigated both higher levels of fill-in (ILU)(k)), as well as less costly classical preconditioners such as Jacobi, Gauss–Sidel, symmetric Gauss–Sidel, SOR, and symmetric SOR. To date, ILU(0) has provided the best compromise between CPU performance and preconditioner effectiveness for this problem when solving the steady state equations. We are currently studying the use of block ILU preconditioners and domain-based preconditioners.

The structure of our Jacobian matrix, resulting from the multiply-connected domain, deserves further discussion. This structure will vary depending on whether i, the poloidal grid index, or j, the radial grid index is the fastest running index when we construct the Jacobian. When banded Gaussian elimination was used as the matrix solver (before the implementation of the multiply-connected domain) the Jacobian was constructed with j the fastest running index (ordering 1). This was done since the number of grid points in the y direction, ny, was always less than the number in the x direction, nx, and this choice generated the minimum bandwidth matrix, which led to reduced memory and CPU requirements. For the multiply-connected domain with a banded solver, it is advantageous to order with i the fastest running index (ordering 2). The banded structure of the Jacobian matrix on a multiply connected domain is shown in Fig. 2 for both orderings. Here we can see that the transport across the cuts in the computational domain, Fig. 1b, can significantly increase the matrix bandwidth. This has been one of the prime motivations to move from a banded direct solver to a preconditioned conjugate gradient-like solver. The storage requirement for our current implementation of ILU(0) TFQMR is twice the number of nonzero diagonals times the dimension of the matrix. The storage requirements for the two different orderings are compared to that of banded Gaussian elimination in Table I as a function of grid dimension. In this table, and Fig. 2, we have set the number of poloidal grid points between each plate and the null point to one quarter of the total. The memory savings is clear. The motivation for not using ILU(1), or higher, is the rapid increase in the number of nonzero diagonals in going from ILU(0) to ILU(1) for our multiply-connected domain. For the five equation system on an orthogonal, staggered grid ILU(0) requires 65 diagonals and ILU(1) requires 240 diagonals. While ILU(1) is a better preconditioner and reduces the required number of TFQMR iterations, the extra cost in forming ILU(1) and using it in forward/backward solves actually increases the CPU cost. However, in the application of inexact Newton methods on simply connected geometries, we have observed CPU savings using ILU(1) or ILU(2) preconditioning [14].

The ILU(0) right-preconditioned TFQMR algorithm is presented below for completeness, where $(\mathbf{Pr}) = (\mathbf{LU})^{-1}$ represents the inverse of the preconditioning matrix. The algorithm solves the system $\mathbf{Ax} = \mathbf{b}$, where **A** represents the Jacobian matrix (**J**), **x** is the Newton update (δ**x**), and **b** is the Newton residual

TABLE I

Linear System Solution Memory Requirement in Megawords

Grid size	ILU(0)	LINPACK	
nx, ny	TFQMR	Ordering 1	Ordering 2
32, 8	0.17	2.4	0.61
64, 16	0.67	39.3	5.0
128, 32	2.67	629	39.3

$(-\mathbf{F}(\mathbf{x}))$. All other new vectors and scalars are self-defined within the algorithm.

(I) *Initialize*

1. *Choose* $\mathbf{x}_0$
2. *Set:*

$$\mathbf{r}_0^{CGS} = \tilde{\mathbf{r}}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0,\ \mathbf{q}_0 = \mathbf{p}_{-1} = \mathbf{d}_0 = \mathbf{0},$$
$$\nu_0 = \eta_0 = n = 0;$$
$$\rho_{-1} = 1,\ \tau_0 = \|\mathbf{r}_0^{CGS}\|;\ (\|\cdot\| \textit{ denotes Euclidean norm})$$

(II) *For* $n = 0, 1, 2, \ldots$, *do: (*n *is not the Newton iteration count)*

1. *Set:*

$$\rho_n = \tilde{\mathbf{r}}_0^T \mathbf{r}_n^{CGS},\ \beta_n = \frac{\rho_n}{\rho_{n-1}},\ \mathbf{u}_n = \mathbf{r}_n^{CGS} + \beta_n \mathbf{q}_n;$$

$$\mathbf{p}_n = \mathbf{u}_n + \beta_n(\mathbf{q}_n + \beta_n \mathbf{p}_{n-1}),\ \mathbf{v}_n = \mathbf{A}(\mathbf{P}_r)\mathbf{p}_n,$$

$$\sigma_n = \tilde{\mathbf{r}}_0^T \mathbf{v}_n,\ \alpha_n = \frac{\rho_n}{\sigma_n};$$

$$\mathbf{q}_{n+1} = \mathbf{u}_n - \alpha_n \mathbf{v}_n,\ \mathbf{v}_n = \alpha_n(\mathbf{P}_r)(\mathbf{u}_n + \mathbf{q}_{n+1}),$$

$$\mathbf{r}_{n+1}^{CGS} = \mathbf{r}_n^{CGS} - \mathbf{A}\mathbf{v}_n;$$

2. *For* $m = 2n + 1, 2n + 2$ *do:*

$$\omega_{m+1} = \left\{ \frac{\|\mathbf{r}_{n+1}^{CGS}\|}{\sqrt{\|\mathbf{r}_n^{CGS}\| \cdot \|\mathbf{r}_{n+1}^{CGS}\|}} \quad \begin{matrix} \textit{if } (m+1) \textit{ is odd} \\ \textit{if } (m+1) \textit{ is even} \end{matrix} \right\},$$

$$\nu_m = \frac{\omega_{m+1}}{\tau_{m-1}},$$

$$c_m = \frac{1}{\sqrt{1 + \nu_m^2}},\ \tau_m = \tau_{m-1}\nu_m c_m,\ \eta_m = c_m^2 \alpha_n;$$

$$\mathbf{y}_m = \left\{ \begin{matrix} \mathbf{u}_n \textit{ if } m \textit{ is odd} \\ \mathbf{q}_n \textit{ if } m \textit{ is even} \end{matrix} \right\},$$

$$\mathbf{d}_m = \mathbf{y}_m + \frac{\nu_{m-1}^2 \eta_{m-1}}{\alpha_n} \mathbf{d}_{m-1};$$

$$\mathbf{x}_m = \mathbf{x}_{m-1} + \eta_m(\mathbf{P}_r)\mathbf{d}_m;$$

Continue until $\mathbf{x}_m$ *has converged*

3.4. *Mesh Sequencing*

Mesh sequencing is used to obtain an initial guess on the finest grid which lies within the radius of convergence of Newton's method. Mesh sequencing is analogous to the first upward cycle of a Full Multigrid (FMG) algorithm [38]. We use Lagrangian interpolation polynomials to move through a series of predefined grids, and the interpolated solution from the previous grid is used as the initial guess on the new grid. We have found that with the edge plasma equations and geometry the accuracy of the interpolation is very sensitive to the interpolation direction. Because of the strong coupling (i.e., transport) along the magnetic field, we interpolate in this direction first.

3.5. *Pseudo-transient Continuation*

Another method for improving the initial guess for Newton's method is to use an elementary continuation procedure. We have implemented the switched evolution relaxation algorithm (SER) [39]. This technique involves including artificial transient terms in the governing equations, specifically, time derivatives of the principal variables. Implementation of this algorithm requires the introduction of a false time step, Δt, in the discretized equations. Δt must be chosen sufficiently small to ensure convergence, yet large enough to obtain efficient steady state calculations. We typically use $\Delta t = 1.0\text{e}{-6}$ s which should resolve ion sound speed but not electron conduction. In our algorithm, Δt is chosen adaptively as

$$\Delta t_k = \eta \frac{\|\mathbf{F}(\mathbf{x}^0)\|_\infty \Delta t_0}{\|\mathbf{F}(\mathbf{x}^k)\|_\infty}, \tag{17}$$

where the $\mathbf{F}(\mathbf{x})$ represent the steady state equations, not the transient equations. The initial time step, Δt_0, is set at the usual explicit stability limits. The time step is further controlled by the scaling parameter, η, which typically is taken to be 10. Equation (17) forces Δt to be small when the transient is far from steady state, but allows it to increase rapidly as steady state is approached. Use of this algorithm requires tracking and monitoring the residuals of the steady equations throughout the pseudo-transient.

4. ALGORITHM PERFORMANCE

In order to demonstrate algorithm performance we model both a high recycling divertor and a gaseous divertor using a grid which represents the DIII-D tokamak at General Atomics in San Diego, CA. A high recycling divertor is one in which the majority (>95%) of the ion flux to the divertor plates is recycled as a neutral source. A gaseous divertor is created by an additional source of neutral gas injected into the divertor region. The purpose of this injected gas is to extract energy from the electrons through atomic line radiation and from the ions through charge exchange. We use te same three grid sequence for both problems; 32×8, 64×16, and 128×32. The 64×16 grid is shown in Fig. 3.

We refer to the computational geometry shown in Fig. 1 to describe the boundary conditions for the high recycling divertor simulation:

(1) Core (2-3)
- $T_i = T_e = 150$ eV, $n = 2 \times 10^{19}$ m^{-3}, $u_\parallel = 0$, $\partial n_0/\partial y = 0$;

(2) Private flux (3-4 and 1-2)

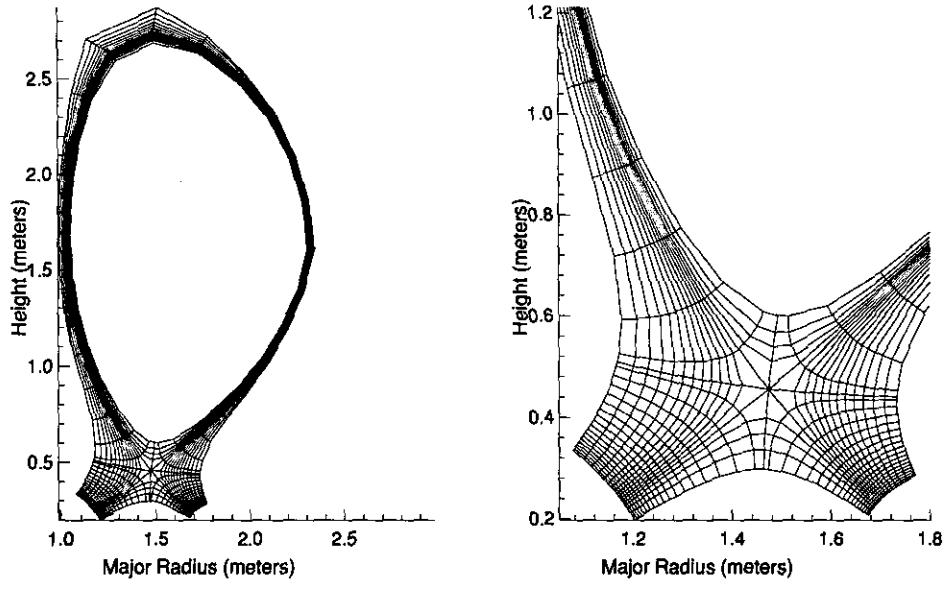

FIG. 3. 64 × 16 DIII-D grid.

- $\partial T_i/\partial y = \partial T_e/\partial y = \partial n/\partial y = \partial_0/\partial y = \partial u_\parallel/\partial y = 0$;

(3) Divertor plate (4-5 and 8-1)
- $u_\parallel = \sqrt{(T_i + T_e)/(m_i + m_e)} = C_s$ (subsonic), $\partial u_\parallel/\partial x = 0$ (supersonic)
- $D_{n_0}(\partial n_0/\partial x) = R(B_x/B)n\, u_\parallel$, $R = 0.98$
- $Q_i = \delta_i(B_x/B)u_\parallel\, nT_i$, $Q_e = \delta_e(B_x/B)u_\parallel\, nT_e$; $\delta_i = 2.5$; $\delta_e = 5.5$
- with $Q = \frac{5}{2}(B_x/B)u_\parallel\, nT - \kappa_x(\partial T/\partial x)$;

(4) Wall (5-8)
- $\partial T_i/\partial y = \partial T_e/\partial y = \partial n/\partial y = \partial u_\parallel/\partial y = 0$
- $-D_{n_0}(\partial n_0/\partial y) = \frac{1}{2}(1 - \gamma)/(1 + \gamma)\sqrt{T_i/m_i}$,

$$\gamma = \begin{cases} 1 & \text{for 6-7} \\ 0.95 & \text{for 5-6, 7-8.} \end{cases}$$

The boundary conditions at the divertor plate are the most complicated and controversial [40–42]. In this paper we use the standard boundary condition arrived at from plasma sheath theory [43]. Here, the neutral flux is assumed to be some fraction of the plasma flux flowing in the opposite direction. This corresponds to ions neutralizing at the plate. The albedo boundary condition on the wall for neutrals is intended to simulate neutral pumping from the machine.

We continue to use standard five-point finite volume differencing on a staggered grid with thermodynamic variables at cell centers and velocities at cell faces. Upwinding is used for convective terms. We are in the process of studying higher order upwinding schemes in the poloidal direction to more accurately resolve the sharp fronts that occur in a radiative/gaseous divertor. Also, in most experiments, the poloidal magnetic field lines do not intersect the divertor plate at an orthogonal angle. Consequently, we are developing a nonorthogonal finite volume stencil and have generated some initial results [44] which will be the subject of another paper.

Figure 4 shows the convergence behavior for this problem on the three-grid sequence. In this case the steady-state equations were solved directly and SER was not used. The benefit of doing the early iterations on a course grid are clearly demonstrated. The total CPU time for this run on a Cray-C90 was 230 s. Approximately 30% of this time was spent forming the Jacobian and 70% was spent solving the linear problem. Approximately one-third of the linear solve time was spent forming the preconditioning matrix. On average, the number

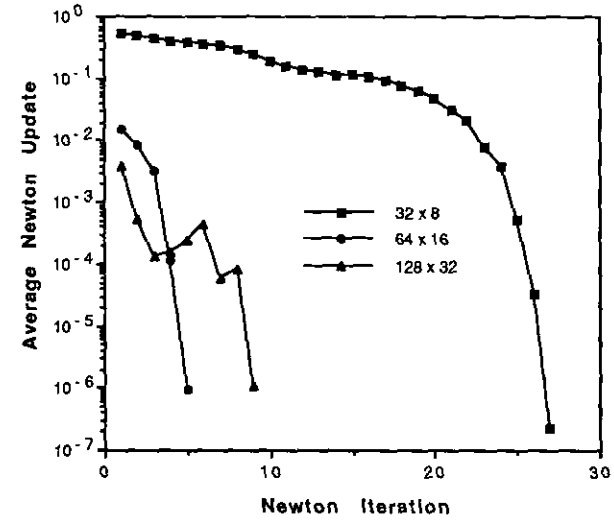

FIG. 4. Average Newton update on three grid sequence for high recycling divertor simulation.

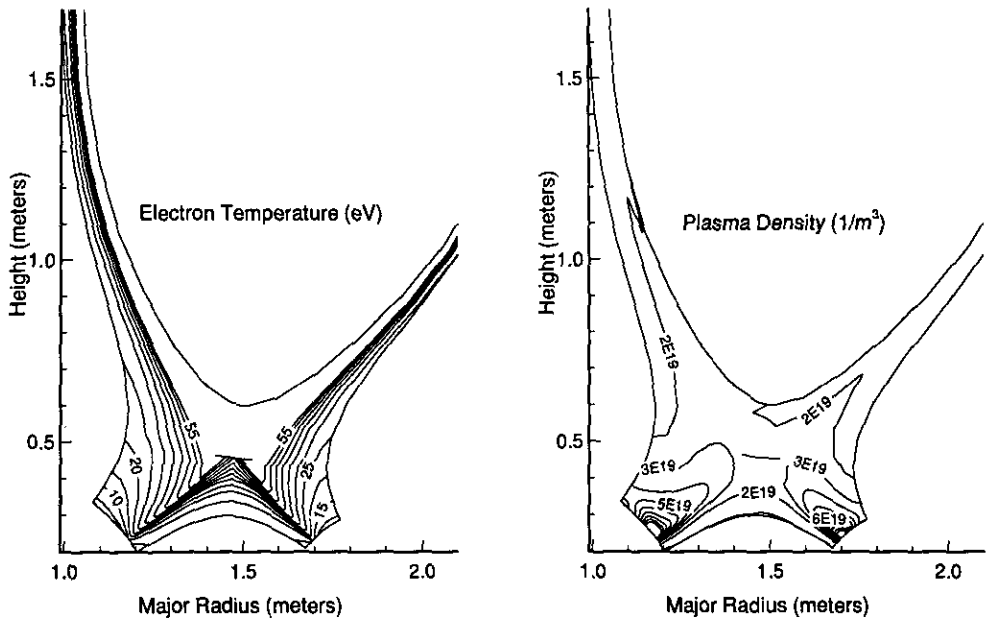

FIG. 5. Electron temperature and plasma density for high recycling simulation.

of TFQMR iterations required per Newton iteration on the 32×8, 64×16, and 128×32 grids were 15, 30, and 60, respectively.

Figure 5 shows the contour plots of electron temperature and plasma density for this problem. The short horizontal line near the null point is a function of the plotting package. We can see the drop in temperature and the increase in density as we move towards the divertor plates poloidally, which characterizes a high recycling divertor solution. We also see the sharp radial gradients in temperature between the private flux region and the divertor region. Figure 6 contains poloidal line plots of ion and electron temperature, parallel Mach number, and plasma and neutral density, all just outside the separatrix on the outboard side. The null point is at $x = 2.7$ and the right boundary

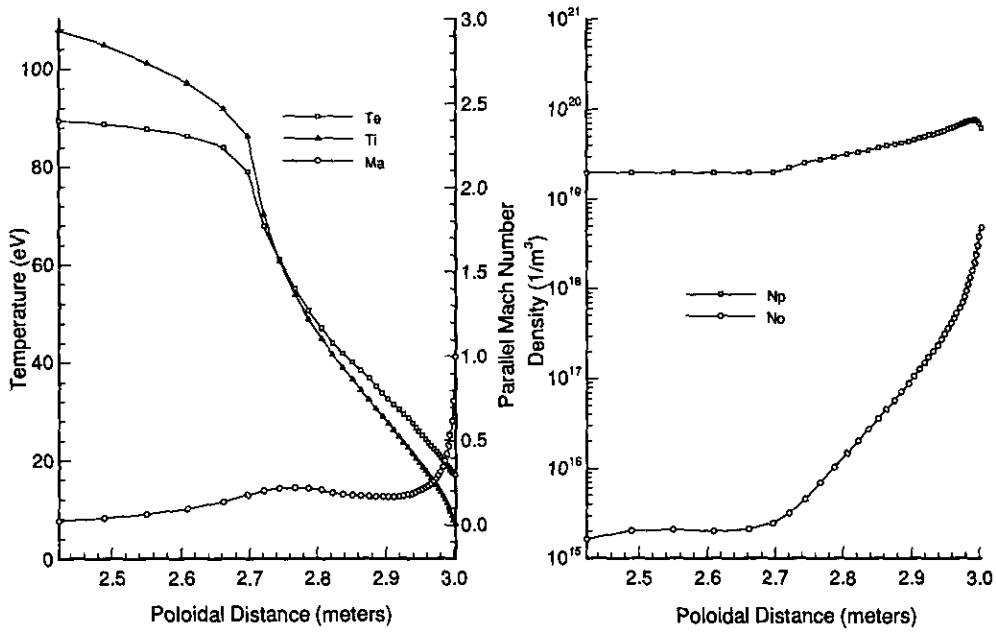

FIG. 6. Electron (T_e) and ion (T_i) temperatures, parallel Mach number (Ma), plasma (N_p) and neutral (N_0) densities along separatrix in outboard divertor region for high recycling simulation.

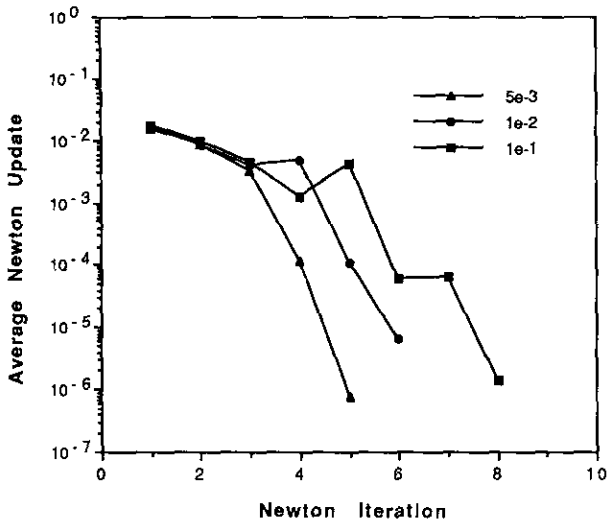

FIG. 7. Average Newton update on 64 × 16 grid for various inexact convergence parameters.

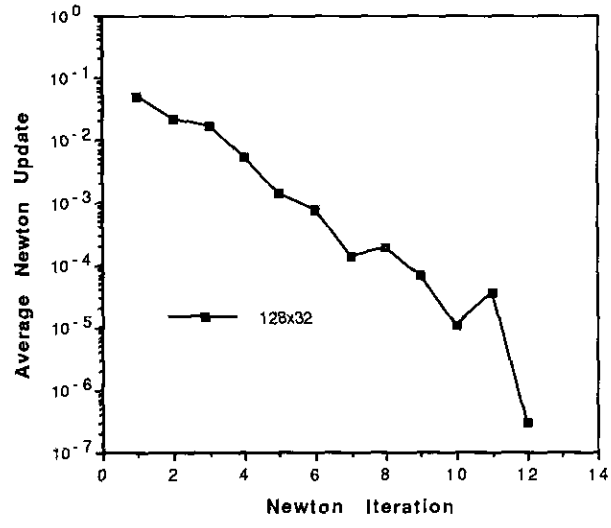

FIG. 8. Average Newton update on 128 × 32 grid for gas puffing simulation.

of the plots is the divertor plate. These plots show the density buildup and the temperature drop after passing the null point. We also see the acceleration to sonic flow right at the plate. Even at its highest value, the neutral density is significantly below the plasma density. This solution has 3.45 MW and 6.0e-21 particles per second crossing the separatrix; 0.85 MW were radiated out of the problem with the remainder of the power going to the plates. On the outboard plate, the peak electron temperature was 18 eV, the peak plasma density was 1.1e-20m^{-3}, and the peak heat flux was 2.5 Mw/m^2.

The convergence results in Fig. 4 were generated using γ_n = 5.0e-3 in Eq. (14). Figure 7 compares the convergence behavior of this choice with results using γ_n = 1.0e-2 and 1.0e-1 on the 64 × 16 grid. We can see the effects of increasing γ_n: the increased number of Newton iterations and the loss of monotonic convergence. Table II shows the effect of varying γ_n on required TFQMR iterations and relative CPU times. This problem appears relatively insensitive to the choice of γ_n with regard to CPU time.

Recently, much attention has been given to the gaseous/radiative divertor concept as a means to reduce the heat flux at the divertor plates [45]. Experiments have been performed on a number of tokamaks [46, 47] to further investigate these ideas. Here, we simulate deuterium gas puffing into the first model problem in order to further demonstrate the performance of this algorithm. The same boundary conditions and grid are used, except that the neutral equation, Eq. (2), has an inward directed flux boundary condition along the 1-2 and 3-4 segments of Fig. 1b. This simulates deuterium puffing from the floor of the machine as shown in Fig. 1a. For this problem a deuterium atom flux of 3.0e-22 s^{-1} is distributed uniformly over segments 1-2 and 3-4. This is approximately three times the puff rate that has been used in DIII-D experiments and is intended solely to demonstrate the capability of the algorithm to converge on cold plasma solutions with an "ionization front." The addition of this gas puffing makes the problem more difficult and a solution for solving the steady state equations directly is not possible without a very good initial guess. We use SER on the first two grids and then solve the steady-state equations on the fine grid. Figure 8 shows the convergence behavior on the 128 × 32 grid. On the 64 × 16 grid the SER algorithm needed 66 total Newton iterations, and 2172 TFQMR iterations. Figure 9 shows the contour plots of the electron temperature and plasma density for this solution. The effect of puffing neutral gas is clearly noticeable. The electrons give up energy to ionize the neutral gas. This causes the observed drop in electron temperature and the increase in plasma density. The energy lost by the electrons through radiation is assumed a loss in the problem; although for ITER-like divertor conditions this may not be a good assumption [48], since much of the resonance radiation may be reabsorbed. Figure 10 shows line plots in the poloidal direction for the outboard divertor leg. Here we see supersonic flow and a significant increase in the neutral density fraction near the plate. The declining plasma density near the plate is due to volume recombination which becomes important

TABLE II

Effect of γ_n in Eq. (14) on a 64 × 16 Grid Solution

γ_n	Total Newton iterations	Total TFQMR iterations	Relative CPU time
5.0e-3	5	147	1.0
1.0e-2	6	146	1.06
1.0e-1	8	116	1.07

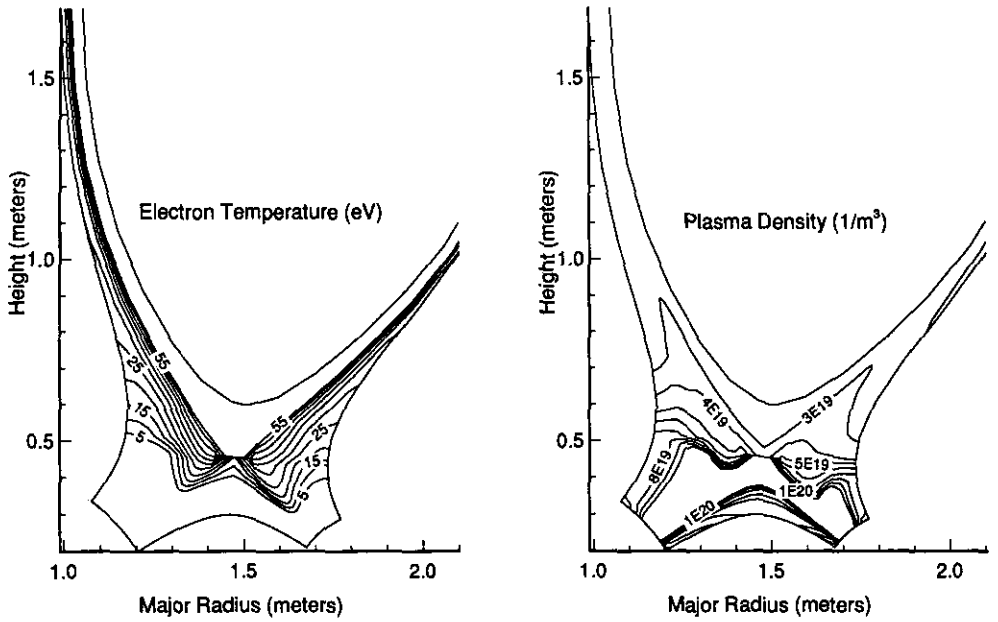

FIG. 9. Electron temperature and plasma density for gas puffing simulation.

below 1 eV. The "ionization front" can be clearly seen in the temperature profiles. This solution had 3.2 MW crossing the separatrix, and 1.1e-22 particles flowing back into the core; 2.3 MW were radiated out of the problem with the remainder of the power going to the plates. On the outboard plate, the peak electron temperature was 0.4 eV, the peak plasma density was 2.25e-20 m^{-3}, and the peak heat flux was 0.1 Mw/m^2.

5. CONCLUSIONS

Significant improvements have been added to the original Newton solution algorithm for the edge plasma fluid equations described in [11, 12]. These improvements include the use of an efficient numerical evaluation of the Jacobian and an advanced iterative solution of the linear system. This new algorithm pro-

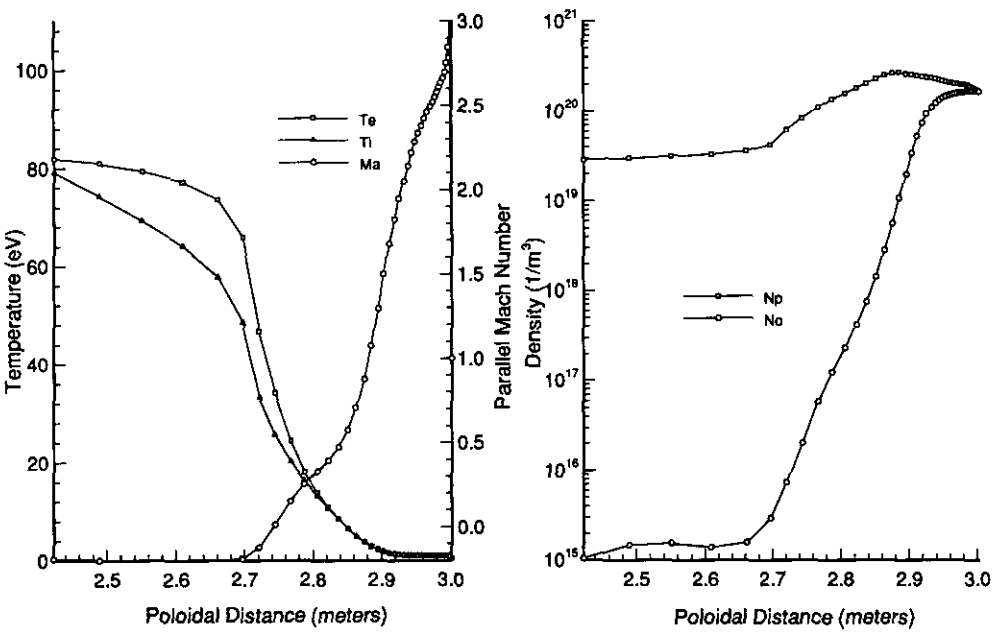

FIG. 10. Electron (T_e) and ion (T_i) temperatures, parallel Mach number (Ma), plasma (N_p) and neutral (N_0) densities along separatrix in outboard divertor region for gas puffing simulation.

vides a robust and efficient method for calculating fully implicit solutions on a multiply-connected domain and retains the unambiguous convergence characteristics. This will allow the inclusion of $\mathbf{E} \times \mathbf{B}$ and diamagnetic fluid drifts on grids of sufficient resolution. It will also allow progression to 2D multifluid radiating impurities. In addition, there is a growing desire to perform transient simulations of the edge plasma. To this end we are studying the performance of less costly preconditioners as well as the use of matrix-free methods [32, 49] for the efficient solution of the transient equations.

ACKNOWLEDGMENTS

Work was supported through the EG&G Idaho Laboratory Directed Research and Development Program under DOE Idaho Field Office Contract DE-AC07-76ID01570.

REFERENCES

1. S. A. Cohen *et al., J. Nucl. Mater.* **196-198,** 50 (1992).
2. S. V. Patankar, *Numerical Heat Transfer and Fluid Flow* (Hemisphere, New York, 1980).
3. B. J. Braams, Ph.D. thesis, University of Utrecht, 1986.
4. D. W. Peacheman and H. H. Rachford, Jr., *J. Soc. Indus. Appl. Math.* **3,** 28 (1955).
5. M. Petravic, *J. Nucl. Mater.* **196-198,** 883 (1992).
6. Y. Xu and M. D. Smooke, *J. Comput. Phys.* **104,** 99 (1993).
7. C. K. Westbrook, *J. Comput. Phys.* **29,** 67 (1978).
8. L. A. Hageman and D. M. Young, *Applied Iterative Methods* (Academic Press, San Diego, 1981).
9. J. Shuen *et al., J. Comput. Phys.* **106,** 306 (1993).
10. D. W. Hewett *et al., J. Comput. Phys.* **101,** 11 (1992).
11. D. A. Knoll, Ph.D. thesis, University of New Mexico, 1991.
12. D. A. Knoll, A. K. Prinja, and R. B. Campbell, *J. Comput. Phys.* **104,** 418 (1993).
13. D. A. Knoll and P. R. McHugh, *J. Nucl. Mater.* **196-198,** 352 (1992).
14. D. A. Knoll and P. R. McHugh, *Int. J. Numer. Methods Fluids,* **19,** 439 (1994).
15. R. W. Freund, *SIAM J. Sci. Stat. Comput.* **14,** 470 (1993).
16. B. M. Averick and J. M. Ortega, *Appl. Numer. Math.* **8,** 443 (1991).
17. R. Dembo *et al., SIAM J. Numer. Anal.* **19,** 400 (1982).
18. R. B. Campbell *et al., J. Nucl. Mater.* **196-198,** 426 (1992).
19. T. D. Rognlien *et al., J. Nucl. Mater.* **196-198,** 347 (1992).
20. R. Zanino, *J. Nucl. Mater.* **196-198,** 326 (1992).
21. R. Vessy *et al., J. Comput. Phys.* **116** (1995).
22. M. Petrivic, *J. Comput. Phys.* **73,** 125 (1987).
23. M. Rensink, private communications.
24. S. I. Braginskii, Transport processes in a plasma, *in Reviews of Plasma Physics,* Vol. I, edited by M. A. Leontovich (Consultants Bureau, New York, 1965).
25. E. L. Vold *et al., Phys. Fluids B* **3,** 3132 (1991).
26. R. A. Hulse, *Nucl. Technol./Fusion* **3,** 259 (1981).
27. R. K. Janev *et al., J. Nucl. Mater.* **121,** 10 (1984).
28. Y. Saad and M. H. Schultz, *Math. Comput.* **44**(170), 417 (1985).
29. S. F. Ashby, T. A. Manteuffel, and P. E. Saylor, *SIAM J. Numer. Anal.* **27,** 1542 (1990).
30. V. Faber and T. Manteuffel, *SIAM J. Numer. Anal.* **21,** 352 (1984).
31. C. W. Gear and Y. Saad, *SIAM J. Sci. Stat. Comput.* **4,** 583 (1983).
32. P. N. Brown and A. C. Hindmarsh, *SIAM J. Numer. Anal.* **23,** 610 (1986).
33. P. N. Brown and Y. Saad, *SIAM J. Sci. Stat. Comput.* **11,** 450 (1990).
34. S. Ashby, T. Manteuffel, and P. Saylor, *Preconditioned Polynomial Iterative Methods, A Tutorial,* University of Colorado at Denver, April 7–8, 1992.
35. Y. Saad and M. H. Schultz, *SIAM J. Sci. Stat. Comput.* **7,** 856 (1986).
36. J. A. Meijerink and H. A. van der Vorst, *Math. Comput.* **31**(137), 148 (1977).
37. P. Langtangen, *Int. J. Numer. Methods Fluids* **9,** 213 (1989).
38. A. Brandt, *Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics,* technical report, von Karman Institute, 1984.
39. W. A. Mulder and B. Van Leer, *Implicit Upwind Methods for the Euler Equations,* AIAA Paper 83-1930, 1983 (unpublished).
40. A. H. Futch *et al., J. Nucl. Mater.* **196-198,** 860 (1992).
41. Yu. L. Igitkhanov *et al., J. Nucl. Mater.* **196-198,** 904 (1992).
42. R. J. Procassini and D. A. Knoll, *J. Nucl. Mater.* **196-198,** 363 (1992).
43. P. C. Stangeby, in *Physics of Plasma Wall Interactions in Controlled Fusion,* edited by Post and Behrisch (Plenum, New York, 1984).
44. D. A. Knoll and P. R. McHugh, *Bull. Am. Phys. Soc.* **38** (1993).
45. W. L. Barr and B. G. Logan, *Fusion Technol.* **18,** 251 (1990).
46. T. W. Petrie *et al., J. Nucl. Mater.* **196-198,** 848 (1992).
47. S. L. Allen *et al., J. Nucl. Mater.* **196-198,** 804 (1992).
48. R. Marchand and J. Lauzon, *Phys. Fluids B* **4,** 924 (1992).
49. P. R. McHugh and D. A. Knoll, *AIAA J.,* in press.