Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 211 (2006) 99–128

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

Solving the 3D MHD equilibrium equations in toroidal geometry by Newton's method

H.J. Oliver [*], A.H. Reiman, D.A. Monticello

Princeton Plasma Physics Laboratory, Princeton University, NJ, USA

Received 11 August 2004; received in revised form 22 April 2005; accepted 18 May 2005
Available online 11 July 2005

Abstract

We describe a novel form of Newton's method for computing 3D MHD equilibria. The method has been implemented as an extension to the hybrid spectral/finite-difference Princeton Iterative Equilibrium Solver (PIES) which normally uses Picard iteration on the full nonlinear MHD equilibrium equations. Computing the Newton functional derivative numerically is not feasible in a code of this type but we are able to do the calculation analytically in magnetic coordinates by considering the response of the plasma's Pfirsch–Schlüter currents to small changes in the magnetic field. Results demonstrate a significant advantage over Picard iteration in many cases, including simple finite-β stellarator equilibria. The method shows promise in cases that are difficult for Picard iteration, although it is sensitive to resolution and imperfections in the magnetic coordinates, and further work is required to adapt it to the presence of magnetic islands and stochastic regions.
© 2005 Elsevier Inc. All rights reserved.

PACS: 52.30.Cv; 52.65.Kj

Keywords: MHD equilibrium; Newton's method; Princeton Iterative Equilibrium Solver; PIES

1. Introduction

Magnetohydrodynamics (MHD) is the basic single-fluid model of macroscopic plasma behavior [1,2]. It describes the effect of magnetic geometry on a plasma's macroscopic equilibrium and stability through the interaction of inertial, pressure, and magnetic forces. When a plasma evolves on timescales much longer

[*] Corresponding author. Present address: National Institute of Water and Atmospheric Research (NIWA), Department of Meteorology, 301 Evans Bay Parade, Greta Point, Private Bag 14901, Wellington, New Zealand. Tel.: +64 4 386 0461.
E-mail addresses: h.oliver@niwa.co.nz (H.J. Oliver), reiman@pppl.gov (A.H. Reiman), monticello@pppl.gov (D.A. Monticello).

0021-9991/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2005.05.007

than the Alfvén time (typically microseconds in a fusion plasma) its behavior is governed by the MHD equilibrium equations,

$$\mathbf{J} \times \mathbf{B} = \nabla P, \tag{1}$$

$$\nabla \times \mathbf{B} = \mathbf{J}, \tag{2}$$

$$\nabla \cdot \mathbf{B} = 0, \tag{3}$$

where P is the scalar pressure, $\mathbf{B}$ is the magnetic field, and $\mathbf{J}$ is the current density. Eq. (1) expresses equilibrium force balance in the plasma and Eq. (2) is Ampere's Law (displacement current is neglected in MHD). Toroidal MHD equilibria are important in modeling magnetic confinement devices for the nuclear fusion program (tokamaks and stellarators) because steady state reactor conditions are desirable. The equilibrium equations are analytically intractable for most realistic plasma configurations, so they must generally be solved by numerical techniques. Newton's method has been applied to simple one- and two-dimensional systems [3,4], confirming its well-known speed advantage and the ability to find some equilibria that raw Picard iteration cannot, but it has never been applied successfully in the general 3D case.

Our Newton equilibrium solver has been implemented in the hybrid spectral finite-difference Princeton Iterative Equilibrium Solver (PIES) [5] which normally uses direct (Picard) iteration of the full 3D equilibrium equations. PIES can be used to study realistic systems with magnetic islands and stochastic regions because it makes no assumptions about magnetic field structure, but a lot of solution blending between successive iterations can be required to get convergence. Newton's method is potentially a much faster direct algorithm, but computing the Newton functional derivative numerically is not feasible in a code of this type, for reasons given in Section 3. By considering the response of the plasma's pressure-driven Pfirsch–Schlüter currents to small magnetic perturbations, however, we are able to derive the functional derivative analytically in magnetic coordinates. Moreover, our version of Newton's method can be seen as a natural extension of the existing Picard formulation of PIES, as discussed in Section 3. We note that the Jacobian-free Newton–Krylov methods [6] are a potentially interesting alternative to the approach taken here.

Section 2 gives an overview of the Picard algorithm for computing 3D MHD equilibria, and its implementation in PIES. Section 3 shows how Newton's method can be formulated to solve the same problem, and how it relates to the Picard scheme. Section 4 gives essential details of our numerical implementation of Newton's method: the radial discretization and how its accuracy is affected by gauge choice for the vector potential, derivation of near origin radial dependence for Fourier coefficients, boundary conditions, and gross structure of the discretized system. Section 5 describes our analytic derivation of the Newton gradient term, and Section 6 gives some results on the performance of the Newton code. Finally, Section 7 summarizes and suggests further research.

2. Overview of the PIES code

The PIES code solves the MHD equilibrium equations in the form

$$\nabla \times \mathbf{B} = \mathbf{J}(\mathbf{B}), \tag{4}$$

where $\mathbf{J}$ is a complicated, nonlinear function of $\mathbf{B}$. Given $\mathbf{B}$, the component of $\mathbf{J}$ perpendicular to $\mathbf{B}$ is determined by the force balance equation, Eq. (1). The component parallel to $\mathbf{B}$, $\mathbf{J} \cdot \mathbf{B}$, is determined by $\nabla \cdot \mathbf{J} = 0$. As described below in this section, the PIES code solves Eq. (4) using Picard iteration,

$$\nabla \times \mathbf{B}_i = \mathbf{J}(\mathbf{B}_{i-1}), \tag{5}$$

where the subscript i denotes iteration number. The remaining sections of this paper will be concerned with the solution of Eq. (4) using Newton's method.

Picard iteration is commonly used on complicated multi-dimensional systems where it may be difficult to apply more sophisticated techniques. It was first suggested for the MHD equilibrium equations in the late 1950s by Spitzer [7] and by Grad and Rubin [8], but computing power was not sufficient for 3D cases until the late 1980s. Numerical methods also had to be developed to solve the *magnetic differential equation* (MDE) that arises when computing the current density from the force balance equation [9–11] (see also Section 2.2). One iteration of the PIES implementation of Eq. (5) can be described as follows: given an estimate of the magnetic field, solve the force balance equation for the current density then, in terms of the new current density, solve Ampere's Law for a new magnetic field estimate,

$$\mathbf{J}_i(\mathbf{B}_{i-1}) \Leftarrow \begin{cases} \mathbf{J}_i(\mathbf{B}_{i-1}) \times \mathbf{B}_{i-1} = \nabla P, \\ \nabla \cdot \mathbf{J}_i(\mathbf{B}_{i-1}) = 0, \end{cases} \tag{6}$$

$$\mathbf{B}_i \Leftarrow \begin{cases} \nabla \times \mathbf{B}_i = \mathbf{J}_i(\mathbf{B}_{i-1}), \\ \nabla \cdot \mathbf{B}_i = 0. \end{cases} \tag{7}$$

In the 2D axisymmetric case this reduces to the Picard-iterative form of the Grad-Shafranov equation [2],

$$\nabla^* \psi^p_{i+1} = J_\phi(\psi^p_i) = R^2 P'(\psi^p_i) + g g'(\psi^p_i), \tag{8}$$

where ψ^p is a magnetic surface label proportional to the poloidal flux. PIES was the first code to demonstrate that the full 3D algorithm converges [12].

Section 2.1 describes the representation of fields within PIES, Section 2.2 discusses the solution of Eq. (6), and Section 2.3 discusses the solution of Eq. (7).

2.1. Representation of fields in PIES

The principal coordinate systems used in PIES are the quasi-magnetic and fixed background toroidal systems (ρ, θ, ϕ), as illustrated in Fig. 1. ρ is the radial coordinate in the poloidal plane, varying between 0 at the center and 1 at the outer surface. The angular coordinates, θ (poloidal) and ϕ (toroidal), both vary between 0 and 2π, but only ϕ is a uniform polar angle. The toroidal background grid is unrelated to the magnetic field except that the outer coordinate surface coincides with the outer magnetic surface. The

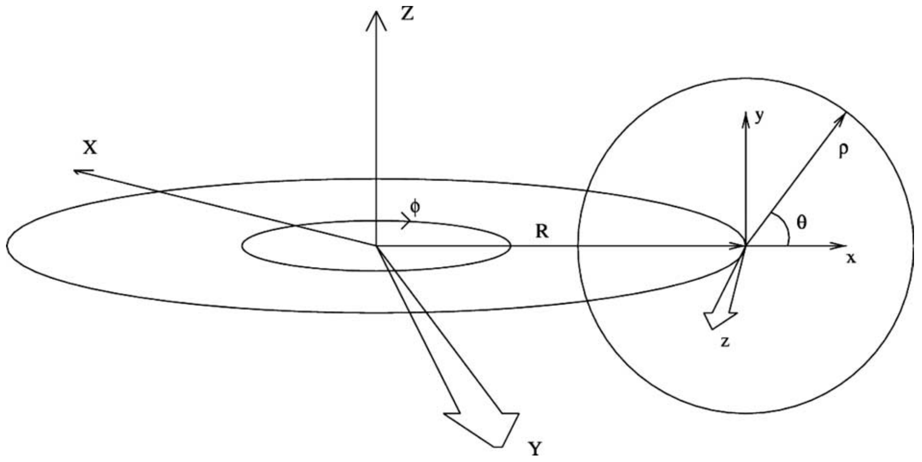


Fig. 1. The coordinate systems used in PIES. Fixed background and quasi-magnetic toroidal coordinates are represented by (ρ, θ, ϕ), shown in relation to laboratory (X, Y, Z) and rotating (x, y, z) Cartesian systems.

quasi-magnetic grid is a straight field line system wherever good magnetic surfaces exist, and is interpolated smoothly across other regions. Straight field-line coordinates are defined by the canonical equation,

$$\mathbf{B} = \nabla\psi \times \nabla\theta + \imath(\psi)\nabla\phi \times \nabla\psi, \tag{9}$$

where $\psi(\rho)$ is the toroidal flux and $\imath(\psi)$ is the rotational transform or winding number of the field.

Fields in PIES are represented by sets of 2D Fourier coefficients on each radial surface ρ_l ($l = 1, \ldots, k$). They are assumed symmetric under the stellarator transformation ($\theta \to -\theta$, $\phi \to -\phi$) which halves the number of modes to be retained and implies strictly odd or even parity,

$$f(\rho, \theta, \phi) = \sum_{m=0}^{\mathscr{M}} \sum_{n=-\mathscr{N}}^{\mathscr{N}} f_{m,n}(\rho)[\cos \mid \sin](nN\phi - m\theta), \tag{10}$$

where m and n are the poloidal and toroidal mode numbers. The number of toroidal periods, N, is omitted from subsequent equations. This is equivalent to considering axisymmetric devices or a single period of a non-axisymmetric device. For other cases, N can easily be restored by replacing ϕ with $N\phi$ and $\partial\phi$ with $N\partial\phi$. Angular resolution is set at run-time by fixing the cutoff mode numbers $\mathscr{M}$ and $\mathscr{N}$. The redundant $m = 0$, $n = \pm v$ coefficients are retained to simplify mode-number loops.

2.2. Computing the current density

An equation for the current density can be obtained from the vector product of the magnetic field and the force balance equation,

$$\mathbf{J} = \mu\mathbf{B} + \frac{\mathbf{B} \times \nabla P}{B^2}, \tag{11}$$

the divergence of which yields a magnetic differential equation for the parallel current density μ,

$$\mathbf{B} \cdot \nabla\mu = -\nabla P \cdot \nabla \times \left(\frac{\mathbf{B}}{B^2}\right). \tag{12}$$

Magnetic differential equations can be solved analytically by Fourier decomposing in the straight field line coordinates of Eq. (9). Accordingly, solving for μ and substituting into Eq. (11), we get

$$\begin{aligned}\mathbf{J}(\psi, \theta, \phi) = {} & \left(I'(\psi) + P'(\psi)\sum_{m,n}{}' \frac{m\mathscr{J}_{m,n}}{(n - \imath m)} \cos(n\phi - m\theta)\right)\nabla\psi \times \nabla\theta \\ & - \left(g'(\psi) - P'(\psi)\sum_{m,n}{}' \frac{n\mathscr{J}_{m,n}}{(n - \imath m)} \cos(n\phi - m\theta)\right)\nabla\phi \times \nabla\psi,\end{aligned} \tag{13}$$

where $I'(\psi)$ and $g'(\psi)$ are the toroidal and poloidal current profiles, $P'(\psi)$ is the pressure profile, and the primes indicate omission of $m = n = 0$ coefficients from sums. The computational effort in solving for the current density is thus largely expended on sophisticated numerical field-line mapping techniques, at the start of each iteration, for constructing the coordinate systems and Fourier decompositions in order to simply evaluate the analytic expression for $\mathbf{J}$ (in fact Eq. (11) is used to compute $\mathbf{J}$ on the background grid, having solved the MDE for μ in magnetic coordinates; see the second to last paragraph of Section 6.4). The field line mapping routines evaluate the Cartesian coordinates (x, y) along lines of magnetic force to construct the Fourier representation of the magnetic to Cartesian transformation (i.e. the Jacobian and metric elements) and the code reconstructs the 2D spectra of various fields, on each radial surface, from 1D spectra obtained by evaluation along the field lines [9,13].

2.3. Solving Ampere's Law for the magnetic field

Once the current density is known, Ampere's Law can be solved as a linear equation for the new magnetic field. This is done in PIES by reducing Eq. (7) to a Poisson equation for a scalar field whose gradient can be related to **B** [12]. The system is discretized by 2D Fourier decomposition and finite-differencing on offset radial grids, and the leading-order near origin radial dependence is factored from Fourier coefficients to improve accuracy (Section 4.1). The resulting sparse linear system is solved by block tridiagonal LU decomposition. The Ampere's Law solver, in contrast to unlike the current density solver, does not have to run in magnetic coordinates, and PIES performs markedly better when Ampere's Law is solved on the background grid (this is of relevance to the Newton algorithm too; see Section 6.3).

3. Newton's method for MHD equilibria

We derive Newton's method for 3D MHD equilibria in this section, and describe its implementation within PIES in Section 4.

Newton's method is a well-known multi-dimensional root-finding technique that is quadratically convergent given a sufficiently good initial guess. The familiar iterative algorithm can be derived by Taylor expansion of the nonlinear equation $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ in the vicinity of an initial guess, $\mathbf{x}_{i-1}$. Discarding second-order and higher terms leaves a linear equation for $\mathbf{x}_i$, the next iterative approximation to the solution vector,

$$\nabla\mathbf{f}(\mathbf{x}_{i-1})\cdot(\mathbf{x}_i-\mathbf{x}_{i-1}) = -\mathbf{f}(\mathbf{x}_{i-1}). \tag{14}$$

Likewise, Newton's method solution of Eq. (4) proceeds by writing

$$\mathbf{B}_i = \mathbf{B}_{i-1} + \delta\mathbf{B}_i \tag{15}$$

and Taylor expanding Eq. (4) to first order:

$$\nabla\times(\mathbf{B}_{i-1}) + \nabla\times(\delta\mathbf{B}_i) = \mathbf{J}(\mathbf{B}_{i-1}) + (\delta\mathbf{J}/\delta\mathbf{B})|_{\mathbf{B}_{i-1}}\cdot\delta\mathbf{B}_i. \tag{16}$$

In the expression $(\delta\mathbf{J}/\delta\mathbf{B})|_{\mathbf{B}_{i-1}}\cdot\delta\mathbf{B}_i$, $\delta\mathbf{B}_i$ is to be regarded as a vector whose components consist of all the Fourier components of **B** on every radial grid surface, and $(\delta\mathbf{J}/\delta\mathbf{B})|_{\mathbf{B}_{i-1}}$ is a matrix operating on all of those components. For a given $\mathbf{B}_{i-1}$, Eq. (16) is a linear equation for $\delta\mathbf{B}_i$. The functional derivative $\delta\mathbf{J}/\delta\mathbf{B}$ is the derivative of each of the Fourier components of $\mathbf{J}(\mathbf{B})$ on each flux surface with respect to each of the components of **B**. Evaluating it numerically would not be feasible in PIES because the computationally intensive 3D field-line mapping code would have to be invoked many times in each iteration to determine the effect of independently varying every Fourier coefficient of the magnetic field. We are able to avoid this by deriving the functional derivative (gradient term) analytically in magnetic coordinates, by considering the response of the plasma's Pfirsch–Schlüter currents to small magnetic field perturbations, in Section 5. Jacobian-free Newton–Krylov methods are a potentially interesting alternative, as noted in the introduction, but our approach fits very well within the existing design of PIES.

At each step of the Newton scheme, Eqs. (6) and (7) are replaced by

$$\mathbf{J}_i(\mathbf{B}_{i-1}) \Leftarrow \begin{cases}\mathbf{J}_i(\mathbf{B}_{i-1})\times\mathbf{B}_{i-1} = \nabla P,\\ \nabla\cdot\mathbf{J}_i(\mathbf{B}_{i-1}) = 0,\end{cases} \tag{17}$$

$$\mathbf{B}_i \Leftarrow \begin{cases}\nabla\times\delta\mathbf{B}_i - \delta\mathbf{J}_i = \mathbf{J}_i(\mathbf{B}_{i-1}) - \nabla\times\mathbf{B}_{i-1},\\ \nabla\cdot\delta\mathbf{B}_i = 0,\end{cases} \tag{18}$$

where $\mathbf{B}_i = \mathbf{B}_{i-1} + \delta\mathbf{B}_i$, and $\delta\mathbf{J}$ is the Newton gradient term,

$$\delta \mathbf{J}_i \equiv \left.\frac{\partial \mathbf{J}(\mathbf{B})}{\partial \mathbf{B}}\right|_{\mathbf{B}_{i-1}} \cdot \delta \mathbf{B}_i = \mathbf{J}_{i+1}(\mathbf{B}_i) - \mathbf{J}_i(\mathbf{B}_{i-1}) + \mathcal{O}(\delta \mathbf{B}_i^2). \tag{19}$$

Note that Eq. (18) looks like its Picard counterpart with one new term added, which raises the possibility of converting PIES to Newton's method with relative ease. The reduction of Ampere's Law to a scalar Poisson equation no longer works in the presence of the extra term though, so we first constructed a new Newton-compatible vector potential Picard Ampere's Law solver,

$$\mathbf{B}_i = \nabla \times \mathbf{A}_i \quad \Leftarrow \quad \nabla \times \nabla \times \mathbf{A}_i = \mathbf{J}_i(\mathbf{B}_{i-1}). \tag{20}$$

The corresponding form for Newton's method is

$$\mathbf{B}_i = \mathbf{B}_{i-1} + \nabla \times \delta \mathbf{A}_i \quad \Leftarrow \quad \nabla \times \nabla \times \delta \mathbf{A}_i - \mathscr{L}_i \cdot \delta \mathbf{A}_i = \mathbf{J}_i(\mathbf{B}_{i-1}) - \nabla \times \mathbf{B}_{i-1}, \tag{21}$$

where $\mathscr{L}_i \cdot \delta \mathbf{A}_i$ is the Newton gradient term expressed as a linear operator $\mathscr{L}$ acting on the new solution vector, δA_i. Through Eq. (20) much of the Newton code can be verified in the context of a simpler algorithm, by comparing the performance of PIES in its original and vector potential Picard modes. In particular, the *curl curl* operator is common to Picard and Newton, and the entire system can be discretized in such a way that the gross structure of the resulting sparse linear system is not changed by addition of the Newton gradient term.

4. Implementation of Newton's method in PIES

The Newton scheme requires the numerical solution of Eq. (21) for the vector potential. First the new vector potential Ampere's Law solver was discretized in a similar way to the original one (Section 2.3): Fourier decomposition, with the leading-order near origin radial dependence factored out of Fourier coefficients to enhance the accuracy of finite-differencing on the offset radial grids. As discussed in this section, the particular considerations that arise in dealing with the near origin radial dependence of the vector potential led us to develop a custom block tridiagonal matrix solver.

Section 4.1 discusses our choice of gauge for the vector potential, and its corresponding near origin behavior. Section 4.2 discusses the discretization of the equations: the finite-differencing scheme, the handling of convolutions, the structure of the resultant sparse linear system, and the custom matrix solver. The derivation of the Newton gradient term is described separately in Section 5.

4.1. Gauge choice and radial dependence of the vector potential

The gradient of any scalar field can be added to a vector potential without changing the magnetic field. We use this gauge freedom to set $A_\rho = 0$, reducing the number of equations, prior to discretization, from three to two. The remaining gauge freedom is

$$\mathbf{A}(\rho, \theta, \phi) \rightarrow \mathbf{A}(\rho, \theta, \phi) + \nabla \chi(\theta, \phi), \tag{22}$$

where χ is an arbitrary function of θ and ϕ. The angular covariant components of this expression, in the representation of Eq. (10), are

$$A_{\theta;m,n}(\rho) \rightarrow A_{\theta;m,n}(\rho) - m\chi_{m,n}, \tag{23}$$

$$A_{\phi;m,n}(\rho) \rightarrow A_{\phi;m,n}(\rho) + n\chi_{m,n}. \tag{24}$$

Other gauge conditions in addition to $A_\rho = 0$ are needed to fix the values of the $(m + 1)(2n + 1) - 1$ arbitrary gauge constants, $\chi_{m,n}$. The choice of these constants is critical to our use of radial factors for the vector potential, as we shall see below, and also determines whether the Newton gradient term is flux-surface local or not.

The finite nature of physical fields and the properties of polar-like coordinate systems dictate that the Fourier coefficients of physical scalar fields, physical vector field components, and coordinate-related fields such as the Jacobian have predictable *analytic* radial dependence near the polar-like origin,

$$F_{m,n}(\rho) \sim \rho^{m+\zeta} \quad \text{as } \rho \to 0, \tag{25}$$

where m is the poloidal mode number and ζ is some small integer. Numerical radial derivatives of high-m modes, which vary rapidly near the origin, are better approximated by finite-difference expressions if their leading-order behavior is factored out. This improved the performance of the PIES Ampere's Law solver, so we derive analogous expressions for the vector potential here.

To derive the leading-order radial dependence of a physical scalar field $f(\rho, \theta)$, consider a *Cartesian-like* coordinate system (ξ, η),

$$\xi = \rho\cos(\theta), \tag{26}$$

$$\eta = \rho\sin(\theta). \tag{27}$$

If $f(\rho, \theta)$ is expanded in a Cartesian-like power series about the origin, $\xi = \eta = 0$, each series coefficient must be finite because the Cartesian-like grid is non-singular, and the field itself is finite. By substitution of Eqs. (26) and (27) and application of standard trigonometric identities, the power series can be put in the form of a Fourier series in the polar-like system, and comparison with the standard representation of Eq. (10) yields a near origin radial power series for each Fourier coefficient:

$$f_m(\rho) = \rho^m\left(f_m^{(0)} + \rho^2 f_m^{(2)} + \rho^4 f_m^{(4)} + \cdots\right) \quad \text{as } \rho \to 0, \tag{28}$$

where the $f_m^{(i)}$ are just constant coefficients of ρ^i. The leading-order radial dependence of the Fourier coefficients of a physical scalar field is therefore ρ^m.

For the polar components of vector fields and other quantities that depend directly on the coordinates, such as the Jacobian and metric elements, the effect of the coordinate basis vectors must also be considered,

$$\nabla\rho = \nabla\xi\cos(\theta) + \nabla\eta\sin(\theta), \tag{29}$$

$$\rho\nabla\theta = \nabla\eta\cos(\theta) - \nabla\xi\sin(\theta). \tag{30}$$

Noting that the *Cartesian-like components* of a *physical vector field* must be analytic, and proceeding as for the scalar field derivation, one can show that the near origin leading-order radial dependence of the Jacobian is ρ^{m+1}, the metric elements behave as shown in Table 1, and the co- and contravariant components of a physical vector field behave as shown in Table 2. As another consequence, there are some relations between the lower-order power series coefficients of polar vector components near the origin. In the contravariant case, for example,

$$V_{0,n}^{\rho,(0)} = +V_{0,n}^{\theta,(0)}, \tag{31}$$

$$V_{1,n}^{\rho,(0)} = -V_{1,n}^{\theta,(0)}, \tag{32}$$

$$\vdots$$

$$V_{m,n}^{\rho,(0)} = -V_{m,n}^{\theta,(0)}, \tag{33}$$

where the superscript i indicates a coefficient of ρ^i.

The vector potential, which is not a physical field, does not have to be analytic. A general three-component analytic vector potential, A_1, does give rise to an analytic magnetic field, but the final near origin radial behavior depends on our subsequent gauge transformation,

Table 1
Near origin leading-order radial exponents, $\rho^{e(m)}$, for metric elements

Field	Parity	$e(0)$	$e(1)$	$e(m>1)$
$(\mathcal{J}g^{\rho\rho})_{m,n}$	Even	1	2	$m-1$
$(\mathcal{J}g^{\rho\theta})_{m,n}$	Odd	2	1	$m-2$
$(\mathcal{J}g^{\rho\phi})_{m,n}$	Odd	2	1	m
$(\mathcal{J}g^{\theta\theta})_{m,n}$	Even	-1	0	$m-3$
$(\mathcal{J}g^{\theta\phi})_{m,n}$	Even	1	0	$m-1$
$(\mathcal{J}g^{\phi\phi})_{m,n}$	Even	1	2	$m+1$
$(\mathcal{J}^{-1}g_{\rho\rho)_{m,n}}$	Even	-1	0	$m-3$
$(\mathcal{J}^{-1}g_{\rho\theta})_{m,n}$	Odd	2	1	$m-2$
$(\mathcal{J}^{-1}g_{\rho\phi})_{m,n}$	Odd	0	-1	$m-2$
$(\mathcal{J}^{-1}g_{\theta\theta})_{m,n}$	Even	1	2	$m-1$
$(\mathcal{J}^{-1}g_{\theta\phi})_{m,n}$	Even	1	0	$m-1$
$(\mathcal{J}^{-1}g_{\phi\phi})_{m,n}$	Even	-1	0	$m-1$

Table 2
Radial behavior, $\rho^{e(m)}$, for components of a physical vector field

Field	$e(0)$	$e(m>1)$
$V^{\rho}_{m,n}$	1	$m-1$
$V^{\theta}_{m,n}$	0	$m-2$
$V^{\phi}_{m,n}$	0	m
$V_{\rho;m,n}$	1	$m-1$
$V_{\theta;m,n}$	2	m
$V_{\phi;m,n}$	0	m

$$\mathbf{A}_1 \to \mathbf{A}_2 = \mathbf{A}_1 + \nabla\zeta \quad \text{such that } A_{2\rho} = 0. \tag{34}$$

First write out the analytic covariant components of $\mathbf{A}_1$ according to Table 2,

$$A_{1\rho;m,n}(\rho) = \begin{cases} \rho^1\left(r^{(0)}_{0,n} + r^{(2)}_{0,n}\rho^2 + \cdots\right), & m = 0, \\ \rho^{m-1}\left(r^{0}_{m,n} + r^{(2)}_{m,n}\rho^2 + \cdots\right), & m \geqslant 1, \end{cases} \tag{35}$$

$$A_{1\theta;m,n}(\rho) = \begin{cases} \rho^2\left(t^{(0)}_{0,n} + t^{(2)}_{0,n}\rho^2 + \cdots\right), & m = 0, \\ \rho^{m}\left(t^{(0)}_{m,n} + t^{(2)}_{m,n}\rho^2 + \cdots\right), & m \geqslant 1, \end{cases} \tag{36}$$

$$A_{1\phi;m,n}(\rho) = \begin{cases} \rho^0\left(p^{(0)}_{0,n} + p^{(2)}_{0,n}\rho^2 + \cdots\right), & m = 0, \\ \rho^{m}\left(p^{(0)}_{m,n} + p^{(2)}_{m,n}\rho^2 + \cdots\right), & m \geqslant 1, \end{cases} \tag{37}$$

where $r^{(i)}_{m,n}$, $t^{(i)}_{m,n}$, and $p^{(i)}_{m,n}$ are constant power series coefficients. The gauge function $\zeta(\rho, \theta, \phi)$ is then determined by the integral of the covariant radial component of Eq. (34),

$$\zeta_{m,n}(\rho) = \begin{cases} \zeta_{0,n}(0) - \rho^2\left(\frac{r^{(0)}_{0,n}}{2} + \frac{r^{(2)}_{0,n}}{4}\rho^2 + \cdots\right), & m = 0, \\ \zeta_{m,n}(0) - \rho^{m}\left(\frac{r^{(0)}_{m,n}}{m} + \frac{r^{(2)}_{m,n}}{m+2}\rho^2 + \cdots\right), & m \geqslant 1. \end{cases} \tag{38}$$

Substituting this back into the gauge equation, using the near origin component relations of Eqs. (31) to (33) eliminate $r^{(0)}_{m,n}$ and annihilate the lowest-order term of the θ-component power series, and aggregating series coefficients, we get the components of $\mathbf{A}_2$,

$$A_{2\rho;m,n}(\rho) = 0, \tag{39}$$

$$A_{2\theta;m,n}(\rho) = \begin{cases} \rho^2\left(\alpha^{(0)}_{0,n} + \alpha^{(2)}_{0,n}\rho^2 + \cdots\right), & m = 0, \\ -m\chi_{m,n} + \rho^{m+2}\left(\alpha^{(0)}_{m,n} + \alpha^{(2)}_{m,n}\rho^2 + \cdots\right), & m \geqslant 1, \end{cases} \tag{40}$$

$$A_{2\phi;m,n}(\rho) = \begin{cases} n\chi_{0,n} + \beta^{0}_{0,n} + \beta^{(2)}_{0,n}\rho^2 + \cdots, & m = 0, \\ n\chi_{m,n} + \rho^{m}\left(\beta^{(0)}_{m,n} + \beta^{(2)}_{m,n}\rho^2 + \cdots\right), & m \geqslant 1, \end{cases} \tag{41}$$

where the $\alpha^{(i)}_{m,n}$ and $\beta^{(i)}_{m,n}$ are constant series coefficients and the $\chi_{m,n}$ are the Fourier coefficients of the gauge function ζ evaluated at the origin. This is clearly non-analytic behavior, and the only way to extract useful radial factors for $\mathbf{A}$ is to choose the exact gauge in which all the $\chi_{m,n}$ are identically zero. This must be done by forcing the appropriate modes of $\mathbf{A}_{m,n}$ to zero at the origin because the radial dependence predictions are only valid as $\rho \to 0$,

$$A_{\theta;m,n}(0) = 0 \;\Rightarrow\; \chi_{m,n} = 0, \quad m \neq 0, \tag{42}$$

$$A_{\phi;0,n}(0) = 0 \;\Rightarrow\; n\chi_{0,n} + \beta^{(0)}_{0,n} = 0, \quad m = 0,\; n \neq 0, \tag{43}$$

$A_{\phi;0,0}(1)$, the flux through the hole in the center of the torus, is physically irrelevant; we can fix its value arbitrarily by setting $A_{\phi;0,0}(0)$ to zero,

$$A_{\phi;0,0}(0) = 0, \quad \Rightarrow \beta^{(0)}_{0,0} = 0, \quad m = n = 0 \tag{44}$$

for a total of $(m + 1)(2n + 1)$ origin conditions. Given our final gauge choice, the near origin leading-order radial dependence of the vector potential is therefore

$$A_{\theta;m,n}(\rho) \sim \rho^{m+2} \quad \forall m, \tag{45}$$

$$A_{\phi;m,n}(\rho) \sim \begin{cases} \rho^2, & m = 0, \\ \rho^m, & m \geqslant 1. \end{cases} \tag{46}$$

In practice, factoring large powers of ρ from $\mathbf{A}$ can cause the $\nabla\times\nabla\times\mathbf{A}$ matrix to become ill-conditioned, so the radial factors are truncated to some maximum value in the code (ρ^2 was found to be a good compromise).

4.2. Discretization of Ampere's Law

PIES calculates the contravariant components of the current density. The divergence-free nature of $\mathbf{J}$ implies a relation between the components of the right side vector, and our gauge choice eliminates the ρ-component of the solution vector. Consequently we solve the two contravariant angular components of Ampere's Law for the covariant angular components of $\mathbf{A}$,

$$\left.\begin{aligned} (\nabla\times\nabla\times\mathbf{A})^{\theta} &= J^{\theta} \\ (\nabla\times\nabla\times\mathbf{A})^{\phi} &= J^{\phi} \end{aligned}\right\} \;\Rightarrow\; A_{\theta},\, A_{\phi}. \tag{47}$$

The discretization is performed on two offset radial grids: the contravariant components of Ampere's Law and the covariant components of $\mathbf{A}$ reside on the integer grid,

$$\rho = l\Delta, \quad l = 0, 1, \ldots, k, \tag{48}$$

and the covariant components of $\nabla \times \mathbf{A}$ reside on an intermediate half-integer grid,

$$\rho = \left(l + \frac{1}{2}\right)\Delta, \quad l = 0, 1, \ldots, k. \tag{49}$$

This effectively doubles the radial resolution, increasing the accuracy of the radial discretization, and leads to finite-difference expressions of the following form:

$$\left(\frac{\partial f}{\partial \rho}\right)^{l}_{m,n} = R'^{l}_{f;m}\left(\frac{\tilde{f}^{l-\frac{1}{2}}_{m,n} + \tilde{f}^{l+\frac{1}{2}}_{m,n}}{2}\right) + R^{l}_{f;m}\left(\frac{\tilde{f}^{l+\frac{1}{2}}_{m,n} - \tilde{f}^{l-\frac{1}{2}}_{m,n}}{\Delta}\right), \tag{50}$$

$$\left(\frac{\partial g}{\partial \rho}\right)^{l+\frac{1}{2}}_{m,n} = R'^{l+\frac{1}{2}}_{g;m}\left(\frac{\tilde{g}^{l}_{m,n} + \tilde{g}^{l+1}_{m,n}}{2}\right) + R^{l+\frac{1}{2}}_{g;m}\left(\frac{\tilde{g}^{l+1}_{m,n} - \tilde{g}^{l}_{m,n}}{\Delta}\right), \tag{51}$$

where $R^{l}_{f;m} = \rho_l^{e(m)}$ is the radial factor for $f_{m,n}(\rho)$ evaluated at radial surface l, from the calculations of Section 4.1.

4.2.1. Truncated analytic convolutions

In products of the solution vector with other quantities, such as metric elements, the Fourier-space convolution contains sums over non-standard index ranges and trigonometric sum and difference terms with indices outside the standard series truncation limits. It can be shown, however, that the coefficients of the solution vector can be extracted from these convolutions in a form consistent with the standard Fourier representation if the coefficients of the multiplying series are redefined in a particular way [14]. Consider the convolution of two Fourier series f and u,

$$fu = \sum_{m_1=0}^{M}\sum_{n_1=-N}^{N}(fu)_{m_1,n_1}(\sin\,|\,\cos)[n_1\phi - m_1\theta]. \tag{52}$$

The coefficients of the product series can be written,

$$(fu)_{m_1,n_1} = \sum_{m_2=0}^{M}\sum_{n_2=-N}^{N}\begin{cases}\left(\overline{f}_{\substack{m_1-m_2,\\ n_1-n_2}} + \overline{f}_{\substack{m_1+m_2,\\ n_1+n_2}}\right)u_{m_2,n_2} & \text{(EE or OE)},\\ \left(\overline{f}_{\substack{m_1-m_2,\\ n_1-n_2}} - \overline{f}_{\substack{m_1+m_2,\\ n_1+n_2}}\right)u_{m_2,n_2} & \text{(EO or OO)},\end{cases} \tag{53}$$

where the labels on the right refer to the parity of f and u, (E)ven or (O)dd, and the coefficients of f have been embedded within a new set, $\overline{f}_{m,n}$:

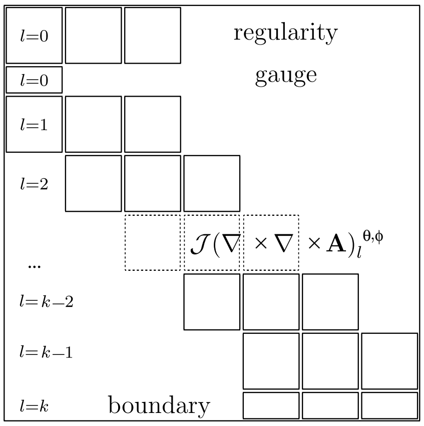
$$\overline{f}_{0,n} = \begin{cases}\frac{1}{2}f_{0,n}, & m_1 = 0,\\ f_{0,n}, & m_1 \neq 0,\end{cases} \tag{54}$$

$$\overline{f}_{m,n} = \begin{cases}\frac{1}{4}f_{m,n}, & m_1 = 0,\\ \frac{1}{2}f_{m,n}, & m_1 \neq 0,\end{cases} \tag{55}$$

$$\overline{f}_{m,n} = 0, \quad \|m\| > M \text{ or } \|n\| > N, \tag{56}$$

where m_1 is the poloidal index from Eq. (53), *and* $\overline{f}$'s index range extends to negative m such that

$$\overline{f}_{-m,-n} = \begin{cases}\overline{f}_{m,n} & \text{(EE or OE)},\\ -\overline{f}_{m,n} & \text{(EO or OO)}.\end{cases} \tag{57}$$



l=0
regularity
gauge
l=0
l=1
l=2
$\mathcal{J}(\nabla\times\nabla\times\mathbf{A})_l^{\theta,\phi}$
...
$l=k-2$
$l=k-1$
$l=k$
boundary

Fig. 2. Block structure of the sparse linear system resulting from the discretization of Ampere's Law in our chosen gauge. The size of each dense square block is $2^2(m+1)^2(2n+1)^2$.

4.2.2. Structure of the sparse linear system

Our Ampere's Law discretization results in $(m + 1)(2n + 1)$ equations at each internal radial surface, or $(k - 1)(m + 1)(2n + 1)$ in total, where $l = k$ is the radial index of the outer surface. The boundary condition

$$\mathbf{B}\cdot\nabla\rho|_{\rho=1} = 0 \tag{58}$$

constrains the outer boundary to be a magnetic flux surface. Discretized, this gives rise to $(m + 1)(2n + 1) - 1$ equations at $l = k$. For the $m = n = 0$ mode, conservation of net external poloidal current gives one more equation,

$$[(\nabla\times\mathbf{A})_\phi]^k_{0,0} = B^k_{\phi;0,0}, \tag{59}$$

where $B^k_{\phi;0,0}$ is one of the code's input parameters. The gauge conditions provide another $(m + 1)(2n + 1)$ equations at the origin. The radial dependence relations allow us to force $A_{\theta;m,n} = 0$ and $A_{\phi;m,n} = 0$ wherever the gauge conditions do not already specify that, and we use regularity conditions to impose the correct near origin radial dependence for the remaining modes, yielding a further $2(m + 1)(2n + 1)$ origin conditions. In total, the discretization of Ampere's Law, with boundary, gauge, and origin regularity conditions, results in $2(k + 1)(m + 1)(2n + 1)$ equations for the same number of unknowns ($A^l_{\theta;m,n}$ and $A^l_{\phi;m,n}$, $l = 0,\ldots,k$). However, there are three times as many equations at the origin as at the outer boundary, a consequence of the gauge conditions that make our radial factoring possible, so the resulting sparse linear system is structured as in Fig. 2. This cannot be transformed into standard block tridiagonal form; we therefore built a custom sparse matrix solver for the system obtained by moving the origin gauge equations to the lower block row (i.e. a block tridiagonal system with an extra block in the lower left corner). The custom solver uses block LU decomposition, processing one block row at a time. The extra corner block results in a dense final row in the lower triangular factor, but the solver's performance is still of order N block operations, where N is the number of block-rows in the matrix.

5. Analytic derivation of the Newton gradient

It would not be feasible to numerically evaluate the large number of terms in the functional derivative of Eq. (19), $(\delta\mathbf{J}/\delta\mathbf{B})|_{\mathbf{B}_{i-1}}$, as explained in 3. Our way around this problem, an analytic derivation of the term in magnetic coordinates, is outlined in this section.

The Newton gradient calculation depends intimately on the magnetic coordinates, which are related to the magnetic field computed at the end of the *previous* iteration. In light of this, and of the complex algebra involved, it is convenient to simplify our subscripting by focusing on a particular iteration and labeling quantities according to the magnetic field with which they are associated, rather than by iteration number. Thus the iteration of interest begins with the *old* (current, or known) field $\mathbf{B}_0$ which defines the old magnetic coordinates (ρ_0, θ_0, ϕ_0), and in terms of which the old current density $\mathbf{J}_0$ is computed. Then at the end of the same iteration the *new* field, $\mathbf{B}_1 = \mathbf{B}_0 + \delta\mathbf{B}$, will be computed by the Ampere's Law solver before being used to define the new magnetic coordinates (ρ_1, θ_1, ϕ_1) in the *next* iteration. In this notation the Newton gradient is written,

$$\delta\mathbf{J}(= \delta\mathbf{J}_0) = \mathbf{J}_1(\mathbf{B}_1) - \mathbf{J}_0(\mathbf{B}_0). \tag{60}$$

On entering the Ampere's Law solver the second term on the right is a known quantity. We therefore need to estimate, to first order in $\delta\mathbf{B}$, the current density solution to the force balance equation for the unknown field $\mathbf{B}_1$. This section of the paper demonstrates two different ways to derive an analytic expression for $\delta\mathbf{J}$.

5.1. Functions in the two magnetic coordinate systems

A potentially confusing aspect of this derivation is that we must work simultaneously with functions, generally representing physical quantities with well-known symbolic labels (e.g. $\mathbf{J}$ for current density), in both the old and new magnetic coordinate systems. These are generally known functions of their native magnetic coordinates so that, for example, $\mathbf{J}_0$ is the same function of (ψ_0, θ_0, ϕ_0) that $\mathbf{J}_1$ is of (ψ_1, θ_1, ϕ_1), as defined by Eq. (13). When working with the old and new versions of a particular quantity in a single coordinate system, however, it must be kept in mind that the two functional forms are different. The new/old subscripting (1/0) therefore also serves as an indicator of functional form: $\mathbf{J}_1(\psi_0, \theta_0, \phi_0)$ is not the same function as $\mathbf{J}_0(\psi_0, \theta_0, \phi_0)$, just as $f(x)$ and $g(x)$ generally represent different functions. The new coordinates themselves can of course be written as functions of the old,

$$\psi_1(\psi_0, \theta_0, \phi) = \psi_0 + \delta\psi(\psi_0, \theta_0, \phi), \tag{61}$$

$$\theta_1(\psi_0, \theta_0, \phi) = \theta_0 + \delta\theta(\psi_0, \theta_0, \phi), \tag{62}$$

$$\phi_1(\psi_0, \theta_0, \phi) = \phi_0, \tag{63}$$

where $\delta\psi$ and $\delta\theta$ are calculated in Sections 5.2.5 and 5.2.6.

5.2. Derivation of $\delta\mathbf{J}$, Method 1

The first method derives the gradient term entirely in terms of the old magnetic coordinates, by making use of the canonical magnetic field and analytic current density expressions, Eqs. (9) and (13). Both $\mathbf{J}_1$ and $\mathbf{B}_1$ are written in their native magnetic coordinates and, assuming that the small change in $\mathbf{B}$ also results in a small change in the coordinates, Eqs. (61)–(63), we make a first-order series expansion about the old coordinates. The current density equation then yields a relation between $\delta\mathbf{J}$ and the change in the coordinates, while the canonical equation yields a relation between the change in the coordinates and the components of $\delta\mathbf{B}$, with everything expressed in terms of the old coordinates. The Newton gradient is then given by

$$\delta\mathbf{J}(\psi_0, \theta_0, \phi_0) = \mathbf{J}_1(\psi_0, \theta_0, \phi_0) - \mathbf{J}_0(\psi_0, \theta_0, \phi_0). \tag{64}$$

It is convenient to work with an alternative form of the expression for $\mathbf{J}$ here,

$$\mathbf{J}(\psi, \theta, \phi) = \left(I'(\psi) - \frac{\partial v}{\partial \theta}\right)\nabla\psi \times \nabla\theta + \left(-g'(\psi) + \frac{\partial v}{\partial \phi}\right)\nabla\phi \times \nabla\psi, \tag{65}$$

where v is the periodic part of a stream function for the current density and satisfies the following magnetic differential equation, which is actually an intermediate step in the derivation of Eq. (13),

$$\mathscr{J}(\mathbf{B}\cdot\nabla)v = \mathscr{J}\frac{\mathrm{d}P}{\mathrm{d}\psi} + \frac{\mathrm{d}g}{\mathrm{d}\psi} + \iota\frac{\mathrm{d}I}{\mathrm{d}\psi}. \tag{66}$$

The sections below outline essential parts of the derivation, while necessarily omitting a lot of algebraic detail.

5.2.1. The radial component of $\delta \boldsymbol{J}$

The radial component of the Newton Ampere's Law is linearly dependent with the other two vector components so it is not used in the numerical algorithm, but δJ^{ψ} is needed as part of a right side term equivalent to the curl of the previous iteration's magnetic field. Expanding $\mathbf{J}_1\cdot\nabla\psi_1 = 0$ about the old quantities, discarding second-order terms, and multiplying by the Jacobian gives

$$\mathscr{J}_0\,\delta J^{\psi} = -\mathscr{J}_0 J_0^{\theta}\frac{\partial\,\delta\psi}{\partial\theta_0} - \mathscr{J}_0 J_0^{\phi}\frac{\partial\,\delta\psi}{\partial\phi_0}, \tag{67}$$

where $\delta\psi$ is calculated in terms of $\delta\mathbf{B}$ in Section 5.2.5.

5.2.2. The poloidal component of $\delta \boldsymbol{J}$

Expanding the contravariant poloidal component of Eq. (65) for $\mathbf{J}_1$ about the unperturbed quantities, discarding second-order terms, and making use of the zero-order equation yields

$$\mathscr{J}_0\,\delta J^{\theta} = \frac{\partial\,\delta v}{\partial\phi_0} - \delta\mathscr{G} - J_0^{\theta}\,\delta\mathscr{J} - \mathscr{J}_0\mathbf{J}_0\cdot\nabla\,\delta\theta, \tag{68}$$

where $\delta\mathscr{G}$ is shorthand for the change in the poloidal current profile,

$$\delta\mathscr{G} \equiv \delta(g') = \delta\left(\frac{\mathrm{d}g}{\mathrm{d}\psi}\right) = \frac{\mathrm{d}g_1}{\mathrm{d}\psi_1} - \frac{\mathrm{d}g_0}{\mathrm{d}\psi_0}. \tag{69}$$

The first term on the right is a function of ψ_1 alone in the new system, but is a function of all the old coordinates (and the opposite applies for the second term). The change in the stream function is calculated in Section 5.2.10, the change in the poloidal current profile in Section 5.2.11, the change in the Jacobian in Section 5.2.4, and the change in the poloidal coordinate, in terms of $\delta\mathbf{B}$, in Section 5.2.6.

5.2.3. The toroidal component of $\delta \boldsymbol{J}$

Expanding the contravariant toroidal component of Eq. (65) for $\mathbf{J}_1$ about the unperturbed quantities, discarding second-order terms, and making use of the zero-order equation yields

$$\mathscr{J}_0\,\delta J^{\phi} = \delta\mathscr{I} - \delta\mathscr{N} - J_0^{\phi}\,\delta\mathscr{J}, \tag{70}$$

where $\delta\mathscr{I}$ is shorthand for the change in the toroidal current profile, and $\delta\mathscr{N}$ is the change in the θ-derivative of the current density stream function,

$$\delta\mathscr{N} \equiv \delta\left(\frac{\partial v}{\partial\theta}\right) = \frac{\partial v_1}{\partial\theta_1} - \frac{\partial v_0}{\partial\theta_0}, \tag{71}$$

$$\delta\mathscr{I} \equiv \delta(I') = \frac{\mathrm{d}I_1}{\mathrm{d}\psi_1} - \frac{\mathrm{d}I_0}{\mathrm{d}\psi_0}. \tag{72}$$

These are derived Sections 5.2.8 and 5.2.12, and the change in the Jacobian in terms of $\delta\mathbf{B}$ in Section 5.2.4.

5.2.4. The change in the Jacobian

Eqs. (68) and (70) require an expression for the perturbed Jacobian. According to the canonical equation the Jacobian is just the inverse of $\mathbf{B}\cdot\nabla\phi$. Doing the usual expansion, and recalling that ϕ is fixed, we get

$$\delta\mathscr{J} = -\mathscr{J}_0^2\,\delta B^\phi \tag{73}$$

and for the inverse Jacobian,

$$\delta(\mathscr{J}^{-1}) = \delta B^\phi. \tag{74}$$

A second expression for $\delta\mathscr{J}$ can be derived from its definition,

$$\mathscr{J}^{-1} = \nabla\psi\cdot(\nabla\theta\times\nabla\phi). \tag{75}$$

Expanding this to first order and subtracting the zero-order equation,

$$\delta\mathscr{J} = \mathscr{J}_0\left(\frac{\partial\,\delta\psi}{\partial\psi_0}+\frac{\partial\,\delta\theta}{\partial\theta_0}\right). \tag{76}$$

5.2.5. The change in the radial coordinate

The radial component of the new canonical equation ($\mathbf{B}_1\cdot\nabla\psi_1 = 0$) expanded about the old coordinates gives the change in the radial coordinate,

$$\delta\psi_{m,n} = \frac{[\mathscr{J}\,\delta B^\psi]_{m,n}}{(n-\imath m)}. \tag{77}$$

The $m = n = 0$ Fourier coefficient, $\delta\psi_{0,0}$, can be derived by equating the two forms of $\delta\mathscr{J}$, Eqs. (73) and (76) and integrating over each of the coordinates,

$$\delta\psi_{0,0}(\psi_0) = \delta\psi_{0,0}(0) + \int_0^{\psi_0}[\mathscr{J}_0\,\delta B^\phi]_{0,0}\mathrm{d}\psi_0. \tag{78}$$

Note that ψ is the toroidal magnetic flux, so it is physically reasonable that $\delta\psi_{0,0}$ at some radial surface depends on $\delta\mathbf{B}_{0,0}$ everywhere inside the surface. This non-flux-surface-local behavior would present a problem if we formulated Ampere's Law in terms of $\delta\mathbf{B}$ (it would generate a dense linear system on discretization), but it is flux-surface local when expressed in terms of our vector potential in the $A_\rho = 0$ gauge. The integrand is

$$[\mathscr{J}_0\,\delta B^\phi]_{0,0} = \left[\frac{\partial\,\delta u}{\partial\psi_0}\right]_{0,0} \tag{79}$$

and the gauge conditions are also such that $\delta u_{0,0} = 0$ at the old magnetic axis ($\psi_0 = 0$), so Eq. (78) becomes

$$\delta\psi_{0,0}(\psi_0) = \delta\psi_{0,0}(0) + \delta u_{0,0}(\psi_0). \tag{80}$$

The constant $\delta\psi_{0,0}(0)$ exists because an arbitrary constant can be added to the radial coordinate without changing the field. We implicitly fix ψ_1 and ψ_0 to zero at their respective magnetic axes, however, by our use of the prescribed profiles $I'(\psi)$ and $P'(\psi)$. Taking the $m = n = 0$ coefficient of Eq. (61) and evaluating it at the old origin shows that $\delta\psi_{0,0}$ is equal to the average value of the new radial coordinate at the old magnetic axis,

$$\delta\psi_{0,0} = [\psi_1]_{0,0}(0). \tag{81}$$

The near axis leading-order radial dependence of $\delta\psi$ can be deduced from that of $\delta\mathbf{A}$ or $\delta\mathbf{B}$ (see Section 4),

$$\delta\psi_{m,n}(\rho) \sim \begin{cases} \rho^2, & m = 0, \\ \rho^m, & m > 0, \end{cases} \tag{82}$$

which shows that $\delta\psi_{0,0}$ goes to zero at the origin, and the axis shift is a second-order quantity that can be neglected,

$$\delta\psi_{0,0}(\psi_0) = \delta u_{0,0}(\psi_0). \tag{83}$$

5.2.6. The change in the poloidal coordinate

The change in the poloidal coordinate is derived directly from the poloidal component of the canonical equation,

$$\delta\theta_{m,n} = \frac{\delta\iota_{m,n} + \iota[\mathscr{J}\,\delta B^\phi]_{m,n} - [\mathscr{J}\,\delta B^\theta]_{m,n}}{(n - \iota m)}, \tag{84}$$

where the change in the rotational transform, $\delta\iota$, is calculated in Section 5.2.7. $\delta\theta$ has odd parity and thus no $m = n = 0$ mode.

5.2.7. The change in the rotational transform

$\delta\iota$ can be derived by considering the integral definition of the new rotational transform in the new coordinates,

$$\iota_1(\psi_1) = \int_0^{2\pi} \mathrm{d}\theta_1 \int_0^{2\pi} \mathrm{d}\phi_1 \; \mathscr{J}_1 \mathbf{B}_1 \cdot \nabla\theta_1. \tag{85}$$

After converting this to a volume integral, by integrating over a radial Dirac delta function, we can transform to the old coordinates, expand everything about the unperturbed values, and do the integration. After much algebra we obtain

$$\iota_1(\psi_1)|_{\psi_1=\psi^*} = [\mathscr{J}\,\delta B^\theta]_{0,0}(\psi^*) - \iota_0(\psi^*)[\mathscr{J}\,\delta B^\phi]_{0,0}(\psi^*) + \iota_0(\psi^*) - \iota_0'(\psi^*)\delta\psi_{0,0}(\psi^*), \tag{86}$$

where the right side is a function of ψ_0 evaluated at $\psi_0 = \psi^*$, and the left side is a function of ψ_1 evaluated at $\psi_1 = \psi^*$. Careful consideration of how these functions relate to each other gives the final result,

$$\delta\iota(\psi_0,\theta_0,\phi_0) = [\mathscr{J}\,\delta B^\theta]_{0,0}(\psi_0) - \iota_0(\psi_0)[\mathscr{J}\,\delta B^\phi]_{0,0}(\psi_0) + \iota_0'(\psi_0)\{\delta\psi(\psi_0,\theta_0,\phi_0) - \delta\psi_{0,0}(\psi_0)\}. \tag{87}$$

On Fourier decomposition a cancellation occurs in the last term, so that the $m = n = 0$ mode has to be treated as a special case:

$$\delta\iota_{0,0} = [\mathscr{J}_0\,\delta B^\theta]_{0,0} - [\iota_0\mathscr{J}_0\,\delta B^\phi]_{0,0}, \tag{88}$$

$$\delta\iota_{m,n} = \iota_0'\,\delta\psi_{m,n}, \quad (m,n) \neq (0,0). \tag{89}$$

$\delta\psi_{0,0}$ cancels out of the equation as it should (only quantities that are directly related to the two prescribed profiles can depend upon the absolute value of the radial coordinate, and hence on $\delta\psi_{0,0}$). Eq. (88) is consistent with the $m = n = 0$ mode of Eq. (84), from the $\delta\theta$ calculation.

5.2.8. The change in the prescribed profiles, P' and I'

The equilibrium free profile functions used in the code are the pressure and toroidal current profiles, $P'(\psi)$ and $I'(\psi)$. Newton's method needs the new profiles $P_1'(\psi_1)$ and $I_1'(\psi_1)$ in terms of the old ones, $P_0'(\psi_0)$ and $I_0'(\psi_0)$. This is quite simple so long as the profiles are specified in terms of the radial flux coordinate ψ and not the uniform radial coordinate ρ. They are actually specified in terms of ρ in the PIES Picard algorithm but this generates non-flux-surface-local terms in the Newton method, because the normalization of ρ implies knowledge of the edge-value at the next iteration. In contrast to all the other quantities we have considered, the functional form of these profiles is the same in both the old and the new coordinate systems; the difference between the perturbed and unperturbed profiles is due solely to the change in the coordinates,

$$\delta\mathscr{P} \equiv \delta(P') = P_0''\delta\psi, \tag{90}$$

$$\delta\mathscr{I} \equiv \delta(I') = I_0''\delta\psi. \tag{91}$$

5.2.9. The change in $\mathscr{J}_{0,0}$

The derivation of the change in the stream function, in Section 5.2.10, requires the $m = n = 0$ mode of the Jacobian in, and with respect to, both the old and the new coordinates. These are related by

$$[\mathscr{J}_1]_{0,0}^{(1)} = [\mathscr{J}_0]_{0,0}^{(0)} + \delta\xi, \tag{92}$$

where $\delta\xi$ is shorthand for $\delta(\mathscr{J}_{0,0})$, which is not the same as the $m = n = 0$ mode of the change in the Jacobian, $\delta\mathscr{J}_{0,0} \equiv (\delta\mathscr{J})_{0,0}$. The parenthetic superscripts indicate whether the Fourier mode numbers refer to the new or old coordinate systems. In the absence of any such superscript, the "0" case is implied, which is the usual situation. The left side of Eq. (92) can be written in terms of the old Jacobian,

$$[\mathscr{J}_1]_{0,0}^{(1)} = [\mathscr{J}_0]_{0,0}^{(1)} + [\delta\mathscr{J}]_{0,0}^{(1)}. \tag{93}$$

To first order, the second term on the right here is just $\delta\mathscr{J}_{0,0}^{(0)}$, which we have already calculated. We need the first term on the right, as a function of the old coordinates. Its integral definition is

$$[\mathscr{J}_0]_{0,0}^{(1)} = \int\int_0^{2\pi} \mathscr{J}_0 \mathrm{d}\theta_1\,\mathrm{d}\phi_1. \tag{94}$$

The method here is similar to that used for $\delta\iota$: integrate over all three coordinates using a radial Dirac delta function, then change integration variables and evaluate the integral in the old coordinates. After some algebra we get

$$\begin{aligned}[\mathscr{J}_1]_{0,0}^{(1)}(\psi_0,\theta_0,\rho_0) = {} & [\mathscr{J}_0]_{0,0}(\psi_0) - \left[\frac{\partial\mathscr{J}_0}{\partial\psi_0}\delta\psi\right]_{0,0}(\psi_0) + \left[\mathscr{J}_0\frac{\partial\,\delta\theta}{\partial\theta_0}\right]_{0,0}(\psi_0) - [\mathscr{J}_0^2\delta B^\phi]_{0,0}(\psi_0) \\ & + \left[\frac{\partial\mathscr{J}_0}{\partial\psi_0}\right]_{0,0}(\psi_0)\delta\psi(\psi_0,\theta_0,\rho_0),\end{aligned} \tag{95}$$

where the left side is a Fourier coefficient with respect to the new coordinates expressed as a function of the old coordinates. The same result can also be derived by considering the volume within a flux surface of $\mathbf{B}_1$ at $\psi_1 = \psi^*$:

$$V(\psi^*) = \int\int_0^{2\pi}\int_0^{\psi^*} \mathscr{J}_1(\psi_1,\theta_1,\phi_1)\,\mathrm{d}\psi_1\,\mathrm{d}\theta_1\,\mathrm{d}\phi_1. \tag{96}$$

The radial derivative of this is exactly what we need,

$$\frac{\mathrm{d}V}{\mathrm{d}\psi_1}(\psi^*) = \int\int_0^{2\pi} \mathscr{J}_1(\psi^*,\theta_1,\phi_1)\,\mathrm{d}\psi_1\,\mathrm{d}\theta_1 = [\mathscr{J}_1]_{0,0}^{(1)}(\psi^*). \tag{97}$$

Rather than change integration variables to the old system, in this case we note that the same volume must result from the following integral in the old coordinates:

$$V(\psi^*) = \int \int_0^{2\pi} \int_0^{\bar{\psi}_0(\psi^*,\theta_0,\phi_0)} \mathscr{J}_0(\psi_0,\theta_0,\rho_0)\,\mathrm{d}\psi_0\,\mathrm{d}\theta_0\,\mathrm{d}\phi_0. \tag{98}$$

The radial integration limit now depends on the angular coordinates because we are integrating out to a constant ψ_1 surface, not to a constant ψ_0 surface. The limit $\bar{\psi}_0$ is therefore the value of ψ_0 on that ψ_1 surface. Now we proceed to take the radial derivative using the technique of differentiating an integral with respect to a parameter. Working through the calculation eventually yields the same result as Eq. (95), from which $\delta\xi$ can be extracted,

$$\delta\xi_{m,n} = \left[\frac{\partial \mathscr{J}_0}{\partial \psi_0}\right]_{0,0} \delta\psi_{m,n}, \tag{99}$$

$$\delta\xi_{0,0} = \mathscr{J}'_{0,0}\,\delta\psi_{0,0} - [\mathscr{J}'_0\,\delta\psi]_{0,0} - [\mathscr{J}_0\,\delta\psi']_{0,0}. \tag{100}$$

$\delta\psi$, and thereby $\delta\mathbf{B}$, can be brought out of the convolution terms on the right by the methods of Section 4.2.1. The first term, proportional to $\delta\psi_{0,0}$, cancels with a term in one of the convolution sums ($\delta\xi$ is just the change in a specific Fourier coefficient of the Jacobian; it should not depend directly on either of the prescribed profiles, and consequently it should not depend on $\delta\psi_{0,0}$). The final expression is rather large, but the rest is just algebra.

5.2.10. The change in the stream function

The defining equation for the periodic part of the current density stream function is an MDE, Eq. (66). Its $m = n = 0$ mode is identically zero, which gives a relation between the three profile quantities on the right side. Substituting that information back into the original equation gives

$$\mathscr{J}(\mathbf{B}\cdot\nabla)v = P'(\mathscr{J} - \mathscr{J}_{0,0}). \tag{101}$$

Writing this in the new system, doing the usual first-order expansion, and Fourier decomposing in the old coordinates gives

$$(n - \imath m)\delta v_{m,n} = P'_0\,\delta\xi_{m,n} + [\delta\mathscr{P}(\mathscr{J} - \mathscr{J}_{0,0})]_{m,n} - [\mathscr{J}\,\delta\mathbf{B}\cdot\nabla v_0]_{m,n} - P'_0\mathscr{J}_{0,0}[\mathscr{J}\,\delta B^{\phi}]_{m,n}. \tag{102}$$

All terms on the right are now known quantities although, once again, much unraveling of analytic convolutions must be done to put $\delta v_{m,n}$ in its final form.

5.2.11. The change in the poloidal current profile

The free equilibrium profiles for pressure and toroidal current are code inputs, and the poloidal current profile is then determined by the $m = n = 0$ mode of Eq. (66):

$$g' + \imath I' + \mathscr{J}_{0,0}P' = 0. \tag{103}$$

The poloidal current profile varies with $\mathbf{B}$ as

$$g'_0 \equiv \frac{\mathrm{d}g_0}{\mathrm{d}\psi_0} \to g'_1 \equiv \frac{\mathrm{d}g_1}{\mathrm{d}\psi_1} = g'_0 + \delta\mathscr{G}. \tag{104}$$

Writing Eq. (103) in the perturbed system gives

$$g'_0 + \delta\mathscr{G} + (\imath_0 + \delta\imath)(I'_0 + \delta\mathscr{I}) + (\mathscr{J}_{0,0} + \delta\xi)(P'_0 + \delta\mathscr{P}) = 0. \tag{105}$$

In terms of quantities we have already calculated, this reduces to

$$\delta\mathscr{G}_{m,n} = -\imath_0\,\delta\mathscr{I}_{m,n} - \mathscr{J}_{0,0}\,\delta\mathscr{P}_{m,n} - I'_0\,\delta\imath_{m,n} - P'_0\,\delta\xi_{m,n}. \tag{106}$$

5.2.12. The change in $\partial v/\partial\theta$

The θ-derivative of the current density stream function varies with **B** as

$$\frac{\partial v_0}{\partial\theta_0} \to \frac{\partial v_1}{\partial\theta_1} = \frac{\partial v_0}{\partial\theta_0} + \delta\mathcal{N}, \tag{107}$$

where we have defined

$$\delta\mathcal{N} \equiv \delta\left(\frac{\partial v}{\partial\theta}\right) = \frac{\partial v_1}{\partial\theta_1} - \frac{\partial v_0}{\partial\theta_0}. \tag{108}$$

$\partial v_1/\partial\theta_1$ can be evaluated in the old coordinates by using a chain rule for the partial derivative and noting that the following is true to first order:

$$\frac{\partial\,\delta\theta}{\partial\theta_1} = \frac{\partial\,\delta\theta}{\partial\theta_0} \tag{109}$$

and similarly for the ψ-derivative. Therefore,

$$\delta\mathcal{N} = \frac{\partial\,\delta v}{\partial\theta_0} - \frac{\partial v_0}{\partial\psi_0}\frac{\partial\,\delta\psi}{\partial\theta_0} - \frac{\partial v_0}{\partial\theta_0}\frac{\partial\,\delta\theta}{\partial\theta_0}, \tag{110}$$

where δv has already been calculated, and the last two terms are analytic convolutions that can be unraveled as per Section 4.2.1.

5.2.13. Method 1 summary

The final expression for $\delta\mathbf{J}_{m,n}(\psi_0)$, obtained by combining the results from Sections 5.2.1 through 5.2.12, is far too large to print here: around 10 pages of text [14]. The algebra becomes very complicated for several reasons: there are interdependencies between the many terms, many convolutions need to be unraveled, the $m = n = 0$ coefficients are generally special cases, everything must be expressed in terms of the components of $\delta\mathbf{A}$ rather than $\delta\mathbf{B}$, radial factors must be extracted from every Fourier coefficient, and all radial derivatives discretized in the manner of Eqs. (50) and (51). The full derivation was done twice, once using the symbolic algebra capability of Mathematica [15] as far as possible, and the two versions were checked exhaustively against each other to verify correctness. To avoid transcription errors we had Mathematica write out the result directly in Fortran code.

5.3. Derivation of $\delta\mathbf{J}$, Method 2

An alternative derivation of the Newton gradient term begins by directly varying the coordinate-independent equation for the current density that satisfies the force balance equation, Eq. (11),

$$\delta\mathbf{J} = \mu\delta\mathbf{B} + \frac{\delta\mathbf{B}\times\nabla P}{B^2} - 2\mathbf{B}\cdot\delta\mathbf{B}\frac{\mathbf{B}\times\nabla P}{B^4} + \mathbf{B}\delta\mu + \frac{\mathbf{B}\times\nabla\delta P}{B^2}. \tag{111}$$

The first three terms here are already of the right form, resulting in matrices operating on coefficients of $\delta\mathbf{B}$, and they can be straightforwardly discretized in *any* coordinate system. The two final terms, however, can only be expressed in terms of $\delta\mathbf{B}$ by solving MDEs for $\delta\mu$ and δP, which still requires specializing to magnetic coordinates. Nevertheless, this method offers some opportunity to devise a Newton algorithm that is less dependent on magnetic coordinates (see Section 6.5), as well as an independent check on the very complicated result of Section 5.2.

The magnetic differential equation for δP arises from the flux-function nature of the scalar pressure ($\mathbf{B}\cdot\nabla P = 0$),

$$\mathbf{B}\cdot\nabla\delta P = -\delta\mathbf{B}\cdot\nabla P, \tag{112}$$

and the MDE for $\delta\mu$ is generated by varying Eq. (12),

$$\mathbf{B}\cdot\nabla\delta\mu = -\delta\mathbf{B}\cdot\nabla\mu - \nabla\delta P\cdot\nabla\times\left(\frac{\mathbf{B}}{B^2}\right) - \nabla P\cdot\nabla\times\left(\frac{\delta\mathbf{B}}{B^2}\right) + 2\nabla P\cdot\times\left(\frac{\mathbf{B}\,\mathbf{B}\cdot\delta\mathbf{B}}{B^4}\right). \tag{113}$$

Following this method, and using similar techniques to those of the first derivation, the expression for $\delta\mathbf{J}$ that emerges is considerably smaller, but the algebra is still extensive and the final result is still far too large to print here. The two versions of the gradient term have been compared to each other to check for errors in the derivations, and both are available for use in the PIES Newton algorithm.

6. Results: convergence properties

We compare the performance of the new Picard and Newton vector potential schemes with the original PIES Picard scheme in this section. Initial Picard iteration tests are described in Section 6.1, then the results of simple linear Newton tests, which can be done in background coordinates, are given in Section 6.2. The basic Newton scheme of Section 3 encountered problems due to noise growth in the magnetic coordinates, prompting us to develop two Newton method variants to recover some of the benefits of solving Ampere's Law in fixed background coordinates. These Newton variants are described in Sections 6.3–6.5, and summarized briefly in Section 6.6. The main Newton code results are presented in Sections 6.7–6.9. PIES can now operate in its original Picard configuration, or in any of the new vector potential modes (Picard iteration for $\mathbf{A}$ or $\delta\mathbf{A}$, the basic Newton scheme of Section 3, or the variant Newton schemes described later in this section).

6.1. Initial Picard iteration tests

The $\nabla\times\nabla\times\mathbf{A}$ discretization was first tested independently of PIES using analytically specified coordinate geometry and Mathematica to verify the second-order accuracy of the radial finite-differencing, etc. This confirmed the quadratic convergence of Fourier modes with respect to radial resolution. Then, within the PIES code, the iterative convergence properties of the new Picard scheme proved to be almost indistinguishable from those of the original scheme, even for complicated 3D equilibria with magnetic islands and stochastic regions. This validated our coding of substantial parts of the Newton algorithm: the $\nabla\times\nabla\times\delta\mathbf{A}$ discretization, gauge choice, radial dependence predictions for $\delta\mathbf{A}$, and custom sparse matrix solver.

6.2. Linear cases in background coordinates

When the plasma pressure vanishes the current density is parallel to the magnetic field: $\mathbf{J} = \mu(\mathbf{B})\mathbf{B}$. Newton's method should converge in a single iteration in such linear cases. Further, if $I'(\psi)$ is flat then μ is constant and the Newton gradient term is vastly simplified,

$$\delta\mathbf{J} = \frac{\partial\mathbf{J}}{\partial\mathbf{B}}\cdot\delta\mathbf{B} = \lambda\delta\mathbf{B} = \lambda\nabla\times\delta\mathbf{A}. \tag{114}$$

$\delta\mathbf{J}$ is entirely coordinate-independent here so Ampere's Law solver, Eq. (21), can run in fixed background coordinates. The test case of Fig. 3, a linear cylindrical equilibrium, shows that the algorithm can indeed converge to machine precision in one iteration. At very low resolution, however, convergence is delayed slightly because the current density is apparently not adequately represented. Picard converges slowly in this case, albeit without any blending. In general, Newton's convergence rate exhibits a strong dependence on resolution, as in the linear toroidal example of Fig. 4, whereas Picard's is unaffected. The saturation in

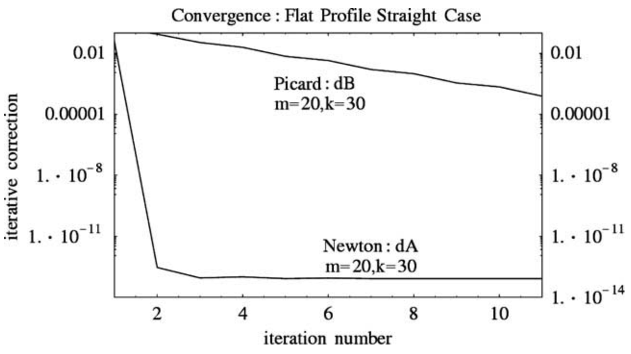


Fig. 3. Convergence of the Newton and Picard codes, as measured by the iterative correction, for a linear test case ($\mathbf{J} = \lambda\mathbf{B}$, the best case scenario for Newton). The equilibrium is cylindrical, aspect ratio = 0.5, poloidal elongation 1.5.

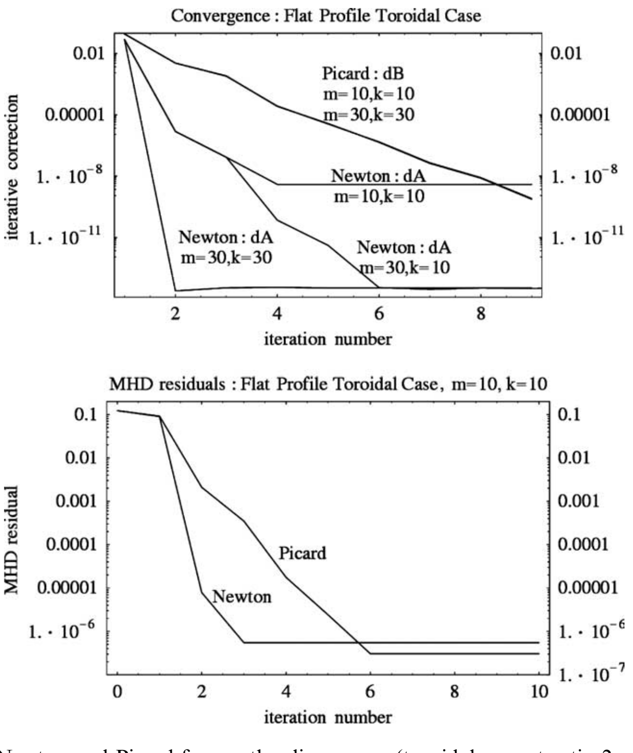


Fig. 4. Upper: convergence of Newton and Picard for another linear case (toroidal, aspect ratio 2, poloidal elongation 1.5). Lower: MHD residuals $\|\mathbf{J} \times \mathbf{B} - \nabla P\|$ for the lowest resolution case.

convergence at low angular resolution shown in the lower figure can be understood as follows. When the *curl* of the old field, on the right side, gets interpolated forward to the new grid at the next iteration, it goes through a transformation from polar vector components on the old grid through background Cartesian components to polar components on the new grid. As the field converges, so do the coordinates, but even when convergence is attained this interpolation is only perfect in the limit of infinite angular resolution because the metric elements (which are convolutions of unit vectors) have full Fourier series even if the

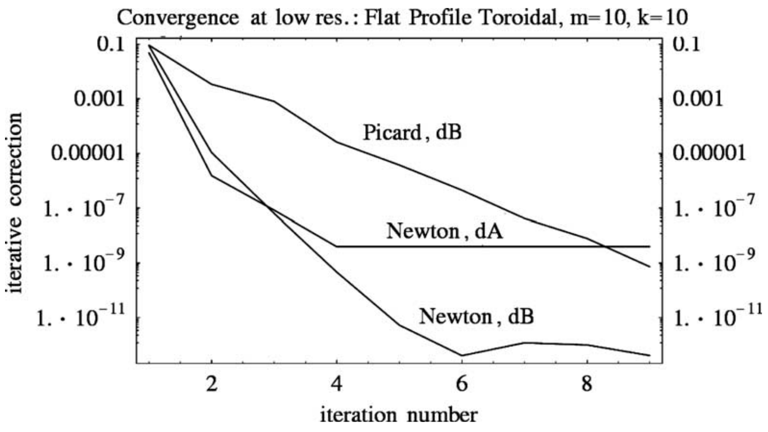


Fig. 5. Magnetic field and vector potential iterative corrections for the lowest resolution case in Fig. 4.

coordinate geometry contains only low-order modes. Thus the minimum size of $\delta\mathbf{A}$ is of the order of the minimum difference between $\mathbf{J}_i$ and $\mathbf{J}_{i-1}$, the angular resolution dependent error accrued by $\mathbf{J}_{i-1}$ in this interpolation. In fact, $\delta\mathbf{A}$ converges to this value, while the magnetic field correction continues converging toward machine precision, as shown in Fig. 5. When the angular resolution is insufficient the final converged equilibrium fields are not exactly the same as those obtained by the Picard code; they are modified by the remaining non-zero $\nabla \times \delta\mathbf{A}$ (the two solutions are mutually convergent with respect to resolution, however).

6.3. Magnetic coordinates and general equilibria

For finite plasma pressure and general current profiles we are forced to use the full Newton gradient term of Section 5 and magnetic coordinates (this is achieved by simply disabling conversion to the background grid after computing $\mathbf{J}$) in solving in Eq. (21). Unfortunately the basic Newton algorithm performed very poorly in magnetic coordinates for anything other than simple circular cylindrical equilibria, in which only the $m = 0$ modes are significant and the magnetic and background grids coincide. Even simple linear toroidal cases, like those of the previous section, failed to converge at all in magnetic coordinates. This could not be due to coding errors because the linear Newton gradient term is quite trivial and its correctness can be absolutely guaranteed. In fact, the Picard algorithm, which converges well in background coordinates in these cases, does not converge in magnetic coordinates either. While this is a known issue and the reason why Picard-PIES now solves Ampere's Law in background coordinates, we had hoped that Newton's stronger convergence might overcome the problem. A major deficiency of magnetic coordinates is the difficultly in getting good angular resolution in certain regions, notably the outer mid-plane, in cases with significant plasma pressure, low aspect ratio, or strong boundary shaping, because of spreading of the lines of constant θ (use of equal arc-length magnetic coordinates might help, but for now the choice of a uniform ϕ-coordinate is deeply entrenched within PIES). In addition, when the entire code runs in magnetic coordinates, the unbroken coupling between the magnetic field and the coordinates leads to numerical noise growth. Solving Ampere's Law is an integrating operation that tends to smooth noise in the solution vector, but any remaining noise ends up in the coordinates at the next iteration, and any noise in the coordinates is not smoothed by integration but is transferred on to the next iteration through the grid interpolation process. The noise problem tends to arise near the origin because the radial finite differencing of high-order Fourier coefficients is less accurate there. After several iterations in magnetic coordinates the noise swamps convergence and begins to cause trouble for the axis-finding routines. Newton's method in fact has even greater potential for noise growth because the smoothing integration operates only on $\delta\mathbf{A}$ rather than the full field.

6.4. The two-pass Newton algorithm

A modified Newton algorithm allows us to recover some of the benefits of using a fixed background coordinate system, by solving Ampere's Law twice in each iteration. The first pass through the Ampere's Law solver is unchanged, in magnetic coordinates, but the solution vector $\delta\mathbf{A}$ is only used to compute the Newton gradient term. $\delta\mathbf{J}$ is then transferred to the right side of the system as a known quantity and we re-solve for the full vector potential in background coordinates. This effectively allows the noise-smoothing integration to operate on the full field in each iteration,

$$\begin{aligned} &\quad\text{(magnetic coordinates)}\\ \mathbf{J}_i(\mathbf{B}_{i-1}) &\Leftarrow \mathbf{J}_i(\mathbf{B}_{i-1}) \times \mathbf{B}_{i-1} = \nabla P_i, \nabla\cdot\mathbf{J}_i(\mathbf{B}_{i-1}) = 0, \qquad (115)\\ &\quad\text{pass 1 (magnetic coordinates)}\\ \underline{\delta\mathbf{J}_i} = \mathscr{L}_i\cdot\delta\mathbf{A}_i &\Leftarrow \nabla\times\nabla\times\delta\mathbf{A}_i - \mathscr{L}_i\cdot\delta\mathbf{A}_i = \mathbf{J}_i - (\mathbf{J}_{i-1} + \delta\mathbf{J}_{i-1}), \qquad (116)\\ &\quad\text{pass 2 (background coordinates)}\\ \mathbf{B}_i = \nabla\times\mathbf{A}_i &\Leftarrow \nabla\times\nabla\times\mathbf{A}_i = \mathbf{J}_i + \underline{\delta\mathbf{J}_i}. \qquad (117) \end{aligned}$$

Either form of $\delta\mathbf{J}$, from Section 5.2 or 5.3, can be used. While this scheme is more difficult to implement than the original single-pass version it does not lead to a large drop in efficiency because the main computational bottleneck is still in the field-line mapping code at the front end of the current density solver.

Note that $\mathbf{J}_{i-1} + \delta\mathbf{J}_{i-1}$ on the right of Eq. (116) is just the *curl* of the old magnetic field. We use it in this form, rather than $\nabla\times\mathbf{B}_{i-1}$ or $\nabla\times\nabla\times\mathbf{A}_{i-1}$, because the other choices involve differentiation of the new field after it has been interpolated forward to the new coordinate grid, which results in detrimental numerical noise growth (this applies to Newton's method and the $\delta\mathbf{B}$ form of the Picard scheme). One could potentially take the derivatives before transforming to the new grid but that is incompatible with our particular offset radial grid discretization and, in addition, our *curl curl* operator would not be valid on the new grid because the vector potential would acquire a radial component in the coordinate transformation. The extra (radial) component of the Newton gradient is needed in order to put $\delta\mathbf{J}_{i-1}$ in the new coordinates but it is at least much simpler to derive than the other two components.

It should also be noted that the *curl* of the old magnetic field must be interpolated forward from the first (magnetic coordinate) pass of Ampere's solver at the previous iteration, not taken from the background pass. Otherwise the algorithm cannot converge because of the small resolution-dependent error incurred in transforming between the background and magnetic grids (whereas the two magnetic grids mutually converge as the field converges). One further modification allows the two-pass Newton algorithm to converge to exactly the same result as the Picard algorithm: instead of taking $\mathbf{J}_i$ directly from the magnetic to the background grid for *pass 2* we calculate the parallel component of the current density μ (Eq. (11)) and then transform it and $\mathbf{B}$ to the background grid. $\mathbf{J}$ can then be reconstructed on the background grid from μ and $\mathbf{B}$. PIES uses $\mathbf{J}$ in this form to facilitate the inclusion of magnetic island physics, and it also yields a smaller MHD residual because the original perpendicular current density cancels from the residual vector product.

The validity of our analytically specified Newton gradient term depends on having good magnetic coordinates, whose accuracy in turn depends on factors such as the field line following distance and tolerance, proximity to resonant surfaces, and the precision of the fast Fourier transforms that yield the Fourier coefficients on each surface. In simple two-dimensional cases there is a clear correlation between

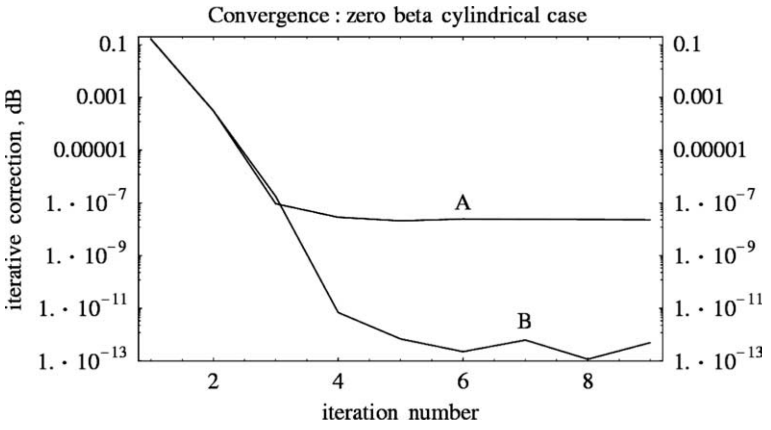


Fig. 6. Newton's method convergence for values of the PIES field-line following parameter `ftfol` (5.0×10^{-5} in case A, and 5.0×10^{-12} in case B). The distance followed is proportional to the logarithm of the inverse of `ftfol`. The case shown is a circular cross-section cylindrical equilibrium with a non-flat current profile.

the field line following distance (determined by a parameter `ftfol` in the code) and the maximum level to which the Newton method converges, as illustrated in Fig. 6. This effect is likely to be even more important with more complex field geometry, but it tends to be obscured by other factors that reduce the level of convergence.

6.5. The two-pass Newton algorithm, Version 2

The second derivation of the Newton gradient, Section 5.3, also allows a two-pass Newton variant with minimal use of magnetic coordinates. Rather than computing the full Newton gradient from the first pass, we can use the solution vector to compute just δP and $\delta\mu$ from the two terms in $\delta\mathbf{J}$ that depend explicitly on the magnetic coordinates. These can then be transferred to the right side as known quantities, leaving the other terms on the left, before solving the system again on the background grid. The positive effect of minimal magnetic coordinate use, however, appears to be countered by the less effective iterative noise reduction when solving for $\delta\mathbf{A}$ instead of $\mathbf{A}$, and consequently both of the two-pass algorithms perform similarly. The second derivation of $\delta\mathbf{J}$ also suggests a modified Picard, or partial Newton scheme: if just the coordinate-independent parts of $\delta\mathbf{J}$ were used as extra linear terms in the Picard Ampere's Law equation, they might help speed convergence in some cases. This has not been extensively tested, however. As coded into PIES, the second two-pass Newton algorithm can be written,

$$\text{(magnetic coordinates)}$$
$$\mathbf{J}_i(\mathbf{B}_{i-1}) \Leftarrow \mathbf{J}_i(\mathbf{B}_{i-1}) \times \mathbf{B}_{i-1} = \nabla P_i, \nabla \cdot \mathbf{J}_i(\mathbf{B}_{i-1}) = 0, \tag{118}$$
$$\text{pass 1 (magnetic coordinates)}$$
$$\underline{\delta\mu_i}, \underline{\delta P_i} \Leftarrow \delta\mathbf{A}_i \Leftarrow \nabla \times \nabla \times \delta\mathbf{A}_i - \mathscr{L}_i \cdot \delta\mathbf{A}_i = \mathbf{J}_i - \nabla \times \mathbf{B}_{i-1}, \tag{119}$$
$$\text{pass 2 (background coordinates)}$$
$$\begin{aligned}\mathbf{B}_i = \mathbf{B}_{i-1} + \nabla \times \delta\mathbf{A}_i \Leftarrow & \nabla \times \nabla \times \delta\mathbf{A}_i - \mu_i \nabla \times \delta\mathbf{A}_i - \frac{\nabla \times \delta\mathbf{A}_i \times \nabla P_i}{B_{i-1}^2} + 2\mathbf{B}_{i-1} \cdot \nabla \times \delta\mathbf{A}_i \frac{\mathbf{B}_{i-1} \times \nabla P_i}{B_{i-1}^4} \\ & = \mathbf{J}_i + \mathbf{B}_{i-1}\underline{\delta\mu_i} + \frac{\mathbf{B}_{i-1} \times \nabla \underline{\underline{\delta P_i}}}{B_{i-1}^2} - \nabla \times \mathbf{B}_{i-1}.\end{aligned} \tag{120}$$

Here, the *curl* of the old magnetic field is given by

$$\nabla \times \mathbf{B}_{i-1} = \mathbf{J}_{i-1} + \mu_{i-1}\nabla \times \delta\mathbf{A}_{i-1} + \frac{\nabla \times \delta\mathbf{A}_{i-1} \times \nabla P_{i-1}}{B_{i-2}^2} - 2\mathbf{B}_{i-2} \cdot \nabla \times \delta\mathbf{A}_{i-1}\frac{\mathbf{B}_{i-2} \times \nabla P_{i-1}}{B_{i-2}^4} + \mathbf{B}_{i-1}\delta\mu_{i-1} + \frac{\mathbf{B}_{i-2} \times \nabla\delta P_{i-1}}{B_{i-2}^2}.$$

6.6. Relative performance of the three Newton variants

To summarize, the original pure magnetic coordinate formulation of Newton's method performed poorly for reasons described in detail in Section 6.3. This motivated the development of the first two-pass Newton algorithm (Section 6.4) which is designed to control magnetic noise growth by solving Ampere's Law twice in each iteration. First we solve for the field increment in magnetic coordinates and use it only to compute the Newton gradient term. The gradient term is then transferred to the right hand side as a known quantity and we re-solve the system for the full field in background coordinates. This dramatically improves the performance of the Newton code, as can be seen from the results to follow. The second two-pass variant (Section 6.5) was tried because it further minimizes the use of magnetic coordinates by taking from the first pass only those sub-terms of the Newton gradient that depend inextricably on the magnetic coordinates. In the second pass, however, we can no longer solve for the full field; this has a competing detrimental effect and the method consequently shows no significant improvement over the first two-pass algorithm. The remaining results presented in the paper were therefore generated with the first two-pass method.

6.7. Zero-beta tokamak equilibria

For two-dimensional cases with no plasma pressure, Newton's method works very well in comparison to Picard iteration. For example, Fig. 7 compares Newton and Picard for a zero-β D III-D tokamak equilibrium (β is the volume averaged ratio of plasma pressure and squared field strength). The slow start by Newton suggests that the initial guess was not sufficiently good. We note here that the processor time per iteration is only fractionally larger for the Newton code, in general, because the Ampere's Law solver in PIES, whether Picard or Newton, is relatively fast in comparison to the field-line mapping code at the start of each iteration.

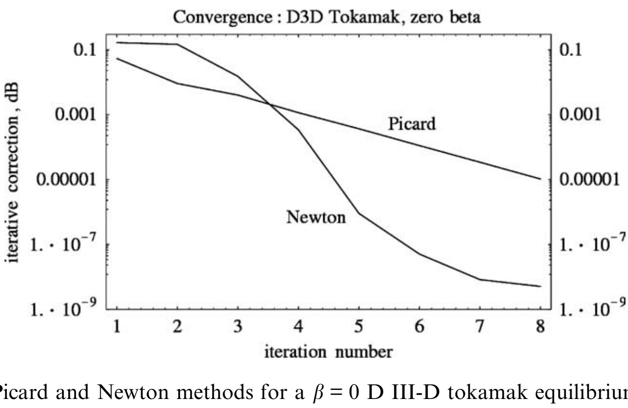


Fig. 7. Convergence of the Picard and Newton methods for a $\beta = 0$ D III-D tokamak equilibrium. Resolution $k = 30$, $m = 30$.

6.8. Finite plasma pressure

Finite plasma pressure makes the equilibrium equations increasingly nonlinear, and it becomes difficult to get good angular resolution at the outer midplane because of distortion in the magnetic coordinates. In toroidal geometry the magnetic axis is also pushed outward, which exacerbates this problem. Pressure-dependent parts of the Newton gradient term, however, can be comprehensively tested on non-toroidal cases because they are only weakly dependent on geometry through the Jacobian (in contrast, the *curl curl* operator is strongly dependent on the geometry, but was easily tested by comparing the two Picard schemes). In fact, β can easily be driven so high in non-toroidal cases that pressure-dependent terms totally dominate. Results for a cylindrical case of ellipticity 1.665 at resolution $k = 20$, $m = 10$ are shown in Figs. 8 and 9. The Newton code's performance deteriorates with increasing pressure, and it ceases to be quadratically convergent, but it is always faster and more stable than raw Picard iteration, which eventually fails at $\beta = 460\%$. Lowering the resolution from that in Figs. 8 and 9 adversely affects the rate of convergence, while moderately higher resolution improves it, but not a lot. This is possibly an indication that the increasing nonlinearity at high β shrinks the radius of quadratic convergence to such an extent that we do not reach it before numerical noise begins to cause problems.

6.9. Simple 3D stellarator equilibria

Stellarators get rotational transform from external coils rather than net toroidal current, and they consequently have vacuum field solutions. The Picard algorithm solves for stellarator vacuum fields in a single

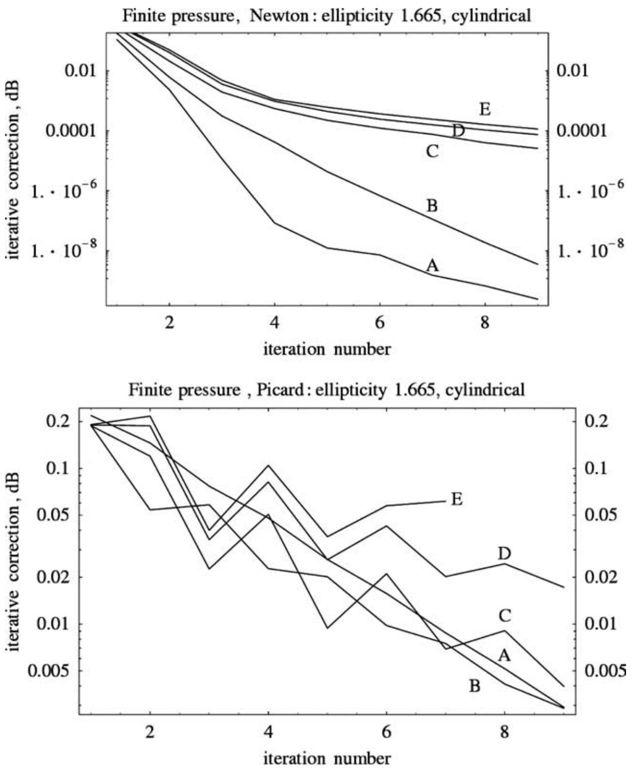
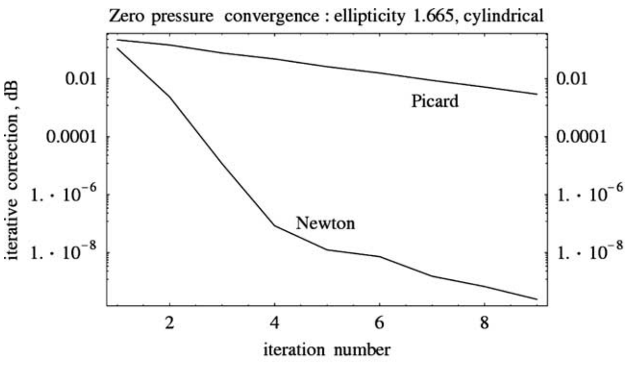


Fig. 8. Comparison of Newton (top) and Picard (bottom) in the elliptical cylindrical case at various high plasma pressures: volume average $\beta = 0$ (A), 85% (B), 276% (C), 398% (D), and 460% (E). Picard fails at iteration 7 of case E.



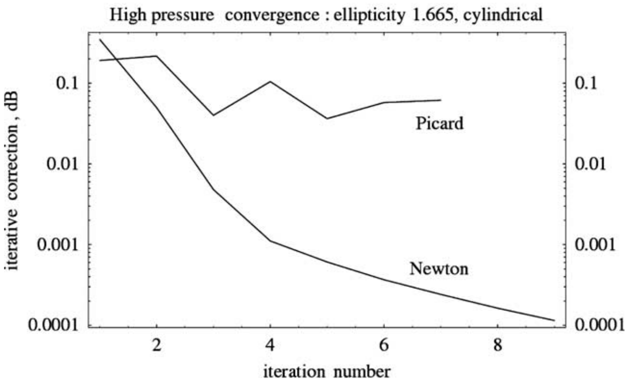


Fig. 9. Newton and Picard compared on the same scale for the two extreme cases (top: $\beta = 0$, and bottom: $\beta = 460\%$) from Fig. 8. The Picard code fails at iteration 7 in the high pressure case.

iteration, but sometimes increasing plasma pressure causes a rapid transition to equilibria that require a lot of blending to get convergence. The Newton method may be useful in these cases. Application of our code to 3D equilibria, however, currently requires that we avoid low-order resonant surfaces where islands chains open up, because the gradient term assumes good magnetic surfaces throughout the plasma (extending the Newton code to deal with magnetic islands should be possible; see Section 7). Low-order resonances can be avoided by adjusting the profiles, but this limits resolution because there are always islands and stochastic regions in 3D equilibria if one looks on a sufficiently fine scale [16]. Coordinate surfaces in close proximity to rational surfaces (where $\iota(\rho) = m/n$) might also cause problems because the Newton gradient term contains many resonant denominators. We have therefore used a resonance broadening technique to modify the denominators,

$$\frac{1}{n - \iota(\rho)m} \rightarrow \frac{n - \iota(\rho)m}{(n - \iota(\rho)m)^2 + \epsilon^2}, \tag{121}$$

where ϵ is a small parameter that can be chosen to smooth the resonance over a few radial grid points. Away from the immediate vicinity of the resonance this accurately represents the original function.

The extinct Wendelstein VII-A stellarator, a relatively uncomplicated five-period device with a large aspect ratio of 20, provides a convenient 3D test for Newton's method. The outer flux surface geometry is specified by

$$x = \frac{1}{2}[(1-e)\cos(2N\phi - \theta) + (1+e)\cos(\theta)], \tag{122}$$

$$y = \frac{1}{2}[(1-e)\sin(2N\phi - \theta) + (1+e)\sin(\theta)], \tag{123}$$

where $e > 1$ is the elliptical elongation. This corresponds to an elliptical poloidal cross-section that rotates in ϕ while the lines of constant θ do not rotate about the axis. We were restricted to very low resolution in order to avoid low-order rational surfaces (above). Low resolution was detrimental in 2D cases, but Newton's method nevertheless showed some promise here. Fig. 10 compares convergence of Newton and Picard for a volume-averaged β of 10%, for a straightened version of the stellarator with cross-sectional ellipticity of 1.6 to give the characteristic rotational transform of about 0.55 on axis. Newton's method provides a factor of two improvement over Picard, in terms of number of iterations, before it saturates due to the very low resolution and insufficient field-line following. Similar convergence results for the five-period toroidal case is shown in Fig. 11 (β = 1%) and Fig. 12 (β = 2%). The poloidal cross-section has an ellipticity of 1.665, giving an on-axis rotational transform of about 0.55, with low shear. The β = 2% case has a 26% magnetic axis shift. It is very much under-resolved at k = 20, m = 5, n = 4 ($\delta\mathbf{A}$ saturates after a few iterations for the reasons given in Section 6.2), but the Newton method still performs better than Picard iteration. Finally, Fig. 13 shows a zero β case with finite net current and lower ellipticity to keep the rotational transform

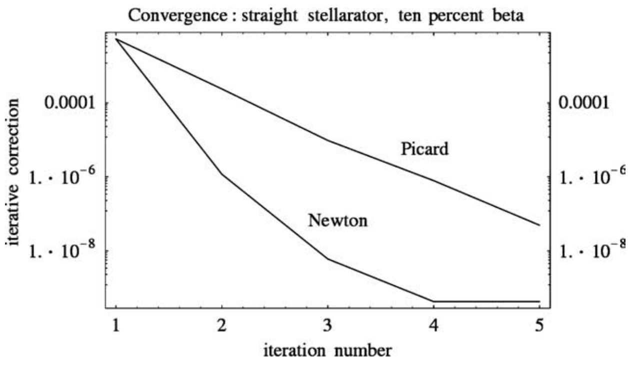


Fig. 10. Convergence results for the straightened W VII-A five period stellarator case, β = 10%. Resolution is low: m = 5, n = 4, k = 20.

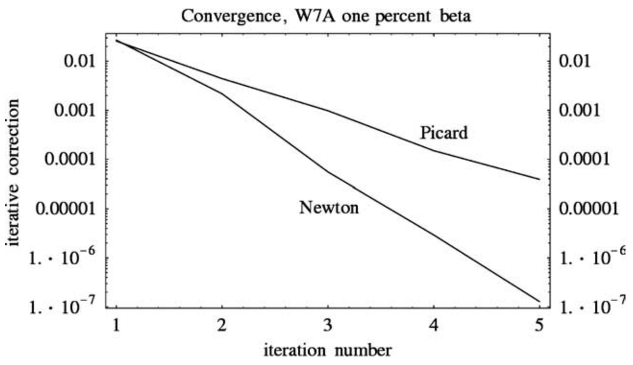


Fig. 11. Convergence to a Wendelstein VII-A equilibrium at β = 1%. Resolution is low k = 20, m = 5, n = 4 (with a factor of two angular padding for dealiasing).

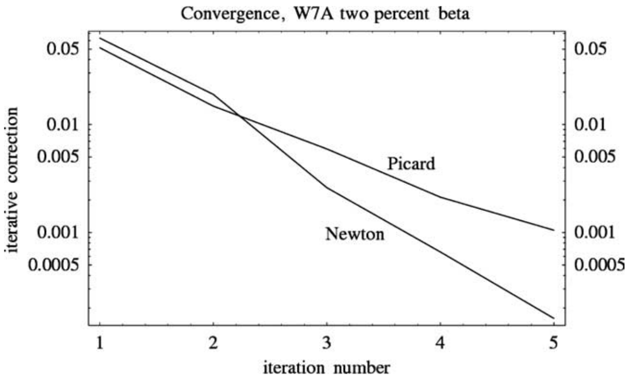


Fig. 12. Convergence to a Wendelstein VII-A equilibrium at $\beta = 2\%$. Resolution is low $k = 20$, $m = 5$, $n = 4$ (with a factor of 2 angular padding for dealiasing).

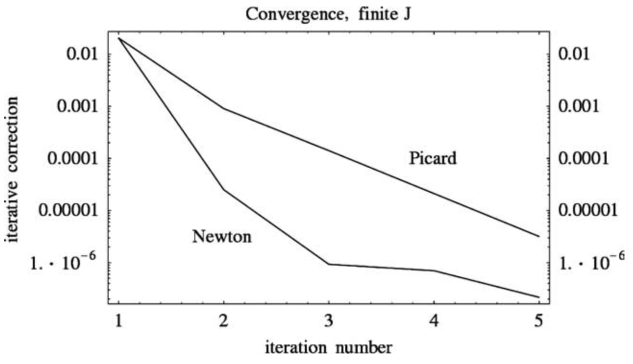


Fig. 13. Convergence to a zero β Wendelstein VII-A equilibrium, with finite net current. Resolution is low $k = 20$, $m = 5$, $n = 4$ (with a factor of two angular padding for dealiasing).

at a reasonable level. The two algorithms converge with the same MHD residuals and, by inspection of the magnetic field, to the same solution. The mutual quadratic convergence of the two solutions with respect to radial resolution has been confirmed for various modes.

7. Conclusions

We have implemented a (**J**, **B**) space Newton method for the 3D MHD equilibrium equations, in toroidal geometry, as an extension to the hybrid spectral/finite-difference Princeton Iterative Equilibrium Solver (PIES). Computing the Newton gradient term (functional derivative) numerically in a hybrid spectral code is not feasible but we avoid this problem through an analytic derivation, in magnetic coordinates, of the way in which the plasma's pressure-driven Pfirsch–Schlüter currents vary with the field. Deriving the gradient term in two different ways was a huge task; the final expressions are too large to print here, but they have been extensively validated with algebraic help from Mathematica, and by using test cases (cylindrical geometry, zero pressure, etc.) that isolate specific terms in the equations. Ampere's Law is formulated in terms of a vector potential in the $A_\rho = 0$ gauge, and discretized on offset radial grids with near origin leading-order radial behavior factored from Fourier coefficients to improve finite difference accuracy.

The resulting sparse linear system is internally block tridiagonal but globally sub-diagonal because of the way in which the vector potential's unusual radial dependence affects the boundary equations. A custom sparse matrix solver was built to solve the system.

In Picard iteration mode and fixed background coordinates the new vector potential algorithm behaves identically to the original PIES code. In the basic Newton mode, it fails to converge because of numerical noise in the magnetic coordinates required by our analytic gradient term (similarly, raw Picard iteration also fails in magnetic coordinates), but a modified "two-pass" Newton scheme recovers some of the benefits of using a fixed background grid, with dramatically improved performance. With sufficient resolution, linear cases converge in a single iteration. More general zero-β equilibria converge in many fewer iterations than Picard. In non-toroidal cases with extremely high plasma pressure the Newton method is a lot faster and more robust than Picard iteration, although its convergence is adversely affected by low resolution and imperfections in the magnetic coordinates. Several finite β 3D toroidal cases were run with parameters typical of the Wendelstein VII-A stellarator. In spite of the extremely low resolution imposed to avoid islands and stochastic regions, Newton converged in approximately half the number of iterations as Picard at $\beta = 1\%$, and was still significantly faster at $\beta = 2\%$. The sensitivity to initial guess typically displayed by Newton's method has not been problematic so far, but we can start the code with one or more Picard iterations if necessary.

The Newton code's performance is tantalizingly good in simple magnetic geometry. Its sensitivity to resolution and coordinate noise is of concern in more complicated cases, but it has the potential to be useful for computing those 3D equilibria that require massive blending to achieve convergence by Picard iteration. The Newton code must first be extended, in future work, to handle magnetic islands and stochastic regions. Magnetic coordinates can become quite distorted near islands but nevertheless this should not prove too difficult because the Newton gradient term is trivial in stochastic regions (where the equilibrium profiles are flat) and one could use the methods of Section 5 to predict the way in which an island's extent varies with the magnetic field.

Acknowledgment

This work was supported by United States DOE contract no. DE-AC02-76-CHO-3073.

References

[1] S.I. Braginskii, in: M.A. Leontovich, Transport Processes in a Plasma, Reviews of Plasma Physics 1, Consultants Bureau, New York, 1965, pp. 205–311.
[2] J.P. Freidberg, Ideal Magnetohydrodynamics, Plenum Press, New York, 1987.
[3] J.M. Finn, L. Sparks, Equilibria of Field Reversed Ion Rings, unpublished, Lab. Plasma Studies, Cornell University, 1977.
[4] G. Miller, V. Faber, A.B. White Jr., Finding plasma equilibria with magnetic islands, J. Comput. Phys. 79 (1988) 417–435.
[5] A.H. Reiman, H.S. Greenside, Calculation of three-dimensional MHD equilibria with islands and stochastic regions, Comput. Phys. Commun. 43 (1986) 157.
[6] D.A. Knoll, D.E. Keyes, Jacobian-free Newton–Krylov methods: a survey of approaches and applications, J. Comput. Phys. 193 (2004) 357–397.
[7] L. Spitzer, The stellarator concept, Phys. Fluids 1 (4) (1958) 253–264.
[8] H. Grad, H. Rubin, MHD equilibrium in an axisymmetric toroid, in: Proceedings of the 2nd UN Conference on the Peaceful Uses of Atomic Energy, Geneva, vol. 31, 1958, p. 190.
[9] A.H. Reiman, H.S. Greenside, Numerical solution of three dimensional magnetic differential equations, J. Comput. Phys. 75 (1988) 423–443.
[10] A.H. Boozer, Establishment of magnetic coordinates for a given magnetic field, Phys. Fluids 25 (1982) 520–521.
[11] A.H. Boozer, Three dimensional stellarator equilibria by iteration, Phys. Fluids 27 (1984) 2110–2114.

[12] H.S. Greenside, A.H. Reiman, A. Salas, Convergence properties of a nonvariational 3D MHD equilibrium code, J. Comput. Phys. 43 (1989) 102–136.
[13] A.H. Reiman, N. Pomphrey, Computation of magnetic coordinates and action angle variables, J. Comput. Phys. 94 (1991) 225–249.
[14] H.J. Oliver, A Newton method for the magnetohydrodynamic equilibrium equations, Ph.D. Dissertation, Princeton University, 1998.
[15] S. Wolfram, The Mathematica Book, third ed., Wolfram Media, 1996.
[16] A.H. Reiman, A.H. Boozer, Island formation and destruction of flux surfaces in three dimensional MHD equilibria, Phys. Fluids 27 (1984) 2446–2454.