

This is a repository copy of *Improved Nonlinear Solvers in BOUT++*.

White Rose Research Online URL for this paper:

<https://eprints.whiterose.ac.uk/75060/>

Version: Submitted Version

Article:

Dudson, Benjamin Daniel orcid.org/0000-0002-0094-4867, Farley, Sean and Curfman McInnes, Lois (Accepted: 2012) *Improved Nonlinear Solvers in BOUT++*. *Computer Physics Communications*. ISSN 0010-4655 (In Press)

Reuse

Items deposited in White Rose Research Online are protected by copyright, with all rights reserved unless indicated otherwise. They may be downloaded and/or printed for private study, or other acts as permitted by national copyright laws. The publisher or other rights holders may allow further reproduction and re-use of the full text version. This is indicated by the licence information on the White Rose Research Online record for the item.

Takedown

If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

Improved Nonlinear Solvers in BOUT++

Ben Dudson^a, Sean Farley^b, Lois Curfman McInnes^c

^a*York Plasma Institute, Department of Physics
University of York
York YO10 5DD, United Kingdom*

^b*Mathematics Department
Illinois Institute of Technology
Chicago, IL 60616, USA*

^c*Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, IL 60439, USA*

Abstract

Challenging aspects of large-scale turbulent edge simulations in plasma physics include robust nonlinear solvers and efficient preconditioners. This paper presents recent advances in the scalable solution of nonlinear partial differential equations in BOUT++, with emphasis on simulations of edge localized modes in tokamaks. A six-field, nonlinear, reduced magnetohydrodynamics model containing the fast shear Alfvén wave and electron and ion heat conduction along magnetic fields is solved by using Jacobian-free Newton-Krylov (JFNK) methods and nonlinear GMRES (NGMRES). Physics-based preconditioning based on analytic Schur factorization of a simplified Jacobian is found to result in an order of magnitude reduction in runtime over unpreconditioned JFNK, and NGMRES is shown to significantly reduce runtime while requiring only the nonlinear function evaluation. We describe in detail the preconditioning algorithm, and we discuss parallel performance of NGMRES and Newton-Krylov methods using the PETSc library.

Keywords: physics-based preconditioning, edge localized modes, Newton-Krylov, nonlinear GMRES

1. Introduction

The international community is committed to the development of fusion energy in order to meet growing world energy needs. This is a complex, international, and costly endeavor; the next experimental magnetic-fusion plasma confinement device, ITER [1], is estimated to cost in excess of \$10B. The handling of the plasma exhaust is a challenge for the operation of ITER and for the design of the future demonstration power plant DEMO. To achieve its design goals, ITER must operate in a high-performance regime with steep pressure gradients close to the plasma edge [2, 3]. While this capability improves the plasma performance, the steep pressure gradients and associated bootstrap current can destabilize peeling-ballooning modes [4–6]. These result in eruptions from the plasma edge, known as edge localized modes (ELMs) (see, e.g., [7]). Scaling from existing experiments indicates that these violent events, if uncontrolled, could produce peak power loads of around 1 GW/m² onto material surfaces in ITER [8, 9] and potentially more in a DEMO power plant. Such loads would result in unacceptable erosion of plasma facing components. Thus, researchers are extremely interested in understanding and controlling ELMs, and work is ongoing to use several nonlinear simulation codes in order to simulate ELMs, predict their size in ITER, and find ways to mitigate them.

Nonlinear simulations of tokamak plasmas are challenging because of the wide range of length scales and timescales present: The characteristic ion cyclotron motion has scales around 1 mm and 10⁻⁷ seconds, while the machine has length scales of ~10 m and transport timescales in seconds. If electron physics is important, then the characteristic scales become smaller by the square root of the mass ratio, a factor of ~60. These small cyclotron scales are due to the strong magnetic field (typically 1 – 5 Teslas), which is

used to confine the plasma in a tokamak. This produces strong anisotropy in the system, as particles are confined perpendicular to the magnetic field but are essentially free to move along it, resulting in a difference of 10^8 in heat conduction rates perpendicular and parallel to the field. The topology of the magnetic field poses challenges for simulation, as the core fieldlines form helices with a pitch angle that varies across the device. At the boundary of the core plasma, in the region we are interested in simulating, a separatrix is formed, outside of which the fieldlines intersect material boundaries rather than forming closed toroidal “flux” surfaces. Furthermore, the edge region of tokamaks is complicated by the interaction with neutral gas and material surfaces, bringing in atomic physics, sputtering, and deposition processes.

Analytic reduction of the full problem, employing asymptotic expansions in small parameters such as the ratio of the Larmor radius to machine size, has been successfully used for many years (see, e.g., [10] and references therein). In this paper we consider fluid equations, which evolve moments of the particle velocity distribution, and remove the smallest scales associated with cyclotron motion [11]. Further simplifications include reduced magnetohydrodynamics (MHD), exploiting the anisotropy of the system in which length scales parallel to the magnetic field are much larger than those perpendicular to the field (see, e.g., [12, 13]). By assuming force balance, this approach allows us to analytically remove the fast magnetosonic wave, which would be the fastest wave in our system. This wave is not important for the relatively slow MHD instabilities we wish to study here, but it is responsible for establishing force balance along the magnetic field. After this reduction, the system of equations still contains two fast timescales that cannot be removed without also removing physics of interest: the shear Alfvén wave and the parallel heat transport. These can severely limit the timestep in numerical simulations, so implicit and semi-implicit methods are widely used in plasma simulation [14]. In BOUT++ edge simulations of ELMs [15–18], such techniques help address challenges in the broad range of temporal scales, complicated geometry with a range of spatial scales, and competing physical effects of the various degrees of freedom.

This paper focuses on efficient and scalable nonlinear solvers for BOUT++ ELM simulations. We construct a *physics-based* preconditioner that reduces the time to solve nonlinear systems with Jacobian-free Newton-Krylov (JFNK) techniques [19]. The term *physics-based preconditioner* has been used to describe various approaches to application-specific preconditioners that incorporate insight into the physics behind the equations being solved [20–26]. Inspired by the approach of Chacón [27], we use the term here to describe our development of a custom preconditioning algorithm based on analytic reduction and Schur factorization of the system Jacobian. Identifying the key physics terms allows a simplification of the resulting equations and enables efficient numerical solution. A similar approach has been previously attempted within the NIMROD code [25], where a Crank-Nicholson (CN) Jacobian-free Newton-Krylov scheme was used along with physics-based preconditioning. In that work numerical instability was encountered and attributed to the CN scheme. In this paper we exploit the field-aligned coordinate system used in BOUT++ [15] to construct a preconditioner that reduces a large 3D problem to the solution of many one-dimensional systems, each of which can be solved independently. This approach greatly improves convergence of Newton-Krylov methods in implicit timestepping schemes, without causing numerical instability.

We also introduce the complementary approach of nonlinear GMRES [28], which offers the advantage of requiring only nonlinear function evaluations from the application, rather than also needing Jacobian-vector products and preconditioning for Newton-based methods. We demonstrate that both of these approaches—physics-based preconditioned Newton-Krylov and nonlinear GMRES—significantly reduce the time for solving the nonlinear ELM systems at each timestep compared with a Newton-Krylov approach with no preconditioning, and we demonstrate good strong scaling on up to 512 processor cores.

The remainder of this paper explains the details of our approach. Section 2 introduces the BOUT++ application and its approach to modeling ELMs. Section 3 describes the algorithms and software that we use to solve the resulting nonlinear systems that arise at each timestep, including both Newton-Krylov methods with physics-based preconditioning and nonlinear GMRES. Section 4 presents experimental results, while Section 5 discusses conclusions and directions for future work.

2. BOUT++ Simulation Overview

The numerical methods studied in this paper are quite general, but are applied here to the solution of reduced MHD for fusion applications within the BOUT++ framework.

2.1. BOUT++ Capabilities

BOUT++, a parallel MPI + OpenMP finite difference code written in C++ [15], simulates a range of plasma fluid equations in curvilinear coordinate systems—in particular, reduced MHD models in coordinates aligned with a strong background magnetic field. Current applications of BOUT++ include the study of ELMs [16–18], plasma turbulence [29], and blob dynamics [30]. In this paper BOUT++ is extended to more efficiently simulate heat conduction parallel to magnetic fields, an important process in all these areas. Parallel heat conduction modifies the linear structure and growth rate of plasma instabilities and plays a crucial role in the loss of energy to material surfaces in tokamak edge simulations of turbulence and ELMs.

2.2. Reduced MHD Equations

Since the linear stability threshold is well described by ideal MHD [5], it is reasonable to start studying ELMs by using fluid models based on MHD. Several initial-value codes including BOUT++ [17], M3D-C1 [31] and NIMROD [32], have now been benchmarked and shown good agreement with linear ideal MHD codes ELITE [5, 6] and GATO [33]. Nonlinearly, ideal MHD theory predicts narrowing structures erupting outwards, leading to a finite time singularity [34]. Long before this happens, additional effects must become important in order to allow plasma to decouple from magnetic fields and relax to a new equilibrium. Nonlinear modeling of ELMs has therefore concentrated on fluid models incorporating a range of different nonideal effects [31, 32, 35–38].

Since each simulation code currently uses different fluid models, nonlinear benchmarking of codes is difficult. In order to enable future comparisons between codes, a set of equations close to those used in JOREK [39] has been implemented in BOUT++, which we will refer to in this paper as the six-field model. Six scalar fields are evolved: mass density ρ ; electron and ion temperatures T_e and T_i , respectively; vorticity $U = \mathbf{b}_0 \cdot \nabla \times \mathbf{v} \simeq \frac{1}{B_0} \nabla_{\perp}^2 \phi$; parallel velocity $v_{\parallel} = \mathbf{b}_0 \cdot \mathbf{v}$; and parallel vector potential $A_{\parallel} = \mathbf{b}_0 \cdot \mathbf{A}$. From these auxiliary quantities are calculated the total plasma pressure $P = \rho(T_e + T_i)$ and parallel current density $J_{\parallel} = \mathbf{b}_0 \cdot \mathbf{J} = -\nabla_{\perp}^2 A_{\parallel}$. The equilibrium magnetic field unit vector is $\mathbf{b}_0 = \mathbf{B}_0/B_0$. Here all quantities with subscript “0” are calculated from the starting equilibrium and are not evolved. The evolving quantities are deviations from equilibrium; for example, the total mass density is $\rho_0 + \rho$. The equations for plasma density and temperatures are

$$\begin{aligned} \frac{\partial \rho}{\partial t} &= -\mathbf{v} \cdot \nabla (\rho + \rho_0) + (\nabla \cdot \mathbf{v}) (\rho + \rho_0) + D_{\perp} \nabla_{\perp}^2 \rho \\ \frac{\partial T_s}{\partial t} &= -\mathbf{v} \cdot \nabla (T_s + T_{s0}) - \frac{2}{3} (\nabla \cdot \mathbf{v}) (T_s + T_{s0}) \\ &\quad + \frac{1}{\rho + \rho_0} [\nabla_{\parallel} \cdot (\chi_{s\parallel} \partial_{\parallel} T_s) + \chi_{s\perp} \nabla_{\perp}^2 T_s] + \frac{2}{3(\rho + \rho_0)} W_s, \end{aligned} \quad (1)$$

where the total plasma velocity $\mathbf{v} = \mathbf{v}_{E \times B} + \mathbf{b}_0 v_{\parallel}$ is given by an $E \times B$ drift perpendicular to magnetic fieldlines $\mathbf{v}_{E \times B} = \frac{1}{B_0} \mathbf{b}_0 \times \nabla \phi$ and parallel flow v_{\parallel} along them. The above equations describe the advection ($\mathbf{v} \cdot \nabla$) and compression ($\nabla \cdot \mathbf{v}$) of density and temperatures ($s = e, i$), collisional diffusion perpendicular to the magnetic field with coefficients D_{\perp} and $\chi_{s\perp}$, and parallel conduction of heat along magnetic fields with coefficient $\chi_{s\parallel}$. Derivatives are split into those parallel to the magnetic field $\nabla_{\parallel} = \mathbf{b}_0 \cdot \nabla$ and those perpendicular $\nabla_{\perp} = \nabla - \mathbf{b}_0 \cdot \nabla$.

The velocity of the plasma is described by the vorticity and parallel flow:

$$\begin{aligned}\frac{\partial U}{\partial t} &= -\mathbf{v} \cdot \nabla U + \frac{1}{\rho + \rho_0} \left[B_0^2 \nabla_{\parallel} \left(\frac{J_{\parallel} + J_{\parallel 0}}{B_0} \right) \right. \\ &\quad \left. + 2\mathbf{b}_0 \times \mathbf{k}_0 \cdot \nabla P + \nu_{\parallel} \partial_{\parallel}^2 U + \nu_{\perp} \nabla_{\perp}^2 U \right] \\ \frac{\partial v_{\parallel}}{\partial t} &= -\mathbf{v} \cdot \nabla v_{\parallel} - \frac{1}{\rho + \rho_0} \nabla_{\parallel} P.\end{aligned}\tag{2}$$

The vorticity equation (evolving U) contains a term responsible for shear Alfvén wave propagation ($B_0^2 \nabla_{\parallel} \frac{J_{\parallel}}{B_0}$), instability drive due to equilibrium magnetic field curvature κ and pressure gradient ∇P , as well as parallel and perpendicular viscosity terms ν_{\parallel} and ν_{\perp} .

The evolution of the magnetic field is described by Ohm's law for the electric field parallel to the magnetic field $\mathbf{b} \cdot \mathbf{E} = \mathbf{b} \cdot \eta \mathbf{J}$,

$$\frac{\partial A_{\parallel}}{\partial t} = -\nabla_{\parallel} \phi - \eta J_{\parallel},\tag{3}$$

where η is the parallel resistivity.

2.3. Timescales for Shear Alfvén Wave and Parallel Heat Conduction

The timescales of interest in equations (1)-(3) are those of the linear growth of peeling-ballooning modes and nonlinear eruption of filaments, typically of the order of $\sim 100 \mu\text{s}$ up to milliseconds. The above set of equations contains timescales much faster than this, which must be preconditioned in an implicit scheme. The vorticity and A_{\parallel} equations contain the terms

$$\frac{\partial U}{\partial t} = \frac{B_0^2}{\rho_0} \nabla_{\parallel} \left(\frac{J_{\parallel}}{B_0} \right) \quad \frac{\partial A_{\parallel}}{\partial t} = -\nabla_{\parallel} \phi,$$

which reduce to

$$\frac{\partial^2 \phi}{\partial t^2} = \nabla_{\perp}^{-2} \left[\frac{B_0^3}{\rho_0} \nabla_{\parallel} (\nabla_{\perp}^2 \nabla_{\parallel} \phi / B_0) \right] \simeq \frac{B_0^2}{\rho_0} \nabla_{\parallel}^2 \phi,$$

corresponding to the shear Alfvén wave traveling along magnetic fieldlines. Given the strong magnetic field B_0 and low mass density ρ_0 in tokamaks, wave speeds of $v_A \sim 10^7 \text{m/s}$ are typical, which for parallel grid spacing of $\sim 10 \text{cm} - 1 \text{m}$ give CFL timesteps $\sim 10^{-8} \text{s}$, 3 to 4 orders of magnitude shorter than timescales of interest.

The second fast timescale is introduced through the ion and electron temperature equation

$$\frac{\partial T_s}{\partial t} = \frac{1}{\rho_0} \nabla_{\parallel} (\chi_{s\parallel} \partial_{\parallel} T_s),$$

which is a diffusion equation along magnetic fieldlines. At the high temperatures relevant to fusion, electron thermal speeds along fieldlines of $10^6 - 10^7 \text{m/s}$ are typical, and hence parallel thermal diffusion is also fast compared with nonlinear evolution timescales.

The above equations contain one other wave along magnetic fieldlines: the sound wave described by

$$\frac{\partial v_{\parallel}}{\partial t} = -\frac{1}{\rho_0} \nabla_{\parallel} P \quad \frac{\partial \rho}{\partial t} = -\rho_0 \nabla_{\parallel} v_{\parallel} \quad \frac{\partial T_s}{\partial t} = -\frac{2}{3} T_{s0} \nabla_{\parallel} v_{\parallel}.$$

The speed of this wave is typically around 10^5m/s for the problems in which we are interested, several orders of magnitude slower than the Alfvén wave and parallel electron heat conduction. It is trivial to extend the preconditioner described below to include this wave, but this extension was found to make little difference to the convergence rate and so is not included here.

In the following sections we outline the implicit timestepping method and construct a preconditioning scheme that can be applied to address both the Alfvén wave and temperature conduction timescales.

3. Scalable Nonlinear Solvers

The coupled set of reduced MHD equations (1)-(3) are discretized by the method of lines, using 4th-order central differencing for diffusive terms, and 3rd order WENO [40] for upwinding. Because this work focuses on nonlinear solvers, we chose the timestepping algorithm to be the same for each test case, namely, a variable order backward differential formulation (BDF) (see, e.g., [41]). This family comprises methods that are implicit and linear multistep, and that allow variable timestepping. At each timestep, by writing the system state as a vector, we obtain from equations (1)-(3) a nonlinear system of the form

$$f(u) = f \begin{pmatrix} \rho \\ T_i \\ T_e \\ U \\ v_{||} \\ A_{||} \end{pmatrix} = 0, \quad (4)$$

where $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$. This requires the solution of a nonlinear system, here using Newton-Krylov and NGMRES methods.

3.1. Newton-Krylov Methods

In a Newton-Krylov method, we solve the nonlinear system (4) at each timestep using a Newton iteration (see, e.g., [42]),

$$u_{k+1} = u_k - [f'(u_k)]^{-1} f(u_k), \quad k = 0, 1, \dots$$

Here u_0 is an initial approximation to the solution, and $f'(u_k)$, the Jacobian, is nonsingular at each iteration. A Krylov iterative method (here GMRES) is then used to solve this linear system at each Newton iteration. In practice, the Newton iteration is implemented by the following two steps:

1. (Approximately) solve $f'(u_k)\Delta u_k = -f(u_k)$.
2. Update $u_{k+1} = u_k + \alpha\Delta u_k$.

Here $0 < \alpha \leq 1$ is a scalar. In these experiments, we use a JFNK variant, where Jacobian-vector products are computed matrix-free [19] by an approximation

$$f'(v)a \approx \frac{f(v + h \cdot a) - f(v)}{h},$$

for a differencing parameter h . By computing finite-difference Jacobian-vector products for the iterative solve, assembling the actual Jacobian is never required.

If the nonlinear system being evolved contains a wide range of timescales, as in most plasma simulations, then it is said to be “stiff,” and the Jacobian can be ill-conditioned. In this case the Krylov method will require an unacceptably large number of iterations. Preconditioning the linearized Newton systems, which have the form $\mathbb{J}x = b$, where \mathbb{J} is the nonsingular $n \times n$ Jacobian matrix and x and b are n -dimensional vectors, transforms these to equivalent systems $\mathbb{P}\mathbb{J}x = \mathbb{P}b$ through the action of a preconditioner, \mathbb{P} , whose inverse action approximates that of the Jacobian but at smaller cost. Choosing an effective approximation is one of the goals of this work, since the preconditioning phase is crucial to achieving low computational cost and scalable parallelism.

3.2. Physics-Based Preconditioning

Physics-based preconditioning [21, 27, 43] is a method by which insight into the physics behind the equations being solved is used to precondition the Newton-Krylov method. After first identifying the stiff terms in the equations, an approximate inverse is constructed with block Gaussian elimination (Schur factorization) of the simplified analytic Jacobian. The resulting matrices have a parabolic form, even if the starting system is hyperbolic, as a result of the implicit timestepping procedure. Multigrid methods are

often used to solve these systems (see, e.g., [27]), but these are difficult to implement in complex geometries and an existing codebase that was not designed with them in mind. Here we describe a simpler scheme that can be used to efficiently solve the factorized Jacobian in field-aligned coordinates without the complexity of a multigrid method.

To derive a preconditioner, we begin by simplifying the nonlinear six-field system, equations (1)-(3), retaining the linear vorticity terms driving perpendicular plasma $E \times B$ motion, the fast parallel Alfvén wave that propagates the motion along fieldlines, the linear terms describing perpendicular advection of density and temperature, and the parallel diffusion of temperature perturbations. This procedure reduces the equations to

$$\begin{aligned}
\frac{\partial \rho}{\partial t} &\simeq -\mathbf{v}_{E \times B} \cdot \nabla_{\perp} \rho_0 \\
\frac{\partial T_s}{\partial t} &\simeq -\mathbf{v}_{E \times B} \cdot \nabla_{\perp} T_{s0} + \frac{1}{\rho_0} \nabla_{\parallel} \cdot (\chi_{s\parallel} \partial_{\parallel} T_s) \\
\frac{\partial U}{\partial t} &\simeq \frac{1}{\rho_0} \left[B_0^2 \nabla_{\parallel} \left(\frac{J_{\parallel}}{B_0} \right) + 2\mathbf{b}_0 \times \mathbf{k}_0 \cdot \nabla P \right] \\
\frac{\partial v_{\parallel}}{\partial t} &\simeq 0 \\
\frac{\partial A_{\parallel}}{\partial t} &\simeq -\nabla_{\parallel} \phi.
\end{aligned} \tag{5}$$

We have therefore neglected all nonlinear terms (those involving two or more evolving quantities), all perpendicular diffusive terms, viscosity, and resistivity. The neglect of nonlinear terms is justified because we wish to follow the nonlinear dynamics and thus should not step over these timescales; instead, we focus on preconditioning the fast linear dynamics. Perpendicular diffusion and resistive effects are also slow relative to the parallel dynamics and so can be dropped. We stress that these simplifications made in the preconditioner do not affect the solution to the full six-field model, only the convergence rate toward it. When the assumptions made here are violated, the preconditioner becomes less effective, but the solution is not compromised (provided that the Krylov method still converges and does not stall).

In an implicit timestepping scheme (Section 3.1) the operator to be inverted is $\mathbb{A} = \mathbb{I} - \gamma \mathbb{J}$, where \mathbb{J} is the system Jacobian; γ is usually proportional to the timestep and is determined by the choice of integration scheme. For preconditioning, a simplified form of the Jacobian is derived from Equations (5) and is given in Equation (6).

$$\mathbb{J} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \frac{1}{B} \mathbf{b} \times \nabla \rho_0 \cdot \nabla \nabla_{\perp}^{-2} [B_0] \\ 0 & \frac{\chi_{i,\parallel}}{\rho_0} \nabla_{\parallel}^2 & 0 & 0 & 0 & \frac{1}{B} \mathbf{b} \times \nabla T_{i0} \cdot \nabla \nabla_{\perp}^{-2} [B_0] \\ 0 & 0 & \frac{\chi_{e,\parallel}}{\rho_0} \nabla_{\parallel}^2 & 0 & 0 & \frac{1}{B} \mathbf{b} \times \nabla T_{e0} \cdot \nabla \nabla_{\perp}^{-2} [B_0] \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\nabla_{\parallel} \nabla_{\perp}^{-2} [B_0] \\ \frac{2}{\rho_0} \mathbf{b} \times \kappa \cdot \nabla [(T_{e0} + T_{i0})] & \frac{2}{\rho_0} \mathbf{b} \times \kappa \cdot \nabla [\rho_0] & \frac{2}{\rho_0} \mathbf{b} \times \kappa \cdot \nabla [\rho_0] & 0 & -\frac{1}{\rho_0} B_0 \nabla_{\parallel} \nabla_{\perp}^2 & 0 \end{bmatrix} \tag{6}$$

Following the methodology of [27], we first factorize the matrix $\mathbb{A} = \mathbb{I} - \gamma \mathbb{J}$ using Schur factorization:

$$\mathbb{A}^{-1} = (\mathbb{I} - \gamma \mathbb{J})^{-1} = \begin{bmatrix} \mathbb{E} & \mathbb{U} \\ \mathbb{L} & \mathbb{D} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbb{I} & \mathbb{E}^{-1} \mathbb{U} \\ 0 & \mathbb{I} \end{bmatrix} \begin{bmatrix} \mathbb{E}^{-1} & 0 \\ 0 & \mathbb{P}_{Schur}^{-1} \end{bmatrix} \begin{bmatrix} \mathbb{I} & 0 \\ -\mathbb{L} \mathbb{E}^{-1} & \mathbb{I} \end{bmatrix}, \tag{7}$$

where \mathbb{L} is a row vector containing the vorticity drive terms due to curvature and parallel current, \mathbb{U} is a column vector containing the linear $E \times B$ advection terms, and \mathbb{E} contains the parallel heat conduction terms, which have the form $\mathbb{I} - \gamma \frac{\chi_{\parallel e,i}}{\rho_0} \nabla_{\parallel}^2$.

The matrix $\mathbb{P}_{Schur} = \mathbb{D} - \mathbb{L} \mathbb{E}^{-1} \mathbb{U}$ is the Schur complement; and since nonlinear terms are neglected, \mathbb{D} is the identity matrix. To simplify the form of \mathbb{P}_{Schur} , we drop terms involving only perpendicular derivatives, leaving only the shear Alfvén wave:

$$\mathbb{P}_{Schur} \simeq \mathbb{I} - \gamma^2 \frac{B_0^2}{\rho_0} \nabla_{\parallel} \nabla_{\perp}^2 \nabla_{\parallel} \nabla_{\perp}^{-2}.$$

In order to remove the perpendicular derivatives, the following approximation is used:

$$\nabla_{\parallel} \nabla_{\perp}^2 \nabla_{\parallel} \nabla_{\perp}^{-2} \simeq \nabla_{\parallel}^2 - \frac{2\nabla_{\parallel} (RB_{\theta})}{RB_{\theta}},$$

where R and B_{θ} are the equilibrium major radius and poloidal magnetic field, respectively.

The above factorization contains three applications of \mathbb{E}^{-1} ; however, it was found that only the second had a significant impact on the convergence rate, and so the first and last were omitted, significantly speeding the calculation. The resulting preconditioning operator can be divided into three stages, $\mathbb{P} = \mathbb{P}_3 \mathbb{P}_2 \mathbb{P}_1$, and written as an operation on a preconditioned state vector $v^{(U)}$, which produces a state vector $v^{(P)} = \mathbb{P}v^{(U)}$. In the following description, superscripts on state vectors or components of state vectors will indicate the stage of the preconditioning operation.

Note that in the simplest backwards Euler implicit scheme when $\mathbb{P} = \mathbb{A}^{-1}$, the vector $v^{(U)}$ is the solution at the last timestep, and $v^{(P)}$ the solution at the next. The preconditioner can therefore be interpreted as a predictor step, calculating an approximation of the state vector at the next timestep from the previous one. The application of \mathbb{P} , Equation (7), is performed as follows.

1. Given an unpreconditioned vector $v^{(U)} = (\rho^{(U)}, T_i^{(U)}, T_e^{(U)}, v_{\parallel}^{(U)}, A_{\parallel}^{(U)}, U^{(U)})$, apply the first matrix in Equation (7):

$$v^{(P1)} = \begin{bmatrix} \mathbb{I} & 0 \\ -\mathbb{L} & \mathbb{I} \end{bmatrix} v^{(U)},$$

where the \mathbb{E}^{-1} operator has been dropped, as discussed above. When this is expanded, only the vorticity field is modified:

$$U^{(P1)} = U^{(U)} + \gamma \frac{B_0}{\rho_0} \nabla_{\parallel} J_{\parallel}^{(U)} + \gamma \frac{1}{\rho_0} \mathbf{2b} \times \kappa \cdot \nabla P^{(U)},$$

where $J_{\parallel}^{(U)} = -\nabla_{\perp}^2 A_{\parallel}^{(U)}$ and $P^{(U)} = \rho_0 (T_i^{(U)} + T_e^{(U)}) + \rho^{(U)} (T_{i0} + T_{e0})$. This operation represents a forward Euler step to predict the vorticity after a timestep γ .

2. Apply the second matrix, solving for parallel temperature conduction and shear Alfvén wave dynamics,

$$v^{(P2)} = \begin{bmatrix} \mathbb{E}^{-1} & 0 \\ 0 & \mathbb{P}_{Schur}^{-1} \end{bmatrix} v^{(P1)},$$

which can be written as

$$T_s^{(P2)} = \left(\mathbb{I} - \gamma \frac{\chi_{\parallel s}}{\rho_0} \nabla_{\parallel}^2 \right)^{-1} T_s^{(U)}$$

$$U^{(P)} = \left(\mathbb{I} + \frac{2\gamma^2 \nabla_{\parallel} (RB_{\theta})}{RB_{\theta} B_0^2} - \gamma^2 \frac{B_0^2}{\rho_0} \nabla_{\parallel}^2 \right)^{-1} U^{(P1)}.$$

Since all these operators are of the form $(a + b\nabla_{\parallel}^2)^{-1}$ with different scalars a and b , the same parallel solver can be used. Since the equations for temperature and vorticity are decoupled, a further optimization, not used in this current work, would be to also solve the three equations at the same time. This operation represents a backward Euler step for the propagation of vorticity and temperature along magnetic fieldlines over a timestep γ and is the most computationally demanding. Efficiently solving these systems is critical to the performance of the preconditioner, so the method used is described in detail below.

3. Apply the third matrix, updating the other fields based on the new vorticity

$$v^{(P)} = \begin{bmatrix} \mathbb{I} & -\mathbb{U} \\ 0 & \mathbb{I} \end{bmatrix} v^{(P2)},$$

where again the \mathbb{E}^{-1} operator has been dropped. This can be written as

$$\begin{aligned}\rho^{(P)} &= \rho^{(U)} - \gamma \mathbf{v}_{E \times B}^{(P)} \cdot \nabla \rho_0 \\ T_s^{(P)} &= T_s^{(P2)} - \gamma \mathbf{v}_{E \times B}^{(P)} \cdot \nabla T_{s0} \\ A_{\parallel}^{(P)} &= A_{\parallel}^{(U)} - \gamma \nabla_{\parallel} \phi^{(P)},\end{aligned}$$

where $\mathbf{v}_{E \times B}^{(P)} = \frac{1}{B_0} \mathbf{b}_0 \times \nabla \phi^{(P)}$ is the perpendicular $E \times B$ velocity calculated by using $\phi^{(P)} = \nabla_{\perp}^{-2} (B_0 U^{(P)})$, the electrostatic potential calculated from the updated vorticity. This represents an implicit update of ρ, T_s, A_{\parallel} , advecting plasma quantities based on an approximate $E \times B$ velocity at the next timestep.

The only variable that is not modified in any of these operations is the parallel velocity, so $v_{\parallel}^{(P)} = v_{\parallel}^{(U)}$.

The preconditioner described above contains the key physics responsible for fast timescales in the system, and it has greatly reduced the complexity and computational cost of calculating \mathbb{P} . These features can be seen by considering the sizes of the matrices involved. Since 6 variables are evolved on N mesh points, the matrix \mathbb{A} to be inverted has $(6N)^2$ elements, most of which are however zero. A typical simulation will use on the order of 100 points in each dimension (between 64 and 512 in most cases, though the radial mesh size N_{ψ} can be several thousand for ITER simulations) and so have $\sim 10^6$ mesh points, requiring the inversion of a $(6 \times 10^6)^2$ matrix at each timestep. By block factorizing and keeping only the fast components, we have reduced this to solving three separate matrices (in \mathbb{E} and \mathbb{P}_{Schur}), which can be performed in parallel, each of which is of size $(N)^2 \simeq (10^6)^2$. This approach in itself does not represent a large savings; but by exploiting the structure of these blocks in the field-aligned coordinate system employed by BOUT++, further simplifications can be made.

Since each block contains only derivatives along magnetic fieldlines after we remove perpendicular derivatives, and equilibrium magnetic fieldlines form helices around a torus covering 2D “flux” surfaces, we can independently solve for each flux surface, immediately reducing the $(10^6)^2$ inversion problem to ~ 100 separate $(10^4)^2$ problems. This assumption of equilibrium flux surfaces will break down if the magnetic field becomes significantly perturbed in the later stages of an ELM crash, but this is not a critical problem because at these times nonlinear dynamics are fast and a smaller timestep is unavoidable.

To further simplify the problem, we exploit the toroidal symmetry of the equilibrium and decompose into toroidal Fourier harmonics. This approach decouples the toroidal and poloidal directions, splitting the 2D problem into a set of one-dimensional equations in poloidal angle, with a complex phase shift representing the pitch of the fieldlines. If we started with N_{ψ} flux surfaces, each containing N_{θ} poloidal points and N_{ζ} toroidal points (so $N = N_{\psi} \times N_{\theta} \times N_{\zeta}$), then we are left with $N_{\psi} \times (N_{\zeta}/2) \sim 10^4$ complex cyclic tridiagonal systems, each of length $N_{\theta} \sim 10^2$. These are then solved efficiently in parallel by using a variant of the cyclic reduction algorithm with interface equations [44]. Since the three equations being solved are also independent, all $3 \times N_{\psi} \times (N_{\zeta}/2)$ tridiagonal systems can be solved simultaneously, resulting in a good overlap of communication and computation on a large number of processors.

3.3. Nonlinear GMRES

As an alternative to preconditioned Newton-Krylov methods, we consider nonlinear GMRES (NGMRES) to solve the nonlinear system (4). NGMRES and other similar methods (quasi-Newton, Anderson mixing) of solution for nonlinear systems are analogous to multistep Krylov methods used for linear systems [28, 45]. At each iteration, an updated solution is constructed as a linear combination of several stored previous solutions and a new candidate solution. A minimum-residual linear combination is found by solving a small minimization problem.

The new candidate solution is constructed by perturbing the current solution by some negative damping of the current residual, typically found through a line search. One may also construct the candidate using a small number of iterations of some other nonlinear solution technique. For the present problem we use a damping factor of -0.001 on the residual to construct the candidate and store 50 previous solutions. By using these Krylov subspace accelerators, we aim to reduce the overall CPU time for each nonlinear solve and also reduce memory usage.

4. Experimental Results

We use the nonlinear solvers in the PETSc library [46, 47] to solve these nonlinear systems over varying numbers of processors. The library provides a simple interface for nonlinear solvers that enables selection of algorithms and parameters at runtime, with no changes to the application code.

Our experiments were run on the Fusion cluster at Argonne National Laboratory. This cluster has 320 compute nodes, each with an Intel Nehalem dual-socket, quad-core 2.53 GHz processor (or 2,560 compute processors), connected by Infiniband.

We specified a relative nonlinear convergence tolerance of 10^{-3} for all methods, which is consistent to what is typically employed for implicit timestepping. For each Newton-Krylov run, we used GMRES with a restart of 30 and a relative linear convergence tolerance of 10^{-2} . The method NGMRES, by definition, has no linear solver associated with it. Consequently, each nonlinear solve using NGMRES has many more iterations, in contrast with the Newton-Krylov approaches, where an average of two or three nonlinear iterations achieved convergence.

4.1. Convergence of Nonlinear Solvers

To test the feasibility of our approach to physics-based preconditioning, we conducted initial experiments with a simpler problem: a two-field subset of the six-field model that contains only the Alfvén wave,

$$\begin{aligned}\frac{\partial A_{\parallel}}{\partial t} &= -\nabla_{\parallel}\phi - \eta J_{\parallel} \\ \frac{\partial U}{\partial t} &= -\mathbf{v}_{E \times B} \cdot \nabla U + \frac{1}{\rho_0} B_0^2 \nabla_{\parallel} \left(\frac{J_{\parallel} + J_{\parallel 0}}{B_0} \right),\end{aligned}\tag{8}$$

with auxiliary equations

$$U = \frac{1}{B} \nabla_{\perp}^2 \phi \quad J_{\parallel} = -\nabla_{\perp}^2 A_{\parallel}.$$

Though greatly simplified, this model can describe the essential physics of magnetic reconnection and island formation in tokamak plasmas [48]. Thus, the efficient solution of this system of equations is of interest in itself. We used a rectangular (slab) grid of dimension $68 \times 32 \times 16$. As seen in Figure 1, physics-based preconditioned JFNK showed a significant reduction in overall runtime in comparison with the default (JFNK with no preconditioning). NGMRES also significantly decreased overall runtime.

For the remaining experiments, we used the six-field model, Equations (1)-(3), on a grid size of $N_{\psi} = 64$, $N_{\theta} = 64$, and $N_{\zeta} = 128$, partitioned only in the $\psi\theta$ -plane because of using the discrete Fourier transform in the ζ -direction. Each example has six coupled independent variables ($\rho, T_e, T_i, U, v_{\parallel}, A_{\parallel}$), as given by Equation (4). One reason for using a relatively small grid is to test the scaling of these algorithms when the number of grid cells per processor becomes small. Since domain decomposition is in ψ and θ , 512 processors (the maximum used here) will divide the mesh into $4 \times 2 \times 128$ blocks. This is a more useful test than simply using a large mesh to demonstrate scaling.

As shown in Figures 2a and 2b, physics-based preconditioning achieves an order of magnitude decrease in overall time for the Newton-Krylov nonlinear solve and nearly two orders of magnitude reduction in the number of Krylov iterations in comparison with the unpreconditioned variant. NGMRES also significantly reduces overall time for the nonlinear solve compared with the BOUT++ default of JFNK with no preconditioning. Note that while the nonlinear relative convergence tolerance for each run is 10^{-3} , the final value of the function norm for the Newton-Krylov runs is much smaller compared with the NGMRES run because of the substantial decrease in function norm at each Newton step. While the physics-based preconditioner code is hand-written for each type of physics model, NGMRES requires no application-specific code and can be activated at runtime, making it an attractive alternative for efficient and scalable nonlinear solves.

At early simulation times ($t = 0$, Figure 2a), nonlinear terms in the evolution equations are negligible, and so the solution is essentially exponential growth. At later times ($t = 40/\omega_A$, Figure 2b), nonlinear terms become important, filaments begin to erupt outwards from the plasma, and the solution becomes more complicated, even turbulent. As a result the number of iterations and wall clock time increase significantly

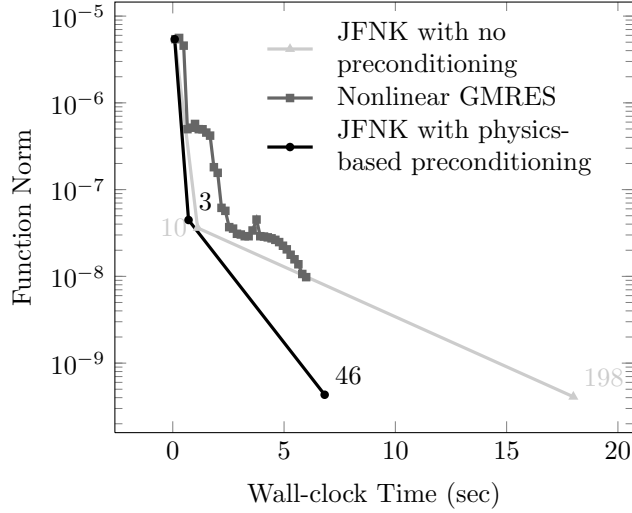


Figure 1: Function norm (l_2) versus wall-clock time (sec) on 16 cores for two-field test case, Equation 8. Labels on the figure indicate cumulative number of linear solver iterations for Newton-Krylov methods.

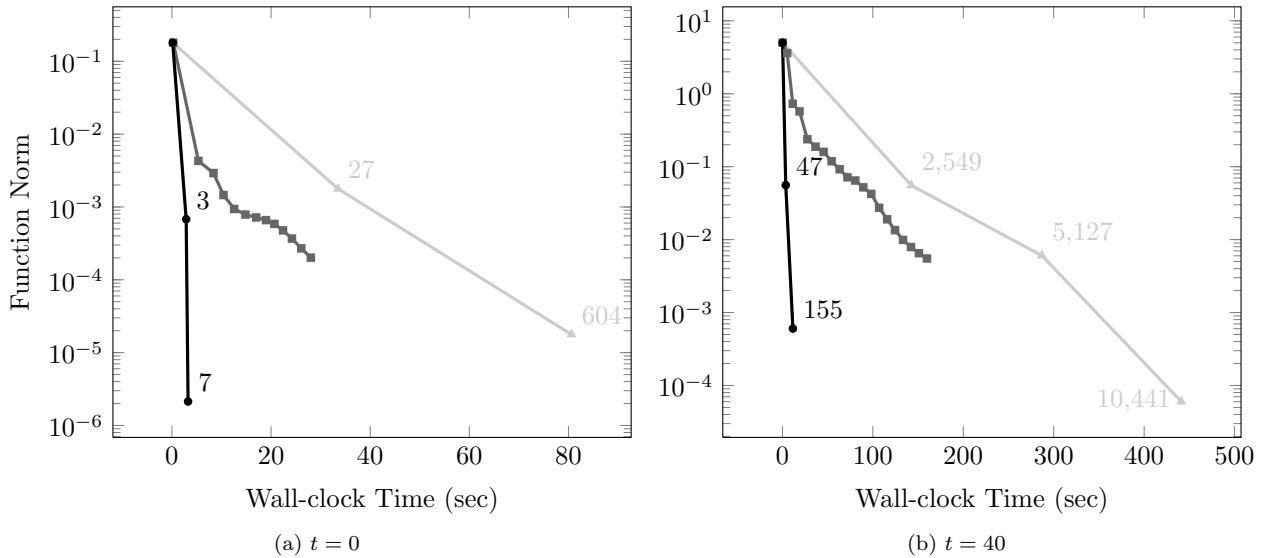


Figure 2: Function norm (l_2) versus wall-clock time (sec) on 512 cores for the full six-field model (Section 2.2) at fixed problem size. *Left:* Results for the initial timestep $t = 0$. *Right:* Results for timestep $t = 40$. Labels indicating the cumulative number of linear solver iterations for Newton-Krylov methods show that the linearized Newton system is relatively easy to solve initially, though it becomes more difficult at timestep $t = 40$.

for later timesteps. The physics-based preconditioner is marginally less effective at later times, reducing the iteration count over unpreconditioned JFNK by a factor of 67 at $t = 40$, compared with a factor of 86 at $t = 0$. This result is expected because nonlinear terms were not included in construction of the preconditioner. Nevertheless, the linear preconditioner remains highly effective, indicating that linear

dynamics are still limiting the timestep even when the plasma evolution is nonlinear.

4.2. Scalability of Nonlinear Solvers

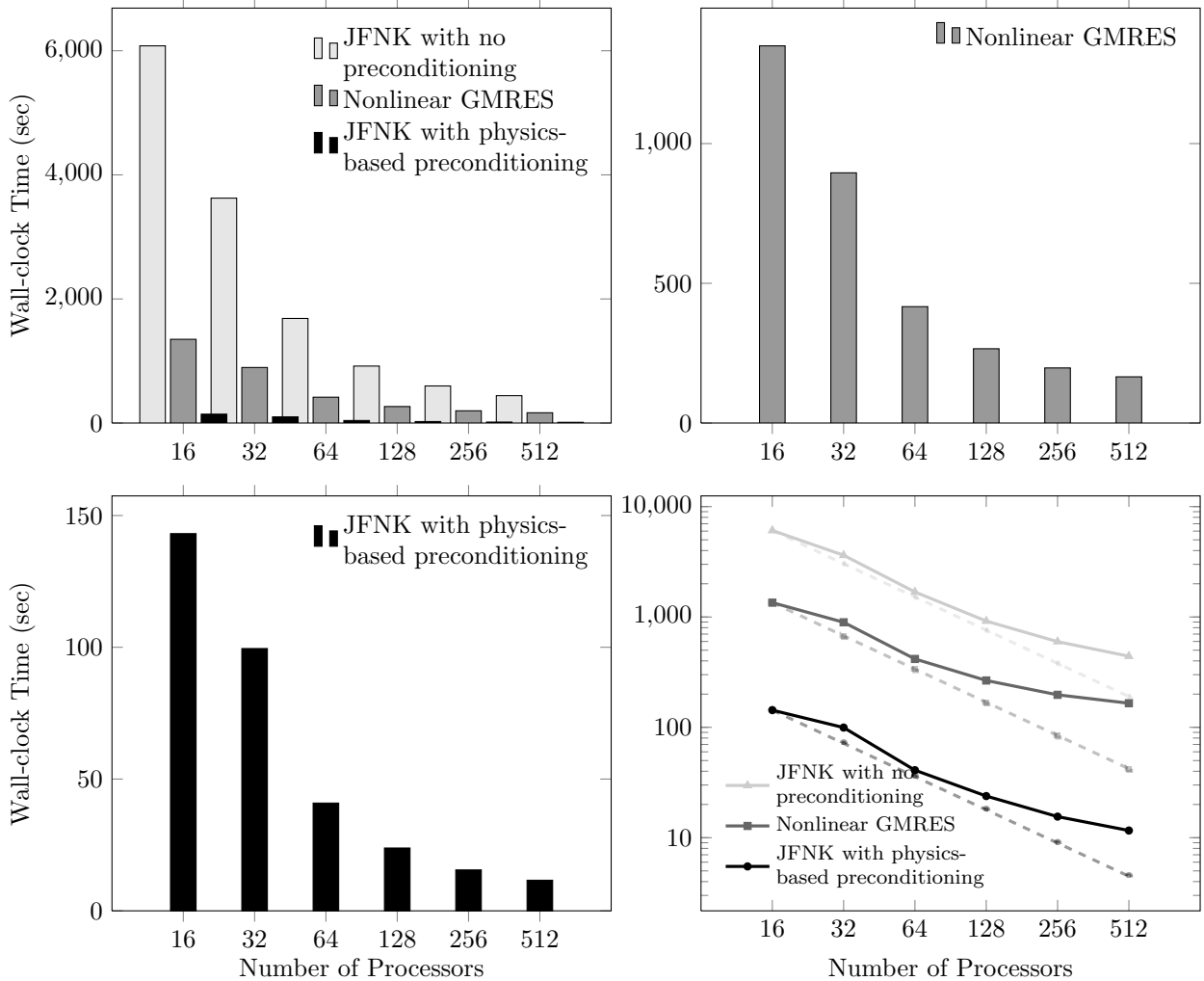


Figure 3: The upper left plot compares the scalability of each algorithm for the full six-field model. The upper right and lower left plots are close-ups of NGMRES and physics-based preconditioning, respectively. The lower right plot compares each algorithm with ideal scaling. We can see that NGMRES and especially physics-based preconditioning significantly reduced overall runtime.

Figure 3 shows the scaling of the JOREK test case with the Jacobian-free Newton-Krylov (no preconditioning), nonlinear GMRES, and Jacobian-free Newton-Krylov (with physics-based preconditioning) algorithms in the first subfigure with lighter bars denoting the comparison with NGMRES and physics-based preconditioning. Notice that the physics-based preconditioning is barely legible in the first subfigure because of the overall runtime being an order of magnitude faster than the default solver. For this reason the individual plots of NGMRES and physics-based preconditioning are also displayed as individual subfigures. All three algorithms then are plotted in a log-log graph in conjunction with ideal scaling (the dashed lines). One can see that all three algorithms have almost identical scaling. This relatively small test case becomes too small for core counts larger than 128, as evidenced by the tapering of scalability.

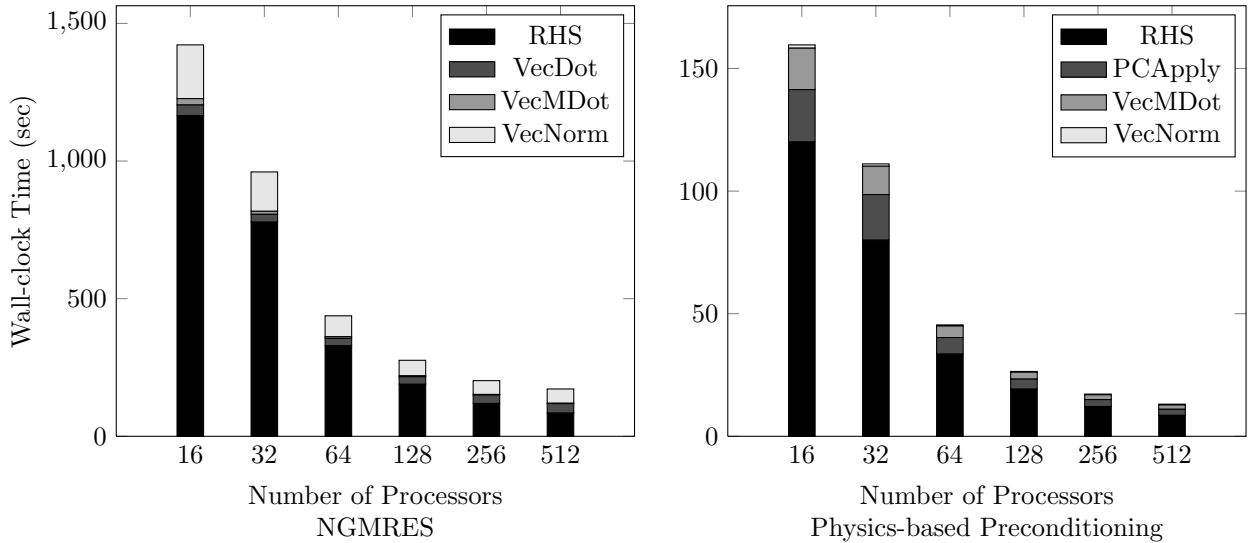


Figure 4: Results showing that function evaluation (RHS) dominates the time for the six-field model

Clearly, using a physics-based preconditioner achieves the fastest runtime. Physics-based approaches, however, require detailed knowledge of the modeling equations being used [21, 27, 43], enforcing a tight coupling between the programmer and physics-based preconditioning analysis. Even if such a coupling exists, potential challenges arise with implementation because of complexities of physics and preconditioning algorithms. In contrast, NGMRES requires no application-specific customization, since the application provides code only for the nonlinear function evaluation.

The bar graphs in Figure 4 show the proportion of time spent in various phases of the nonlinear solve. For each run, the evaluation of the function (or right-hand side, RHS) dominates the runtime. For Jacobian-free variants, a function evaluation is required for each Jacobian-vector product (one per Krylov iteration). The preconditioned variant of JFNK requires fewer function evaluations because of better-conditioned linear systems. NGMRES, alternatively, needs roughly ten times more function evaluations than does preconditioned JFNK. Also, we can see that the time needed for computing the search direction in NGMRES (VecNorm) is nontrivial. This result further indicates that the parallel scaling shown in Figure 3 is due primarily to the scaling of the nonlinear function evaluation; improving this scaling is the subject of ongoing work.

5. Conclusions

We have introduced a physics-based preconditioner that leverages insight into the physics of the equations being solved in order to improve convergence of matrix-free Newton-Krylov methods in BOUT++ ELM simulations. We also introduced the use of a complementary approach, nonlinear GMRES, which requires no Jacobian-vector products or preconditioning. We demonstrated that both approaches reduce time for solving the nonlinear systems that arise at each timestep of an implicit ELM simulation.

The physics-based preconditioning scheme described in this paper is straightforward to construct once the analytic Jacobian is known, can be efficiently solved in magnetic field-aligned coordinate systems commonly used in plasma simulations, and is highly effective for both hyperbolic (Alfvén wave) and parabolic (heat conduction) problems. The preconditioning steps have intuitive interpretations, making it straightforward to experiment with adding and removing terms in the equations in order to optimize the preconditioner. By applying this method to a six-field reduced MHD model, order-of-magnitude reductions in wall clock time have been achieved, even in regimes where nonlinear terms in the evolution equations are important. The application of this improved code to ELM simulations, and predictions for ITER, will be reported in a separate paper.

Using a relatively small problem size, we have demonstrated good scaling up to 512 processors. Since the costly matrix inversion step in the preconditioner is trivially parallelized in two dimensions (ψ and ζ), increasing the problem size should allow scaling to a much larger number of processors. Testing this will be the subject of future work.

Other directions of future work include the development of additional physics-based preconditioning variants, such as fieldsplits for fast Alfvén waves and thermal conductivity along the fieldlines. We are also exploring broader issues in time-dependent simulations, including the use of implicit-explicit (IMEX) approaches for flexible timestepping. Work is under way to incorporate BOUT++ into the FACETS framework [49, 50] in order to facilitate research on core-edge-wall coupling.

Acknowledgments

We gratefully acknowledge use of the Fusion cluster in the Laboratory Computing Resource Center at Argonne National Laboratory. S. Farley and L. McInnes were supported by the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract DE-AC02-06CH11357. B. Dudson was supported by EFDA WP11 and HPC-FF computing resources. This work was carried out largely as part of the FACETS SciDAC project supported by DOE. We also thank L. Chacon for his presentation and insightful discussions at the 2011 BOUT++ workshop, which provided the impetus for this work.

References

- [1] ITER homepage, <http://www.iter.org> (2007).
- [2] M. Keilhacker, F. Becker, K. Bernhardt, et al., *Plasma Phys. Control. Fusion* 26 (1984) 49.
- [3] E. J. Doyle, et al., in: *22nd IAEA Fusion Energy Conference*, 2008.
- [4] J. W. Connor, R. J. Hastie, H. R. Wilson, R. L. Miller, *Physics of Plasmas* 5 (1998) 2687.
- [5] P. B. Snyder, et al., *Physics of Plasmas* 9 (2002) 2037.
- [6] H. R. Wilson, et al., *Physics of Plasmas* 9 (2002) 1277.
- [7] C. F. Maggi, *Nucl. Fusion* 50 (2010) 066001.
- [8] A. W. Leonard, et al., *J. Nucl. Materials* 266-269 (1999) 109–117.
- [9] G. Federici, A. Loarte, G. Strohmayer, *Plasma Phys. Control. Fusion* 45 (2003) 1523–1547.
- [10] J. A. Krommes, *Annual Review of Fluid Mechanics* 44 (1) (2012) 175–201. doi:10.1146/annurev-fluid-120710-101223.
- [11] S. I. Braginskii, Transport processes in a plasma, *Reviews of Plasma Physics* 1 (1965) 205.
- [12] H. R. Strauss, *Phys. Fluids* 19 (1976) 134.
- [13] R. D. Hazeltine, J. D. Meiss, *Plasma Confinement*, 2nd Edition, Dover, 2003.
- [14] S. C. Jardin, *J. Comput. Phys.* 231 (2012) 822–838.
- [15] B. Dudson, M. Umansky, X. Xu, P. Snyder, H. Wilson, *Comp. Phys. Comm.* 180 (9) (2009) 1467–1480.
- [16] X. Q. Xu, B. D. Dudson, P. B. Snyder, M. V. Umansky, H. R. Wilson, *Phys. Rev. Lett.* 105 (2010) 175005.
- [17] B. D. Dudson, X. Q. Xu, M. V. Umansky, P. B. Snyder, H. R. Wilson, *Plasma Phys. Control. Fusion* 53 (2011) 054005.
- [18] X. Xu, B. Dudson, P. Snyder, M. Umansky, H. Wilson, T. Casper, *Nuclear Fusion* 51 (10) (2011) 103040.
- [19] D. A. Knoll, D. E. Keyes, *J. Comput. Phys.* 193 (2004) 357–397.
- [20] V. A. Mousseau, D. A. Knoll, *J. Comput. Phys.* 190 (1) (2003) 42–51. doi:10.1016/S0021-9991(03)00252-3.
- [21] L. Chacón, D. A. Knoll, J. M. Finn, *J. Comput. Phys.* 178 (1) (2002) 15–36. doi:10.1006/jcph.2002.7015.
- [22] E. Santilli, A. Scotti, *J. Comput. Phys.* 230 (23) (2011) 8342–8359. doi:10.1016/j.jcp.2011.06.022.
- [23] K. N. Premnath, M. J. Pattison, S. Banerjee, *J. Comput. Phys.* 228 (3) (2009) 746–769. doi:10.1016/j.jcp.2008.09.028.
- [24] S. P. Hirshman, R. Sanchez, C. R. Cook, *Physics of Plasmas* 18 (6) (2011) 062504.
- [25] B. Jamroz, S. Kruger, T. Austin, in: *APS*, 2010.
- [26] M. McCourt, T. D. Rognlien, L. C. McInnes, H. Zhang, *Computational Science and Discovery* 5 (014012). doi:10.1088/1749-4699/5/1/014012.
- [27] L. Chacón, *Physics of Plasmas* 15 (2008) 056103.
- [28] C. Oosterlee, T. Washio, *SIAM Journal on Scientific Computing* 21 (5) (2000) 1670–1690.
- [29] B. Friedman, et al., arXiv (2012) plasm-ph/1205.2337.
- [30] J. R. Angus, M. V. Umansky, S. I. Krasheninnikov, *Phys. Rev. Lett.* 108 (2012) 215002.
- [31] N. M. Ferraro, S. C. Jardin, P. B. Snyder, *Physics of Plasmas* 17 (10) (2010) 102508. doi:10.1063/1.3492727.
- [32] B. J. Burke, et al., *Physics of Plasmas* 17 (2010) 032103. doi:10.1063/1.3309732.
- [33] L. C. Bernard, F. J. Helton, R. W. Moore, *Comp. Phys. Comm.* 24 (3-4) (1981) 377–380.
- [34] H. R. Wilson, S. C. Cowley, *Phys. Rev. Lett.* 92 (2004) 175006.
- [35] S. Pamela, G. Huysmans, S. Benkadda, *Plasma Phys. Control. Fusion* 52 (2010) 075006.
- [36] M. Hoelzl, et al., arXiv (2012) 1201.5765.

- [37] L. E. Sugiyama, the M3D team, *J. Phys.: Conf. Ser.* 180 (2009) 012060.
- [38] S. C. Jardin, J. Breslau, N. Ferraro, *J. Comput. Phys.* 226 (2007) 2146.
- [39] G. T. A. Huysmans, S. Pamela, E. van der Plas, P. Ramet, *Plasma Phys. Control. Fusion* 51 (12) (2009) 124012.
- [40] G.-S. Jiang, C.-W. Shu, *J. Comput. Phys.* 126 (1996) 202–228.
- [41] U. Ascher, L. Petzold, *Society for Industrial Mathematics*, 1998.
- [42] J. Nocedal, S. J. Wright, *Numerical Optimization*, Springer-Verlag, New York, 1999.
- [43] L. Chacon, D. A. Knoll, J. M. Finn, *J. Comput. Phys.* 188 (2002) 15–36.
- [44] T. M. Austin, M. Berndt, J. D. Moulton, *Tech.Rep. LA-UR 03-4149*, LANL, USA (2004).
- [45] D. Anderson, *Journal of the ACM (JACM)* 12 (4) (1965) 547–560.
- [46] S. Balay, et al., *Tech. Rep. ANL-95/11 - Revision 3.1*, Argonne National Laboratory (2010).
- [47] S. Balay, W. D. Gropp, L. C. McInnes, B. F. Smith, in: E. Arge, A. M. Bruaset, H. P. Langtangen (Eds.), *Modern Software Tools in Scientific Computing*, Birkhauser Press, 1997, pp. 163–202.
- [48] R. Fitzpatrick, *Physics of Plasmas* 5 (9).
- [49] J. Cary, et al., *J. Phys.: Conf. Ser.* 180 (2009) 012056.
- [50] J. Cary, et al., in: *23rd annual IAEA Fusion Energy Conference (FEC 2010)*, Daejeon, Republic of Korea, 2010.

Government License. The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory (“Argonne”). Argonne, a U.S. Department of Energy Office of Science laboratory, is operated under Contract No. DE-AC02-06CH11357. The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government.