An optimal, parallel, fully implicit Newton–Krylov solver for three-dimensional viscoresistive magnetohydrodynamics[a)]

L. Chacón[b)]
Theoretical Division, Los Alamos National Laboratory, P.O. Box 1663, Los Alamos, New Mexico 87545, USA

(Received 9 November 2007; accepted 3 January 2008; published online 12 February 2008)

The conceptual development and implementation of a scalable nonlinear solver for the time-dependent three-dimensional (3D) compressible resistive magnetohydrodynamics model (MHD) is discussed. The approach is based on Jacobian-free Newton–Krylov technology, preconditioned with multigrid methods for algorithmic scalability. The key to the approach is the reformulation of the hyperbolic MHD system into a parabolic one, which is amenable to multigrid techniques. Such reformulation (parabolization) aims to render the modified system block diagonally dominant (unlike the original MHD system, which is diagonally submissive for implicit time steps Δt larger than the explicit Courant–Friedrichs–Lewy time step). The algorithm has been tested on a variety of 2D and 3D configurations which demonstrate its excellent algorithmic scalability properties. In particular, it is shown that, serially, the CPU time for a given simulation scales linearly with the number of unknowns, and that large CPU gains (~ 30) are attainable with respect to an explicit approach even for moderate grids (256×256). In parallel, the algorithm features excellent scalability properties up to thousands of processors (4096) and millions of unknowns ($\sim 1.3 \times 10^8$). © *2008 American Institute of Physics*. [DOI: 10.1063/1.2838244]

I. INTRODUCTION

The three-dimensional (3D) compressible viscoresistive magnetohydrodynamic (MHD) equations present a formidable challenge for efficient implicit differencing due to the disparity of time scales, the nonlinear couplings present in the equations, and the fine grids needed to resolve evolving current sheets. Partially implicit time-differencing schemes have been employed in multidimensional MHD calculations, which allow time steps larger than the explicit Courant–Friedrichs–Lewy (CFL) stability limit Δt_{CFL} while requiring less work than a direct implicit integration for a given time step. Such partially implicit techniques include alternating direction implicit (ADI) methods,[1,2] Harned–Kerner (H-K) semi-implicit techniques,[3–9] and implicit-explicit techniques.[10–14] In ADI, the semi-implicit operator decouples the integration in the different directions of the problem (p. 172 in Ref. 15). Consequently, it is possible to restrict the implicit integration to one direction while the others remain explicit, alternating directions every time step. However, in MHD, the approach suffers from complications due to the nonlinear couplings and the presence of mixed derivatives, requiring further simplifications to render the problem tractable. As a result, ADI methods in MHD problems only allow time steps $\sim 10-20$ times larger than the explicit numerical stability limit for stability.[1,2]

In H-K semi-implicit methods, the semi-implicit operator slows down the propagation of the fastest waves in the problem, thus achieving unconditional numerical stability for a sufficiently large semi-implicit coefficient. However, the accuracy of the numerical solution obtained with MHD semi-implicit techniques is strongly dependent on the choice of the form of the semi-implicit operator.[5,6] As a consequence, the time step in H-K semi-implicit methods is typically limited by *accuracy* considerations. For instance, in Ref. 5, a time step limit of $\gamma \Delta t \leq 0.05$ was reported for the $m=1$, $n=-2$ tearing mode, with γ the growth rate. In general, time-consuming convergence studies are required to find the time-step accuracy limit. Nevertheless, the approach has been favored by implicit MHD practitioners (such as in the fusion community), and is currently employed in well-known simulation tools such as the NIMROD code.[8,16] Recently, a novel implementation of the H-K semi-implicit idea that improves on the temporal accuracy issue has been proposed in Ref. 9 in the context of a low-β extended reduced MHD model. The algorithm is iterative (and shown to converge to the full nonlinear problem if enough iterations are used), and features temporal error control and robust damping for the dissipative terms. Preliminary results employing a pseudospectral spatial representation indicate that the approach is accurate, and competitive with scalable fully implicit methods for similar extended MHD models.[17]

Implicit-explicit methods only treat implicitly certain terms, while others are treated in an explicit fashion. Terms treated implicitly are usually chosen to remove a source of numerical stiffness in the problem, either parabolic (diffusion) or hyperbolic (waves), or both. The approach is attractive because the implicit treatment of the selected terms renders the algorithm manageable while still being able to provide efficiency benefits (thus its acceptance among implicit MHD practitioners[10–14]). However, the approach is conditionally stable (i.e., only those time scales treated implicitly are stabilized), and is usually first-order accurate in time (and hence may suffer from significant numerical diffusion).

There have been recent attempts at devising nonlinear

[a)]Paper BI2 3, Bull. Am. Phys. Soc. **52**, 23 (2007).
[b)]Invited speaker.

implicit algorithms for the 3D MHD equations.[18,19] In Ref. 18, a block Gauss-Seidel technique is employed to invert an approximate Jacobian matrix (and LU-factorization techniques to invert the blocks) to obtain convergence including the nonlinear terms in the equations. However, the simplifications in the Jacobian (required to render it manageable) limit the applicability of the method, because they introduce CFL restrictions for large and moderate viscosities and resistivities. A nonlinear iteration is featured also in Ref. 19 for incompressible MHD, which revolves around the SIMPLE (p. 126 in Ref. 20) concept for incompressible flows.

Despite their poor scalability properties, and as an alternative to poorly converging iterative techniques, direct inversion methods have been used by several authors to invert the ill-conditioned matrices resulting from the (partially or fully) implicit discrete MHD model. In particular, well-known simulation tools such as M3D (Refs. 13 and 14) and NIMROD (Ref. 8) currently employ direct methods in some (crucial) step of their time integration algorithms. Some authors even employ direct methods within a Newton–Raphson nonlinear iteration.[21] The advantages of direct methods versus iterative ones is their robustness in the face of matrix ill-conditioning. Their disadvantage is, of course, the poor scaling with the total number of unknowns, and with the number of processors (when sufficiently large).

Modern linear algebra inversion techniques such as preconditioned Krylov subspace iterative methods[22] have been explored in the context of MHD in Refs. 10, 11, 17, and 23–27. In Ref. 23, an implicit MHD solver is proposed based on a implicit-operator-split (IOS) approach, developed earlier for hydrodynamic applications.[28] The IOS algorithm employs Krylov solvers for required inversions in each split step, and is iterated upon in a Gauss-Seidel manner to achieve some degree of nonlinear consistency. However, for large implicit time steps, large numerical errors are reported possible in transient calculations with this algorithm unless enough nonlinear iterations are taken.[23] In Refs. 10 and 11, incomplete-LU-preconditioned Krylov methods are employed to invert the linearly implicit (and also implicit-explicit) set of MHD equations in the context of the Versatile Advection Code. CPU speedups versus explicit of ~ 40 are reported.[10] An unpreconditioned Newton–Krylov solver for 3D compressible MHD is explored in Ref. 25, which also reports order-of-magnitude speedups versus explicit approaches for fine enough grids. More recently, the same researchers have developed an "operator-based" parallel preconditioner for 3D MHD, based on directional splitting of the implicit operator and followed by a characteristic decomposition of the resulting directional PDE operators.[29] Reference 26 explores a Newton–Krylov–Schwarz parallel approach for the reduced Hall MHD model, where gains of an order of magnitude with respect to explicit approaches and good parallel scalability are reported. Finally, Refs. 17 and 27 develop optimal preconditioning strategies for a fully implicit Newton–Krylov treatment of 2D reduced resistive and Hall MHD. These studies report speedups of an order of magnitude or more, and provide the foundation for the approach pursued in this paper.

The present document explores efficient paths for a parallel, fully implicit, fully nonlinear, unsplit differencing of the 3D resistive compressible MHD equations. The objective is to demonstrate a scalable algorithm under grid refinement (defined as featuring a linear scaling of the CPU with the number of unknowns) and with the number of processors (in a weak scaling sense) in massively parallel computers. Convergence of the nonlinear system is achieved using the Newton–Raphson iterative algorithm. Krylov iterative techniques,[22] implemented Jacobian-free[30,31] (i.e., without ever forming and storing the Jacobian matrix) for memory efficiency, are employed for the required algebraic matrix inversions. Here, Flexible Generalized Minimal RESiduals (FGMRES) (Refs. 32 and 33) is employed as the Krylov solver of choice, due to the lack of symmetry in the algebraic system of interest. The flexible character of FGMRES relaxes some of the constraints in the preconditioner step of regular GMRES, which we have found useful in our implementation. In particular, FGMRES allows the preconditioner to change between successive GMRES iterations. For parallelization, we employ the Parallel Extensible Toolkit for Scientific computing[34–36] (PETSc) library.

The efficiency and scalability of Krylov methods depend strongly on adequate preconditioning.[22] Here, we extend previous work on the 2D reduced resistive[27] and Hall[17] MHD models (which pioneered the multilevel "physics-based" preconditioning approach) to the 3D compressible MHD model. As in those references, a suitable multigrid-based preconditioner is developed around the parabolization concept, whereby a hyperbolic system is reformulated as a parabolic one, which is in turn amenable to a multilevel treatment. The connection between the parabolization procedure and the Schur block decomposition, outlined first in Ref. 17, is of the essence in the context of 3D MHD to formulate such a preconditioner.

The rest of the paper is organized as follows: Sec. II introduces the base model equations. Section III introduces the Krylov methods and the specifics of the Jacobian-free implementation. In Sec. IV, the physics-based preconditioner for this particular application is discussed. Details of our parallel implementation are given in Sec. V. Serial and parallel efficiency results of the resulting implicit algorithm in various 2D and 3D configurations are presented in Sec. VI, where it is numerically demonstrated that:

1. The algorithm performs optimally under grid refinement (i.e., CPU $\sim N$, with N the number of unknowns). In fact, for some of the problems considered, superoptimal convergence rates are observed.
2. The algorithm features a favorable scaling of the number of Krylov iterations with increasing implicit time steps, which results in larger CPU gains when larger implicit time steps are employed.
3. The algorithm performs optimally in parallel, with little deviation from the ideal parallel scaling (in a weak sense) for the number of processors tested (up to 4096 processors and 134 million unknowns).

Finally, we conclude in Sec. VII.

II. 3D VISCORESISTIVE MHD MODEL

We consider the compressible viscoresistive MHD model, given by

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v} - D \nabla \rho) = 0, \tag{1}$$

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times \mathbf{E} = 0, \quad \mathbf{E} = -\mathbf{v} \times \mathbf{B} + \frac{\eta}{\mu_0} \nabla \times \mathbf{B}, \tag{2}$$

$$\frac{\partial (\rho \mathbf{v})}{\partial t} + \nabla \cdot \left[\rho \mathbf{v}\mathbf{v} - \frac{\mathbf{B}\mathbf{B}}{\mu_0} + \mathbb{I}\left(p + \frac{B^2}{2\mu_0} \right) + \Pi \right] = 0, \tag{3}$$

$$\frac{\partial T_e}{\partial t} + \mathbf{v} \cdot \nabla T_e + (\gamma - 1) \left[T_e \nabla \cdot \mathbf{v} + \frac{\nabla \cdot \mathbf{q} - Q}{\alpha_T n} \right] = 0, \tag{4}$$

with $\mathbb{I}$ the identity operator, $p = n(T_i + T_e) = nT_e(1 + \alpha_T)$ the pressure, $\alpha_T = T_i / T_e$ the ion/electron temperature ratio, and n the particle density. In these equations, $\rho = m_i n$ is the ion mass density, $\mathbf{v}$ is the plasma velocity, $\mathbf{B}$ is the magnetic field, η is the resistivity, D is a particle diffusivity that models cross-field particle diffusion [such particle diffusivity D is not self-consistently considered in Eq. (3), and hence should be regarded as *ad hoc* in nature], and $Q = \eta j^2 - \Pi : \nabla \mathbf{v}$ contains the joule and viscous heating sources. Simple closures for the heat flux $\mathbf{q} = -\kappa \nabla T_e$ and the viscous stress tensor $\Pi = -\rho \nu \nabla \mathbf{v}$ are considered at this stage. More accurate closures (such as parallel electron heat transport and gyroviscous stresses) will be considered in future work.

The energy $E = \rho v^2/2 + B^2/2\mu_0 + p/(\gamma - 1)$ will not be conserved in this model to numerical round-off due to discretization errors in the temperature equation. Alternatively to the temperature equation, one can consider the energy equation,

$$\frac{\partial E}{\partial t} + \nabla \cdot \left[\mathbf{v} \left(\rho \frac{v^2}{2} + \frac{\gamma p}{\gamma - 1} \right) + \mathbf{E} \times \mathbf{B} + \Pi \cdot \mathbf{v} + \mathbf{q} \right] = 0. \tag{5}$$

The temperature equation is advantageous to preserve numerical accuracy when modeling low-β plasmas (as in many magnetic fusion applications), since in that case the thermal energy is a small contributor to the total energy inventory of the plasma. On the other hand, the energy equation is advantageous when exact conservation of energy is critical, such as in the presence of shocks.

In Alfvénic units (i.e., Alfvén speed $v_A = B_0 / \sqrt{\mu_0 \rho_0}$, Alfvén time $\tau_A = L / v_A$, where B_0, ρ_0, and L are the reference magnetic field, density, and length, respectively), the dimensionless form of the set of equations above reads

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v} - D \nabla \rho) = 0, \tag{6}$$

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \cdot (\mathbf{v}\mathbf{B} - \mathbf{B}\mathbf{v}) + \nabla \times (\eta \nabla \times \mathbf{B}) = 0, \tag{7}$$

$$\frac{\partial (\rho \mathbf{v})}{\partial t} + \nabla \cdot \left[\rho \mathbf{v}\mathbf{v} - \mathbf{B}\mathbf{B} - \rho \nu \nabla \mathbf{v} + \mathbb{I}\left(p + \frac{B^2}{2} \right) \right] = 0, \tag{8}$$

$$\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T + (\gamma - 1) \left[T \nabla \cdot \mathbf{v} - \frac{\kappa \nabla^2 T + Q}{\alpha_T \rho} \right] = 0, \tag{9}$$

and $p = (1 + \alpha_T) \rho T$. At this point all quantities are dimensionless, and η, ν are the reciprocals of the Lundquist number and the Reynolds number, respectively.

The system in Eqs. (6)–(9) is hyperbolic, supporting a variety of linear normal modes (such as the fast and slow magnetosonic waves, and the shear Alfvén wave; see Chap. 11 in Ref. 37). In some applications (notably, in magnetic fusion confinement), dynamical time scales of interest are much slower than those associated with these normal modes. In such contexts, an implicit approach that steps over the normal-mode time scales to resolve those of interest is useful. This is the subject of this paper. However, as we shall see, the hyperbolic character of the MHD model makes the task of developing an optimal, scalable solver a difficult task.

A word about the discretization of Eqs. (6)–(9) is in order. Spatially, the system is discretized using finite volumes, as documented in detail in Ref. 38. Such spatial discretization has proved to be conservative, solenoidal in the magnetic field to numerical round-off, and remarkably robust in the absence of physical and/or numerical dissipation. Temporally, we employ a θ-scheme, with $\theta = 0.5$ (second-order Crank-Nicolson). This choice results in a set of nonlinear algebraic equations $\mathbf{G}(\mathbf{x}) = \mathbf{0}$, with $\mathbf{x}^T = (\rho, T, \mathbf{B}, \mathbf{v})$, that needs to be inverted every time step. For this, we employ preconditioned Newton–Krylov methods. The next section introduces these methods and some of their properties.

III. JACOBIAN-FREE NEWTON–KRYLOV METHODS

We proceed to give a brief introduction to Jacobian-free Newton–Krylov methods (JFNK). The motivated reader can find extensive discussions on this approach elsewhere.[39] Newton's method solves the nonlinear system $\mathbf{G}(\mathbf{x}) = \mathbf{0}$ iteratively by inverting linear systems of the form

$$\left. \frac{\partial \mathbf{G}}{\partial \mathbf{x}} \right|_k \delta \mathbf{x}_k = -\mathbf{G}(\mathbf{x}_k),$$

with $\mathbf{x}_{k+1} = \mathbf{x}_k + \delta \mathbf{x}_k$. Nonlinear convergence is achieved when

$$\|\mathbf{G}(\mathbf{x}_k)\|_2 < \epsilon_a + \epsilon_r \|\mathbf{G}(\mathbf{x}_0)\|_2 = \epsilon_t, \tag{10}$$

where $\|\cdot\|_2$ is the ℓ_2-norm (Euclidean norm), $\epsilon_a = \sqrt{N} \times 10^{-15}$ (with N the total number of degrees of freedom) is an absolute tolerance to avoid converging below round-off, ϵ_r is the Newton relative convergence tolerance (set to 10^{-4} in this work), and $\mathbf{G}(\mathbf{x}_0)$ is the initial residual.

Such linear systems are solved iteratively with Krylov methods, which only require matrix-vector products to proceed. Because the linear system matrix is a Jacobian matrix, such matrix-vector products can be implemented Jacobian-free using the Gateaux derivative,

$$\left. \frac{\partial \mathbf{G}}{\partial \mathbf{x}} \right|_k \mathbf{v} = \lim_{\epsilon \to 0} \frac{\mathbf{G}(\mathbf{x}_k + \epsilon \mathbf{v}) - \mathbf{G}(\mathbf{x}_k)}{\epsilon}, \tag{11}$$

where in practice a small but finite ϵ is employed (p. 79 in Ref. 39). Thus, the evaluation of the Jacobian-vector product only requires the function evaluation $\mathbf{G}(\mathbf{x}_k + \epsilon \mathbf{v})$, and there is

no need to form or store the Jacobian matrix. This, in turn, allows for a memory-efficient implementation.

An inexact Newton method[40] is used to adjust the convergence tolerance of the Krylov method at every Newton iteration according to the size of the current Newton residual as follows:

$$\|J_k \delta \mathbf{x}_k + \mathbf{G}(\mathbf{x}_k)\|_2 < \zeta_k \|\mathbf{G}(\mathbf{x}_k)\|_2, \tag{12}$$

where ζ_k is the inexact Newton parameter and

$$J_k = \frac{\partial \mathbf{G}}{\partial \mathbf{x}}\Big|_k$$

is the Jacobian matrix. Thus, the convergence tolerance of the Krylov method is loose when the Newton state vector $\mathbf{x}_k$ is far from the nonlinear solution, but tightens as $\mathbf{x}_k$ approaches the solution. Hence, the linear solver works the hardest when the Newton state vector is closest to the nonlinear root. Superlinear convergence rates of the inexact Newton method are possible if the sequence of ζ_k is chosen properly (p. 105 in Ref. 39). Here, we employ the same prescription as in Ref. 17,

$$\zeta_k^A = \gamma \left[\frac{\|\mathbf{G}(\mathbf{x}_k)\|_2}{\|\mathbf{G}(\mathbf{x}_{k-1})\|_2} \right]^\alpha,$$

$$\zeta_k^B = \min[\zeta_{\max}, \max(\zeta_k^A, \gamma \zeta_{k-1}^\alpha)],$$

$$\zeta_k = \min\left[\zeta_{\max}, \max\left(\zeta_k^B, \gamma \frac{\epsilon_t}{\|\mathbf{G}(\mathbf{x}_k)\|_2} \right) \right],$$

with $\alpha=1.5$, $\gamma=0.9$, and $\zeta_{\max}=0.8$. The convergence tolerance ϵ_t is defined in Eq. (10). In this prescription, the first step ensures superlinear convergence (for $\alpha>1$), the second avoids volatile decreases in ζ_k, and the last avoids oversolving in the last Newton iteration.

A further advantage of Krylov methods is that they can be preconditioned by considering the alternate (but equivalent) systems $J_k P_k^{-1} P_k \delta \mathbf{x}_k = -\mathbf{G}_k$ (right preconditioning) or $P_k^{-1} J_k \delta \mathbf{x}_k = -P_k^{-1} \mathbf{G}_k$ (left preconditioning). Such preconditioned systems can be straightforwardly and efficiently implemented in the Krylov algorithm as two consecutive matrix-vector products. A crucial feature of preconditioning is that, while it can substantially improve the convergence properties of the Krylov iteration if $P_k^{-1} \approx J_k^{-1}$, it does not alter the solution of the Jacobian system upon convergence (because the solution $\delta \mathbf{x}_k$ of the preconditioned system is the same as that of the original system). Therefore, one can explore suitable approximations in the preconditioner for efficiency purposes without compromising the accuracy of the converged result.

This paper concentrates on the development of a scalable, robust preconditioning strategy for the Newton–Krylov inversion of the 3D compressible resistive MHD model outlined in Eqs. (6)–(9). Right preconditioning is favored here, because the right-hand side of the Jacobian system $-\mathbf{G}_k$ (which measures nonlinear convergence) is unaffected by the preconditioner. In practice, right preconditioning is implemented in two steps, namely, a linear solver $J_k P_k^{-1} \delta \mathbf{y}_k = -\mathbf{G}_k$, and *a posteriori* application of the preconditioner to find $\delta \mathbf{x}_k = P_k^{-1} \delta \mathbf{y}_k$. We use FGMRES (Ref. 33) as the Krylov method of choice for its robustness in nonsymmetric, indefinite systems (which is the case for compressible MHD due to the presence of flow) and the added flexibility in the preconditioning stage.

We proceed to discuss our approach to preconditioning.

IV. PRECONDITIONING STRATEGY

While implementing an unpreconditioned JFNK solver is fairly straightforward, it is not algorithmically scalable, and some form of preconditioning is required to accelerate convergence. Our goal here is to develop a preconditioner effective enough that results in optimal convergence rates (i.e., independent of the number of unknowns and the number of processors considered). To achieve this goal, we employ multigrid methods (MG) in the preconditioner stage, which have been shown in a variety of applications[17,27,41] that they can lead to such optimal JFNK convergence rates.

In particular, our work builds on previous developments in 2D resistive[27] and Hall[17] reduced MHD, where such optimal behavior has been demonstrated. In these references, the key for an effective multigrid implementation was the parabolization of otherwise hyperbolic PDEs. The approach, which was termed "physics-based," aimed at reformulating the semidiscrete (temporally discrete, spatially continuous) set of PDEs into a diagonally dominant set, amenable to classical smoothing (based on stationary iterative techniques) in a MG setting.

MG methods employ a divide-and-conquer approach where multiple grids of varying refinement are employed.[42,43] The underlying idea is that oscillatory components of the error can be readily attacked at a given grid level (with a so-called smoother), but smooth ones are difficult. In its simplest form (a MG V-cycle), the procedure then involves smoothing the error on a given level, and then coarsening (restricting) the smooth components to the next coarse grid level. In the new level, some of the smooth components will appear oscillatory, and therefore can be subjected to further smoothing. The process is performed recursively until the coarsest grid level is reached, at which point a direct solver can be performed very cheaply. The solution is then interpolated (prolongated) up the grid hierarchy, until the finest level is reached. While variations of the basic V-cycle are possible to improve the convergence rate of a given MG implementation (p. 47 in Ref. 42), one or several V-cycles are generally enough for preconditioning purposes. As can be understood from the previous description, the crucial element for a working MG solver is the availability of a smoother. While smoothers can be found fairly easily for diagonally dominant systems (in a point or block sense; p. 96 in Ref. 43), it is remarkably hard otherwise.

Hyperbolic systems (such as MHD) can be shown to be diagonally submissive when time steps larger than the explicit CFL stability constraint are employed (see Ref. 27 for

an in-depth explanation of this issue). This, in turn, hinders the task of finding a suitable smoother for MG. However, as has been shown in the context of 2D reduced MHD,[17,27] hyperbolic systems can be conveniently parabolized in an implicit time-stepping setting. The basic idea is to produce a well-conditioned (diagonally dominant) parabolic operator from an ill-conditioned hyperbolic system of equations.[27] The procedure can be understood easily with a first-order coupled hyperbolic linear system,

$$\partial_t u = \partial_x v,$$

$$\partial_t v = \partial_x u.$$

Differencing implicitly in time (with backward Euler for simplicity) but keeping the continuum spatial representation, we have

$$u^{n+1} = u^n + \Delta t \partial_x v^{n+1}, \tag{13}$$

$$v^{n+1} = v^n + \Delta t \partial_x u^{n+1}. \tag{14}$$

It is now possible to substitute the second equation into the first to obtain the following parabolic equation:

$$(\mathbb{I} - \Delta t^2 \partial_{xx}) u^{n+1} = u^n + \Delta t \partial_x v^n, \tag{15}$$

which is equivalent to Eqs. (13) and (14), but much better conditioned because the parabolic operator is diagonally dominant. A similar idea is behind the method of differential approximations,[44] and the semi-implicit solvers developed in the MHD context[3,4,6] (although, in these references, the parabolic operator is obtained and implemented in an ad hoc manner).

It is useful at this point, using this simple example, to establish an important connection between the parabolic operator obtained above and the Schur complement. We start by writing Eqs. (13) and (14) in matrix form,

$$\begin{bmatrix} \mathbb{I} & -\Delta t \partial_x \\ -\Delta t \partial_x & \mathbb{I} \end{bmatrix} \begin{bmatrix} u^{n+1} \\ v^{n+1} \end{bmatrix} = \begin{bmatrix} u^n \\ v^n \end{bmatrix}.$$

The matrix can be block-factorized as follows:

$$\begin{bmatrix} \mathbb{I} & -\Delta t \partial_x \\ -\Delta t \partial_x & \mathbb{I} \end{bmatrix} = \begin{bmatrix} \mathbb{I} & -\Delta t \partial_x \\ 0 & \mathbb{I} \end{bmatrix} \begin{bmatrix} P_{\mathrm{SC}} & 0 \\ 0 & \mathbb{I} \end{bmatrix} \times \begin{bmatrix} \mathbb{I} & 0 \\ -\Delta t \partial_x & \mathbb{I} \end{bmatrix},$$

where $P_{\mathrm{SC}} = \mathbb{I} - \Delta t^2 \partial_x^2$ is the so-called Schur complement. The connection between the parabolic operator an the Schur complement is now obvious. The factorized form can be processed easily for inversion. Pre- and post-triangular matrices are easily (and exactly) invertible (they correspond to forward elimination and backward substitution, respectively), and yield Eqs. (14) and (15) as a result. Only the inversion of P_{SC} requires an iterative treatment. The usefulness of this association for the present application is demonstrated in the next section.

A. Parabolization of the resistive MHD model

As discussed earlier, the preconditioner stage should approximate the inverse of the Jacobian matrix for the compressible MHD system. Neglecting heat sources and density factors in selected places, the coupling structure of the linearized MHD system in terms of the linear updates $\delta\rho$, δT, $\delta\mathbf{B}$, and $\delta\mathbf{v}$ reads

$$\delta\rho = \mathcal{L}_\rho(\delta\rho, \delta\mathbf{v}),$$

$$\delta T = \mathcal{L}_T(\delta T, \delta\mathbf{v}),$$

$$\delta\mathbf{B} = \mathcal{L}_B(\delta\mathbf{B}, \delta\mathbf{v}),$$

$$\delta\mathbf{v} = \mathcal{L}_v(\delta\mathbf{v}, \delta\mathbf{B}, \delta\rho, \delta T),$$

where the $\mathcal{L}_i$ represent linear operators. Clearly, while density, temperature, and magnetic-field updates only couple to themselves and the velocity update, the velocity update equation couples all of them. In matrix form, the Jacobian structure that results is

$$J\delta\mathbf{x} = \begin{bmatrix} D_\rho & 0 & 0 & U_{\rho\mathbf{v}} \\ 0 & D_T & 0 & U_{T\mathbf{v}} \\ 0 & 0 & D_\mathbf{B} & U_{\mathbf{B}\mathbf{v}} \\ L_{\mathbf{v}\rho} & L_{\mathbf{v}T} & L_{\mathbf{v}\mathbf{B}} & D_\mathbf{v} \end{bmatrix} \begin{pmatrix} \delta\rho \\ \delta T \\ \delta\mathbf{B} \\ \delta\mathbf{v} \end{pmatrix}.$$

The diagonal blocks are given by

$$D_\rho \delta\rho = \frac{\delta\rho}{\Delta t} + \theta[\nabla\cdot(\mathbf{v}_0\delta\rho) - D\nabla^2\delta\rho],$$

$$D_T\delta T = \left[\frac{1}{\Delta t} + \theta(\gamma-1)\nabla\cdot\mathbf{v}_0\right]\delta T + \theta\left(\mathbf{v}_0\cdot\nabla\delta T - \frac{\kappa}{\alpha_T\rho_0}\nabla^2\delta T\right),$$

$$D_\mathbf{B}\delta\mathbf{B} = \frac{\delta\mathbf{B}}{r\Delta t} + \theta[\nabla\cdot(\mathbf{v}_0\delta\mathbf{B} - \delta\mathbf{B}\mathbf{v}_0) - \eta\nabla^2\delta\mathbf{B}],$$

$$D_\mathbf{v}\delta\mathbf{v} = \rho_0[\delta\mathbf{v}/\Delta t + \theta(\mathbf{v}_0\cdot\nabla\delta\mathbf{v} + \delta\mathbf{v}\cdot\nabla\mathbf{v}_0 - \nu\nabla^2\delta\mathbf{v})],$$

where, for $D_\mathbf{v}$, we have used the nonconservative form of the momentum equation [Eq. (8)]. Here, θ is the time centering parameter. These blocks only contain advection-diffusion terms, which can be readily inverted using MG if upwinding is employed for the advective terms (only in the preconditioning stage[17,27]). Off-diagonal blocks L and U contain all relevant hyperbolic couplings, and are given by

$$U_{\rho\mathbf{v}}\delta\mathbf{v} = \theta\nabla\cdot(\delta\mathbf{v}\rho_0),$$

$$U_{T\mathbf{v}}\delta\mathbf{v} = \theta[\delta\mathbf{v}\cdot\nabla T_0 + (\gamma-1)T_0\nabla\cdot\delta\mathbf{v}],$$

$$U_{\mathbf{B}\mathbf{v}}\delta\mathbf{v} = \theta\nabla\cdot(\delta\mathbf{v}\mathbf{B}_0 - \mathbf{B}_0\delta\mathbf{v}),$$

$$L_{\mathbf{v}\rho}\delta\rho = \theta(1+\alpha_T)\nabla(T_0\delta\rho) + \delta\rho\left(\frac{\mathbf{v_0}}{\Delta t} + \theta\mathbf{v_0}\cdot\nabla\mathbf{v_0} - \nu\theta\nabla^2\mathbf{v_0}\right),$$

$$L_{\mathbf{v}T}\delta T = \theta(1+\alpha_T)\,\nabla\,(\rho_0\delta T),$$

$$L_{\mathbf{vB}}\delta\mathbf{B} = -\,\theta[\mathbf{j}_0\times\delta\mathbf{B} + (\nabla\times\delta\mathbf{B})\times\mathbf{B}_0].$$

The Jacobian matrix has an "arrow" structure, which suggests considering the following 2×2 block structure for analysis purposes:

$$J\delta\mathbf{x} = \begin{bmatrix} M & U \\ L & D_{\mathbf{v}} \end{bmatrix}\begin{pmatrix} \delta\mathbf{y} \\ \delta\mathbf{v} \end{pmatrix},$$

where $\delta\mathbf{y}=(\delta\rho,\delta T,\delta\mathbf{B})^T$, and

$$M = \begin{pmatrix} D_\rho & 0 & 0 \\ 0 & D_T & 0 \\ 0 & 0 & D_{\mathbf{B}} \end{pmatrix}.$$

The block M can be easily invertible, since M is a block diagonal, and, as stated earlier, the blocks themselves are amenable to MG techniques. The Schur factorization of the inverse of the 2×2 block Jacobian matrix yields

$$\begin{bmatrix} M & U \\ L & D_{\mathbf{v}} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbb{I} & -M^{-1}U \\ 0 & \mathbb{I} \end{bmatrix}\begin{bmatrix} M^{-1} & 0 \\ 0 & P_{\mathrm{Schur}}^{-1} \end{bmatrix} \times\begin{bmatrix} \mathbb{I} & 0 \\ -LM^{-1} & \mathbb{I} \end{bmatrix},$$

where $P_{\mathrm{Schur}}=D_{\mathbf{v}}-LM^{-1}U$ is the Schur complement, which contains all the information from the off-diagonal blocks L and U (similarly to the simple coupled-wave-equation example). At this point, the MHD system has been effectively parabolized. The Schur factorization translates into the following 3-step *exact* inversion algorithm (with $\mathbf{G}$ the nonlinear function residual),

$$\begin{aligned} &\text{Predictor: } \delta\mathbf{y}^* = -M^{-1}\mathbf{G}_{\mathbf{y}}, \\ &\text{Velocity update: } \delta\mathbf{v} = P_{\mathrm{Schur}}^{-1}(-\mathbf{G}_{\mathbf{v}} - L\delta\mathbf{y}^*), \\ &\text{Corrector: } \delta\mathbf{y} = \delta\mathbf{y}^* - M^{-1}U\delta\mathbf{v}. \end{aligned} \tag{16}$$

We note that, at this point, no approximations have been introduced, and the exact Jacobian inverse only requires finding M^{-1} and P_{Schur}^{-1}.

B. Physics-based preconditioner: Small-flow approximation

As it stands, inverting P_{Schur} is impractical due to the presence of M^{-1}. (A matrix-free inversion of P_{Schur} using preconditioned Krylov methods is in principle possible if one inverts M accurately enough at every matrix-vector function call; however, this will be expensive, and will defeat the purpose of finding the cheapest working approximation to the Jacobian inverse.) However, a suitable simplification is possible if one considers the *small flow limit*. The key realization is that an implicit approach for the dynamic time integration of the compressible MHD model is most useful when considering small flows (as compared to other speeds in the system such as the Alfvén speed or the sound speed). This is the case in many applications of interest, such as in most magnetic fusion plasmas (where *global* flows are typically limited to a fraction of the Alfvén speed; locally, flows may become Alfvénic, such as in the vicinity of reconnection sites, but in practice this is not found to be an issue. See the tearing mode example in Sec. VI). If large flows were present, motion of features through the mesh would need to be temporally resolved. This, in turn, would impose an effective time step limit for accuracy which would be comparable to other (fast) normal mode time scales in the system, thereby neutralizing the advantages of an implicit approach. [We note, however, that the advantages of implicit techniques do remain even with large flows if one is solving for the steady state (see e.g. Ref. 23), or if the MHD model is extended to include two-fluid electron physics (which supports time scales much faster than ion-related ones). The extension of the current approach to include such electron physics is out of the scope of this study, and is left for future work.]

In the small-flow limit, and neglecting transport terms (transport coefficients in hot, fully ionized plasmas are typically small, with the exception of parallel electron transport, which is not considered here), we can approximate

$$M^{-1} \approx \Delta t\mathbb{I}. \tag{17}$$

Introducing this approximation in steps 2 and 3 of the exact Schur algorithm in Eq. (16), the approximate solver results,

$$\begin{aligned} &\delta\mathbf{y}^* = -M^{-1}\mathbf{G}_{\mathbf{y}}, \\ &\delta\mathbf{v} = P_{\mathrm{SF}}^{-1}[-\mathbf{G}_{\mathbf{v}} - L\delta\mathbf{y}^*], \\ &\delta\mathbf{y} = \delta\mathbf{y}^* - \Delta tU\delta\mathbf{v}, \end{aligned} \tag{18}$$

where $P_{\mathrm{SF}}=D_{\mathbf{v}}-\Delta tLU$ is the small-flow Schur-complement. Taking L and U from the linearized form of Eqs. (6)–(9) [and using the nonconservative form of Eq. (8)], the operator P_{SF} acting on $\delta\mathbf{v}$ reads

$$\begin{aligned} P_{\mathrm{SF}}\delta\mathbf{v} = {}& \rho_0[\delta\mathbf{v}/\Delta t + \theta(\mathbf{v}_0\cdot\nabla\delta\mathbf{v} + \delta\mathbf{v}\cdot\nabla\mathbf{v}_0 - \nu\nabla^2\delta\mathbf{v})] \\ &+ \Delta t\theta^2 W(\mathbf{B}_0,p_0)\delta\mathbf{v}, \end{aligned}$$

where the operator $W(\mathbf{B}_0,p_0)$ results from the composition of hyperbolic off-diagonal operators, and reads

$$\begin{aligned} W(\mathbf{B}_0,p_0)\delta\mathbf{v} = {}& \mathbf{B}_0\times\nabla\times\nabla\times[\delta\mathbf{v}\times\mathbf{B}_0] - \mathbf{j}_0\times\nabla \\ &\times[\delta\mathbf{v}\times\mathbf{B}_0] - \nabla[\delta\mathbf{v}\cdot\nabla p_0 + \gamma p_0\nabla\cdot\delta\mathbf{v}] \end{aligned} \tag{19}$$

with $\mathbf{j}_0=\nabla\times\mathbf{B}_0$. For a homogeneous plasma, the operator $W(\mathbf{B}_0,p_0)$ is the ideal MHD wave propagator and through it all relevant wave propagation information is now contained in P_{SF} (a consequence of the parabolization procedure). P_{SF} couples all the velocity components $\delta\mathbf{v}$, and a coupled (system) solver is necessary to invert it. However, P_{SF} is block diagonally dominant by design, and therefore amenable to a system multigrid treatment (as is demonstrated in Sec. VI).

A few comments are in order. First, the approximation outlined in Eq. (18) can be shown to be equivalent to splitting the $\delta\mathbf{v}$ terms in the linearized equations for $\delta\rho$, δT, and $\delta\mathbf{B}$, and performing the substitutions of the resulting predictor equations in the linearized equation of motion. We point

out the connection to provide some insight into the nature of the approximation in Eq. (17). Secondly, the electromagnetic component of the wave propagator W can be reformulated in a conservative form in terms of the divergence of a symmetric tensor as

$$W_{EM}(\mathbf{B}_0,p_0) = \nabla\cdot[\mathbf{A}\mathbf{B}_0 + \mathbf{B}_0\mathbf{A} - I\mathbf{A}\cdot\mathbf{B}_0],$$

with $\mathbf{A}=-\nabla\times[\delta\mathbf{v}\times\mathbf{B}_0]=\nabla\cdot[\delta\mathbf{v}\mathbf{B}_0-\mathbf{B}_0\delta\mathbf{v}]$. This form can be obtained directly by considering the conservative form of the Lorentz force $\mathbf{j}\times\mathbf{B}$ in the equation of motion, and has proved advantageous from the numerical implementation and multigrid performance standpoints. Finally, we note that the operator P_{SF} is discretized spatially using finite volumes in a 27-point stencil in 3D (9-point stencil in 2D).

V. PARALLEL IMPLEMENTATION

For parallelization, we employ the well-known PETSc library,[34–36] which provides the required nonlinear and linear solvers for this application. While PETSc also provides suitable parallel multigrid solvers, we have chosen to parallelize our own MG implementation (albeit still relying on PETSc distributed array construct for communication). The multigrid solver employed this work features a matrix-light implementation,[17] in which only the diagonal of the system of interest is stored (for smoothing purposes). Coarse operators are found via rediscretization, and required residuals in the MG iteration are found in a matrix-free manner. Spatially varying coefficients required to form the matrix-free operators are not communicated; rather, they are interpolated to coarser grids and extrapolated to ghost-cells in a second-order accurate fashion. This avoids costly communication, and has not resulted in appreciable algorithmic degradation in our numerical tests. For smoothing, we use pointwise weighed Jacobi, which again parallelizes very well. As a result, and in addition to serial algorithmic scalability, the matrix-light MG approach features excellent parallel performance (as demonstrated in the next section).

Our MG implementation does not currently feature a coarse-grid solver for the processor mesh. We expect this to affect the performance of the solver for a sufficiently large number of processors. Results in the next section will confirm that this is the case for more than 1000 processors. Future work will concentrate on adding this capability.

VI. NUMERICAL RESULTS

In what follows, we present efficiency results for two test problems in Cartesian geometry: A tearing mode (also employed in Ref. 38), and the island coalescence problem.[45,46] These problems have the potential of benefiting from an implicit treatment, since the dynamical time scale of interest is independent of the grid resolution, and slower than fast MHD normal modes. We also report serial and parallel three-dimensional results for a 3D setup of the island coalescence problem, where the attracting currents (islands) are also perturbed in the z-direction (leading to nontrivial three-dimensional dynamics).

Unless otherwise specified, we use one multigrid V-cycle to approximate M^{-1} and P_{SF}^{-1} where required. Restriction and prolongation employ second-order splines (of local processor data only). As a smoother, we employ four passes of weighed Jacobi (p. 10 in Ref. 42; p. 118 in Ref. 43), with weight $\omega=0.7$, for both the restriction and the prolongation steps. In MG jargon, such V-cycle is identified as V(4,4), where the two integers indicate restriction and prolongation smoothing steps, respectively. Also, the explicit solver employed in this work as a reference to calculate implicit CPU speedup is a second-order predictor-corrector method, which requires two function evaluations per time step.

TABLE I. Serial time-step convergence study for the tearing mode problem on a 128×128 grid. Results are averaged over ten time steps. The table shows the average number of Newton and FGMRES iterations per time step, the total CPU time, the CPU speedup vs and explicit code ($\mathrm{CPU_{exp}}/\mathrm{CPU}$, with $\mathrm{CPU_{exp}}$ the CPU of the explicit approach), and the implicit time step size relative to the explicit CFL limit ($\Delta t/\Delta t_{\mathrm{CFL}}$).

Δt	Newton/Δt	FGMRES/Δt	CPU (s)	$\mathrm{CPU_{exp}}/\mathrm{CPU}$	$\Delta t/\Delta t_{\mathrm{CFL}}$
0.5	5.0	8.0	526	8.6	333
0.75	5.5	9.5	600	11.4	500
1.0	5.0	11.2	685	13.3	667
1.5	5.6	14.6	875	15.6	1000

A. 2D tearing mode

The tearing mode equilibrium is defined by a uniform density and temperature, no flow, and a force-free magnetic Harris-sheet configuration given by[38]

$$B_{x0}=0, \quad B_{y0}(x)=\tanh[(x-0.5)/\lambda], \quad B_{z0}=\sqrt{1-B_{y0}^2(x)},$$

with $\lambda=0.2$. The domain is rectangular of dimensions with $x\in[0,1]$ and $y\in[0,4]$. The computation is performed with $\eta=10^{-3}$, $\nu=10^{-3}$, $D=0$, and $\gamma=5/3$. The boundary conditions are periodic in y and perfect conductor ($B_x=0$), no stress ($\partial_x v_y=0$), and impenetrable wall ($v_x=0$) boundary conditions in x. Homogeneous Neumann boundary conditions are imposed for both ρ and T. We perturb the equilibrium with a sinusoidal perturbation in ρ, $\delta\rho=10^{-3}\sin(2\pi y/L_y)\cos(2\pi x/L_x)$.

The results of a serial time-step convergence study for a fixed 128×128 grid are given in Table I. In this study, the time step has been increased threefold from $\Delta t=0.5$ to $\Delta t=1.5$. However, as the performance results demonstrate, the CPU time (as well as the total number of FGMRES iterations per time step, FGMRES/Δt) only increases by a factor of 1.6. Stated otherwise, for a fixed final simulation time, the largest time step in the table would result in a CPU savings of $\approx40\%$ versus the smallest one. It follows that using a larger time step is advantageous algorithmically. The table also shows that the CPU gain versus an explicit approach improves with larger time steps, and gains of ≈16 are obtained for the largest implicit time step considered in the study. This is despite the fact that the problem size (128×128) is still fairly small. Larger gains are expected for larger problems.

Table II shows the results of a serial grid convergence study, performed with a fixed implicit time step $\Delta t=1$. The study demonstrates that the performance of the solver not only does not degrade with refinement, but that it actually

TABLE II. Serial grid convergence study for the tearing mode problem with $\Delta t=1$. Results are averaged over ten time steps. In this table, N is the grid size. Other quantities reported are defined as in Table I.

N	Newton/Δt	FGMRES/Δt	CPU	$\text{CPU}_{\text{exp}}/\text{CPU}$	$\Delta t/\Delta t_{\text{CFL}}$
32×32	5.1	14	58	2.43	159
64×64	5.0	11.8	184	5.8	322
128×128	5.0	11.2	685	13.3	667
256×256	5.0	11.4	2835	28.5	1429

improves with it (at least for the grids considered in the table). The reason for this is subtle, and has to do with differences in the spatial truncation errors between the nonlinear function [which solves a discrete form of Eqs. (6)–(9) as described in Ref. 38, without further approximations] and the preconditioner [which solves a discrete form of the parabolized version of Eqs. (6)–(9)]. While both formulations are equivalent in the continuum in the small-flow limit, their discrete spatial representations will feature different spatial truncation error terms.

While computing such truncation error differences is an arduous task for the compressible MHD system, we can gain significant insight from their study in the context of the simple coupled-wave-equation system introduced in Sec. IV. A modified equation analysis of the discrete formulations of the hyperbolic system in Eqs. (13) and (14) and of their parabolized version in Eq. (15) yields that the spatial truncation error of the parabolized equation contains the additional high-order term $\Delta t\Delta x^2/4\,\partial_x^4 u$ (the coefficient is consistent with a first-order backward-Euler implicit treatment of the temporal terms, and second-order finite differences for the spatial ones). While such high-order term will not affect the large scale (small wavenumber) components of the solution, it will significantly do so for the oscillatory (largest wavenumber) ones. The implication for parabolization-based preconditioners is important; such preconditioners are expected to lose effectiveness whenever the nonlinear solver gets to nonlinear residual levels comparable to spatial truncation errors. Figure 1 depicts the residual ℓ_2-norm history of the compressible MHD system versus the FGMRES iteration number for a single linear solve employing different grid refinements. It is clear from this figure that, early in the iteration and for all grid sizes considered, the parabolized preconditioner is very effective, being able to decrease the error by several orders of magnitude in a few iterations. However, as the iteration proceeds and the residual gets smaller, the preconditioner loses effectiveness, and this occurs earlier in the iteration for coarser grids. All these traits are consistent with the truncation error analysis just presented.

From the previous discussion we conclude that the preconditioning approach presented is expected to lose performance when the nonlinear tolerance drives the nonlinear residual at or below spatial truncation error levels. This, however, should not be considered as a serious limitation of the algorithm, since in practice there is little motivation to converge the nonlinear residual to such levels.

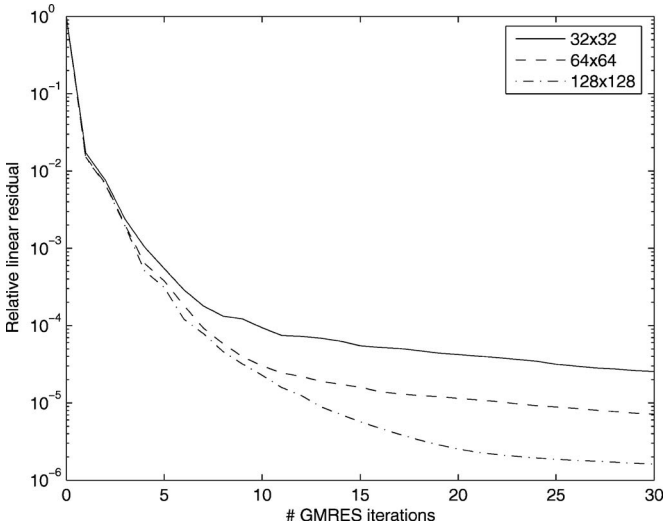

FIG. 1. Residual ℓ_2-norm history of the compressible MHD system vs the FGMRES iteration number for the tearing mode problem using different grid refinements. A fixed implicit time step $\Delta t=1$ has been employed for all traces.

B. 2D island coalescence

The island coalescence equilibrium is a modification of the Harris sheet equilibrium used in the previous section. It features a uniform density, no flow, and magnetic field and pressure given by[45,46]

$$\Psi(x,y) = -\lambda \log\left[\cosh\left(\frac{x}{\lambda}\right) + \epsilon\cos\left(\frac{y}{\lambda}\right)\right],$$

$$p(x,y) = p_0 + \frac{(1-\epsilon^2)}{2}\left[\cosh\left(\frac{x}{\lambda}\right) + \epsilon\cos\left(\frac{y}{\lambda}\right)\right]^{-2},$$

where Ψ is the poloidal flux (from which the magnetic field can be found as $\mathbf{B}=\mathbf{z}\times\nabla\Psi$), and p is the pressure. The domain is square, of dimensions $x,y\in[-1,1]$. The computation is performed with $\eta=10^{-3}$, $\nu=10^{-3}$, $D=0$, $\gamma=5/3$, $p_0=1$, $\lambda=L_y/4\pi$, and $\epsilon=0.2$. The boundary conditions are the same as for the tearing mode problem. We perturb the equilibrium with a solenoidal magnetic field perturbation of the form $\delta\mathbf{B}=\mathbf{z}\times\nabla\delta\Psi$, with $\delta\Psi=10^{-3}\sin(\pi x/L_x)\cos(2\pi y/L_y)$. Figure 2 depicts Ψ and p at $t=0$ (top), and $t=6.5$ (in Alfvén units; bottom). The latter time coincides with the point of maximum reconnection rate, well into the coalescence process.

The island coalescence instability is driven by an ideal process, namely, the attraction of two co-directed current channels. Accordingly, time scales are ideal, and the coalescence process occurs in a time scale comparable to the Alfvén time (consistent with the maximum reconnection rate occurring at $t=6.5$). This ideally driven process is quite different from the previous (tearing mode) problem, which is driven by resistivity and occurring in a hybrid time scale $\sim\sqrt{\tau_\eta\tau_A}\gg\tau_A$, where $\tau_\eta\sim v_A/\eta$ is the resistive time, and $\tau_A\sim L/v_A$ is the Alfvén time.[47] Thus, while the tearing mode leaves ample room for large time steps (and hence to the usefulness of an implicit approach), the island coalescence time scale is limited by ideal (Alfvénic) time scales, and too

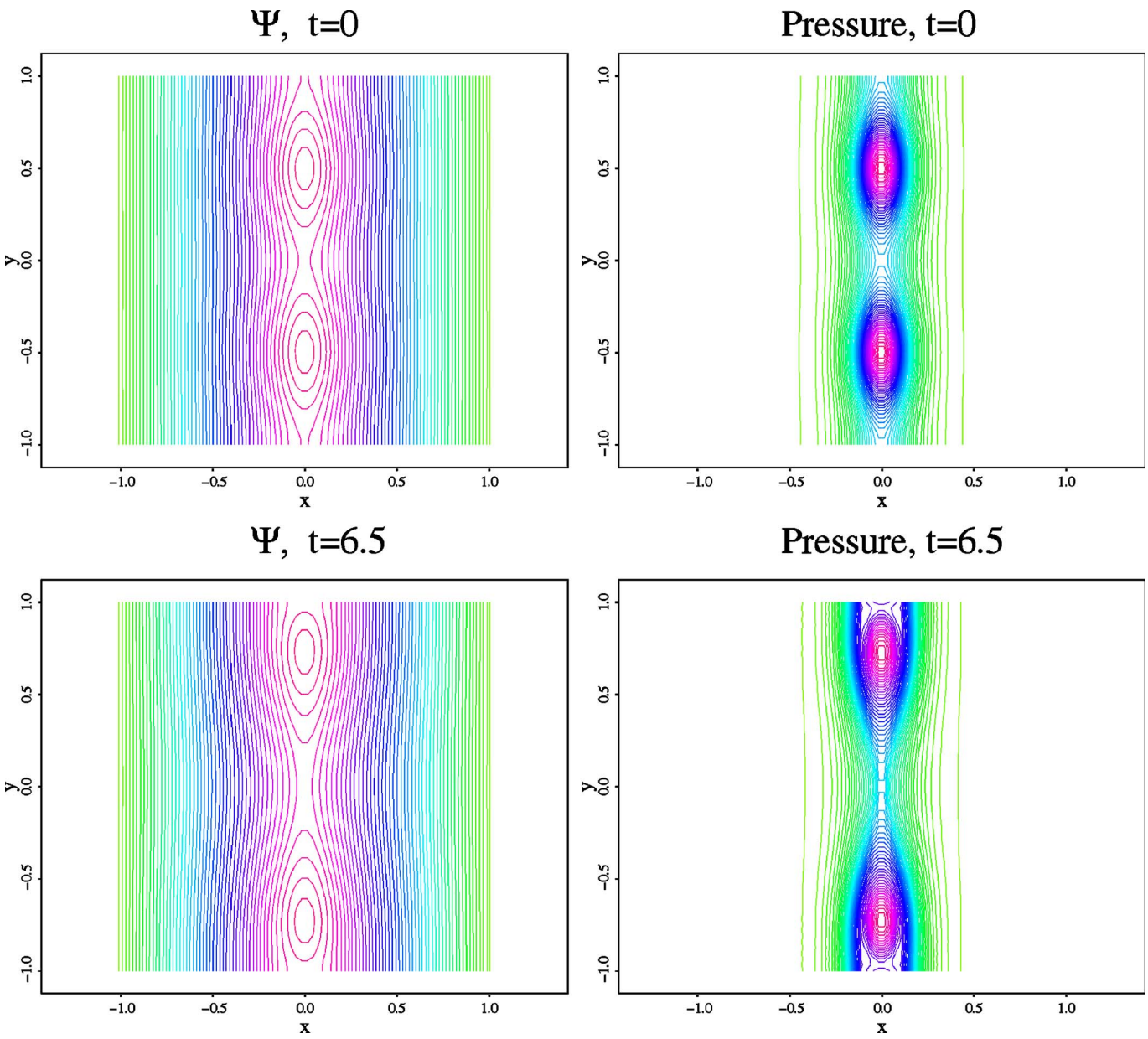

FIG. 2. (Color online) Poloidal flux Ψ and pressure p in the island coalescence problem at equilibrium (top), and at the point of maximum reconnection rate (bottom).

large an implicit time step will easily violate the temporal accuracy condition $\gamma\Delta t < 1/2$,[27] with $\gamma \sim \mathcal{O}(1)$ the ideal instability growth rate.

Yet, because γ is grid-independent, implicit methods do remain useful for such problems if a sufficiently fine mesh is employed. This is demonstrated in Table III, where the results of a grid convergence study with a fixed implicit time step $\Delta t = 0.1$ for different grid refinements are given. For this grid convergence study, we have specified the convergence of the multigrid solves in the preconditioner by either a relative reduction of 10^{-2} or four V(3,3) cycles, whichever comes first (this flexibility is allowed by FGMRES). Performance results are averaged over ten time steps. Several important conclusions can be drawn from this test. First, as in the tearing mode case, the performance of the implicit solver improves with grid refinement. As a result, the CPU scales superoptimally with N [i.e., faster than $\mathcal{O}(N)$]. In contrast, the explicit solver scales as $\mathcal{O}(N^{1.5})$, as expected from a CFL-limited approach. This is direct proof that, as advertised, the implicit solver is indeed algorithmically scalable.

TABLE III. Serial grid convergence study for the island coalescence problem with $\Delta t = 0.1$. Results are averaged over ten time steps. This table reports similar quantities to Table II.

N	Newton/Δt	FGMRES/Δt	CPU	CPU_{exp}/CPU	$\Delta t/\Delta t_{\text{CFL}}$
64×64	4.8	7.4	225	0.25	16
128×128	4.3	5.1	631	0.72	33
256×256	3.9	4.3	2258	1.8	71
512×512	3.9	4.2	10449	3.4	141

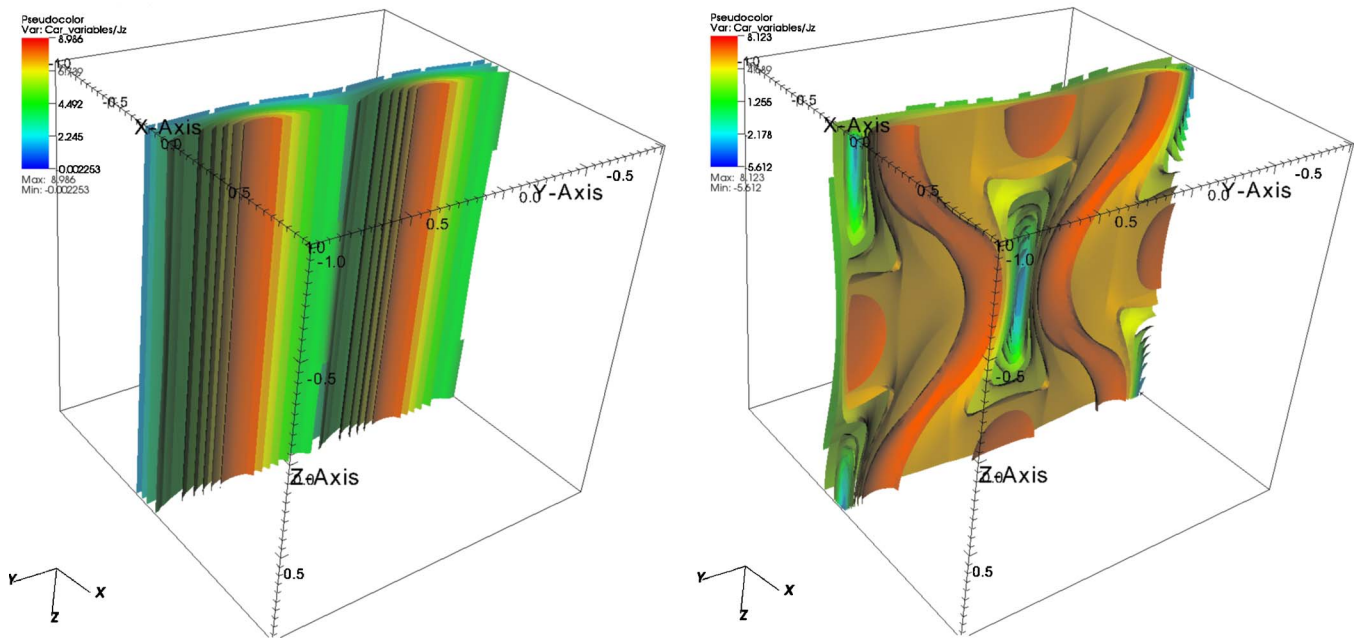

FIG. 3. (Color online) Cut of j_z for 3D island coalescence problem at $t=0$ (left), and at a later stage into the simulation (right), well in the nonlinear regime.

And, secondly, even though implicit speedups are not realized for coarse meshes, they indeed occur for the finer ones (albeit they are moderate for the resolutions considered). The crossover point in grid resolution for the CPU speedup is between 128×128 and 256×256.

C. 3D island coalescence

For a 3D test, we consider a square domain of dimensions $x,y,z\in[-1,1]$, and we extend the previous 2D island coalescence equilibrium to 3D by replicating the two-dimensional equilibrium in every z-slice. However, to provide nontrivial three-dimensional dynamics, we perturb the in-plane magnetic fields according to

$$\delta\Psi = 10^{-3}\sin\left(\frac{\pi x}{L_x}\right)\cos\left(\frac{2\pi y}{L_y}\right)\cos\left(\frac{2\pi z}{L_z}\right).$$

Figure 3 depicts a cut of the current configuration at equilibrium (left) and during the nonlinear stage of the reconnection process (right), which shows its highly kinked state.

Serial performance data of the algorithm for this problem is presented in the grid convergence study in Table IV, which has also been performed with a fixed implicit time step $\Delta t=0.1$. These results confirm previous 2D trends, namely, that the implicit algorithm is scalable under grid refinement, showing no convergence rate degradation as N increases (by a total factor of 64 for the grids considered). The CPU time scaling does degrade some from the theoretical (optimal) factor of 8 per refinement to a factor of 10, but this is consistent with previous studies,[27] and has to do mostly with memory cache effects as the problem size grows.

TABLE IV. Serial grid convergence study for the 3D island coalescence problem with $\Delta t=0.1$. Results are averaged over ten time steps. This table reports similar quantities to Table I.

Grid	Newton/Δt	FGMRES/Δt	CPU
16^3	4.5	5.5	81
32^3	4.9	7.9	1176
64^3	4.7	7.0	11135

Parallel performance results of the algorithm are given in Fig. 4. The top figure shows traces of the wall-clock CPU and the CPU per FGMRES iteration. The bottom figure shows the averaged number of FGMRES per time step (averaged over 10 fixed implicit time steps $\Delta t=0.1$). These parallel performance data have been obtained in a weak sense, using a base resolution of 16^3 grid points per processor. Such small base resolution is unfavorable from a parallel scaling standpoint, since the surface-to-volume ratio (which is a measure of the percentage of points of the local domain that need to be communicated, and is estimated here as $6\times N_p^{-1/3}$, with N_p the total number of mesh points per processor) is fairly large ($\sim37\%$ for $N_p=16^3$). Nevertheless, the algorithm performs well in parallel up to thousands of processors. From Fig. 4 top, it is clear that the scaling of the wall-clock CPU is optimal up to 1000 processors, showing some degradation after that. However, such degradation is not due to parallel bottlenecks in the algorithm; the CPU per FGMRES iteration does not increase, and communication costs remain small for all processor numbers considered. Rather, it is a loss of algorithmic performance, manifested in the growth of FGMRES iterations past the 1000-processor level (Fig. 4 bottom), which is the root cause of the increase. Such growth is attributed to the lack of a coarse-grid solver, as pointed out in Sec. V, and to the fact that the problem is getting harder as the number of processors increases (N

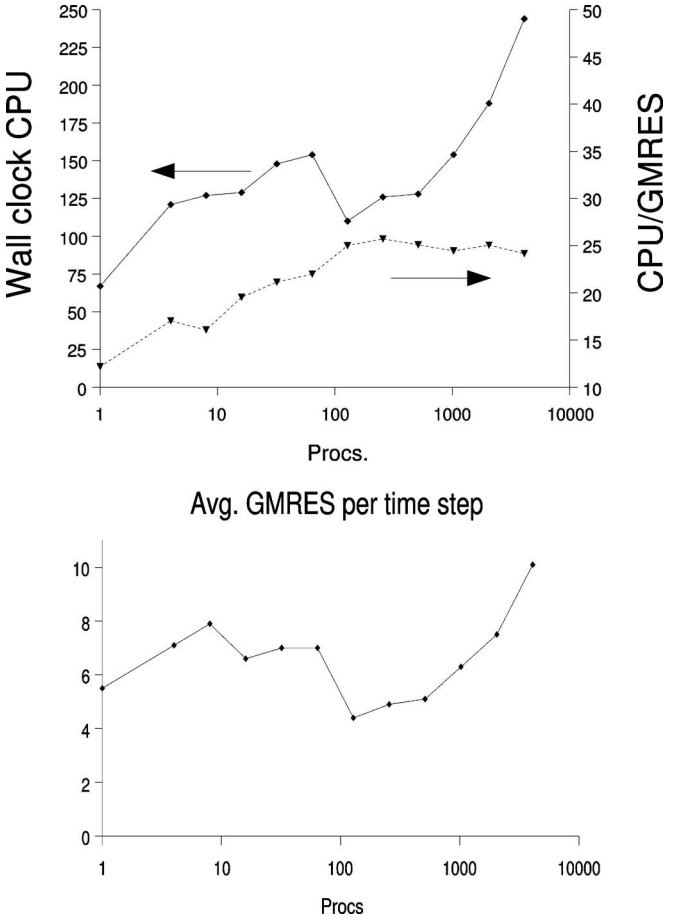

FIG. 4. Parallel performance (in a weak scaling sense) of the implicit 3D MHD algorithm using the 3D island coalescence problem. For this test, we have considered up to 4096 processors, and 134 million unknowns. Data obtained with Franklin at NERSC.

scales linearly with the number of processors n_p, $N \propto n_p$, and the explicit CFL scales as $\Delta t_{\mathrm{CFL}} \sim N^{-1/3}$; therefore, $\Delta t / \Delta t_{\mathrm{CFL}} \sim N^{1/3} \sim n_p^{1/3}$).

VII. CONCLUSIONS

We have developed a parallel, scalable multigrid-based physics-based preconditioning strategy for the 3D compressible viscoresistive MHD equations. The physics-based preconditioning approach is based on two approximations: (1) The small-flow assumption, and (2) the parabolization of the hyperbolic couplings of the MHD model, which renders the MHD system block-diagonally dominant, and thus amenable to a multigrid treatment. A Schur factorization of the Jacobian matrix provides a straightforward, systematic way of performing such parabolization, and in the compressible MHD case it leads to a coupled system of equations for the fluid velocity components. To test the approach, we have considered three relevant tests: A 2D tearing mode, a 2D island coalescence problem, and a generalization of the latter to 3D dimensions (albeit with nontrivial dynamics). These tests demonstrate the excellent scalability properties of the algorithm in terms of grid refinement, implicit time step size, and number of processors (up to 4096 and 134 millions of unknowns have been considered here).

Future work will focus on extending the current algorithm in two directions: (1) The adaptation of the multigrid solver to work in singular coordinate systems (with application to cylindrical and toroidal geometries, of interest for fusion plasma simulation), and (2) the extension of the algorithm to include Hall electron physics (extended MHD).

ACKNOWLEDGMENTS

We acknowledge useful discussions with D. A. Knoll, J. N. Shadid, R. Pawlowski, and N. Krasheninnikova. We thank the PETSc team for technical support.

This work was funded by the Department of Energy at Los Alamos National Laboratory under Contract No. DE-AC52-06NA25396 and the DOE Office of ASCR program in Applied Mathematical Sciences. This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

[1] I. Lindemuth and J. Killeen, J. Comput. Phys. **13**, 181 (1973).
[2] D. Schnack and J. Killeen, J. Comput. Phys. **35**, 110 (1980).
[3] D. S. Harned and W. Kerner, J. Comput. Phys. **60**, 62 (1985).
[4] D. S. Harned and D. D. Schnack, J. Comput. Phys. **65**, 57 (1986).
[5] D. D. Schnack, D. C. Barnes, D. S. Harned, and E. J. Caramana, J. Comput. Phys. **70**, 330 (1987).
[6] D. S. Harned and Z. Mikic, J. Comput. Phys. **83**, 1 (1989).
[7] D. D. Schnack, I. Lottati, Z. Mikic, and P. Satyanarayana, J. Comput. Phys. **140**, 71 (1998).
[8] C. R. Sovinec, A. H. Glasser, T. A. Gianakon, D. C. Barnes, R. A. Nebel, S. E. Kruger, D. D. Schnack, S. J. Plimpton, A. Tarditi, and M. S. Chu, J. Comput. Phys. **195**, 355 (2004).
[9] N. F. Loureiro and G. W. Hammett, "An iterative semi-implicit scheme with robust damping," J. Comput. Phys (submitted).
[10] G. Tóth, R. Keppens, and M. A. Botchev, Astron. Astrophys. **332**, 1159 (1998).
[11] R. Keppens, G. Tóth, M. A. Botchev, and A. V. D. Ploeg, Int. J. Numer. Methods Fluids **30**, 335 (1999).
[12] A. Y. Aydemir and D. C. Barnes, J. Comput. Phys. **59**, 108 (1985).
[13] W. Park, J. Breslau, J. Chen, G. Y. Fu, S. C. Jardin, S. Klasky, J. Menard, A. Pletzer, B. C. Stratton, D. Stutman, H. R. Strauss, and L. E. Sugiyama, Nucl. Fusion **43**, 483 (2003).
[14] S. C. Jardin and J. A. Breslau, Phys. Plasmas **12**, 056101 (2005).
[15] J. C. Strikwerda, *Finite Difference Schemes and Partial Differential Equations* (Chapman and Hall, New York, 1989).
[16] A. H. Glasser, C. R. Sovinec, R. A. Nebel, T. A. Gianakon, S. J. Plimpton, M. S. Chu, and D. D. Schnack, Plasma Phys. Controlled Fusion **41**, A747 (1999).
[17] L. Chacón and D. A. Knoll, J. Comput. Phys. **188**, 573 (2003).
[18] O. S. Jones, U. Shumlak, and D. S. Eberhardt, J. Comput. Phys. **130**, 231 (1997).
[19] J. Schmalzl and U. Hansen, Phys. Earth Planet. Inter. **120**, 339 (2000).
[20] S. V. Patankar, *Numerical Heat Transfer and Fluid Flow* (Taylor and Francis, New York, 1980).
[21] A. M. Popov, V. S. Chan, M. S. Chu, Y. Q. Liu, B. W. Rice, and A. D. Turnbull, Phys. Plasmas **8**, 3605 (2001).
[22] Y. Saad, *Iterative Methods for Sparse Linear Systems* (PWS, Boston, 1996).
[23] A. Hujeirat, Mon. Not. R. Astron. Soc. **298**, 310 (1998).
[24] A. Hujeirat and R. Rannacher, New Astron. Rev. **45**, 425 (2001).
[25] D. R. Reynolds, R. Samtaney, and C. S. Woodward, J. Comput. Phys. **219**, 144 (2006).
[26] S. Ovtchinnikov, F. Dobrian, X.-C. Cai, and D. Keyes, J. Comput. Phys. **225**, 1919 (2007).
[27] L. Chacón, D. A. Knoll, and J. M. Finn, J. Comput. Phys. **178**, 15 (2002).
[28] A. Hujeirat and R. Rannacher, Int. J. Numer. Methods Fluids **28**, 1 (1998).

[29]D. R. Reynolds, R. Samtaney, and C. S. Woodward, "Operator-based preconditioning of stiff hyperbolic MHD systems," SIAM J. Sci. Comput. (submitted).
[30]T. F. Chan and K. R. Jackson, SIAM (Soc. Ind. Appl. Math.) J. Sci. Stat. Comput. **5**, 533 (1984).
[31]P. N. Brown and Y. Saad, SIAM (Soc. Ind. Appl. Math.) J. Sci. Stat. Comput. **11**, 450 (1990).
[32]Y. Saad and M. Schultz, SIAM (Soc. Ind. Appl. Math.) J. Sci. Stat. Comput. **7**, 856 (1986).
[33]Y. Saad, SIAM J. Sci. Comput. (USA) **14**, 461 (1993).
[34]S. Balay, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang, Technical Report No. ANL-95/11, Revision 2.1.5, Argonne National Laboratory (2004).
[35]S. Balay, V. Eijkhout, W. D. Gropp, L. C. McInnes, and B. F. Smith, in *Modern Software Tools in Scientific Computing*, edited by E. Arge, A. M. Bruaset, and H. P. Langtangen (Birkhäuser, Berlin, 1997), pp. 163–202.
[36]S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang, *PETSc Web page*, http://www.mcs.anl.gov/petsc (2001).
[37]T. H. Stix, *Waves in Plasmas* (Springer-Verlag, Berlin, 1992).
[38]L. Chacón, Comput. Phys. Commun. **163**, 143 (2004).
[39]C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations* (SIAM, Philadelphia, 1995).
[40]R. Dembo, S. Eisenstat, and R. Steihaug, SIAM (Soc. Ind. Appl. Math.) J. Numer. Anal. **19**, 400 (1982).
[41]D. A. Knoll and D. E. Keyes, J. Comput. Phys. **193**, 357 (2004).
[42]W. L. Briggs, *A Multigrid Tutorial* (SIAM, Philadelphia, 1987).
[43]P. Wesseling, *An Introduction to Multigrid Methods* (Wiley, Chichester, 1992).
[44]E. J. Caramana, J. Comput. Phys. **96**, 484 (1991).
[45]V. M. Fadeev, I. F. Kvartskhava, and N. N. Komarov, Nucl. Fusion **5**, 202 (1965).
[46]J. M. Finn and P. K. Kaw, Phys. Fluids **20**, 72 (1977).
[47]H. P. Furth, J. Killeen, and M. N. Rosenbluth, Phys. Fluids **6**, 459 (1963).