Confidential: not for distribution. Submitted to IOP Publishing for peer review 19 October 2012

Application of PDSLin to the magnetic reconnection problem

Xuefei Yuan[1], Xiaoye S. Li[1], Ichitaro Yamazaki[2], Stephen C. Jardin[3,4], Alice E. Koniges[1] and David E. Keyes[5,6]

[1] Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA
[2] Innovative Computing Laboratory, University of Tennessee, Knoxville, TN 37996, USA
[3] Theory and Computation Department, Princeton Plasma Physics Laboratory, Princeton, NJ 08540, USA
[4] Department of Astrophysical Sciences, Princeton University, Princeton, NJ 08544, USA
[5] Division of Computer, Electrical and Mathematical Sciences and Engineering, King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia
[6] Department of Applied Physics and Applied Mathematics, Columbia University, New York, NY 10027, USA

E-mail: xyuan@lbl.gov

Abstract. Magnetic reconnection is a fundamental process in a magnetized plasma at both low and high magnetic Lundquist numbers (the ratio of the resistive diffusion time to the Alfvén wave transit time), which occurs in a wide variety of laboratory and space plasmas, e.g., magnetic fusion experiments, the solar corona and the Earth's magnetotail. An implicit time advance for the two-fluid magnetic reconnection problem is known to be difficult because of the large condition number of the associated matrix. This is especially troublesome when the collisionless ion skin depth is large so that the Whistler waves, which cause the fast reconnection, dominate the physics [1].

For small system sizes, a direct solver such as SuperLU can be employed to obtain an accurate solution as long as the condition number is bounded by the reciprocal of the floating-point machine precision. However, SuperLU scales effectively only to hundreds of processors or less. For larger system sizes, it has been shown that physics-based [2] or other preconditioners can be applied to provide adequate solver performance.

Recently, we have been developing a new algebraic hybrid linear solver, PDSLin (Parallel Domain decomposition Schur complement based Linear solver) [3, 4]. In this work, we compare numerical results from a direct solver and the proposed hybrid solver for the magnetic reconnection problem and demonstrate that the new hybrid solver is scalable to thousands of processors while maintaining the same robustness as a direct solver.

PACS numbers: 52.35.Vd

1. Introduction

Most of the visible universe is in the state of plasma, and plasma phenomena are of major importance in space, solar and ionospheric physics. A *plasma* is an ionized gas, which consists of positively charged ions and negatively charged electrons. In a plasma, the microscopic processes are dominated by collective charged particles interactions: charge separation between ions and electrons gives rise to electric fields, charged particles motions result in currents and, consequently, magnetic fields. Electric and magnetic fields configurations can be quite intricate and provide foundations for a wide range of phenomena of overwhelming complexity. The mathematical description of plasma appropriate for describing global dynamics is MagnetoHydroDynamics (MHD) [5].

The simplest form of MHD is ideal MHD, where fluid has so little resistivity that it can be treated as a perfect conductor [6, 7]. The topology of magnetic fields is fixed due to this small resistivity and energy can be stored in moving fluids. The release of energy can happen when the condition of ideal MHD breaks down, thus resistive MHD is considered [8, 9, 10]. Resistive MHD describes magnetized fluids with non-zero electrical resistivity that leads to a breaking in magnetic topology, and the presence of the Hall parameter introduces Whistler waves into the equations [5, 11, 12].

Among the multitude of plasma phenomena, magnetic reconnection problem deserves special attention [13]. *Magnetic reconnection* is a fundamental process in a magnetized plasma: in the reconnection process, two magnetic flux tubes come close together at some point, and they are broken and reconnected in a different way due to the effect of finite resistivity and other non-ideal effects, where the overall topology of the magnetic field is changing and the magnetic field energy is converting into particle heat and bulk kinetic energy over a relatively short period of time [11]. Such phenomena occur in a wide range of laboratory and space plasmas, e.g., magnetic fusion experiments, the solar corona, and the Earth's magnetotail [11, 14, 15].

To fully capture the change of magnetic field topology in magnetic reconnection, we focus on a four-field Hall MHD model valid in the low guide-field limit, where Whistler waves are the dominant two-fluid (ion and electron) effect in the current work. This set of extended MHD equations with hyper-resistivity terms is derived from a set of two-dimensional basic MHD equations describing incompressible, two-fluid, quasi-neutral plasma [16, 17, 18]. These equations are a subset of the full, compressible two-fluid MHD equations which have been studied in two [19] and three [20, 21] dimensions.

However, simulation of magnetically confined, reconnecting plasma presents numerical challenges [18, 22, 19, 23]. This is a result of many factors, including the complexity of models that accurately represent burning plasmas, as well as the resolution of the large range of spatial-temporal scales at which significant physical processes occur [24]. Even in the simpler ideal MHD model, a symmetric hyperbolic system that is a subset of the two-fluid or extended MHD equations, there are three distinct wave types with a wide separation of propagation speeds and with complex polarizations when applied to magnetized plasma conditions typical of fusion plasmas. When discretized

on a finite difference or finite element mesh, these alone lead to a range of timescales and accuracy requirements that are a challenge to address with a single simulation [25].

A key result of temporal stiffness is that traditional explicit methods used for solution to such models may require prohibitively small time step restrictions compared to the dynamical scales of macroscopic stability and plasma fueling. The Courant-Friedrichs-Lewy (CFL) condition [26, 27] imposes the time step size limits for numerical stability that grow quadratically with the mesh increment sizes. Implicit schemes alleviate those issues related to the time step size and the mesh increment sizes.

We present implicit numerical methods in a Backwards Difference Formula (BDF)-Newton solution framework. The Partial Differential Equations (PDE) system becomes a set of nonlinear finite difference equations, $\mathbf{F}(\mathbf{u}) = \mathbf{0}$, after discretization. For such a set of nonlinear algebraic equations, inexact Newton methods [28, 29, 30] are applied on each implicit time step to iterate to a solution through a sequence of linear problems (Newton update equations) from an initial guess of the solution from the previous time step, and some linear solvers are used for the Newton equations. Moreover, such implicit time advance for the two-fluid magnetic reconnection problem is known to be difficult because of the large condition number of the associated matrix. This is especially troublesome when the collisionless ion skin depth is large so that the Whistler waves, which cause the fast reconnection, dominate the physics [1].

Large-scale sparse linear systems of equations, such as the Newton update equations system within numerical simulations of magnetic reconnection, become more interesting in large-scale numerical simulations because their complexity grows superlinearly with traditional direct techniques. If such a system is solved via a direct solver by factorization, the memory requirement is extremely large; if such a system is solved via an iterative solver, the preconditioning techniques are highly required, in which parameters tuning are quite complex to provide a quick convergence [1].

Recently, a new library called PDSLin (Parallel Domain decomposition Schur complement based Linear solver) [3, 4] is introduced as a hybrid direct/iterative solver based on Schur complement methods. The Krylov method (Generalized Minimal RESidual (GMRES) method) with the exact LU factors of the approximation of the Schur complement as the preconditioner in PETSc [31] is used to solve the Schur complement system. Finally, the interior systems are solved in parallel, using the already-computed LU factors of the subdomains. PDSLin is implemented in C with a Fortran interface, and uses MPI for message passing on distributed memory machines.

In Section 2, we state the four-field extended MHD equations including hyper-resistivity terms. Section 3 describes the inexact Newton methods. In Section 4, the hybrid linear solver library PDSLin is introduced. The numerical experiments, comparisons for the direct solver and the hybrid solver are discussed in Section 5. Section 6 concludes the current work.

2. The mathematical model for the magnetic reconnection problem

The reduced two-fluid MHD equations in two-dimensions in the limit of zero electron mass can be written [18, 32] as

$$\left\{\begin{array}{rcrcl} \frac{\partial}{\partial t}\nabla^2\phi & + & \mathbf{V}\cdot\nabla(\nabla^2\phi) & = & [\nabla^2\psi,\psi]+\mu\nabla^4\phi, \\ \frac{\partial V}{\partial t} & + & \mathbf{V}\cdot\nabla V & = & [B,\psi]+\mu\nabla^2 V-\mu h\nabla^4 V, \\ \frac{\partial \psi}{\partial t} & + & \mathbf{V}\cdot\nabla\psi & = & d_i[\psi,B]+\eta\nabla^2\psi-\nu\nabla^4\psi, \\ \frac{\partial B}{\partial t} & + & \mathbf{V}\cdot\nabla B & = & [V,\psi]+d_i[\nabla^2\psi,\psi]+\eta\nabla^2 B-\nu\nabla^4 B. \end{array}\right. \quad (1)$$

Here, ϕ and ψ are stream functions for the in-plane components of the ion velocity and magnetic field, respectively, and V and B are z components of the ion velocity and magnetic field, respectively. Hence, the ion velocity and the magnetic field are expressed as $\mathbf{V}=\nabla\phi\times\hat{z}+V\hat{z}$ and $\mathbf{B}=\nabla\psi\times\hat{z}+B\hat{z}$, η is the electrical resistivity, d_i is the collisionless ion skin depth, μ is the fluid viscosity, ν is the hyper-resistivity (or electron viscosity), and h is the hyperviscosity coefficient added to damp spurious oscillations that might otherwise develop. It must be verified that the physical results converge to a unique value independence of those coefficients over some range [18]. The Poisson bracket $[f,g]\equiv\nabla f\times\nabla g\cdot\hat{z}=\frac{\partial f}{\partial x}\frac{\partial g}{\partial y}-\frac{\partial f}{\partial y}\frac{\partial g}{\partial x}$, $\mathbf{V}\cdot\nabla f=-[\phi,f]$ ($f=\nabla^2\phi$, V, ψ, B), and the out-of-plane current density j is the negative Laplacian of the magnetic flux $j=-\nabla^2\psi$.

It has been shown that Eqs. (1) are valid in the low guide-field limit in which Whistler waves are the dominant two-fluid effect [17], and that a very similar set of equations is valid in the high guide-field limit in which the kinetic Alfvén wave is prominent [16]. Thus, we take Eqs. (1) to be typical of the extended MHD equations in two dimensions. The hyper-viscosity term is present just to damp grid scale oscillations. However, the hyper-resistivity term is necessary for the equations to be mathematically well behaved in the neighborhood of the reconnection layer. It has been shown that a unique converged result will be obtained if the hyper-resistivity decreases as the square of the typical zone size h^2 as h approaches 0 [19, 33].

The finite difference approximation to Eqs. (1) is obtained by applying standard second order space-centered finite difference operators in spatial discretization and variable order BDF methods in temporal discretization [1]. Our physical domain is $\Omega=[-\frac{L_x}{2},\frac{L_x}{2}]\times[-\frac{L_y}{2},\frac{L_y}{2}]$, $L_x=25.6$, $L_y=12.8$ [18] with periodic boundary conditions in the x-direction and Dirichlet boundary conditions at $y=\pm\frac{L_y}{2}$. However, we take the first quadrant $\tilde{\Omega}=[0,\frac{L_x}{2}]\times[0,\frac{L_y}{2}]$ as the computational domain and solutions in Ω are obtained by mirroring solutions from this first quadrant. This is because ϕ, B in Eqs. (1) are anti-symmetric along the x-axis and y-axis and V,ψ are symmetric along the x-axis and y-axis. Therefore, boundary conditions of Eqs. (1) in $\tilde{\Omega}$ become (i) Dirichlet at $y=\frac{L_y}{2}$ and (ii) anti-symmetric for ϕ and B, symmetric for ψ and V at $y=0$, $x=0$, and $x=\frac{L_x}{2}$.

We define a Harris equilibrium and perturbation similar to the one used in the Geospace Environmental Modeling (GEM) magnetic reconnection challenge [34], and

take it as the initial condition for ψ. The other three fields (ϕ, V, B) are initialized to zero:

$$\psi(x, y, 0) = \frac{1}{2} \ln \cosh 2y + \epsilon \cos k_x x \cos k_y y, \quad k_x = \frac{2\pi}{L_x}, \quad k_y = \frac{\pi}{L_y}, \quad \epsilon = 0.1. \quad (2)$$

The GEM initial conditions also included a perturbation in the fluid density, which we take to be constant in this four-field model.

3. Nonlinear solver: inexact Newton methods

The nonlinear partial differential equations (1) become a set of nonlinear algebraic equations through finite difference approximation: $\mathbf{F}(\mathbf{u}) = \mathbf{0}$ with $\mathbf{F} = (F_\phi, F_V, F_\psi, F_B)$ and $\mathbf{u} = (\phi, V, \psi, B)$ after spatial and temporal discretizations. When we advance the system in time with the notation $(\phi^k, V^k, \psi^k, B^k)$ and $(\phi^{k-1}, V^{k-1}, \psi^{k-1}, B^{k-1})$ as the solutions for (ϕ, V, ψ, B) obtained at time level k, $k-1$, respectively, a high-order BDF method requires sufficient solution history to be accumulated at the beginning of the time integration process. In our approach, the time integration process begins, starting from an initial guess at $t = 0$ with a BDF method of order one (backward Euler), gradually increasing the order up to a desired value as more and more solution history becomes available ‡.

For the residual evaluation system $\mathbf{F}(\mathbf{u}) = \mathbf{0}$, we can do multivariable Taylor expansion about a current point $\mathbf{u}^m$:

$$\mathbf{F}(\mathbf{u}^{m+1}) = \mathbf{F}(\mathbf{u}^m) + \mathbf{F}'(\mathbf{u}^{m+1} - \mathbf{u}^m) + \text{h.o.t.},$$

where h.o.t. means higher order terms. Let the right-hand side be zero and neglect h.o.t. to derive a strict Newton iteration over a sequence of linear systems:

$$\begin{cases} \mathbf{J}(\mathbf{u}^m)\delta\mathbf{u}^m &= -\mathbf{F}(\mathbf{u}^m) \\ \mathbf{u}^{m+1} &= \mathbf{u}^m + \delta\mathbf{u}^m, \quad m = 0, 1, \cdots, \end{cases} \quad (3)$$

for a given $\mathbf{u}^0$. Here, $\mathbf{F}(\mathbf{u})$ is a vector-valued function of nonlinear residuals, $\mathbf{J} = \frac{\partial \mathbf{F}}{\partial \mathbf{u}}$ is its Jacobian matrix, $\mathbf{u}$ is the vector to be found, and m is the nonlinear iteration index.

We posit that the vector-valued function $\mathbf{F}(\mathbf{u}^m)$ has following properties: there exists an $\mathbf{u}^*$ with $\mathbf{F}(\mathbf{u}^*) = \mathbf{0}$; $\mathbf{F}$ is continuously differentiable in a neighborhood of $\mathbf{u}^*$; and the Jacobian matrix $\mathbf{J}(\mathbf{u}^*)$ is nonsingular.

The Newton iteration stops based on a required drop of the norm of the nonlinear residual

$$\|\mathbf{F}(\mathbf{u}^m)\| < \epsilon_r \|\mathbf{F}(\mathbf{u}^0)\|,$$

and/or a sufficiently small Newton update

$$\|\delta\mathbf{u}^m\| < \epsilon_s.$$

Newton methods are attractive because they converge rapidly from any sufficiently good initial guess $\mathbf{u}^0$. In transient problems, a good initial guess at each stage is the solution

‡ The highest order available in the hand-coded program is 4th order, and numerical experiments are carried to 2nd order in time.

at the previous stage. However, there is a drawback of Newton's method when solving the Newton correction (Newton update equation) at each stage: it is very expensive to compute the exact solution using a direct method such as Gaussian elimination if the number of unknowns is large and may not be justified when $\mathbf{u}^m$ is far from $\mathbf{u}^*$. Therefore, it is reasonable to use iterative methods to solve the Newton equation only approximately.

There is a class of inexact Newton methods that computes an approximate solution of Newton equations in a manner such that

$$\frac{\|\mathbf{r}^m\|}{\|\mathbf{F}(\mathbf{u}^m)\|} \leq \eta^m,$$

where the residual $\mathbf{r}^m$ is given by

$$\mathbf{r}^m \equiv \mathbf{J}(\mathbf{u}^m)\delta\mathbf{u}^m + \mathbf{F}(\mathbf{u}^m),$$

and the nonnegative forcing sequence $\{\eta^m\}$ is used to control the level of accuracy [30].

Then the inexact Newton method is

$$\left\{ \begin{array}{rcll} \mathbf{J}(\mathbf{u}^m)\delta\mathbf{u}^m & = & -\mathbf{F}(\mathbf{u}^m) + \mathbf{r}^m, & \text{where} \frac{\|\mathbf{r}^m\|}{\|\mathbf{F}(\mathbf{u}^m)\|} \leq \eta^m \\ \mathbf{u}^{m+1} & = & \mathbf{u}^m + \delta\mathbf{u}^m, & m = 0, 1, \cdots. \end{array} \right. \tag{4}$$

Here η^m may depend on $\mathbf{u}^m$. When $\eta^m = 0$, we recover Newton method.

4. The parallel domain decomposition schur complement based linear solver: PDSLin

At each Newton (nonlinear) iteration, we need to solve linear system (3) for Newton methods or (4) for inexact Newton methods. In this section, we introduce the parallel domain decomposition schur complement based linear solver PDSLin.

The hybrid linear software library PDSLin is designed to solve a large-scale linear system:

$$Ax = b, \tag{5}$$

where A is a square real or complex general matrix, b is a given right-hand-side vector, and x is the solution vector. It uses a non-overlapping domain decomposition technique called the Schur complement method [35]. The original linear system is first reordered into a 2×2 block system of the following form:

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}, \tag{6}$$

where A_{11} are *interior subdomains*, A_{22} are *separators*, and A_{12} and A_{21} are the *interfaces* between A_{11} and A_{22}. To eliminate the unknowns associated with A_{11}, an equivalent system

$$\begin{pmatrix} A_{11} & A_{12} \\ 0 & S \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ \hat{b}_2 \end{pmatrix}, \tag{7}$$

is obtained. In this system, the Schur complement S is defined as

$$S = A_{22} - A_{21}A_{11}^{-1}A_{12}, \tag{8}$$

and the right hand side

$$\hat{b}_2 = b_2 - A_{21}A_{11}^{-1}b_1. \tag{9}$$

The solution of the global system can be achieved by solving the Schur complement system

$$Sx_2 = \hat{b}_2 \tag{10}$$

first and then solving the interior system

$$A_{11}x_1 = b_1 - A_{12}x_2. \tag{11}$$

If k interior subdomains are extracted, the coefficient matrix of (6) has the form of

$$\left(\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right) = \left(\begin{array}{cccc|c} D_1 & & & & E_1 \\ & D_2 & & & E_2 \\ & & \ddots & & \vdots \\ & & & D_k & E_k \\ \hline F_1 & F_2 & \cdots & F_k & A_{22} \end{array}\right), \tag{12}$$

where D_l is the l-th subdomain, and E_l and F_l are the interfaces between D_l and A_{22}. There are two processor groups g_l and g_s: processors in g_l factorize the subdomain D_l and rows of D_l, E_l and columns of F_l are distributed among these processors §; processors in g_s solve the Schur Complement system (10).

In parallel, the Schur complement S in (8) is computed as

$$S = A_{22} - \sum_{l=1}^{k} F_l D_l^{-1} E_l = A_{22} - \sum_{l=1}^{k} (U_l^{-T} F_l^T)^T (L_l^{-1} E_l) = A_{22} - \sum_{p=1}^{n_p} W^{(p)} G^{(p)} \tag{13}$$

for an LU factorization $D_l = L_l U_l$. Here, n_p is the number of cores used to solve the entire system, and the matrices $W^{(p)}$ and $G^{(p)}$ are given by ‖

$$W^{(p)} = W(:, j_p : (j_{p+1} - 1)), \quad G^{(p)} = G(j_p : (j_{p+1} - 1), :), \tag{14}$$

such that the p-th processor owns the j_p-th through $(j_{p+1} - 1)$-th columns of $W = (U_{11}^{-T} A_{21}^T)^T$ and rows of $G = L_{11}^{-1} A_{12}$, where $A_{11} = L_{11}U_{11}$ given by L_l and U_l, $l = 1, \cdots, k$.

When computing $W^{(p)}$ and $G^{(p)}$, their approximations $\tilde{W}^{(p)}$ and $\tilde{G}^{(p)}$ are calculated by discarding nonzeros with magnitudes less than a prescribed drop tolerance σ_1, and the approximate update matrix $\tilde{T}^{(p)} = \tilde{W}^{(p)}\tilde{G}^{(p)}$. If the p-th processor belongs to the processor group g_s, to compute its local portion of the approximate Schur complement, it gathers the corresponding rows of $\tilde{T}^{(q)}$ from all processors, and compute

$$\hat{S}^{(p)} = A_{22}^{(p)} - \sum_q \tilde{T}^{(q)}(i_p : (i_{p+1} - 1), :),$$

§ The nonzeros of D_l and E_l are stored in the Compressed Row Storage (CRS) format and the nonzeros of F_l are stored in the Compressed Column Storage (CCS) format.

‖ Fortran notation is used here.

where p-th processor owns the i_p-th through $(i_{p+1} - 1)$-th row of A_{22}. Moreover, small nonzeros are dropped from $\hat{S}^{(p)}$ to form its approximation $\tilde{S}^{(p)}$ with dropping tolerance σ_2. In the process of computing the approximate Schur complement $\tilde{S}$, several performance-enhancing techniques are employed [3].

There are three main phases: extracting and factorizing the interior subdomains A_{11}; computing an approximate Schur complement $\tilde{S}$ of S in (8); and computing the solution. During those phases, challenges are existed for developing such a robust, efficient, and general purpose hybrid solver for thousands of processors with a parallel implementation.

When the parallel nested dissection algorithm implemented in PT-SCOTCH [36] is used to extract interior subdomains, multiple processors are assigned to each interior subdomain to allow us to increase the processor count without increasing neither the number of subdomains nor the size of the Schur complement. This *two-level* parallel approach is different from the general *one-level* approach ¶ and does not need a large number of subdomains to use large numbers of processors; therefore, the size of the Schur complement does not increase. To compute an approximate Schur complement as a preconditioner, we have to deal with the load imbalance and communication both in an *intra-processor group* assigned to the same subdomain and the *inter-processor groups* assigned to different subdomains [3]. When computing the solution, a preconditioned Krylov method in PETSc [31] is used to solve the Schur complement system, and the preconditioner is the exact LU factorization of the approximate Schur complement $\tilde{S}$ via SuperLU [37]. Finally, the interior systems are solved in parallel, using the already-computed LU factors of the subdomains.

The most challenging phase of the parallel implement is to compute an approximate Schur complement $\tilde{S}$, especially for a two-level parallel framework, where multiple processors are assigned to one subdomain. The benefit of using such a two-level approach is to limit the size of the Schur complement when using thousands of processors; however, it is hard to deal with the load imbalance and communication in these two aspects: an intra-processor group assigned to the same subdomain and the inter-processor groups assigned to different subdomains.

There are other two hybrid linear solvers HIPS [38] and MaPHyS [39] that are also based on the Schur complement methods. However, they have different approaches in parallel implementations. The one-level approach is used in HIPS, where multiple subdomains are assigned to one single processor, and PDSLin and MaPHyS use the two-level approach. As a result, HIPS has a larger Schur complement system when the number of cores increases. The Schur complement system is treated as a global system in PDSLin and HIPS, but a local system [+] in MaPHyS. When solving the Schur complement system, HIPS uses the block level-based ILU and MaPHyS uses additive

¶ In a *one-level* parallel approach, a single processor is assigned to factorize one or more interior subdomains.

[+] MaPHyS computes the local Schur complements associated with the subdomains explicitly to construct a set of parallel local preconditioners.

Schwarz as preconditioners; PDSLin uses the LU of the approximation of the Schur complement $\tilde{S}$ as the preconditioner, which provides a robust preconditioning for solving highly-indefinite or ill-conditioned systems.

5. Numerical results

Our numerical experiments are carried on the Cray XE6 *Hopper*, a leading petascale supercomputing system at the National Energy Research Scientific Computing Center (NERSC). *Hopper* is NERSC's first petascale system with a peak performance of 1.28 Petaflops/sec, 153,216 processors cores for running scientific applications, 212 TB of memory, and 2 Petabytes of online disk storage. *Hopper* has 6,384 compute nodes made up of 2 twelve-core AMD 'MagnyCours' 2.1-GHz processors per node, in which there are 6000 nodes have 32GB DDR3 1333-MHz memory per node and 384 nodes have 64GB DDR3 1333-MHz memory per node [40]. All calculations are carried out with 64 bit arithmetic.

There are five physical parameters: the electrical resistivity η, the fluid viscosity μ, the hyper-viscosity $h = C_1 h_x h_y$, the hyper-resistivity $\nu = C_2 h_x h_y \eta$, and the collisionless ion skin depth d_i. The cell size is $h_x \times h_y$. We choose as defaults the values used given in the GEM problem specification: $\eta = 0.005$, $\mu = 0.05$, $C_1 = 4.0$, $C_2 = 1.0$, and $d_i \in [0, 1]$ [18, 34].

For the nonlinear solver: the relative convergence tolerance $\epsilon_r = 10^{-8}$; the convergence tolerance in terms of the norm of the change in the solution between steps $\epsilon_s = 10^{-7}$.

5.1. Condition numbers

Starting from initial states (2), the system (1) evolves in time. When the collisionless ion skin depth number $d_i = 0.0$ (resistive MHD), because $\phi = V = B = 0.0$ at $t = 0$, the second and fourth equations in (1) imply that V and B remain unchanged as time advances, and the out-of-plane current density has large gradient in the mid-plane. As we expect, there is a thin current layer in the mid-plane, known as the Sweet-Parker layer [41, 42] and no x-point shows up. As d_i changes from 0.0 to 1.0, the reconnection region has essentially changed character from a y-point to an x-point as expected [43] (see Figure 1). For the system (1), the reconnected magnetic flux is defined as $\Psi(t) = \frac{1}{2}[\psi(0, 0, t) - \psi(L_x/2, 0, t)]$, and the reconnection rate is the time derivative of $\Psi(t)$: $R(t) = \partial\Psi(t)/\partial t$.

The implicit time advance for this two-fluid magnetic reconnection problem is known to be difficult because of the large condition number of the associated matrix. This is especially troublesome when the collisionless ion skip depth d_i is large so that the Whistler waves, which cause the fast reconnection, dominate the physics [1].

Figure 2(a) and Figure 2(b) show comparisons between $\Psi(t)$ and $R(t)$ for the collisionless ion skin depth $d_i = 0.1$, 0.5, and 1.0 with $dt = 0.1$. When $d_i = 1.0$,

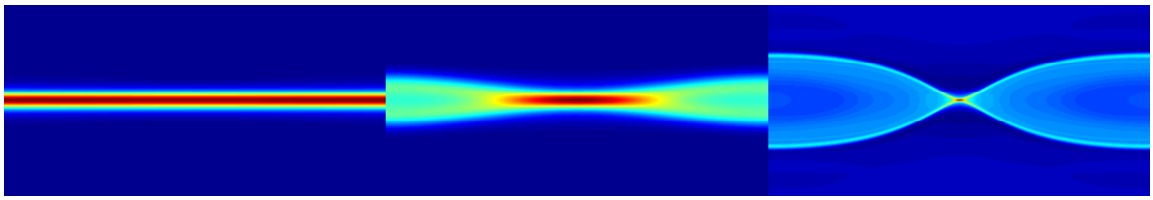
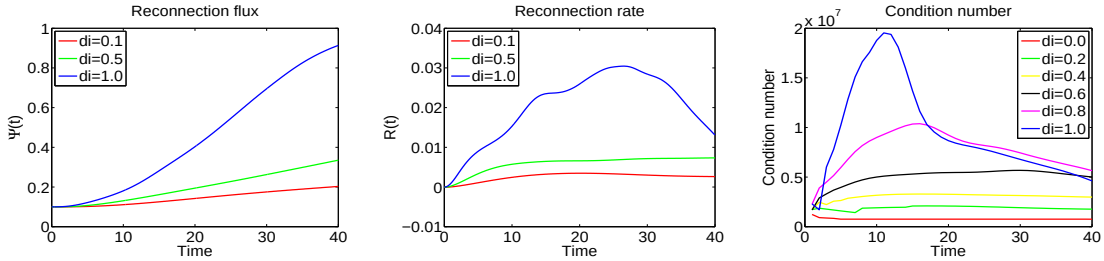


Figure 1. The plots of the negative current density $-j$ in $\Omega = [-\frac{L_x}{2}, \frac{L_x}{2}] \times [-\frac{L_y}{2}, \frac{L_y}{2}]$ at time $t = 0.0$ (left), time $t = 40.0$ with $d_i = 0.0$ (middle) and time $t = 40.0$ with $d_i = 1.0$ (right).



(a) Reconnection flux (b) Reconnection rate (c) Condition number

Figure 2. The reconnected magnetic flux $\Psi(t)$ (a), the reconnection rate $R(t)$ (b) for $d_i = 0.1, 0.5, 1.0$ with $dt = 0.1$, and the condition numbers (c) for $d_i = 0.0, 0.2, 0.4, 0.6, 0.8, 1.0$ with $dt = 1.0$ from time $t = 0.0$ to $T = 40.0$ for $\eta = 0.005$, $\mu = 0.05$, $C_1 = 4.0$, $C_2 = 1.0$ on a 128×128 grid.

the reconnection rate reaches its maximum at $R(t = 26.6) = 0.03$. Figure 2 (c) shows comparisons of condition numbers of different $d_i = 0.0, 0.2, 0.4, 0.6, 0.8$ and 1.0 with $dt = 1.0$. When $d_i = 1.0$, the condition number reaches its maximum $1.95e + 07$ at $t = 11.0$.

Moreover, the condition number increases when the problem size increases or the time step size increases [1]. Table 1 lists condition numbers of associated matrices of the time-dependent nonlinear system for different problem sizes. The problem size $(N_x \times N_y)$, the size of the associated matrix (size(A)), the nonzeros in the matrix (nnz(A)), the nonzeros in the matrix $L + U$ (nnz$(L + Z)$), the fill-ratio, the condition number (cond(A)) and the memory usage (mem) in Gigabyte are listed in the table. When the problem size increases, the fill-ratio increases, therefore the memory usage increases. The maximum memory per node on *Hopper* is 64GB, which is not enough for evaluating the condition number for the associated matrix for the 1024×1024 size problem: if the fill-ratio is 100, the estimate requirement of memory is about 98GB. The sequential direct solver package SuperLU_4.3 [44] is used to get these condition numbers.

Two matrices are chosen for numerical experiments: the first one has a size of $1,048,576$ with $23,970,514$ nonzeros, and we call the related linear system *mcomp*; the

$N_x \times N_y$	size(A)	nnz(A)	nnz($L+U$)	fill-ratio	cond(A)	mem(GB)
64×64	$16,384$	$358,674$	$13,698,180$	38	$6.30e+04$	0.15
128×128	$65,536$	$1,470,802$	$71,554,872$	48	$1.48e+06$	0.88
256×256	$262,144$	$5,956,050$	$401,569,028$	67	$2.41e+07$	5.00
512×512	$1,048,576$	$23,970,514$	$2,133,808,772$	89	$3.93e+08$	28.54

Table 1. The condition numbers of associated matrices of the magnetic reconnection problem for $t = 0.0$ with $dt = 0.5$, $d_i = 1.0$, $\eta = 0.005$, $\mu = 0.05$, $C_1 = 4.0$, $C_2 = 1.0$.

other one has a size of $4,194,304$ with $96,175,314$ nonzeros, and we call the related linear system *bcomp*.

5.2. Hybrid linear solver: PDSLin

In this section, we use PDSLin library as the linear solver for the associated matrices in the magnetic reconnection problem and present some numerical results and parallel performance of this hybrid linear solver. In PDSLin, the SuperLU_DIST 2.4 [37] is used as a direct solver for interior subdomains and the Schur complement systems are solved using a preconditioned Krylov method in PETSc * [31]. The stopping criterion for the Krylov solver is to check the l_2-norm of the initial residual, if it is reduced by at least 10^{-12}, we consider the solution to be converged ♯.

The first experiment is to compare the total time required by the direct solver and the hybrid solver (both the one-level parallel framework and the two-level parallel framework) to solve the *mcomp* linear system. In the one-level approach, the number of the interior subdomains k is set to be the same as the total number of cores, the cores used for solving the Schur complement system is the half of the total number of cores; in the two-level approach, the number of the interior subdomains is fixed at 32, and the processors are distributed equally among all interior subdomains.

To enhance the performance of the hybrid solver, the drop tolerance σ_1 is used to enforce the sparsity of $\tilde{E}$ and $\tilde{F}$, and the drop tolerance σ_2 is used to enforce the sparsity of $\tilde{S}$. Figure 3 shows comparisons of the total time with $\sigma_1 = 10^{-6}$, $\sigma_2 = 10^{-5}$, and the number of cores $n_p = 32, 64, \cdots, 4096$: PDSLin scales better than SuperLU_DIST, and the scaling of the one-level and two-level approaches is similar. For example, SuperLU_DIST does not scale after 128 cores, while PDSLin scales well till 512 cores for the one-level approach and 2048 cores for the two-level approach.

Although similar scaling of the one-level and two-level approaches is observed here, it is not always true. For matrices in other applications [3], the two-level approach has a better scaling than the one-level approach. In the one-level approach, the number of the interior subdomains k increases as the total number of cores increases, thus the

* The version is 3.1.09

♯ When using the Krylov solver to solve the system 10, very accurate solution (x_2) is required as it is used to achieve x_1 in the system 11 for the whole solution of the linear system $Ax = b$.

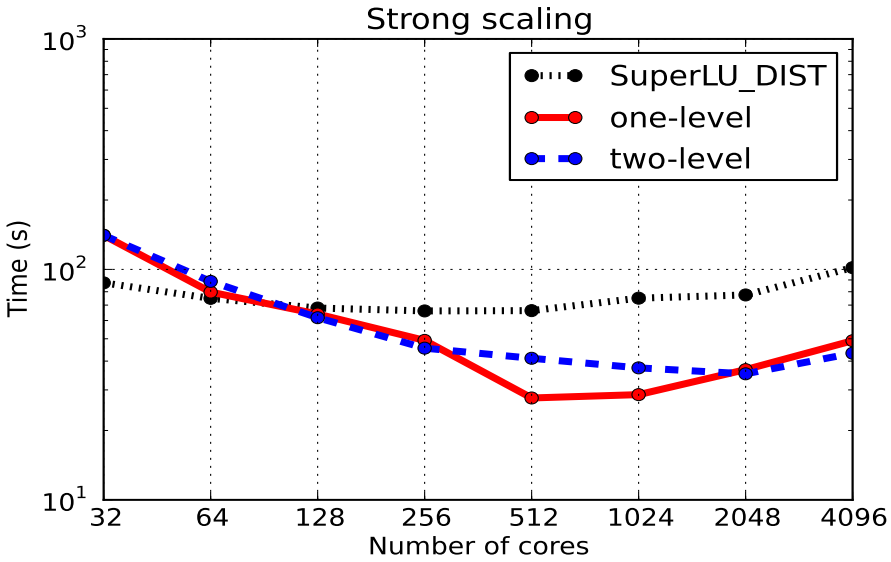


Figure 3. The total time required to solve the linear system *mcomp*: the direct solver SuperLU_DIST_3.0 (black dotted line), the one-level approach (red solid line) and the two-level approach (blue dashed line). The tolerances $\sigma_1 = 10^{-6}$ and $\sigma_2 = 10^{-5}$.

size of the Schur complement increases. In general, the iterative solver has difficulty in convergence in large size Schur complement systems: the iteration number increases as the size of the Schur complement increases. Thus for most matrices, the scaling of the one-level approach is not as good as that of the two-level approach, where the two-level approach has a fixed number of subdomains, and a fixed size of the Schur complement when the number of cores increases.

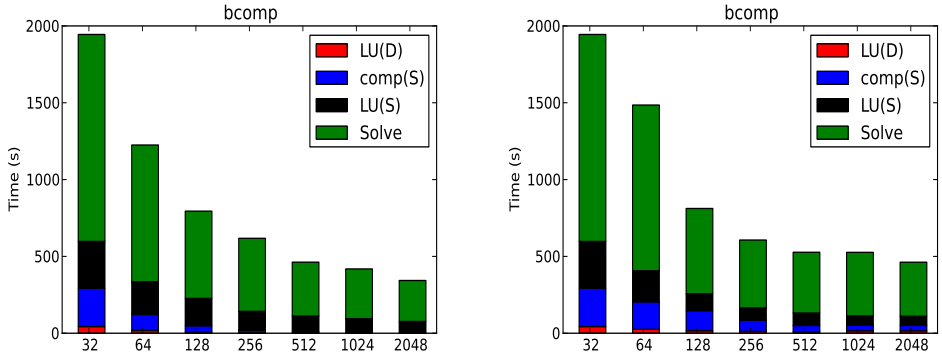
k	size	nnz$(LU(\tilde{S}))$	its	k	size	nnz $(LU(\tilde{S}))$	its
32	38,056	280,975,663	70	512	128,236	575,865,038	62
64	53,610	365,279,060	85	1024	178,925	671,580,408	75
128	80,780	554,788,925	124	2048	251,229	788,066,929	80
256	110,672	637,663,812	120	4096	346,378	860,595,342	93

Table 2. The Schur complement system in the one-level approach: the number of subdomains, the size of the Schur complement, the number of nonzeros and the iteration numbers in iterative solver.

In the two-level approach of solving *mcomp* linear system, the size of the Schur complement is 38056×38056 with average nonzeros $303,189,610$ for a fixed 32 subdomains, and the number of iterations is 70; in the one-level approach of solving *mcomp* linear system, the size of the Schur complement increases. In Table 2, the number of subdomains (k: equivalent to the number of cores n_p), the size of the Schur complement (size), the nonzeros (nnz), and the iteration numbers of the iterative solver for the Schur complement system (its) are listed.

The nonzeros in Schur complement increases slower than the increment of the size of the Schur complement: while the size increases 128 times from 32 to 4096, the size of the Schur complement only increases about 9 times from $38,056$ to $346,378$, and the nonzeros only increases about 3 times from $280,975,663$ to $860,595,342$. Moreover, there is a dip of nonzeros when the number of cores increases from 256 to 512; therefore, the iterations required has decreases from 120 to 62. These explain why the one-level approach and the two-level approach has a similar scaling in the current reconnection model.

Similar scaling is also found in solving *bcomp* linear system in both one-level and two-level approaches. Moreover, we check the times (in seconds) for the LU decomposition of the interior subdomain D_l ($LU(D)$), the computation of the approximate Schur complement $\tilde{S}$ ($comp(S)$), the LU decomposition of $\tilde{S}$ ($LU(S)$), and the computation of the solution vector ($Solve$) for *bcomp*. Figure 4 shows 7 cases where the number of cores $n_p = 32, 128, \cdots, 2048$ for both one-level and two-level approaches for solving *bcomp* linear system.



(a) one-level (b) two-level

Figure 4. The time for LU factorization of the interior domains LU(D), computation of the approximate Schur complement comp(S), LU of the approximate Schur complement LU(S), solution of the system for the one-level (a) and the two-level (b) approaches for *bcomp*.

In general, $LU(S)$ increases with the number of cores because doubling the number of cores doubles the number of subdomains and typically doubles the size of the Schur complement. Hence, $LU(S)$ typically increases when twice many cores are used on a twice larger Schur complement. This is unfortunate because one-level parallelization will not scale on a large number of cores. However, we can use two-level parallelism to scale to a larger number of cores by fixing a small number of subdomains. We can see from Figure 4 (a) (the one-level parallelization for *bcomp*) and Figure 4 (b) (the two-level parallelization for *bcomp*) that all the times are scaling well, and the good scaling of $LU(S)$ allows one-level parallelization to scale to thousands of cores.

6. Conclusions

In this paper, we have presented implicit numerical methods in a BDF-Newton solution framework as the nonlinear solver for the four-field extended MHD equations (magnetic reconnection problem). Two different types of linear solvers are compared for the linear system (Newton update equations): the direct solver and the hybrid solver. Numerical experiments show that in solving the linear system associated with the magnetic reconnection problem, the parallel hybrid solver (both the one-level parallelization and the two-level parallelization) can scale up to thousands of processors. Moreover, the one-level parallelization can scale as well as the two-level parallelization, which is usually difficult to see from other linear systems, for example, the linear systems from the numerical simulation of an accelerator cavity designs [3]. The flexibility of having both the one-level parallelization and the two-level parallelization enables PDSLin to solve linear systems with different properties. When solving the nonlinear system in time, we expect that PDSLin can provide a better scaling than the direct solver while maintaining the same robustness as a direct solver.

Acknowledgments

A majority of the work in this paper was supported by the Petascale Initiative in Computational Science at the National Energy Research Scientific Computing Center (NERSC) and the Center for Simulation of RF Wave Interactions with Magnetohydrodynamics (CSWIM), which is funded by the U.S. Department of Energy, Office of Science. The computational sources were provided by NERSC, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

References

[1] Xuefei Yuan, Stephen C. Jardin, and David E. Keyes. Numerical simulation of four-field extended magnetohydrodynamics in dynamically adaptive curvilinear coordinates via Newton-Krylov-Schwarz. *Journal of Computational Physics*, 231:5822–5853, 2012.

[2] Luis Chacón and Dana A. Knoll. A 2d high-β Hall MHD implicit nonlinear solver. *Journal of Computational Physics*, 188:573–592, 2003.

[3] Ichitaro Yamazaki and Xiaoye S. Li. On techniques to improve robustness and scalability of the schur complement method. In *Proc. of VECPAR 2010*, pages 421–434, 2010.

[4] Ichitaro Yamazaki, Xiaoye S. Li, and Esmond G. Ng. PDSLin Users Guide. Technical report, Lawrence Berkeley National Laboratory, 2011.

[5] Stephen C. Jardin. *Computational Methods in Plasma Physics*. Taylor and Francis, 2010.

[6] Peter A. Sturrock. *Plasma Physics*. Cambridge, 1994.

[7] S. I. Braginskii. *Reviews of Plasma Physics*, volume 1. Consultants Bureau, 1965.

[8] M. Ottaviani and F. Porcelli. Fast nonlinear collisionless magnetic reconnection. *Physics of Plasmas*, 2(11):4104–4117, 1995.

[9] M. Ottaviani and F. Porcelli. Nonlinear collisionless magnetic reconnection. *Physical Review Letters*, 71(23):3802–3805, 1993.

[10] Thomas H. Stix. Plasma transport across a braided magnetic field. *Nuclear Fusion*, 18:353–358, 1978.

[11] Dieter Biskamp. *Magnetic Reconnection in Plasmas*. Cambridge, 2000.

[12] D. Grasso, F. Pegoraro, F. Porcelli, and F. Califano. Hamiltonian magnetic reconnection. *Plasma Physics and Controlled Fusion*, 41(12):1497–1515, 1999.

[13] Dieter Biskamp. *Nonlinear Magnetohydrodynamics*. Cambridge University Press, 1993.

[14] Kazunari Shibata. New observational facts about solar flares from Yohkoh studies-evidence of magnetic reconnection and a unifield model of flares. *Advances in Space Research*, 17:9–18, 1996.

[15] Daniel N. Baker, T. I. Pulkkinen, V. Angelopoulos, W. Baumjohann, and R. L. McPherron. Neutral line model of substorms: Past results and present view. *Journal of Geophysical Research*, 101:12975–13010, 1996.

[16] Richard Fitzpatrick. Scaling of forced magnetic reconnection in the Hall-magnetohydrodynamical Taylor problem with arbitrary guide field. *Physics of Plasmas*, 11(8):3961–3968, 2004.

[17] Richard Fitzpatrick. Scaling of forced magnetic reconnection in the Hall-magnetohydrodynamical Taylor problem. *Physics of Plasmas*, 11(3):937–946, 2004.

[18] Stephen C. Jardin and J. A. Breslau. Implicit solution of the four-field extended-magnetohydrodynamic equations using high-order high-continuity finite elements. *Physics of Plasmas*, 12:056101, 2005.

[19] Stephen C. Jardin, J. Breslau, and N. Ferraro. A high-order implicit finite element method for integrating the two-fluid magnetohydrodynamic equations in two dimensions. *Journal of Computational Physics*, 226:2146–2174, 2007.

[20] N. Ferraro J. Breslau and Stephen C. Jardin. Some properties of the M3D-C(1) form of the three-dimensional mhd equations. *Physics of Plasmas*, 16(092503), 2009.

[21] J. Breslau Stephen C. Jardin, N. Ferraro and J. Chen. Multiple timescale calculations of sawteeth and other global macroscopic dynamics of tokamak plasmas. *Computational Science & Discovery*, 16(014002), 2012.

[22] Luis Chacón, Dana A. Knoll, and J. M. Finn. An implicit, nonlinear reduced resistive MHD solver. *Journal of Computational Physics*, 178:15–36, 2002.

[23] Dieter Biskamp. Magnetic reconnection via current sheets. *Physics of Fluids*, 29:1520–1531, 1986.

[24] Daniel R. Reynolds, Ravi Samtaney, and Carol S. Woodward. A fully implicit numerical method for single-fluid resistive magnetohydrodynamics. *Journal of Computational Physics*, 219:144–162, 2006.

[25] Ralf Gruber and Jacques Rappaz. *Finite element methods in linear ideal MHD*. Springer, 1985.

[26] Richard Courant, Kurt Friedrichs, and Hans Lewy. Uber die parteillen differenzengleichungen der mathematischen physik. *Mathematische annalen*, 100(1):32–74, 1928.

[27] Richard Courant, Kurt Friedrichs, and Hans Lewy. On the partial difference equations of mathematical physics. *IBM Journal*, pages 215–234, 1967.

[28] Xiao-Chuan Cai and David E. Keyes. Nonlinearly preconditioned inexact Newton algorithms. *SIAM Journal on Scientific Computing*, 24(1):183–200, 2002.

[29] Ron S. Dembo, Stanley C. Eisenstat, and Trond Steihaug. Inexact Newton methods. *Journal of Numerical Analysis*, 19:400–408, 1982.

[30] Stanley C. Eisenstat and Homer F. Walker. Choosing the forcing terms in an inexact Newton method. *SIAM Journal on Scientific Computing*, 17(1):16–32, 1996.

[31] Satish Balay, Jed Brown, Kris Buschelman, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc Web page, 2011. http://www.mcs.anl.gov/petsc.

[32] Xuefei Yuan, Stephen C. Jardin, and David E. Keyes. Moving grids for magnetic reconnection via Newton-Krylov methods. *Computer Physics Communications*, 182:173–176, 2011.

[33] Josh A. Breslau and Stephen C. Jardin. Global extended MHD studies of fast magnetic reconnection. *Physics of Plasmas*, 10(5):1291–1298, 2003.

[34] J. Birn, J. F. Drake, M. A. Shay, B. N. Rogers, R. E. Denton, M. Hesse, M. Kuznetsova, Z. W. Ma, A. Bhattacharjee, A. Otto, and P. L. Pritchett. Geospace Environmental Modeling (GEM) magnetic reconnection challenge. *Journal of Geophysical Research*, 106(A3):3715–3719, 2001.

[35] Barry F. Smith, Petter E. Bjorstad, and William D. Gropp. *Domain Decomposition. Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press, 1996.

[36] C. Chevalier and F. Pellegrini. PT-SCOTCH: a tool for efficient parallel graph ordering. *Parallel Computing*, pages 318–331, 2008.

[37] Xiaoye S. Li and James W. Demmel. SuperLU_DIST: A scalable distributed-memory sparse direct solver for unsymmetric linear systems. *ACM Trans. Mathematical Software*, 29(2):110–140, June 2003.

[38] Jérémie Gaidamour and Pascal Hénon. HIPS: a parallel hybrid direct/iterative solver based on a schur complement. 2008.

[39] Luc Giraud, Azzam Haidar, and L. T. Watson. Parallel scalability study of hybrid preconditioners in three dimensions. *Parallel Computing*, pages 363–379, 2008.

[40] www.nersc.gov/systems/hopper-cray xe6/.

[41] Eugene N.Parker. Sweet's mechanism for merging magnetic fields in conducting fluids. *Journal of geophys. res.*, 62(4):509–520, 1957.

[42] Russell M. Kulsrud. Magnetic reconnection: Sweet-Parker versus Petschek. *Earch Planets Space*, 53:417–422, 2001.

[43] D. Biscamp, E. Schwarz, and James F. Drake. Two-fluid theory of collisionless magnetic reconnection. *Physics of Plasmas*, 4:1002–1009, 1997.

[44] James W. Demmel, Stanley C. Eisenstat, John R. Gilbert, Xiaoye S. Li, and Joseph W. H. Liu. A supernodal approach to sparse partial pivoting. *SIAM J. Matrix Analysis and Applications*, 20(3):720–755, 1999.