BOUT++ preconditioning

B.Dudson, University of York

December 17, 2013

Contents

1 Introduction

This manual describes some of the ways BOUT++ could (and in some cases does) support preconditioning, Jacobian calculations and other methods to speed up simulations. This manual assumes that you're familiar with how BOUT++ works internally.

Some notation: The ODE being solved is of the form

$$\frac{\partial \mathbf{f}}{\partial t} = \mathbf{F}\left(\mathbf{f}\right)$$

Here the state vector $f = (f_0, f_1, f_2, \ldots)^T$ is a vector containing the evolving (3D) variables $f_i(x, y, z)$.

The Jacobian of this system is then

$$\mathbb{J} = \frac{\partial \mathbf{F}}{\partial \mathbf{f}}$$

The order of the elements in the vector $\mathbf{f}$ is determined in the solver code and SUNDIALS, so here just assume that there exists a map $\mathbb{I}$ between a global index k and (variable, position) i.e. (i, x, y, z)

$$\mathbf{I} : (i, x, y, z) \mapsto k$$

and it's inverse

$$\mathbf{I}^{-1} : k \mapsto (i, x, y, z)$$

Some problem-specific operations which can be used to speed up the timestepping

1. Jacobian-vector multiply: Given a vector, multiply it by $\mathbb{J}$
2. Preconditioner multiply: Given a vector, multiply by an approximate inverse of $\mathbb{M} = \mathbb{I} - \gamma\mathbb{J}$
3. Calculate the stencils i.e. non-zero elements in $\mathbb{J}$
4. Calculate the non-zero elements of $\mathbb{J}$

2 Physics problems

Some interesting physics problems of increasing difficulty

2.1 Resistive drift-interchange instability

A "simple" test problem of 2 fields, which results in non-trivial turbulent results. Supports resistive drift wave and interchange instabilities.

$$\begin{aligned}
\frac{\partial N_i}{\partial t} + \mathbf{v}_E \cdot \nabla N_i &= 0 \\
\frac{\partial \omega}{\partial t} + \mathbf{v}_E \cdot \nabla \omega &= 2\omega_{ci}\mathbf{b} \times \kappa \cdot \nabla P + N_i Z_i e \frac{4\pi V_A^2}{c^2} \nabla_{||} j_{||} \\
\nabla_\perp^2 \omega / N_i &= \phi \\
0.51 \nu_{ei} j_{||} &= \frac{e}{m_e} \partial_{||} \phi + \frac{T_e}{N_i m_e} \partial_{||} N_i
\end{aligned}$$

2.2 Reduced 3-field MHD

This is a 3-field system of pressure P, magnetic flux ψ and vorticity U:

$$\mathbf{f} = \begin{pmatrix} P \\ \psi \\ U \end{pmatrix}$$

$$\begin{aligned} \frac{\partial \psi}{\partial t} &= -\frac{1}{B_0}\nabla_{||}\phi \\ &= -\frac{1}{B_0}\left[\mathbf{b}_0 - (\mathbf{b}_0 \times \nabla\psi)\right]\cdot\nabla\phi \\ &= -\frac{1}{B_0}\mathbf{b}_0\cdot\nabla\phi - \frac{1}{B_0}(\mathbf{b}_0\times\nabla\phi)\cdot\nabla\psi \\ \Rightarrow \frac{d\psi}{dt} &= -\frac{1}{B_0}\mathbf{b}_0\cdot\nabla\phi \end{aligned}$$

The coupled set of equations to be solved are therefore

$$\frac{1}{B_0}\nabla_\perp^2\phi = U \tag{1}$$

$$\left(\frac{\partial}{\partial t} + \mathbf{v}_E\cdot\nabla\right)\psi = -\frac{1}{B_0}\mathbf{b}_0\cdot\nabla\phi \tag{2}$$

$$\left(\frac{\partial}{\partial t} + \mathbf{v}_E\cdot\nabla\right)P = 0 \tag{3}$$

$$\begin{aligned} \left(\frac{\partial}{\partial t} + \mathbf{v}_E\cdot\nabla\right)U &= \frac{1}{\rho}B_0^2\left[\mathbf{b}_0 - (\mathbf{b}_0\times\nabla\psi)\right]\cdot\left(\frac{J_{||0}}{B_0} - \frac{1}{\mu_0}\nabla_\perp^2\psi\right) \\ &+ \frac{1}{\rho}\mathbf{b}_0\times\kappa_0\cdot\nabla P \end{aligned} \tag{4}$$

$$\mathbf{v}_E = \frac{1}{B_0}\mathbf{b}_0\times\nabla\phi \tag{5}$$

The Jacobian of this system is therefore:

$$\mathbb{J} = \left[\begin{array}{c|c|c} -\mathbf{v}_E\cdot\nabla & 0 & \left[\mathbf{b}_0\times\nabla\left(P_0+P\right)\cdot\nabla\right]\nabla_\perp^{-2} \\ \hline 0 & -\mathbf{v}_E\cdot\nabla & \left(\mathbf{b}_0\cdot\nabla\right)\nabla_\perp^{-2} \\ \hline 2\mathbf{b}_0\times\kappa_0\cdot\nabla & \begin{array}{l} -\frac{B_0^2}{\mu_0\rho}\left(\mathbf{b}_0 - \mathbf{b}_0\times\nabla\psi\right)\cdot\nabla\nabla_\perp^2 \\ +\frac{B_0^2}{\rho}\left[\mathbf{b}_0\times\nabla\left(\frac{J_{||0}}{B_0}\right)\right]\cdot\nabla \\ +\frac{B_0^2}{\mu_0\rho}\nabla\left(\nabla_\perp^2\psi\right)\cdot\left(\mathbf{b}_0\times\nabla\right) \end{array} & -\mathbf{v}_E\cdot\nabla \end{array}\right] \tag{6}$$

Where the blue terms are only included in nonlinear simulations.

This Jacobian has large dense blocks because of the Laplacian inversion terms (involving $\nabla_\perp^{-2}$ which couples together all points in an X-Z plane. The way to make $\mathbb{J}$ sparse is to solve ϕ as a constraint (using e.g. the IDA solver) which moves the Laplacian inversion to the preconditioner.

2.3 Solving ϕ as a constraint

The evolving state vector becomes

$$\mathbf{f} = \begin{pmatrix} P \\ \psi \\ U \\ \phi \end{pmatrix}$$

2.4 UEDGE equations

The UEDGE benchmark is a 4-field model with the following equations:

$$\begin{aligned}
\frac{\partial N_i}{\partial t} + V_{||}\partial_{||} N_i &= -N_i \nabla_{||} V_{||} + \nabla_\psi \left(D_\perp \partial_\psi N_i \right) \\
\frac{\partial \left(N_i V_{||} \right)}{\partial t} + V_{||}\partial_{||} \left(N_i V_{||} \right) &= -\partial_{||} P + \nabla_\psi \left(N_i \mu_\perp \partial_\psi V_{||} \right) \\
\frac{3}{2}\frac{\partial}{\partial t}\left(N_i T_e \right) &= \nabla_{||} \left(\kappa_e \partial_{||} T_e \right) + \nabla_\psi \left(N_i \chi_\perp \partial_\perp T_e \right) \\
\frac{3}{2}\frac{\partial}{\partial t}\left(N_i T_i \right) &= \nabla_{||} \left(\kappa_i \partial_{||} T_i \right) + \nabla_\psi \left(N_i \chi_\perp \partial_\perp T_i \right)
\end{aligned}$$

This set of equations is good in that there is no inversion needed, and so the Jacobian is sparse everywhere. The state vector is

$$\mathbf{f} = \begin{pmatrix} N_i \\ V_{||} \\ T_e \\ T_i \end{pmatrix}$$

The Jacobian is:

$$\mathbb{J} = \left(\begin{array}{c|c|c|c} -V_{||}\partial_{||} - \nabla_{||} V_{||} + \nabla_\psi D_\perp \partial_\psi & -\partial_{||} N_i - N_i \nabla_{||} & 0 & 0 \\ -\frac{1}{N_i}\frac{\partial V_{||}}{\partial t} - \frac{1}{N_i} V_{||} \mathbb{J}_{N_i N_i} & & & \end{array} \right) \tag{7}$$

If instead the state vector is

$$\mathbf{f} = \begin{pmatrix} N_i \\ N_i V_{||} \\ N_i T_e \\ N_i T_i \end{pmatrix}$$

then the Jacobian is

2.5 2-fluid turbulence

3 Jacobian-vector multiply

This is currently implemented into the CVODE (SUNDIALS) solver.

4 Preconditioner-vector multiply

4.1 Reduced 3-field MHD

The matrix $\mathbb{M}$ to be inverted can therefore be written

$$\mathbb{M} = \begin{bmatrix} \mathbb{D} & 0 & \mathbb{U}_P \\ 0 & \mathbb{D} & \mathbb{U}_\psi \\ \mathbb{L}_P & \mathbb{L}_\psi & \mathbb{D} \end{bmatrix} \tag{8}$$

where

$$\mathbb{D} = \mathbb{I} + \gamma \mathbf{v}_E \cdot \nabla$$

For small flow velocities, the inverse of $\mathbb{D}$ can be approximated using the Binomial theorem:

$$\mathbb{D}^{-1} \simeq \mathbb{I} - \gamma \mathbf{v}_E \cdot \nabla \tag{9}$$

Following [**?**, **?**], $\mathbb{M}$ can be re-written as

$$\mathbb{M} = \begin{bmatrix} \mathbb{E} & \mathbb{U} \\ \mathbb{L} & \mathbb{D} \end{bmatrix} \qquad \mathbb{E} = \begin{bmatrix} \mathbb{D} & 0 \\ 0 & \mathbb{D} \end{bmatrix} \qquad \mathbb{U} = \begin{pmatrix} \mathbb{U}_P \\ \mathbb{U}_\psi \end{pmatrix} \qquad \mathbb{L} = \begin{pmatrix} \mathbb{L}_P & \mathbb{L}_\psi \end{pmatrix}$$

The Schur factorization of $\mathbb{M}$ yields [**?**]

$$\mathbb{M}^{-1} = \begin{bmatrix} \mathbb{E} & \mathbb{U} \\ \mathbb{L} & \mathbb{D} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbb{I} & -\mathbb{E}^{-1}\mathbb{U} \\ 0 & \mathbb{I} \end{bmatrix} \begin{bmatrix} \mathbb{E}^{-1} & 0 \\ 0 & \mathbb{P}_{Schur}^{-1} \end{bmatrix} \begin{bmatrix} \mathbb{I} & 0 \\ -\mathbb{L}\mathbb{E}^{-1} & \mathbb{I} \end{bmatrix}$$

Where $\mathbb{P}_{Schur} = \mathbb{D} - \mathbb{L}\mathbb{E}^{-1}\mathbb{U}$ is the Schur complement. Note that this inversion is exact so far. Since $\mathbb{E}$ is block-diagonal, and $\mathbb{D}$ can be easily approximated using equation 9, this simplifies the problem to inverting $\mathbb{P}_{Schur}$, which is much smaller than $\mathbb{M}$.

A possible approximation to $\mathbb{P}_{Schur}$ is to neglect:

- All drive terms
  - the curvature term $\mathbb{L}_P$
  - the $J_{||0}$ term in $\mathbb{L}_\psi$
- All nonlinear terms (blue terms in equation 6), including perpendicular terms (so $\mathbb{D} = \mathbb{I}$)

This gives

$$\begin{aligned} \mathbb{P}_{Schur} &\simeq \mathbb{I} + \gamma^2 \frac{B_0^2}{\mu_0 \rho} \left(\mathbf{b}_0 \cdot \nabla\right) \nabla_\perp^2 \left(\mathbf{b}_0 \cdot \nabla\right) \nabla_\perp^{-2} \\ &\simeq \mathbb{I} + \gamma^2 V_A^2 \left(\mathbf{b}_0 \cdot \nabla\right)^2 \end{aligned} \tag{10}$$

Where the commutation of parallel and perpendicular derivatives is also an approximation. This remaining term is just the shear Alfvén wave propagating along field-lines, the fastest wave supported by these equations.

5 Stencils

6 Jacobian calculation

The (sparse) Jacobian matrix elements can be calculated automatically from the physics code by keeping track of the (linearised) operations going through the RHS function.

For each point, keep the value (as usual), plus the non-zero elements in that row of $\mathbb{J}$ and the constant: result = Ax + b Keep track of elements using product rule.

```
1 class Field3D {
2   data[ngx][ngy][ngz]; // The data as now
3
4   int JacIndex; // Variable index in Jacobian
5   SparseMatrix *jac; // Set of rows for indices (JacIndex,*,*,*)
6 };
```

JacIndex is set by the solver, so for the system

$$\mathbf{f} = \begin{pmatrix} P \\ \psi \\ U \end{pmatrix}$$

`P.JacIndex = 0`, ψ`.JacIndex = 1` and `U.JacIndex = 2`. All other fields are given `JacIndex = -1`.

SparseMatrix stores the non-zero Jacobian components for the set of rows corresponding to this variable. Evolving variables do not have an associated `SparseMatrix` object, but any fields which result from operations on evolving fields will have one.

References

[1] L Chacón. An optimal, parallel, fully implicit newton-krylov solver for three-dimensional viscoresistive magnetohydrodynamics. *Physics of Plasmas*, 15:056103, 2008.

[2] L Chacón, D A Knoll, and J M Finn. An implicit, nonlinear reduced resistive mhd solver. *J. Comput. Phys.*, 178:15–36, 2002.