NEPTUNE: 2047353-TN-01
Priority Equations and Test Cases: Preconditioning Milestones M1.1 and M2.1

Sue Thorne

Version 1: January 2021; Version 2: 22 February 2021

1 Introduction

Within the NEPTUNE Programme, there are a wide variety of interesting problems [1] but, due to the length of the Preconditioning project, it is important that we prioritise a small number of equations and test cases.

2 Elliptic Problems

For elliptic problems, System 2-2 from [1] is our priority, which consists of a 2-D elliptic solver in complex geometry. BOUT++ [2] already has some test cases and Nektar++ [4] also has some suitable cases. BOUT++ has finite difference examples whilst Nektar++ uses finite and spectral/hp elements for its discretizations. There is a solver in BOUT++ that sets up a matrix problem and calls PETSc [5], and another implementation of the same problem which calls HYPRE [6]. Since BOUT++ uses finite differences and not finite or spectral elements, there are plans within the NEPTUNE Programme to implement the Nektar++ version over the next 6 months or so.

3 Hyperbolic Problems

For hyperbolic problems, System 2-3 from [1] is our priority case. There is already a BOUT++ test case called SD1D [3] that models the dynamics along the magnetic field, uses finite differences and is a matrix-free implementation that uses SUNDIALS [7]. A version of this test problem is going to be formed using Nektar++ during the next few months.

For the dynamics across the magnetic field, there are 2D problems like Hasegawa-Wakatani, which are a similar to incompressible fluid dynamics (and are a simplified version of the equations shown in Ben Dudson's talk at the

NEPTUNE Kick-Off Meeting):

$$\begin{aligned}
\frac{\partial n}{\partial t} &= -\left[\phi, n\right] + \alpha\left(\phi - n\right) - \kappa\frac{\partial \phi}{\partial z} + D_n\nabla_\perp^2 n, && (1)\\
\frac{\partial \omega}{\partial t} &= -\left[\omega, n\right] + \alpha\left(\omega - n\right) + D_\omega\nabla_\perp^2 \omega, && (2)\\
\nabla^2\phi &= \omega && (3)
\end{aligned}$$

where the equations are solved for the plasma density n and vorticity $\omega = \mathbf{b}_0 \cdot \nabla \times \mathbf{v}$, where $\mathbf{v}$ is the $E \times B$ drift velocity velocity in a constant magnetic field and $\mathbf{b}_0$ is the unit vector in the direction of the equilibrium magnetic field. The Poison bracket is represented by $[\cdot, \cdot]$.

For simplicity, if the backward Euler method is used to discretise the time derivative and we assume that n, ω and ϕ are discretized using the same discretization basis, then we obtain the following discretized, non-linear equations:

$$\begin{aligned}
0 &= M\left(\underline{n}^i - \underline{n}^{i-1}\right) + \Delta t\left(diag(L_x\underline{\phi}^i)L_z\underline{n}^i - diag(L_z\underline{\phi}^i)L_x\underline{n}^i\right. && (4)\\
&\quad \left. -\alpha M\left(\underline{\phi}^i - \underline{n}^i\right) - \kappa L_z\underline{\phi} - D_n K\underline{n}^i\right),\\
0 &= M\left(\underline{\omega}^i - \underline{\omega}^{i-1}\right) + \Delta t\left(diag(L_x\underline{\phi}^i)L_z\underline{\omega}^i - diag(L_z\underline{\phi}^i)L_x\underline{\omega}^i\right. && (5)\\
&\quad \left. -\alpha M\left(\underline{\phi}^i - \underline{n}^i\right) - D_\omega K\underline{\omega}^i\right),\\
0 &= K\underline{\phi}^i - M\omega, && (6)
\end{aligned}$$

where M is the mass matrix associated with the discretization basis, L_x and L_z are matrices that contain the discretized forms of $\frac{\partial}{\partial x}$ and $\frac{\partial}{\partial z}$, respectively, K is the discretized Laplacian matrix and $diag(\underline{x})$ denotes a diagonal matrix with the (p, p) entry equal to $\underline{x}(p)$.

The Jacobian of equations (4)-(6) with respect to the unknowns $\underline{n}$, $\underline{\omega}$ and $\underline{\phi}$ can be expressed in the following block matrix format:

$$\left[\begin{array}{ccc} A & 0 & E \\ B & C & G \\ 0 & -M & K \end{array}\right], \qquad (7)$$

where

$$\begin{aligned}
A &= M + \Delta t\left(diag(L_x\underline{\phi}^i)L_z - diag(L_z\underline{\phi}^i)L_x + \alpha M - D_n K\right), && (8)\\
B &= \alpha\Delta t M, && (9)\\
C &= M + \Delta t\left(diag(L_x\underline{\phi}^i)L_z - diag(L_z\underline{\phi}^i)L_x - D_\omega K\right), && (10)\\
E &= \Delta t\left(diag(L_z\underline{n}^i)L_x - diag(L_x\underline{n}^i)L_z - \alpha M - \kappa L_z\right), && (11)\\
G &= \Delta t\left(diag(L_z\underline{\omega}^i)L_x - diag(L_x\underline{\omega}^i)L_z\right). && (12)
\end{aligned}$$

If K, L_x, L_z and M are all sparse, then the Jacobian will be sparse.

Alternatively, we can remove $\underline{\phi}^i$ from (4) and (5) by substituting in $\underline{\phi}^i = K^{-1}\underline{\omega}^i$ from (6). The Jacobian for the reduced equations is

$$\left[\begin{array}{cc} P & R \\ Q & S \end{array}\right], \qquad (13)$$

where

$$
\begin{aligned}
P &= M + \Delta\left(diag(L_x K^{-1} M\underline{\omega}^i)L_z - diag(L_z K^{-1} M\underline{\omega}^i)L_x + \alpha M - D_n K\right), \\
Q &= \alpha\Delta t M, \\
R &= \Delta t\left(diag(L_z\underline{n}^i)L_x K^{-1} M - diag(L_x\underline{n}^i)L_z K^{-1} M - \alpha M K^{-1} M \right. \\
&\quad \left. -\kappa L_z K^{-1} M\right), \\
S &= M + \Delta t\left(diag(L_x K^{-1} M\underline{\omega}^i)L_z + diag(L_z\underline{\omega}^i)L_x K^{-1} M - \alpha M K^{-1} M \right. \\
&\quad \left. -D_\omega K\right).
\end{aligned}
$$

Note that the inverse of the elliptic matrix K is normally a dense matrix, particularly for the finite and spectral element discretizations of interest to this project. Hence, for the discretizations of interest, the Jacobian of the reduced system is not sparse. Therefore, one of the questions is whether to (a) use the reduced system but have a dense Jacobian, or (b) treat the elliptic problem as a constraint but have a sparse, larger and more ill-conditioned Jacobian. Option (a) is normally done but experiments have also been done in BOUT++ (b) in order to use PETSc's matrix coloring to extract an approximate matrix for preconditioning from our matrix-free code. Nektar++ also has an implementation of the Hasegawa-Wakatani problem.

4 General Comments

When possible, Nektar++ examples should have higher priority than BOUT++ due to the expectation that finite element and spectral discretizations will be the method of choice for future simulations. In addition, we should expect adaptive hp-refinement to be used. Nektar++ is moving towards matrix-free implementations. For Nektar++, we need to keep in contact with David Moxey.

5 Acknowledgements

I would like to thank Ben Dudson for his help in prioritising the equations and test cases.

References

[1] W. Arter. Equations for NEPTUNE Proxyapps, 2020.

[2] B. Dudson et al. BOUT++ v4.3.2, March 2020.

[3] B. Dudson et al. Sd1d: One-dimensional plasma-neutral simulation for sol and divertor studies. `https://github.com/boutproject/SD1D`, 2020.

[4] D. Moxey et al. Nektar++ website. `https://www.nektar.info`, 2020.

[5] Satish Balay et al. PETSc Web page. https://www.mcs.anl.gov/petsc, 2019.

[6] Lawrence Livermore National Laboratory. HYPRE: Scalable Linear Solvers and Multigrid Methods. https://computing.llnl.gov/projects/hypre-scalable-linear-solvers-multigrid-methods/software.

[7] Lawrence Livermore National Laboratory. SUNDIALS: SUite of Nonlinear and DIfferential/ALgebraic Equation Solvers. https://computing.llnl.gov/projects/sundials.