Excalibur-Neptune report 2047356-TN-02-2

Task 1.1 Elliptic solver tests

Ben Dudson, Peter Hill, Ed Higgins, David Dickinson, and Steven Wright

University of York

David Moxey

University of Exeter

June 16, 2021

Contents

1 Executive summary

This report gives a brief overview of the origin of elliptic problems in plasma physics models, and the tokamak geometry they are solved in. Drawing mainly on experience with BOUT++, approaches to testing of both correctness and performance, and the pros and cons of them are discussed. A series of geometries which can be used for testing are described, starting with slabs of increasing complexity, and then tokamak geometries. Finally two simple time-dependent sets of equations are suggested, one for the shear Alfvèn wave, and the other the Geodesic Acoustic Wave (GAM). These provide ways to test the long-term numerical stability of the elliptic solver, together with the boundary conditions parallel and perpendicular to the magnetic field.

2 Elliptic solvers in plasma equations

Elliptic problems appear in plasma equations typically in the electromagnetic fields, in the limit where the displacement current is neglected and light speed goes to infinity. Two common components are solving for the electrostatic potential ϕ, and the parallel component of the vector potential $A_{||} = \mathbf{b}\cdot\mathbf{A}$ where $\mathbf{b} = \mathbf{B}/\left|\mathbf{B}\right|$ is the unit vector along the magnetic field $\mathbf{B}$.

The electrostatic potential is calculated from a fluid vorticity ω, or analogously in gyro-fluid and gyro-kinetic models, from ion and gyro-centre density differences. These give rise to an equation of the form

$$\nabla\cdot\left(\frac{n}{B^2}\nabla_\perp\phi\right) = \omega \tag{1}$$

where n is the plasma density, which varies in time and space. This system is often simplified by replacing n by a constant. By analogy with buoyancy-driven flows, this is called the Boussinesq approximation. The operator $\nabla_\perp = \nabla - \mathbf{b}\mathbf{b}\cdot\nabla$ is the component of the gradient perpendicular to the magnetic field. It arises because the ion polarisation drift, which ultimately gives rise to ω, depends on the electric field perpendicular to the magnetic field.

The electric field parallel to the magnetic field is determined by quite different physics, being mainly determined by the electron dynamics rather than the ions. The electromagnetic potential $A_{||} = \mathbf{b}\cdot\mathbf{A}$ is related to the current along the magnetic field:

$$\begin{aligned}
\mathbf{B} &= \nabla\times\mathbf{A} \\
\mathbf{J} &= \frac{1}{\mu_0}\nabla\times\mathbf{B} \\
&= \frac{1}{\mu_0}\nabla\times\nabla\times\mathbf{A} \\
&= \frac{1}{\mu_0}\left[\nabla\left(\nabla\cdot\mathbf{A}\right) - \nabla^2\mathbf{A}\right]
\end{aligned}$$

The Coulomb gauge is chosen, setting $\nabla\cdot\mathbf{A} = 0$. The current along the magnetic field is therefore

$$J_{||} = \mathbf{b}\cdot\mathbf{J} = -\frac{1}{\mu_0}\mathbf{b}\cdot\nabla^2\mathbf{A} \tag{2}$$

This elliptic operator is slightly different from equation 1 above, but often the following approximation is used:

$$\mathbf{b}\cdot\nabla^2\mathbf{A} \simeq \nabla\cdot\left(\nabla A_{||}\right) \tag{3}$$

In addition derivatives of $A_{||}$ along the magnetic field are often neglected, so that the same operator as equation 1 can be used.

The potential $A_{||}$ appears in models through the perturbed magnetic field

$$\delta\mathbf{B} = \nabla\times\left(\mathbf{b}A_{||}\right) \tag{4}$$

It also appears in the component of the electric field parallel to the magnetic field:

$$E_{||} = \mathbf{b}\cdot\mathbf{E} = -\mathbf{b}\cdot\nabla\phi - \frac{\partial A_{||}}{\partial t} \tag{5}$$

which appears in an Ohm's law equation for the current along the magnetic field.

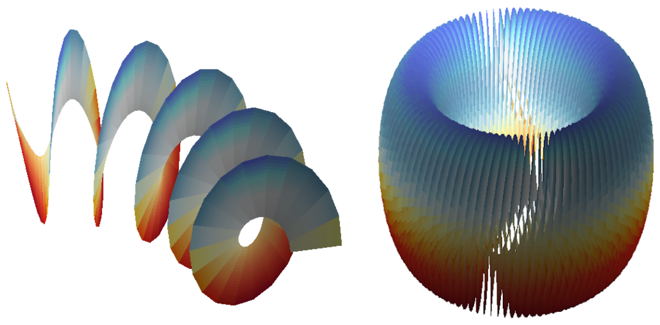

Figure 1: Domain perpendicular to the magnetic field in a torus. For illustration only. Left: A part of the domain; Right: Space filling after many toroidal turns.

2.1 Embedded domain

The elliptic equation to be solved for ϕ in equation 1 may appear to be a 2D problem, because it depends on the component of the electric field perpendicular to the magnetic field. Indeed if the magnetic field were constant in space then this would be a 2D system. In a tokamak however the magnetic field varies in space, and generally has components in both toroidal and poloidal directions. This means that the 2D domain does not in general close on itself, but forms a spiral which fills the toroidal volume. This is illustrated in figure 1. In many situations the tilt of the 2D plane in the toroidal direction can be neglected, using the fact that variation along the magnetic field (which is predominantly toroidal) is generally small. Some important exceptions are:

- Low-n (toroidal mode number) instabilities and waves. For these the assumption that parallel gradients are small relative to perpendicular can break down, and the corrections become important.
- In low aspect-ratio (“spherical“) tokamaks, the pitch of the magnetic field can become quite large: At the outboard midplane of MAST and MAST-U, the pitch can be nearly 45^o.

In either of these cases solving for the potential may require a 3D solve, rather than a decoupled set of 2D solves. It seems likely that in most cases the corrections should be small, and that 2D solves would be a good preconditioner for solving the full 3D problem.

3 Tests of elliptic solvers

As for many components of scientific high performance codes, tests need to address both the correctness and performance of the implementation.

3.1 Correctness tests

In all correctness tests it is important to define a figure of merit. This should be quantitative, and sufficiently well defined that it can be automated. This enables a pass/fail criterion to be established, and the test integrated into continuous integration workflows.

It is usually useful to combine both a measure of a global average error (such as root-mean-square, l_2 norm), and a measure which is sensitive to large localised errors (such as maximum absolute error, l_∞ norm). This enables both the overall accuracy of the scheme to be assessed, and also helps identify problems with specific areas of the mesh such as boundaries.

Round-trip tests: The simplest tests to implement are those which use a forward operator to check the accuracy of an inversion method. This type of test is often fast, and useful as check while developing the code. There are some subtleties in this which can lead to spurious issues: The forward operator must use the exact same discretisation as the inverse operator being tested. The forward operator itself should also be checked for correctness. Manual inspection of a forward operator is usually more straightforward than an inverse operator, but is not a substitute for an actual quantitative test.

Analytic solution tests: In simple geometries (such as slabs), and for particular choices of the density n in equation 1, an analytic solution can be found. In these cases the numerical solution can be compared against the analytic, to calculate the error. To properly check the method, a convergence test should be performed, using multiple grids with different resolutions, and the order of convergence compared against the theoretical order of the method used. Testing the order of convergence in this way can be challenging in some cases: Depending on the system, it can be that numerical round-off begins to affect the error, before the error enters the asymptotic regime.

Manufactured solution tests: Simple analytic tests have the benefit of being easy to implement, but are typically limited in their thoroughness: In a slab, for example, many metric tensor components are zero which in a realistic simulation would be non-zero. In that case the slab test is not able to tell whether the parts of the code which depend on those metric components are implemented correctly. The challenge for testing is that realistic cases which exercise all parts of the code typically do not have analytic solutions (otherwise there would be little point in using the code). Fortunately since the system to be tested is relatively straightforward, an analytic solution can be chosen (manufactured) for ϕ, differentiated analytically to calculate the ω function. Values for ω can then be calculated on a grid, inverted and compared to the original expression. As part of this, a non-trivial geometry needs to be generated analytically. This need not necessarily be of the same form as real simulations (e.g. a tokamak), provided that it exercises the same parts of the code (preferably, all the code).

3.2 Performance tests

The elliptic solver is often the main bottleneck to parallel scaling of plasma physics codes, certainly in existing codes such as BOUT++. This is because it involves global communications, or at least communications over a large region of the domain, and this inversion is done frequently as part of the time advance.

Elliptic solvers are critical to many areas of scientific computing, certainly not unique to plasma physics. Experience with BOUT++ however, has been that the plasma use case is different from others in ways which are important for performance: Libraries are often optimised for solving very large systems of equations (millions of d.o.f), for applications where a relatively small number of such large systems may need to be solved. In typical plasma turbulence simulations, however, the size of each system may be relatively small (10^4 d.o.f for a 2D system; 100s of degrees of freedom per solve if decomposed into 1D systems), but a turbulence simulation may require 10^5 or 10^6 such solves. Since the systems to be solved are part of the time evolution, these 10^5 or 10^6 solves are essentially serial unless parallel-in-time techniques are used, and must be solved efficiently. Since the quantity being solved for is evolving in time, solutions tend to be close to previous solutions, and so iterative schemes tend to be effective.

It is more important that performance tests use problems which are close to those which will be solved in production systems, than it is for correctness tests. Different problem sizes hit different thresholds in cache sizes, message sizes, work intensity, network capacity etc. Problems in slab geometry, which are simpler to solve than realistic situations, may have different convergence characteristics, and not exercise the preconditioner, changing the proportion of time spent in parts of the solver. As discussed above, a characteristic of typical problems is that they are solving systems which start from a generally good initial guess, the value at the previous time step. This should be taken into account in the performance testing.

Specific solution algorithms, their scaling, theoretical and measured performance are not discussed in this report beyond the comments above. These will be addressed in task 1.2 and 1.3.

4 Tests in slab geometries

A series of slab simulations of gradually increasing complexity can be used in build a code up step-by-step, and help to identify problems early before moving on to more complex systems.

- A 2D domain, with fixed (Dirichlet) boundary conditions on all sides, and the magnetic field directed perpendicular to the domain. This is a simple problem which is probably included as an example in most finite difference or finite element packages.
- The magnetic field vector **b** is not perpendicular to the plane the potential ϕ is being solved on. This introduces some non-zero metric terms, and can be made more challenging by varying the angle of the magnetic field with position.
- The boundaries can be modified so that the mesh is periodic in one direction, but continues to have boundaries in the other. In a tokamak the periodic direction would correspond to the poloidal or toroidal directions (depending on the coordinate system chosen), and the other direction to the radial direction from inside (core) to outside (edge).

- A variation on the above is to introduce a corner into the problem, so that the domain is periodic over part of the boundary (in the core), but has fixed boundaries over the rest of the boundary (the scrape-off layer).

4.1 Slab with a pole

It was found in developing a BOUT++ implementation, that the above slab geometries were not sufficiently challenging to be able to exercise a GPU preconditioner (Hypre). To make the problem more challenging, and closer to realistic tokamak geometry, a test case was developed which has a pole in the coordinate system close to (but outside) the mesh edge. This mimics the singularity at the X-point which occurs in field-aligned coordinates.

In field-aligned coordinates one of the coordinates is aligned with the magnetic field. In this case the y coordinate basis vector $\mathbf{e}_y$ is aligned to $\mathbf{B}$, so therefore the gradients of the x and z coordinates are perpendicular to $\mathbf{B}$. A choice used in BOUT++ and other plasma simulation codes is a Clebsch coordinate system, characterised by $\mathbf{B} = \nabla z \times \nabla x$. For details see the BOUT++ manual section on field-aligned coordinates [1].

The radius from a pole, r is used to set non-zero metric tensor components

$$\begin{aligned} g_{xx} &= \mathbf{e}_x \cdot \mathbf{e}_x = 1/r^2 & (6)\\ g_{yy} &= \mathbf{e}_y \cdot \mathbf{e}_y = 1 + 1/r^2 & (7)\\ g_{zz} &= \mathbf{e}_z \cdot \mathbf{e}_z = 1 & (8)\\ g_{yz} &= \mathbf{e}_y \cdot \mathbf{e}_z = 1/r & (9) \end{aligned}$$

and so coordinate Jacobian $J = 1/r$.

In BOUT++ the input file for this configuration is shown in figure 2.

5 Tests in tokamak geometries

These tests approach realistic production cases, introducing a toroidal geometry with sheared magnetic field in doubly-periodic domains, and then by introducing

```
# Mesh size
Lx = 10
Ly = 10

# Number of grid cells
nx = 20
ny = 32

# mesh spacing
dx = Lx / (nx - 4) # Account for 4 guard cells in X
dy = Ly / ny

# Location of the pole in the coordinates
pole_x = Lx + 1.0
pole_y = Ly + 1.0

# Distance from the pole
# Note here "x" is normalised to [0,1] on the grid; "y" normalised
    to [0,2*pi]
r = sqrt((pole_x - x * Lx)^2 + (pole_y - y * Ly / (2*pi))^2)

# This mimics the metric tensor close to the X-point in a tokamak
# by here setting poloidal field Bp ~ r

g11 = r^2
g22 = 1
g33 = 1 + 1 / r^2
g12 = 0
g13 = 0
g23 = -1/r

J = 1 / r

g_11 = 1 / r^2
g_22 = 1 + 1 / r^2
g_33 = 1
g_12 = 0
g_13 = 0
g_23 = 1 / r
```

Figure 2: Part of a BOUT++ input file for a slab with a pole. See BOUT++ manual for details of the code [2].

an X-point into the domain. For each geometry two kinds of tests can be performed:

- A single solve (for correctness), or small number of solves with slowly varying input (for performance).
- Time-evolving a relatively simple system of equations, in which the elliptic solve is a key part. The purpose of this is to identify potential issues with

slowly accumulating errors. These errors may be below tolerance in a single solve, but interact with the time evolving system in a way which leads to numerical instability.

5.1 Tokamak geometries

Tokamak grids are typically generated from a numerical equilibrium, and typically those are provided from experiment at low resolution (e.g. 64x64 for the whole poloidal cross-section). Higher resolution inputs can be generated using a free boundary Grad-Shafranov solver, but generating a sequence of simulation meshes from Grad-Shafranov solutions for a convergence test remains challenging. Not conceptually difficult, but the capability to do it has not been implemented and would require some considerable effort to do in a rigorous way.

For convergence tests, the easiest approach would seem to be to use an analytic solution, either to the Grad-Shafranov solution itself, or a simplified solution which is not a Grad-Shafranov solution but shares important characteristics (such as an X-point).

5.1.1 Numerical tokamak equilibria

If an analytic equilibrium and convergence to small tolerances is not required, then numerical solutions are available for many different tokamaks, both real and conceptual. A common starting point is circular cross-section geometries, which don't have an X-point. Examples include the 'cbm18' series of equilibria generated by P.Snyder (GA). Alternatively an equilibrium can be generated by a free-boundary equilibrium code such as EFIT, EFIT++, Helena, or FreeGS[3].

5.1.2 Analytic tokamak equilibria

The Grad-Shafranov equation is a nonlinear partial differential equation, and so finding analytic solutions is non-trivial. Some have however been found: The Soloviev solutions are widely used, but only include closed flux surfaces (the plasma core), not the separatrix and scrape-off layer. In addition these solutions typically have jumps or current sheets at the edge which make them problematic to extend into the vacuum

Cerfon & Freidberg found a set of analytic equilibria which include a separatrix and vacuum region outside the plasma [4]. That algorithm has been implemented in Python by John Omotani [5]. This could be used as input to a mesh generator for convergence studies.

6 Time-evolving systems

These are simple systems of equations which are intended to be relatively quick to simulate, but which can help identify problems with numerical methods or boundary conditions at an early stage.

Since we wish to identify numerical instabilities and error accumulation, we choose systems which are stable: These systems of equations support waves which are either stable or damped. Any growing mode can therefore be identified as a numerical artefact.

6.1 Shear Alfvèn wave

This is an electromagnetic wave along the magnetic field. Due to the variation of the magnetic field in a tokamak, there is a radial (across the field) phase velocity, and an initially coherent mode will tend to mix and dissipate due to the model and numerical damping.

The set of equations evolves the vorticity U and parallel vector potential $A_{||}$. It appears in normalised form as:

$$
\begin{aligned}
\frac{\partial U}{\partial t} &= \nabla_{||} j_{||} && (10)\\
\frac{\partial A_{||}}{\partial t} &= -\partial_{||}\phi - \eta j_{||} + \mu_{||e}\nabla_{||}^2 j_{||} && (11)\\
j_{||} &= -\frac{2}{\beta_e}\nabla_\perp^2 A_{||} && (12)\\
\nabla_\perp^2 \phi &= U && (13)
\end{aligned}
$$

where $\nabla_{||} f = \nabla \cdot (\mathbf{b} f)$ is the divergence of a flow along the magnetic field, and $\partial_{||} = \mathbf{b} \cdot \nabla$ is the gradient along the magnetic field. The factor β_e is the ratio of electron pressure to magnetic pressure, and is typically around 10^{-4} in the plasma edge. Dissipation is included in the form of resistivity η and parallel electron viscosity $\mu_{||e}$. The resistivity would typically be small ($< 10^{-2}$), and electron viscosity much smaller than that (usually neglected).

6.1.1 Energy conservation

The equations for this wave contain only energy sinks (dissipation), and so oscillations can only grow if there are sources of energy from either numerical instability or boundary fluxes. To calculate the energy in the system, multiply the vorticity equation by ϕ and the $A_{||}$ equation by $j_{||}$, then integrate over the simulation volume.

$$\phi U = \phi \nabla \cdot \nabla_\perp \phi = \nabla \cdot (\phi \nabla_\perp \phi) - \underbrace{\nabla \phi \cdot \nabla_\perp \phi}_{|\nabla_\perp \phi|^2} \tag{14}$$

$$\frac{\partial}{\partial t}(\phi U) = \phi \frac{\partial U}{\partial t} + U \frac{\partial \phi}{\partial t} \tag{15}$$

$$= \phi \nabla_{||} j_{||} + \nabla_\perp^2 \phi \frac{\partial \phi}{\partial t} \tag{16}$$

$$\nabla_\perp^2 \phi \frac{\partial \phi}{\partial t} = \nabla \cdot \left(\frac{\partial \phi}{\partial t} \nabla_\perp \phi\right) - \underbrace{\nabla \frac{\partial \phi}{\partial t} \cdot \nabla_\perp \phi}_{\frac{1}{2}\frac{\partial}{\partial t}\left(|\nabla_\perp \phi|^2\right)} \tag{17}$$

and so

$$\frac{\partial}{\partial t}\left[\nabla \cdot (\phi \nabla_\perp \phi) - |\nabla_\perp \phi|^2\right] = \phi \nabla_{||} j_{||} + \nabla \cdot \left(\frac{\partial \phi}{\partial t} \nabla_\perp \phi\right) - \frac{1}{2}\frac{\partial}{\partial t}\left(|\nabla_\perp \phi|^2\right) \tag{18}$$

Putting these together gives an equation for the evolution of the kinetic energy in the E×B motion:

$$\frac{1}{2}\frac{\partial}{\partial t}\left(|\nabla_\perp \phi|^2\right) = -\phi \nabla_{||} j_{||} + \nabla \cdot \left(\phi \nabla_\perp \frac{\partial \phi}{\partial t}\right) \tag{19}$$

On the left is the energy in the E×B motion, since the factors of ion mass and density are constant here, and have been normalised out. The first term on the right is a transfer of energy from electromagnetic energy, and the second term is a flux of energy from the boundary. Integrating this equation over a volume, the divergence term on the right will become a surface integral over the boundary:

$$\frac{\partial}{\partial t}\int_V \frac{1}{2}\left|\nabla_\perp\phi\right|^2 dV = -\int_V \phi\nabla_{||}j_{||}dV + \oint_S \phi\nabla_\perp\frac{\partial\phi}{\partial t}\cdot d\mathbf{S} \tag{20}$$

and so the flux of energy through the boundary is zero if ϕ is zero, or if the component of $\nabla_\perp\phi$ normal to the boundary is constant in time.

Similarly, the $A_{||}$ equation gives:

$$\begin{aligned}
j_{||}A_{||} &= \frac{2}{\beta_e}A_{||}\nabla_\perp^2 A_{||} = \frac{2}{\beta_e}\nabla\cdot\left(A_{||}\nabla_\perp A_{||}\right) - \frac{2}{\beta_e}\underbrace{\nabla A_{||}\cdot\nabla_\perp A_{||}}_{\left|\nabla_\perp A_{||}\right|^2} && (21)\\
\frac{\partial}{\partial t}\left(j_{||}A_{||}\right) &= j_{||}\frac{\partial A_{||}}{\partial t} + A_{||}\frac{\partial j_{||}}{\partial t} && (22)\\
&= j_{||}\left(\partial_{||}\phi + \eta j_{||}\right) + \frac{2}{\beta_e}A_{||}\nabla\cdot\left(\nabla_\perp\frac{\partial A_{||}}{\partial t}\right) && (23)\\
&= j_{||}\partial_{||}\phi + \eta j_{||}^2 + \frac{2}{\beta_e}\nabla\cdot\left(A_{||}\nabla_\perp\frac{\partial A_{||}}{\partial t}\right) - \frac{2}{\beta_e}\frac{1}{2}\frac{\partial}{\partial t}\left|\nabla_\perp A_{||}\right|^2 && (24)
\end{aligned}$$

and so:

$$\frac{\partial}{\partial t}\left[\frac{2}{\beta_e}\nabla\cdot\left(A_{||}\nabla_\perp A_{||}\right) - \frac{2}{\beta_e}\left|\nabla_\perp A_{||}\right|^2\right] = j_{||}\partial_{||}\phi + \eta j_{||}^2 + \frac{2}{\beta_e}\nabla\cdot\left(A_{||}\nabla_\perp\frac{\partial A_{||}}{\partial t}\right) - \frac{2}{\beta_e}\frac{1}{2}\frac{\partial}{\partial t}\left|\nabla_\perp A_{||}\right|^2 \tag{25}$$

$$\frac{1}{\beta_e}\frac{\partial}{\partial t}\left|\nabla_\perp A_{||}\right|^2 = -j_{||}\partial_{||}\phi - \eta j_{||}^2 + \frac{2}{\beta_e}\nabla\cdot\left(\frac{\partial A_{||}}{\partial t}\nabla_\perp A_{||}\right) \tag{26}$$

Integrating over the volume:

$$\frac{\partial}{\partial t}\int_V \frac{1}{\beta_e}\left|\nabla_\perp A_{||}\right|^2 dV = -\int_V j_{||}\partial_{||}\phi dV - \int_V \eta j_{||}^2 dV + \oint_S \frac{2}{\beta_e}\frac{\partial A_{||}}{\partial t}\nabla_\perp A_{||}\cdot d\mathbf{S} \tag{27}$$

Like the E×B energy equation (eq 20), the flux of energy through the boundary is zero if $A_{||} = \text{const}$ or $\nabla_\perp A_{||}\cdot\mathbf{S} = 0$.

The exchange of energy between E×B and electromagnetic forms is through

$\phi\nabla_{||}j_{||}$ and $j_{||}\partial_{||}\phi$. These should balance for energy to be conserved:

$$\phi\nabla_{||}j_{||} = \phi\nabla\cdot\left(\mathbf{b}j_{||}\right) = \nabla\cdot\left(\mathbf{b}\phi j_{||}\right) - j_{||}\underbrace{\mathbf{b}\cdot\nabla\phi}_{\partial_{||}\phi} \tag{28}$$

Hence there are no fluxes of energy through the parallel boundary if $j_{||}$ or ϕ are zero at the boundary.

From this we conclude:

- $\phi = 0$ or $\nabla_\perp\phi\cdot\mathbf{S} = \text{const}$ for conservation of energy
- $A_{||} = \text{const}$ or $\nabla_\perp A_{||}\cdot\mathbf{S} = 0$ for conservation of energy
- There is no requirement that the form of $\nabla^2_\perp$ in vorticity and $j_{||}$ equations is the same, since the only term which is common to both equations is $\phi\nabla_{||}j_{||} \leftrightarrow j_{||}\partial_{||}\phi$

6.2 Geodesic Acoustic Wave

The Geodesic Acoustic Mode (GAM) is an axisymmetric ($n = 0$) sound wave which occurs through oscillations in the radial current. It involves an up-down ($m = 1$) asymmetry in the density, together with a potential which is approximately constant on flux surfaces ($m = 0$). It can arise from a simple system of equations, but because it involves currents across the magnetic field, it exercises the radial boundary conditions. If an annulus is simulated, so that there is a boundary to the inner core, then treatment of that core boundary can be important, to ensure that there is no net current, or that radial boundary layers don't form.

6.2.1 Simplified model

The GAM oscillation arises because radial gradients of zonal ($n = 0$, $m \simeq 0$) electrostatic potential causes E×B advection of density in the poloidal direction. Because the magnetic field strength varies between inboard and outboard sides, E×B advection is faster on the outboard side than the inboard. This leads to rarefaction and compression at the top and bottom of the device (depending on flow direction). The change in pressure at top and bottom of the device alters the diamagnetic current across the magnetic field, which then modifies the electrostatic potential.

We require equations for current continuity and density n:

$$\begin{aligned}\nabla\cdot\left[\frac{m_i n}{B^2}\frac{\partial\nabla_\perp\phi}{\partial t}\right] &= \nabla\cdot\left[p\nabla\times\left(\frac{\mathbf{b}}{B}\right)\right]+\nabla_{||}j_{||} &\quad (29)\\ \frac{\partial n}{\partial t} &= -\nabla\cdot\left[n\frac{\mathbf{b}\times\nabla\phi}{B}\right] &\quad (30)\end{aligned}$$

The first (vorticity) equation includes a current along the magnetic field, $j_{||}$. This is not required to derive the wave analytically, but is required in a numerical simulation to ensure that ϕ remains approximately constant along the magnetic field. For this purpose a simple electrostatic Ohm's law can be used:

$$j_{||} = -\frac{1}{\eta}\mathbf{b}\cdot\nabla\phi \qquad (31)$$

Decreasing η will make the potential relax more quickly to a constant along flux surfaces.

The usual approach to solving this is to use

$$\nabla\times\frac{\mathbf{b}}{B}\simeq\frac{2}{B}\mathbf{b}\times\kappa$$

and to split the $E\times B$ advection term into a divergence-free advection term, and a divergence term:

$$\frac{\partial n}{\partial t} = -\frac{1}{B}\mathbf{b}\times\nabla\phi\cdot\nabla n - n\nabla\cdot\left(\frac{1}{B}\mathbf{b}\times\nabla\phi\right) \qquad (32)$$

then approximate

$$\nabla\cdot\left(\frac{1}{B}\mathbf{b}\times\nabla\phi\right)\simeq\frac{2}{B}\mathbf{b}\times\kappa\cdot\nabla\phi \qquad (33)$$

It is usual to neglect the poloidal derivative (y) terms in the $E\times B$ advection operator. In Clebsch coordinates this term looks like

$$\frac{1}{B}\mathbf{b}\times\nabla\phi\cdot\nabla n = \frac{\partial\phi}{\partial x}\frac{\partial\phi}{\partial z}-\frac{\partial\phi}{\partial z}\frac{\partial\phi}{\partial x}$$

For axisymmetric flows the z derivatives are zero, so this term vanishes, leaving only the compression term. This leaves a minimal GAM model:

$$\begin{aligned}\frac{\partial\Omega}{\partial t} &= \frac{2}{B}\mathbf{b}\times\kappa\cdot\nabla p + \nabla_{||}j_{||} & (34)\\ \frac{\partial n}{\partial t} &= -n\frac{2}{B}\mathbf{b}\times\kappa\cdot\nabla\phi & (35)\\ \Omega &= \frac{m_i n_0}{B_0^2}\nabla_\perp^2\phi & (36)\end{aligned}$$

where n_0 and B_0 are constants in space and time, and an isothermal approximation is used here:

$$p = enT_0 \tag{37}$$

Note also that here ϕ is assumed to be approximately constant on flux surfaces, which will need to be enforced numerically using an Ohm's law.

6.2.2 Analytic solution

Starting with the density equation, we look for solutions of the form

$$n\left(x,\theta,t\right) = \hat{n}\left(\theta\right)e^{ikx-iwt}$$

and potential ϕ:

$$\phi\left(x,\theta,t\right) = \hat{\phi}e^{ikx-iwt}$$

Linearising equation 35, assuming a simple circular cross-section, large aspect-ratio, and keeping only the poloidal flow

$$\mathbf{b}\times\kappa\cdot\nabla \rightarrow \frac{1}{R}\sin\theta\frac{\partial}{\partial x} \tag{38}$$

we get

$$\hat{n} = n_0\frac{2k}{BR\omega}\sin\theta\hat{\phi} \tag{39}$$

Hence if we assume $\hat{\phi}$ is independent of θ then $\hat{n}$ has a $\sin\theta$ dependence.

The parallel current term will act to equalise potential over a flux surface, but provided this occurs sufficiently rapidly we can remove it from the analysis and assume that ϕ is constant on flux surfaces. To remove the parallel current term from the vorticity equation, average over a flux surface by defining $\langle\cdot\rangle$ so that $\left\langle\nabla_{||}\right\rangle = 0$. For large aspect ratio:

$$\langle\cdot\rangle = \oint \cdot d\theta$$

This gives:

$$\begin{aligned}\frac{\partial}{\partial t}\langle\Omega\rangle &= \left\langle\frac{2}{B}\mathbf{b}\times\kappa\cdot\nabla p\right\rangle && (40)\\ -i\omega\frac{m_in_0}{B^2}\left(-k^2\right)\left\langle\hat{\phi}\right\rangle &= \left\langle 2\frac{eT_0}{BR}\sin\theta\left(ik\right)\hat{n}\right\rangle && (41)\\ \omega k^2\frac{m_in_0}{B^2} &= \left\langle 4\frac{k^2en_0T_0}{\omega B^2R^2}\sin^2\theta\right\rangle && (42)\end{aligned}$$

Most terms cancel, leaving

$$\omega^2 = \frac{eT_0}{m_i}\frac{2}{R^2}$$

i.e. the frequency depends on the sound speed $c_s = \sqrt{eT_0/m_i}$ and major radius R. This is the correct dependency for GAMs in the simple electrostatic, large aspect ratio limit when parallel flows are neglected.

6.2.3 Mass and energy conservation

For accurate and stable numerical simulations the mass and energy of the system should be conserved over long times.

Starting by multiplying the vorticity equation by ϕ

$$\phi\Omega = \phi\frac{m_in_0}{B_0^2}\nabla_\perp^2\phi = \frac{m_in_0}{B_0^2}\left[\nabla\cdot\left(\phi\nabla_\perp\phi\right) - \left|\nabla_\perp\phi\right|^2\right]$$

$$\begin{aligned}\frac{\partial}{\partial t}\left(\phi\Omega\right) &= \phi\frac{\partial\Omega}{\partial t} + \Omega\frac{\partial\phi}{\partial t} && (43)\\ &= \phi\frac{2}{B}\mathbf{b}\times\kappa\cdot\nabla p + \phi\nabla_{||}j_{||} + \Omega\frac{\partial\phi}{\partial t} && (44)\end{aligned}$$

$$\Omega\frac{\partial\phi}{\partial t} = \frac{m_in_0}{B_0^2}\left[\nabla\cdot\left(\frac{\partial\phi}{\partial t}\nabla_\perp\phi\right) - \frac{1}{2}\frac{\partial}{\partial t}\left|\nabla_\perp\phi\right|^2\right]$$

$$\frac{\partial}{\partial t}\left[\frac{1}{2}\frac{m_i n_0}{B_0^2}\left|\nabla_\perp\phi\right|^2\right] = -\phi\frac{2}{B}\mathbf{b}\times\kappa\cdot\nabla p - \phi\nabla_{||}j_{||} \tag{45}$$

$$+ \frac{m_i n_0}{B_0^2}\left[\frac{\partial}{\partial t}\nabla\cdot\left(\phi\nabla_\perp\phi\right) - \nabla\cdot\left(\frac{\partial\phi}{\partial t}\nabla_\perp\phi\right)\right] \tag{46}$$

This equation describes the change of $E \times B$ kinetic energy. The first line (eq 45) contains transfer terms from other forms of energy, whilst the second term describes fluxes from the boundaries. By setting either $\phi = 0$ or $\nabla_\perp\phi\cdot\mathbf{S} = 0$ at the boundary the fluxes go to zero at the boundary.

Many choices for the parallel current are possible, but a simple form is

$$E_{||} = -\partial_{||}\phi = \eta j_{||} \tag{47}$$

where η is the resistivity. In this case

$$\nabla_{||}\left(\phi j_{||}\right) = \phi\nabla_{||}j_{||} + j_{||}\partial_{||}\phi \tag{48}$$

$$= \phi\nabla_{||}j_{||} - \eta j_{||}^2 \tag{49}$$

$$-\phi\nabla_{||}j_{||} = -\eta j_{||}^2 - \nabla_{||}\left(\phi j_{||}\right) \tag{50}$$

and so this term is always a sink of energy, and boundary fluxes go to zero if $\phi = 0$ or $j_{||} = 0$ at the boundary.

The curvature transfer term can be derived by starting from

$$\nabla\cdot\left(\phi p\frac{2}{B}\mathbf{b}\times\kappa\right) = \phi\frac{2}{B}\mathbf{b}\times\kappa\cdot\nabla p + p\frac{2}{B}\mathbf{b}\times\kappa\cdot\nabla\phi + p\phi\nabla\cdot\left(\frac{2}{B}\mathbf{b}\times\kappa\right) \tag{51}$$

$$\frac{\partial}{\partial t}\left[\frac{1}{2}\frac{m_i n_0}{B_0^2}\left|\nabla_\perp\phi\right|^2\right] = p\frac{2}{B}\mathbf{b}\times\kappa\cdot\nabla\phi - \eta j_{||}^2 \tag{52}$$

$$+ \frac{m_i n_0}{B_0^2}\left[\frac{\partial}{\partial t}\nabla\cdot\left(\phi\nabla_\perp\phi\right) - \nabla\cdot\left(\frac{\partial\phi}{\partial t}\nabla_\perp\phi\right)\right] \tag{53}$$

$$- \nabla\cdot\left(\phi p\frac{2}{B}\mathbf{b}\times\kappa\right) - \nabla_{||}\left(\phi j_{||}\right) \tag{54}$$

$$+ p\phi\nabla\cdot\left(\frac{2}{B}\mathbf{b}\times\kappa\right) \tag{55}$$

Multiplying equation 35 by the constant eT_0 we get

$$\frac{\partial p}{\partial t} = -p\frac{2}{B}\mathbf{b}\times\kappa\cdot\nabla\phi \tag{56}$$

It can be seen that this term balances the first term on the right of equation 52. The total energy:

$$E = \int_V dV\left[\frac{1}{2}\frac{m_i n_0}{B_0^2}\left|\nabla_\perp\phi\right|^2 + p\right] \tag{57}$$

is conserved (apart from resistive losses) so long as the terms in equations 53, 54 and 55 vanish.

1. Equations 53 and 54 are divergences, and all go to zero at the boundaries if $\phi = 0$ at the boundary.

2. Equation 55 is not a divergence, so can produce sources and sinks of energy in the domain, not just at the boundary. Since p and ϕ can be arbitrary functions, the curvature vector must satisfy

$$\nabla\cdot\left(\frac{2}{B}\mathbf{b}\times\kappa\right) = 0 \tag{58}$$

   The toroidal component of this curvature vector $\frac{2}{B}\mathbf{b}\times\kappa$ doesn't affect the conservation properties; the radial x component is essential for the GAM. Either the x component has to be constant, or the y component must also be included.

7 Conclusions

A set of test cases have been described, starting from simple slabs and progressing first to more complex geometries, and then to integrating the elliptic solver as part of a time-evolving system. Criteria for correctness have been specified: l_2 and l_∞ error norms for tests with an analytic solution (manufactured in complex cases). For time-evolving problems energy conservation is a useful measure, because it has a strong connection to numerical stability and physical correctness of the result. These tests will be used in future reports to test elliptic solver implementations.

8 References

[1] BOUT++ contributors. BOUT++ manual: Field-aligned coordinates. `https://bout-dev.readthedocs.io/en/latest/user_docs/coordinates.html#id1`.

[2] BOUT++ contributors. BOUT++ manual. `https://bout-dev.readthedocs.io/`.

[3] FreeGS contributors. FreeGS: Free boundary Grad-Shafranov solver in python. `https://github.com/bendudson/freegs`.

[4] Antoine J. Cerfon and Jeffrey P. Freidberg. "One size fits all" analytic solutions to the Grad–Shafranov equation. *Phys. Plasmas*, 17:032502, 2010, doi:10.1063/1.3328818.

[5] John Omotani. Cerfon-freidberg geometry generator in python. `https://github.com/johnomotani/CerfonFreidbergGeometry`.