Hexa User's Guide – V2.0

Gordon Gibb
School of Mathematics & Statistics
University of St Andrews
gpsg@st-andrews.ac.uk

August 31, 2015

Contents

1 Introduction **2**
1.1 Directory Structure . . . 2

2 Magnetofriction **2**

3 Quick Start Guide **3**
3.1 Magnetogram Time-Series . . . 3
3.2 Data Preparation . . . 5
3.3 Running Hexa . . . 5
3.4 Generating Potential Fields . . . 5
3.5 Analysis Routines . . . 5

4 In Depth Description of the Codes' Functions and Usage **6**
4.1 Data Preparation & Cleanup . . . 6
4.1.1 Cleanup . . . 6
4.1.2 Data Preparation . . . 7
4.1.3 Generation of Hexa Files . . . 8
4.2 Hexa . . . 9
4.3 Generation of Potential Fields . . . 9
4.4 Analysis Routines . . . 10
4.5 Hexa.pro . . . 10
4.5.1 Cheung.pro . . . 11

A Data Preparation Routine Flowcharts **12**

B Hexa **16**
B.1 The Numerical Grid . . . 16
B.2 Solution of the Induction Equation . . . 17
B.3 Calculation of $\mathbf{B}$ . . . 18
B.4 Calculation of $\mathbf{j}$ . . . 20
B.5 Magnetofrictional Velocity . . . 21
B.6 Non-Ideal Terms . . . 21
B.6.1 Ohmic Diffusion . . . 22
B.6.2 Hyperdiffusion . . . 22
B.6.3 Diffusion of $\nabla \cdot \mathbf{A}$ . . . 22
B.7 Flux Imbalanced Lower Boundary Condition . . . 23

1 Introduction

This document describes the fortran code Hexa, which allows the data-driven simulation of an active region's coronal magnetic field to be carried out using normal-component magnetograms as a lower boundary condition. Hexa utilises the magnetofrictional method to produce a series of time-evolving quasi-static non-linear force-free (NLFF) equilibria. The coronal evolution is driven by the evolution of the photospheric magnetic field, as prescribed from magnetogram observations. Hexa is a parallel code (parallelisation implemented by MPI) and is designed to be run on a supercomputer, however low resolution runs can be carried out on a desktop machine relatively quickly (utilising 2 or 4 cores).

1.1 Directory Structure

Along with Hexa, a series of data preparation and analysis routines are provided. Below, the directory structure and the functions of the codes contained within them is described.

- data_prep/
  Contained within this directory are the magnetogram cleanup routines, and the codes used to construct the lower boundary condition and an initial condition.
- potenMPI/
  This directory contains a fortran code that produces a potential field for each magnetogram observation. These potential fields are used to calculate the free magnetic energy and relative magnetic helicity built up in the simulation.
- potenFourier/
  This directory contains an IDL code capable of producing potential fields from each magnetogram with an open top boundary and/or periodic side boundary conditions.
- hexaf90/
  This directory contains the hexa fortran codes which carry out the simulations of the evolution of the active region.
- analysis_dir/
  This directory contains the analysis codes which calculate the free magnetic energy, helicity and currents built up in the simulation.
- hexaIDL/
  This directory contains an IDL program that can be used to visualise the output from Hexa. In particular it allows magnetic field lines to be plotted, and can be used to locate magnetic flux ropes produced in the simulation.

2 Magnetofriction

Since the magnetic field within the corona cannot presently be routinely measured, it must be inferred from simulations or from extrapolations of the measured photospheric magnetic field. The corona is frequently treated as being force-free ($\mathbf{j}\times\mathbf{B}=0$), whereby the coronal field is constrained to be either potential ($\mathbf{j}=0$), linear force-free ($\mathbf{j}=\alpha\mathbf{B}$), or most realistically, non-linear force-free ($\mathbf{j}=\alpha(\mathbf{r})\mathbf{B}$).

The magnetofrictional method (Yang et al. 1986) is a relaxation technique that can be used to produce a NLFF field. Once a magnetic field has been perturbed, the magnetofrictional method acts to relax the magnetic field into an equilibrium state, which is generally NLFF. The magnetofrictional method evolves the 3D magnetic field, $\mathbf{B}$, according to the induction equation,

$$\frac{\partial \mathbf{A}}{\partial t} = \mathbf{v}\times\mathbf{B} + \mathbf{D} \tag{1}$$

where $\mathbf{A}$ is the magnetic vector potential such that $\mathbf{B} = \nabla \times \mathbf{A}$, $\mathbf{v}$ is the magnetofrictional velocity and $\mathbf{D}$ represents any non-ideal terms.

In the magnetofrictional approach, the equation of motion in MHD is modified to include an artificial frictional term of the form $\nu'\mathbf{v}$, where ν' is a frictional coefficient:

$$\rho\frac{D\mathbf{v}}{Dt} = \rho\mathbf{g} - \nabla p + \mathbf{j} \times \mathbf{B} - \nu'\mathbf{v} \tag{2}$$

Under the force-free approximation (steady state, with gravity and pressure being neglected), the equation of motion reduces to:

$$\mathbf{j} \times \mathbf{B} - \nu'\mathbf{v} = 0, \tag{3}$$

where $\mathbf{j} = \nabla \times \mathbf{B}$. The magnetofrictional velocity $\mathbf{v}$ can thus be prescribed as:

$$\mathbf{v} = \frac{\mathbf{j} \times \mathbf{B}}{\nu'}. \tag{4}$$

It is important to note that this is a relaxation velocity, not a physical velocity. The form of the frictional coefficient can be taken to be:

$$\nu' = \nu B^2 \tag{5}$$

where ν is a constant, which results in the frictional relaxation velocity being independent of the magnetic field strength. This aids the field to relax more quickly in low-field regions, The relaxation velocity is aligned in the direction of the Lorentz force and acts to restore any non-equilibrium perturbed field towards a force-free equilibrium.

The magnetofrictional method (Yang et al. 1986) was originally designed to relax a prescribed magnetic field to a NLFF equilibrium (e.g. Valori et al. (2005), van Ballegooijen (2004), Savcheva et al. (2012)). It may also be used to generate a continuous time series of NLFF fields. This is carried out by evolving an initial coronal field through continuously varying the photospheric magnetic field according to a time series of synthetic or observed magnetograms (van Ballegooijen et al. 2000, Mackay & van Ballegooijen 2006, Yeates et al. 2007, Mackay et al. 2011, Cheung & DeRosa 2012, Meyer et al. 2013, Gibb, Mackay, Green & Meyer 2014, Gibb, Jardine & Mackay 2014). Using this method, the time series of NLFF fields is now dependent upon previous magnetic configurations and field line connectivity. This allows the buildup of free magnetic energy and helicity to be studied – in addition to the continuous evolution of the magnetic field. This is the method the code Hexa uses to drive the evolution of the coronal field.

3 Quick Start Guide

This section provides a quick start guide to set up and carry out a simulation, then analyse the results. For more comprehensive information on the usage of the codes, please refer to Section 4. Figure 1 displays a flowchart outlining the main steps involved in running and analysing a simulation from Hexa. In the following section we assume the simulation run name is `run1`. In addition to this quick start guide, every directory listed in Section 1.1 contains a `README` file with basic usage instructions.

3.1 Magnetogram Time-Series

The input into the data preparation routines is a time series of n_t normal component magnetograms. These must be de-rotated magnetograms, containing n_x and n_y pixels in the x and y directions respectively. Each magnetogram frame should be separated by the same time, Δt, and the pixel sizes Δx and Δy should be the same ($\Delta x = \Delta y$). This time series should be stored in a IDL float array called `data`, where `data=fltarr(nx,ny,nt)`. This array should be placed in an IDL savefile called `data.sav`.

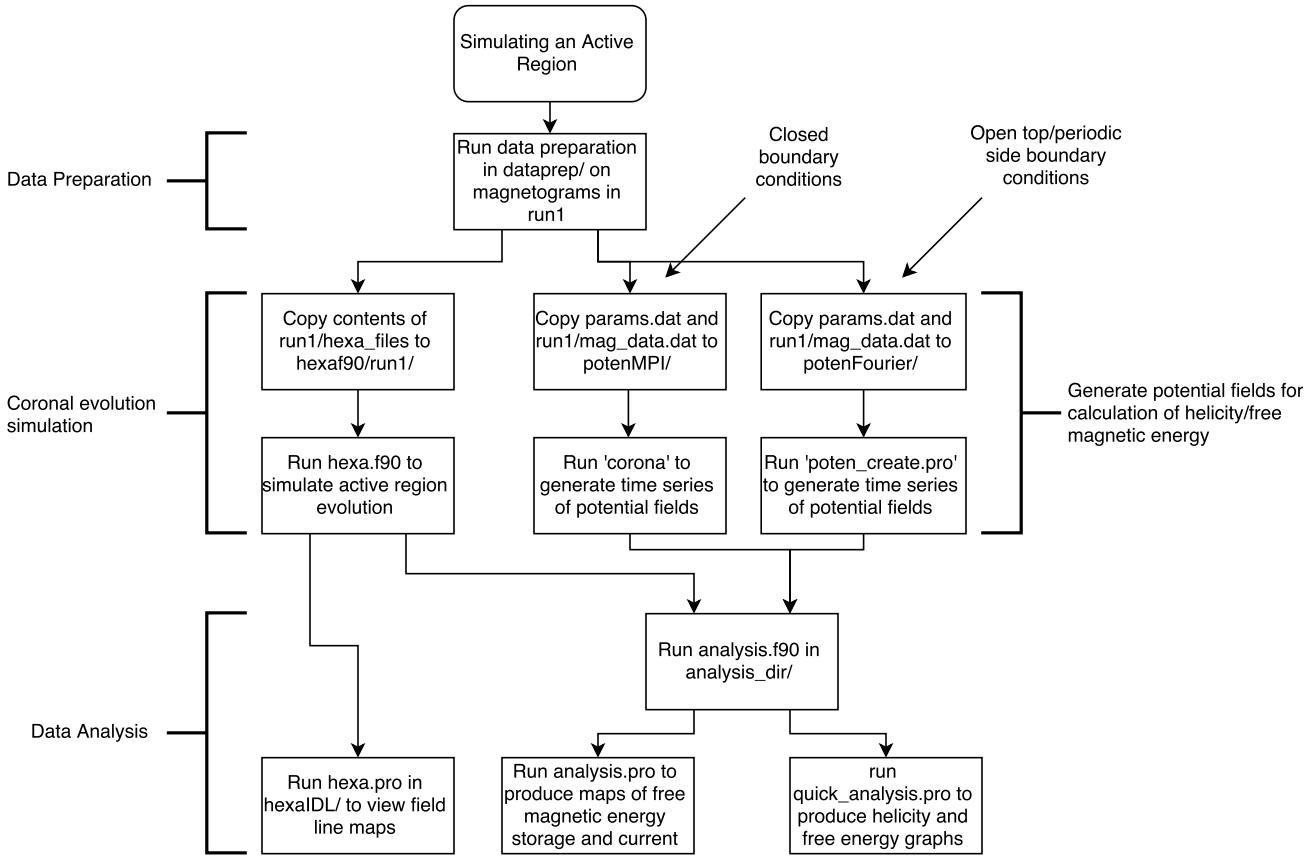

Figure 1: Flowchart explaining how to run a simulation of an active region (with a simulation name of 'run1'.)

3.2 Data Preparation

In the directory `dataprep` create a directory called `run1` and inside it place `data.sav`. Type `make` to compile the fortran codes (if they are not already compiled). Open up the file `required_info` and set `dir='run1/'`, set `deltx` to Δx (in centimetres) and `dtime` to Δt (in seconds).

Then launch IDL and type `@Start` to set up the routines, then type `data_prep` to start the data preparation routines. Follow the onscreen instructions. The data preparation routines will produce graphs of the active region properties, allow various clean-up operations to be applied, allow the user to choose the simulation resolution, whether they wish to have and open/closed top boundary, periodic/closed side boundaries and the kind of flux balancing they wish to have applied. It will then produce all the files required by Hexa into the directory `dataprep/run1/hexa_files`, in addition to the file `dataprep/params.dat`, and the file `dataprep/run1/mag_data.dat` which is required to generate the evolving boundary condition and the initial condition field.

3.3 Running Hexa

Create a directory `run1` in `hexaf90` and copy the files contained in `dataprep/run1/hexa_files/` to `hexaf90/run1/`. In `hexa_main.f90` change lines 15 and 16 to

```
dir='run1'
root='run1'
```

Type `make` to compile Hexa. Run Hexa using 2^n processors, where $n \geq 1$. The idl routine `diag.pro` can be run to view some diagnostics of the simulation, such as the free magnetic energy, the timestep, the amount of current in the simulation and the amount of energy dissipated by the magnetofriction. Hexa produces the files `run1_#####p`, where '`#####`' corresponds to the magnetogram frame number.

3.4 Generating Potential Fields

There are two different ways to produce potential fields, depending upon the kinds of boundary conditions the user has chosen. If the user has chosen closed top and side boundaries the codes in the `potenMPI/` directory must be used. If the user has chosen to have an open top boundary condition and/or periodic side boundary conditions, the codes in the `potenFourier/` directory must be used. If the wrong code is used to attempt to generate potential fields, it will stop and tell the user to use the correct code.

In `potenMPI/` create the directory `run1` and place the the file `dataprep/run1/mag_data.dat` in it. Also, place `dataprep/params.dat` into `potenMPI/`. Type `make` to compile the fortran code, then run it (on any number of processors) to generate potential fields, which will be placed in `potenMPI/run1/` with filenames of the form `poten_#####p`, where '`#####`' corresponds to the magnetogram frame number.

In `potenFourier/` create the directory `run1` and place the the file `dataprep/run1/mag_data.dat` in it. Also, place `dataprep/params.dat` into `potenFourier/`. Run IDL and run `poten_create.pro` to construct the potential fields.

3.5 Analysis Routines

In `analysis_dir`, open up `analysis.f90` and change lines 9–12 to

```
dir='../hexaf90/run1'
dirp='../potenMPI/run1'
root='run1'
rootp='poten'
```

then type `make` to compile the code. Run `./analysis`. After `analysis` has run, use `quick_analysis.pro` to generate plots of the active region's free magnetic energy with time. `analysis.pro` can be used to create images of the line of sight integrated free magnetic energy storage (e.g. see Mackay et al. (2011) and Gibb,

Mackay, Green & Meyer (2014)) and the emission proxy (Gibb, Mackay, Green & Meyer 2014). Make sure to change line 3 of both `analysis.pro` and `quick_analysis.pro` to `dir='../hexaf90/run1'`.

In addition to the analysis routines in `analysis_dir`, the program `hexa.pro` contained in `hexaIDL/` can be used to plot field lines, and locate flux ropes in the simulated corona. Please refer to Section 4 for usage instructions.

4 In Depth Description of the Codes' Functions and Usage

An in depth description and discussion of the various codes and their options will now be carried out. Flowcharts pertaining to the data preparation routines are attached in Appendix A.

4.1 Data Preparation & Cleanup

The data preparation code, located in `dataprep/`, takes the magnetograms and applies cleaning operations – as selected by the user – to them, prepares them for use as a lower boundary condition, then generates all files required for Hexa to be run. Below the various options available to the user will be discussed, and the operations applied will be described.

4.1.1 Cleanup

The user has several options to clean-up the magnetograms. They can choose from various time averaging methods, small feature removal, and isolated feature removal.

Firstly, the user is asked whether they would like to apply time averaging to the magnetograms. In order to carry out the time averaging, the following operation is applied to each magnetogram frame:

$$C_i = \left(\sum_{j=1}^{N} W_{ij} F_j \right) \left(\sum_{j=1}^{N} W_{ij} \right)^{-1} \tag{6}$$

Where C_i is the i^{th} cleaned frame, F_j is the j^{th} raw frame, and W_{ij} is the weighting function. Each cleaned frame is a linear combination of the N raw frames. This reduces the noise, since assuming that the noise is random, averaging frames together will average the noise term towards zero. The user has three options, boxcar (uniform), Gaussian, or no time averaging. Boxcar (uniform) averaging is when a certain number of frames either side of the i^{th} frame are combined with equal weighting. For the cleanup applied by the code, each cleaned frame is an average of three raw frames. The weighting function that corresponds to this is:

$$W_{ij} = \sum_{k=i-1}^{i+1} \delta_{jk}, \tag{7}$$

where δ_{jk} is the Kronneker delta, such that $C_i = (F_{i-1} + F_i + F_{i+1})/3$. For Gaussian time averaging, the weighting function is

$$W_{ij} = \exp\left(- \left[\frac{i-j}{\tau} \right]^2 \right). \tag{8}$$

This means that the weighting of frames falls off as a Gaussian away from the i^{th} frame. τ is the separation at which the weighting falls to $1/e$, and is 2 frames. Lastly, for the case of no time averaging, $W_{ij} = \delta_{ij}$, or simply, $C_i = F_i$. *It is recommended – especially for noisy data – that time-averaging is applied. Gaussian time-averaging is the recommended time-averaging method as it is most effective at removing noise, whilst retaining the large-scale properties of the active region.*

Next, the user is given the option to apply low flux removal. This sets all pixels where the absolute value of the flux density is less than 25 Mx to zero. This in effect removes the background flux values, related

to the 'quiet' regions around the active region. *Low-flux removal is recommended as it removes the flux contributions from low-field regions that are not associated with the active region.*

Lastly, the user is asked to apply small-feature removal. This process removes small scale features such as network magnetic elements found at the boundaries of super-granular cells and any noise. Small-scale removal is achieved on a pixel-by-pixel basis, where for each pixel its eight neighbours are considered. If fewer than four of its neighbours have the same sign of flux as it then its value is set to zero. Pixels along the edges of the magnetograms - who have fewer than eight neighbours - are set to zero. *Small-feature removal only has a significant effect on the magnetograms if they are particularly noisy.* For more details on the cleanup processes, please refer to Gibb, Mackay, Green & Meyer (2014).

4.1.2 Data Preparation

The data preparation involves the user choosing the simulation resolution, how the magnetograms are embedded into the 'photosphere', the flux balancing method applied and what range of frames to use.

To begin with, the user must choose the resolution of the simulation. Three options are available – 128^3, 256^3 and 512^3. A lower resolution simulation will run more quickly (and is recommended for carrying out a quick simulation to get a feel for the active region's coronal properties) however features such as flux ropes may not be resolved so well, and the effects of numerical diffusion will be greater due to the lower resolution. A higher resolution simulation will allow features such as flux ropes to be resolved, however these simulations run much more slowly and use up considerably more computer memory/data storage. *It is important to note that if a simulation is run at a lower resolution than the magnetograms then the active region's evolution may not be followed correctly as the magnetograms must be downscaled to fit into the computational box, thus removing information from the photospheric evolution.*

Secondly the user must choose how to place the magnetograms within the box. The options are to scale the magnetogram up/down to fit the box, or to apply a custom scaling. For the custom scaling, the user can choose to place the magnetogram in the box at its current size (i.e. apply no rescaling to the magnetogram) or to place the magnetogram in the box with an arbitrary scaling. The custom scale is useful to apply if the user would like to have a certain amount of blank space around the magnetogram in order to attempt to reduce boundary effects, or to apply a flux-balancing halo (see below).

Now the user is asked if they want to use periodic side boundaries. It is not recommended to do this, as the active region may interact with itself through the side boundaries. The user is then asked if they wish to have an open top boundary. If an open top boundary is selected, the magnetograms need not be flux balanced. It is recommended to have a closed top boundary, however under some circumstances it may be advisable to use an open top boundary. Such a case is when the magnetogram has a strong flux imbalance, and the user feels that applying flux balancing will significantly change the photospheric flux distribution.

Next, the user is prompted to choose the flux balancing method to be applied. If the user has selected a closed top boundary, flux balancing is required to ensure $\nabla \cdot \mathbf{B} = 0$ in the coronal volume. If the user has selected an open top boundary, they need not apply flux balancing, however they can if they choose to. Two flux balancing options are available: adjusting pixels or adding a halo. When adjusting the pixels, the number of pixels whose absolute value is greater than 25G is counted. The flux imbalance is then divided by this number of pixels, and is subtracted from each of these pixels. The halo method of flux balancing adds a halo of pixels around the edge of the computational box whose total flux is equal to the imbalance, such that the total flux through the base is zero. A halo can only be used if there is sufficient blank space (magnetogram cannot take up more than 75% of the base of the computational box) around the magnetogram. *When adjusting pixels, the correction per pixel may become greater than 25 G. In this case, some pixels may change sign. If this occurs, the user will be warned about this, and will be given the option to continue, or to go back and choose another flux balancing method/magnerogtam scaling. It should also be noted that the halo flux balancing method is most useful when there is a large flux imbalance (such that adjusting individual pixels is impractical) or when the imbalance is always of the same sign (active region is lying in a unipolar flux region).*

Lastly, the user is given the option if they wish to use the whole magnetogram time series in the simulation, or just a portion. The user may only want to use a portion of the time series if, for example, the first few

observations are of low quality due to the active region's proximity to the solar limb at these times.

4.1.3 Generation of Hexa Files

The various parameter files for Hexa must now be generated, in addition to the lower boundary condition, and an initial coronal field. The parameter files, `param1` and `*_setup`, where '`*`' is the run name, contain various parameters for Hexa to be run. These are automatically generated by the data preparation routine.

Hexa uses the magnetic vector potential, $\mathbf{A}$ as a primary variable to ensure $\nabla \cdot \mathbf{B} = 0$. A time series of the vector potential on the base, A_{xb} and A_{yb}, which reproduce the B_z from the magnetograms, must thus be constructed. These are constructed as follows:

1. Each of the cleaned magnetograms, $B_z(x, y, k)$, are taken, where k represents the discrete observations.

2. Next the horizontal components of the vector potential at the base, $z = 0$, are written in the form,

$$A_{xb}(x, y, k) = \frac{\partial \Phi}{\partial y},$$
$$A_{yb}(x, y, k) = -\frac{\partial \Phi}{\partial x},$$

   where Φ is a scalar potential.

3. For each discrete time index k, the equation

$$B_z = \frac{\partial A_{yb}}{\partial x} - \frac{\partial A_{xb}}{\partial y},$$

   then becomes,

$$\frac{\partial^2 \Phi}{\partial x^2} + \frac{\partial^2 \Phi}{\partial y^2} = -B_z, \tag{9}$$

   which is solved using a multigrid numerical method (using the Fotran codes if closed top and side boundaries are selected – details of this method can be found in the papers by Finn et al. (1994) and Longbottom (1998) and references therein) or a Fourier transform method (if open top boundary and/or periodic top boundaries are selected.)

These vector potentials make up the evolving boundary condition, and are stored in the file `*_evolve`, where '`*`' is the run name, produced by the code, `evolve.f90` which is called by the data preparation routine.

An initial coronal field must also be produced. If the top and side boundaries are closed, a linear force-free (LFF) initial condition is produced, and the user is given the option to specify the force-free parameter, α. The user can choose a value of α that ranges between [-3.3,3.3]. A choice of $\alpha = 0$ produces a potential field. The field is calculated by solving

$$\nabla^2 \mathbf{A} = -\alpha \mathbf{B} \tag{10}$$

(Finn et al. 1994), by the code, `corona.f90`, which is called automatically by the data preparation routines. The LFF initial condition is saved in the file `*_00000p`, where '`*`' is the run name. *IMPORTANT: constructing a* 512^3 *LFF initial condition uses approximately 8GB of RAM, so ensure that the machine that is running the data preparation routines has at least 8GB of RAM installed!* If the side boundaries are periodic and/or the top boundary is open a potential field is generated using a Fourier transform method.

It is important to note that the Fourier tranform method requires the input magnetograms to be in flux balance in order to construct the lower boundary condition/potential field. If the magnetogram is unbalanced (with an open top boundary condition) then the fourier transform method determines the flux imbalance per pixel, subtracts this from the frame, calculates the vector potentials, then adds a constant gradient in the x*-direction to* A_y *to add the flux imbalance back to the field.*

4.2 Hexa

Hexa solves the un-curled induction equation,

$$\frac{\partial \mathbf{A}}{\partial t} = \mathbf{v} \times \mathbf{B} + \mathbf{D} \tag{11}$$

Where $\mathbf{v}$ is the magnetofrictional velocity, expressed by

$$\mathbf{v} = \frac{1}{\nu} \frac{\mathbf{j} \times \mathbf{B}}{B^2}, \tag{12}$$

and $\mathbf{D}$ are any non-ideal terms (described below and in Appendix B). For a full description of Hexa, please see Appendix B and Gibb (2015).

There are some additional options that Hexa may be run with. Firstly, the user may include some non-ideal terms in the induction equation, namely Ohmic diffusion and Hyperdiffusion. These can be switched on by changing the lines

```
etaia=0.00
eta41=0.00
```

in the `*_setup` file, that set the Ohmic and hyperdiffusion coefficients respectively to the desired value in km^2 s^{-1} and km^4 s^{-1} respectively. *It is recommended to keep these values at zero. Hyperdiffusion may be needed if the field lines produced by hexa appear to be quite jagged.* More details on the Ohmic and Hyperdiffusive terms can be found in Appendix B.

Secondly, if Hexa has been stopped mid-run, it may be restarted from the previous snapshot that it has produced by changing the `nstrt` parameter in the `*_setup` file to be the snapshot number.

Lastly, the frictional coefficient can be changed. Within Hexa, the frictional coefficient is `FRC`, defined as

$$\frac{1}{\nu}. \tag{13}$$

This is set in `hexa_var.f90` through the variable `frc_coef` and is set to 3000 $\mathrm{km}^2\mathrm{s}^{-1}$ by default. This value is chosen because it produces the best match between the simulated coronal field and coronal observations. A larger frictional coefficient will produce a more relaxed (lower energy, flux ropes have fewer turns) corona, and a lower frictional coefficient will produce a less relaxed (higher energy, flux ropes have more turns) corona. It is recommended to keep the frictional coefficient at 3000 $\mathrm{km}^2\mathrm{s}^{-1}$.

4.3 Generation of Potential Fields

In order to generate the potential fields for each magnetogram, the same process is used as when producing the LFF initial condition (see Section 4.1.3). If the multigrid method is used to construct the potential fields (closed side and top boundary conditions), this version of `corona.f90` is parallelised in order to expedite generating the potential fields for each magnetogram. To achieve this, the code takes the N magnetograms, and loops over each of them, calculating a potential field for each magnetogram. If it is run on n processors, then it will give each process N/n magnetograms to process. If N/n is not an integer, then the final processor will get the remainder of magnetograms to process. *IMPORTANT: for calculating potential fields with a resolution of* 512^3*, keep in mind that each process will use ∼8GB of RAM. Ensure that each node does not run more processes than it has memory for!*

The Fourier transform method (open top and/or periodic sides) does not need to be run on a supercomputer, and can be run on a desktop machine. This sequentially runs through every frame and constructs a potential field.

4.4 Analysis Routines

The fortran analysis program calculates the free magnetic energy,

$$E_f = \frac{1}{8\pi}\int (B^2 - B_p^2) dV, \tag{14}$$

where B_p is the magnetic field from a potential field, and the relative magnetic helicity,

$$H_r = \int (\mathbf{A} - \mathbf{A}_p) \cdot (\mathbf{B} - \mathbf{B}_p) dV, \tag{15}$$

where $\mathbf{A}$ and $\mathbf{A}_p$ are the vector potentials corresponding to the magnetic field from hexa and the magnetic field from a potential magnetic field respectively (Finn & Antonsen 1985). In addition to this, it also calculates the current density $\mathbf{j} = \nabla \times \mathbf{B}$ and writes all of these to the files `fenergy_#####p`.

`quick_analysis.pro` allows the free magnetic energy and helicity to be plotted as a function of time, whilst `analysis.pro` allows maps of the free magnetic energy storage and the emission proxy to be produced (see Mackay et al. (2011) and Gibb, Mackay, Green & Meyer (2014) for further details.)

4.5 Hexa.pro

`hexa.pro` is a set of IDL routines to read in and visualise the fields produced by Hexa. Namely it allows field lines to be plotted. Below are a set of basic usage instructions. Please note that these are not exhaustive.

To use `hexa.pro`:

1. Type in `.r hexa` to compile hexa.pro
2. Type `hexa`. This produces a graphical interface to the program.
3. Under the main menu, select `Setup Directory and Grid` (or alternatively type `prog_grid` into the IDL prompt). The user will then be prompted to type in the directory containing the output files from the simulation. This will set up the program to use the appropriate hexa grid.
4. Under the main menu select `Restore 3D Model from File` (or alternatively type `prog_restore` into the IDL prompt). The user will then be prompted to type in the filename of the file you want read in.
5. Then select either `Show 1D Display`, `Show 2D Display` or `Show 3D Display` to display the data.

Below all the menu options are listed and their functions are described:

- Set HEXA Home Directory (`prog_home`):
  Sets the IDL program's default path. The user is asked to enter the path they would like to use.
- Setup Directory and Grid (`prog_grid`):
  Sets the directory the simulation files are stored in and sets up the grid. The user is asked to enter the path where the simulation files are stored.
- Restore 3D Model from File (`prog_restore`):
  Reads in the output file specified by the user.
- Restore Current File (`prog_restore_cur`):
  Reads in the previously read in file.
- Recompute magnetic field (`prog_field`):
  Computes (B, J and v) from the vector potentials. This is automatically called by `prog_restore` and `prog_restore_cur`.

- Show 1D Display (`prog_plot1d`):
  Plots various quantities (e.g. $\mathbf{B}$, $\mathbf{j}$, $\mathbf{v}$, α etc...) along a user defined 1D cut.

- Show 2D Display (`prog_plot2d`):
  Plots a 2D cut through the 3D volume. The cut can be visualised as a contour plot (or greysclale) of the component normal to the cut, and vectors denote the component in the plane of the cut. If the user clicks on the plot area, a field line is drawn from the location they click. The user can also locate flux ropes by clicking 'Find Flux Ropes' (or typing in `frfinder[,threshold=...]`) which locates the positions of the flux rope axes in the volume where the field strength is greater than a certain threshold (default: 50G). Clicking 'VIEW' will then plot field lines from the flux rope axis locations.

- Show 3D Display (`prog_plot3d`):
  Allows the field lines drawn to be viewed in 3D, by the user selecting the view angle. The user can also produce an animation of the volume being rotated (via the 'ROTATE' button) and view a time sequence of the evolving field (through the 'TimeSeq' button).

- Show Surface Evolution (`prog_evolve`):
  Displays an animation of the photospheric boundary condition.

- Save Field Lines (`prog_save_lines`):
  Saves the starting points of the current set of field lines to file. The user is asked to enter the number to be used in the field line save file.

- HEXA Colour Table (`hexa_colour`):
  Sets up the Hexa colour tables and plot settings. This procedure is automatically called during startup, but may occasionally require to be called to reset the plot settings.

- About HEXA:
  Displays information about Hexa and its developers

- Quit:
  Exits hexa.

4.5.1 Cheung.pro

This code produces an emission proxy using the method described in Cheung & DeRosa (2012). This traces many (thousands of) field lines and for each determines the mean current along the field line. The emission at every point in space is set to be proportionate to the mean current running through the field line at that point in space.

To run it, first have the snapshot you wish to construct the image from read into hexa.pro. Then, make sure you have compiled cheung.pro by typing `.r cheung.pro`, then type `cheung`. By default the code plots 10,000 field lines. To change this, when running the code, include the optional argument `nlines=#####` where `#####` is the number of field lines you wish to use.

A Data Preparation Routine Flowcharts

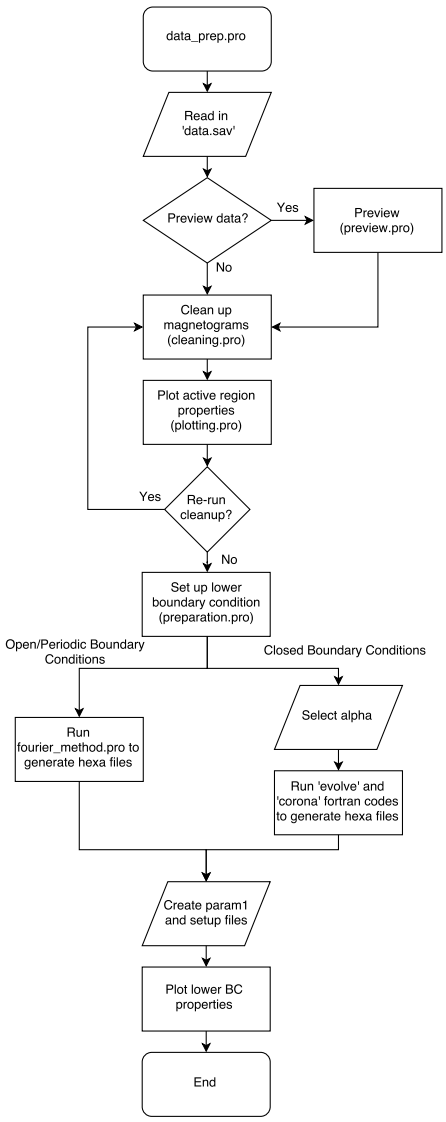

Figure 2: Flowchart of the data preparation routine. The following figures outline each process outlined in this flowchart in more detail.

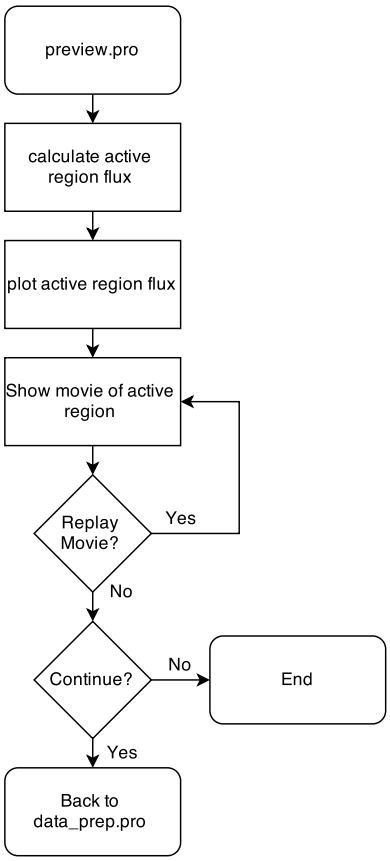

Figure 3: preview.pro

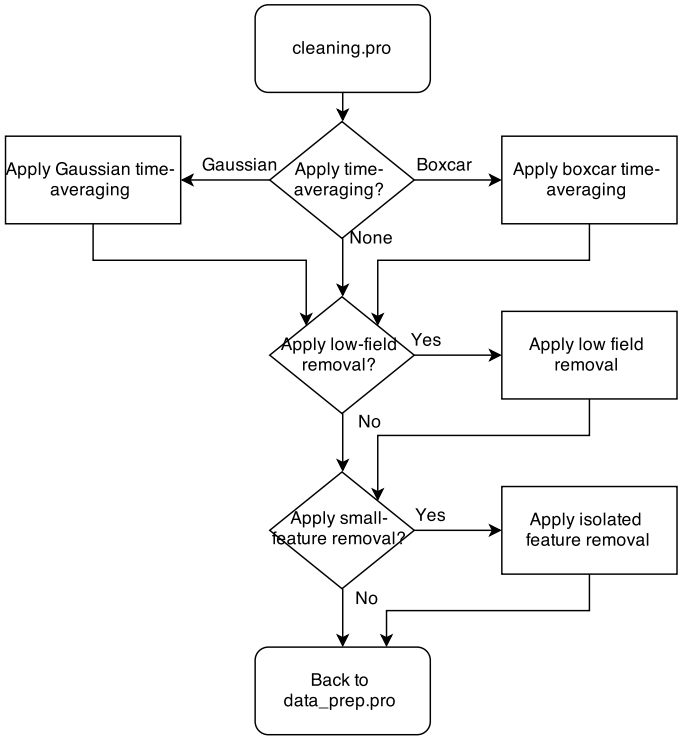

Figure 4: cleaning.pro

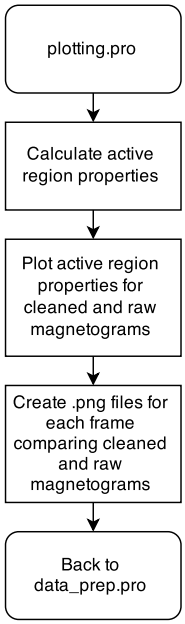

Figure 5: plotting.pro

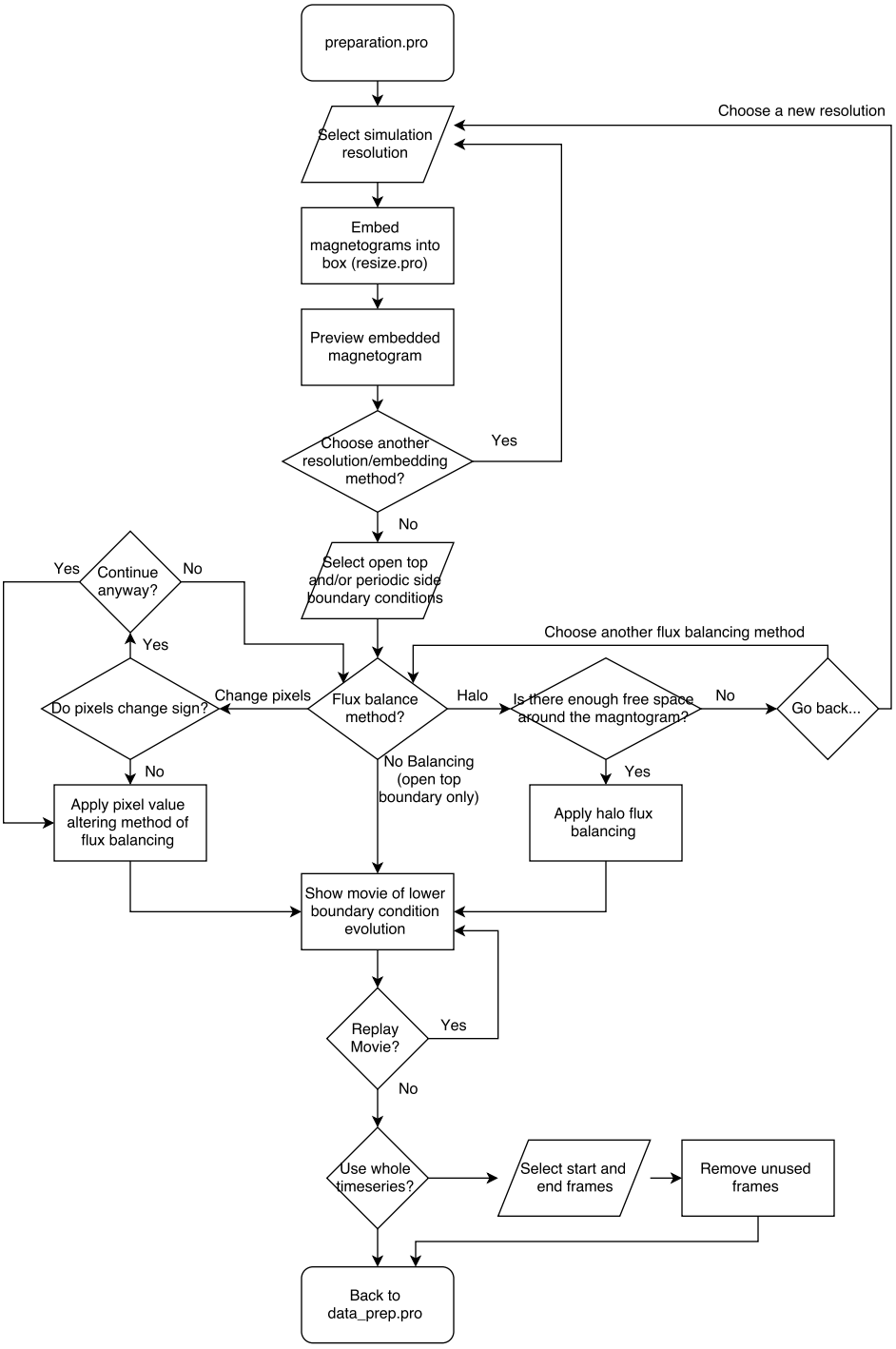

Figure 6: preparation.pro

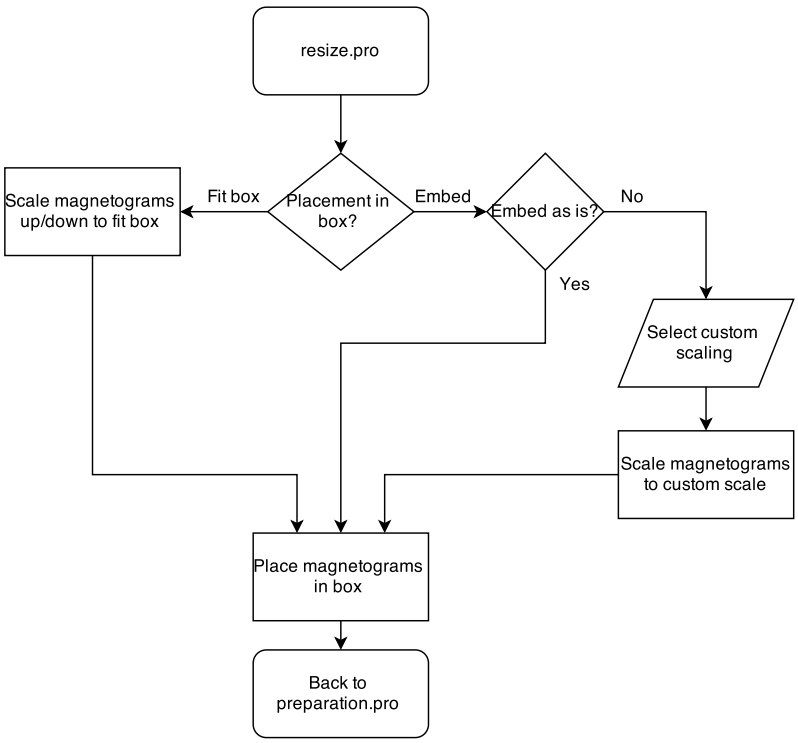

Figure 7: resize.pro

B Hexa

Hexa is a code that applies the magnetofrictional method to evolve a coronal field through a time series of NLFF equilibria in a Cartesian frame of reference. As such it is designed to simulate a small portion of the Sun whose size (l_0) is much smaller than the solar radius ($l_0 \ll R_\odot$). This small portion may be considered as cartesian, with a flat photosphere. It was written by Aad van Ballegooijen (CfA, Harvard), and was later parallelised by Duncan Mackay (Mathematical Institute, St Andrews). Finally, Gordon Gibb (Mathematical Institute, St Andrews) altered Hexa to use a variable timestep, produce a diagnostics file, and use dimensional units.

B.1 The Numerical Grid

Hexa uses a staggered grid to ensure second order accuracy and ensure $\nabla \cdot \mathbf{B} = 0$. Variables are either located on cell faces, cell corners, or cell ribs. The magnetic vector potential and currents are located on cell ribs. The magnetofrictional velocity is located on the cell corners, and the magnetic field is located on the cell faces. Figure 8 displays the faces, ribs and corners for the z-components of variables. The primary variable is the magnetic vector potential, $\mathbf{A}$, as its use on a staggered grid automatically ensures that $\nabla \cdot \mathbf{B} = 0$ at cell centres, since $\mathbf{B} = \nabla \times \mathbf{A}$ and the divergence of a curl is zero. All the other variables ($\mathbf{v}, \mathbf{B}, \mathbf{j}$) are derived from the magnetic vector potential.

Hexa is defined to have N_x, N_y and N_z grid cells in the x, y and z-directions respectively. The grid thus has $N_x + 1, N_y + 1$ and $N_z + 1$ corners in the x, y and z directions respectively. In this chapter, the coordinates of cell corners within the grid will have integer values that range from $1 : N_{X_i} + 1$, where X_i refers to either x, y or z. Cell centres have integer $+\frac{1}{2}$ values, ranging from $1 + \frac{1}{2} : N_{X_i} + \frac{1}{2}$. By default, the ranges of x, y and z are scaled to be $[0, 6]$. The grid separations $(\Delta x, \Delta y, \Delta z)$ are thus $(6/N_x, 6/N_y, 6/N_z)$. Figure 9 outlines the positions of all the variables and their indices in the plane of the z face of a grid cell

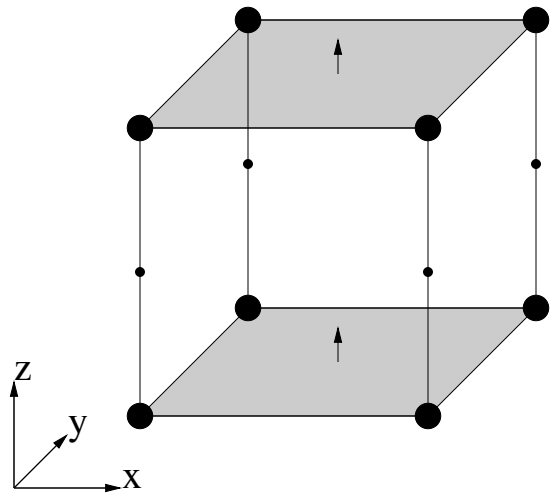

Figure 8: The locations of the z-components of variables. Corner variables are shown by the "•" symbols (for example v_z), ribs are shown by the "·" symbols (for example A_z) and the faces are shaded in grey, with the centres indicated by the arrows (for example B_z).

whose bottom left corner is located at (i, j, k).

In solving the induction equation, spatial derivatives must be taken. For cells located at the edge of the grid, this cannot be achieved without the inclusion of ghost cells (a.k.a. halo cells), which are cells located outside the boundary of the grid. The values within these cells must be specified using boundary conditions, which are discussed in Section B.3. Ghost cells are located either at rib or cell face locations, and have coordinates of either $\frac{1}{2}$ or $N_{X_i} + \frac{3}{2}$ in the direction that they extend beyond the grid.

B.2 Solution of the Induction Equation

Hexa solves the induction equation using the Eulerian method of numerical integration:

$$\mathbf{A}(t + \Delta t) = \mathbf{A}(t) + [\mathbf{v}(t) \times \mathbf{B}(t) + \mathbf{D}(t)]\,\Delta t \tag{16}$$

Where for each timestep (Δt) the RHS of Equation 1 must be calculated. In brief, this is carried out by:

1. A_x and A_y, corresponding to B_z at the photosphere, are updated on the base ($z = 0$) according to the prescribed photospheric evolution.
2. The 3D magnetic field is calculated on the cell faces from the 3D vector potential, boundary conditions are applied and then the magnetic field is averaged onto cell corners.
3. The 3D current density is calculated on cell ribs from the 3D magnetic field on the cell faces and is then averaged onto cell corners.
4. The 3D magnetofrictional velocity is calculated at cell corners.
5. $\mathbf{v} \times \mathbf{B}$ is calculated at cell corners and averaged onto the cell ribs.
6. Non-ideal terms are calculated on cell ribs.
7. The vector potential is updated by: $\mathbf{A} \rightarrow \mathbf{A} + (\mathbf{v} \times \mathbf{B} + \mathbf{D})\Delta t$.

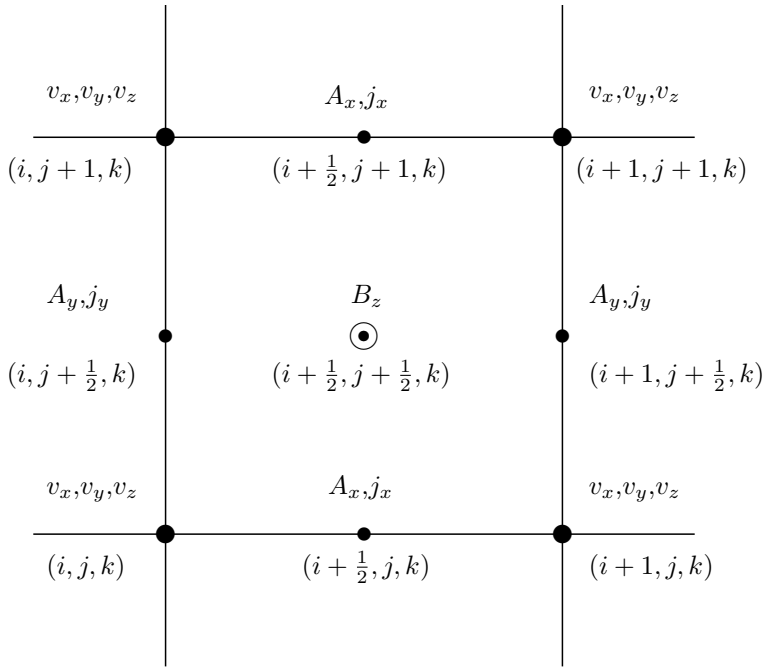

Figure 9: The location of variables located in the plane of the z face of a grid cell. The coordinates of the variables are given in parenthesis. Cell corners ($\bullet$) have coordinates of the form (i, j, k), cell centres ($\odot$) of $(i+\frac{1}{2}, j+\frac{1}{2}, k)$, y-ribs ($\cdot$) of $(i, j+\frac{1}{2}, k)$ and x-ribs ($\cdot$) of $(i+\frac{1}{2}, j, k)$.

The timestep is set according to the CFL condition, whereby the crossing time for the magnetofrictional velocity (or the diffusion timescale of hyperdiffusion/Ohmic diffusion over one grid cell) cannot be any smaller than $5\Delta t$.

The following subsections detail the above calculation steps. The steps will be described briefly, and in general will only provide a description of the calculation of the z variables so as to demonstrate the process.

B.3 Calculation of B

Firstly the magnetic field is calculated on the cell faces within the grid (i.e. neglecting ghost cells). In order to calculate this, we take the spatial derivatives of the magnetic vector potentials. For example, we calculate B_z by calculating

$$B_z = \frac{\partial A_y}{\partial x} - \frac{\partial A_x}{\partial y}. \tag{17}$$

Since in Hexa the variables are located on a grid, they are discrete rather than continuous. Therefore, to calculate derivatives, a finite difference scheme is used. Under this, B_z is calculated by

$$B_z(i+\tfrac{1}{2}, j+\tfrac{1}{2}, k) = \frac{A_y(i+1, j+\frac{1}{2}, k) - A_y(i, j+\frac{1}{2}, k)}{\Delta x} - \frac{A_x(i+\frac{1}{2}, j+1, k) - A_x(i+\frac{1}{2}, j, k)}{\Delta y}, \tag{18}$$

for $i = 1 : N_x$, $j = 1 : N_y$ and $k = 1 : N_z + 1$. Figure 10 outlines the calculation of B_z from A_x and A_y, showing the positions of the variables on the grid.

Now the boundary conditions must be applied. These are set such that the current density parallel to the boundary planes is zero. Since the magnetofrictional velocity, $\mathbf{v}$, is proportional to $\mathbf{j} \times \mathbf{B}$ this ensures that the

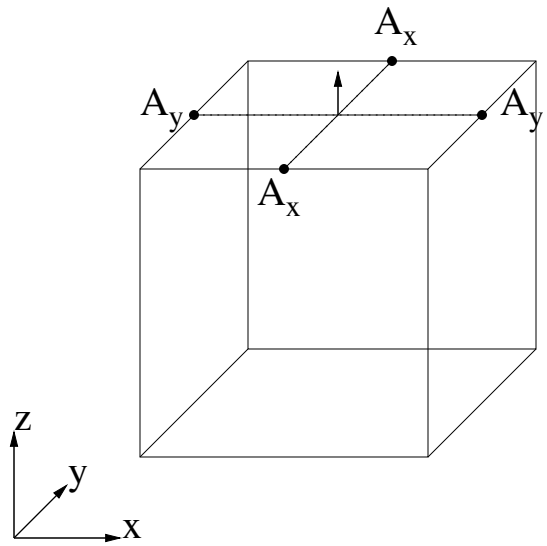

Figure 10: The calculation of B_z from A_x and A_y, outlined in Equation 18. B_z (indicated by the upward facing arrow) is calculated on the cell face from the values of A_x and A_y on the ribs surrounding the face. Δx and Δy are represented by the dotted lines joining the A_y values and A_x values respectively.

component of the magnetofrictional velocity normal to the boundary planes is zero. The magnetofrictional velocity cannot, therefore, transport any magnetic field in or out of the computational box. For the closed boundaries, we set the derivative of the magnetic field in the direction normal to the boundary plane to be zero. For example, if the top boundary ($z = 6$) is closed, we set

$$\frac{\partial B_x}{\partial z} = \frac{\partial B_y}{\partial z} = 0 \tag{19}$$

This condition ensures that j_x and j_y are

$$j_x = \frac{\partial B_z}{\partial y} \tag{20}$$

$$j_y = -\frac{\partial B_z}{\partial x} \tag{21}$$

If the initial condition coronal field has $B_z = 0$ on the upper boundary (closed boundary condition) then j_x and j_y are zero on the boundary plane, ensuring that $\mathbf{v}$ lies in the boundary plane. At the lower boundary (and the top boundary if it is open), B_z is not necessarily zero, and we instead specify

$$\frac{\partial B_y}{\partial z} = \frac{\partial B_z}{\partial y} \tag{22}$$

$$\frac{\partial B_x}{\partial z} = \frac{\partial B_z}{\partial x} \tag{23}$$

which ensures that j_x and j_y are zero on the lower boundary, resulting in $\mathbf{v}$ being parallel to the boundary.

Lastly the magnetic field is averaged onto cell corners. The magnetic field at the cell corner is defined to be the average of the magnetic field from the corner's four surrounding faces. Figure 11 illustrates this averaging process for B_z.

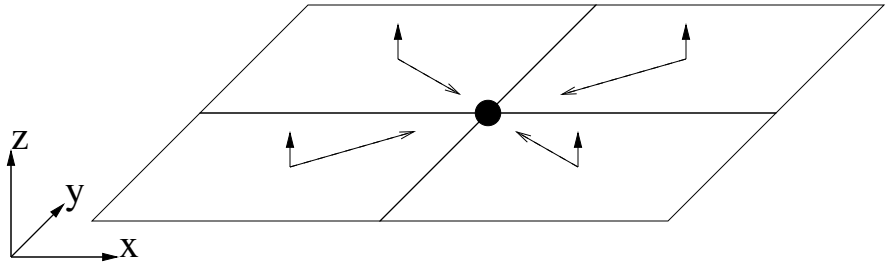

Figure 11: Averaging B_z from cell faces onto cell corners. The values at the four faces (denoted by the upward pointing arrows) surrounding the corner (•) are averaged together.

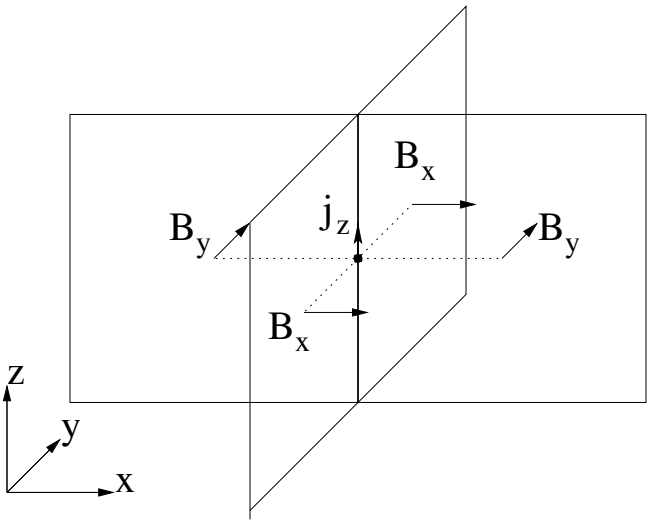

Figure 12: Calculating j_z from B_x and B_y. j_z (upward pointing arrow) is calculated on the z-rib from the values of B_x and B_y on the faces surrounding that rib. Δx is represented by the dotted line joining the two B_y values, and Δy is represented by the dotted line joining the two B_x values.

B.4 Calculation of j

The current density is calculated by $\mathbf{j} = \nabla \times \mathbf{B}$. It is calculated from the face values of $\mathbf{B}$, and is calculated onto the ribs - including ribs located in the ghost cells. For example, the z component of the current is calculated by

$$j_z(i,j,k+\tfrac{1}{2}) = \frac{B_y(i+\tfrac{1}{2},j,k+\tfrac{1}{2}) - B_y(i-\tfrac{1}{2},j,k+\tfrac{1}{2})}{\Delta x} - \frac{B_x(i,j+\tfrac{1}{2},k+\tfrac{1}{2}) - B_x(i,j-\tfrac{1}{2},k+\tfrac{1}{2})}{\Delta y} \quad (24)$$

for $i = 1 : N_x + 1$; $j = 1 : N_y + 1$ and $k = 0 : N_z + 1$. Figure 12 illustrates the calculation of j_z.

The current density is then averaged onto the cell corners. Each component of $\mathbf{j}$ at the cell corner is calculated from the average of the two nearest rib values. This process is illustrated for j_z in Figure 13.

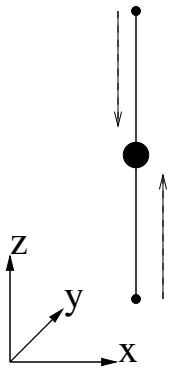

Figure 13: Averaging j_z onto cell corners. The two rib values of j_z ($\cdot$) either side of the cell corner are averaged onto the cell corner ($\bullet$).

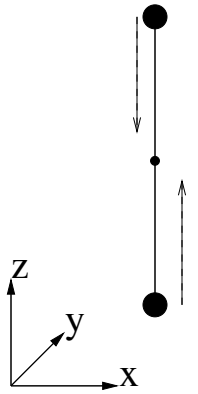

Figure 14: Averaging the z-component of the advective term from the cell corners onto the cell ribs. The two corner values ($\bullet$) either side of the rib ($\cdot$) are averaged together, and the averaged value is located at the rib.

B.5 Magnetofrictional Velocity

The magnetofrictional velocity is defined as

$$\mathbf{v} = \frac{1}{\nu}\frac{\mathbf{j}\times\mathbf{B}}{B^2}, \tag{25}$$

and is calculated on the cell corners. B^2 is calculated at cell corners. The maximum value of B^2, $B^2_{\max}$ is determined. For any point on the grid where B^2 is less than $0.0001B^2_{\max}$, the value of B^2 is set to $0.0001B^2_{\max}$. This is to prevent dividing by very small numbers, causing the magnetofrictional velocity to become anomalously large.

We then calculate the advective term in the induction equation ($\mathbf{v}\times\mathbf{B}$) at the cell corners, and average this quantity onto cell ribs. Figure 14 illustrates how the z-component of the advective term is averaged onto the cell rib.

B.6 Non-Ideal Terms

There are three different non-ideal terms that are implemented by Hexa. These are Ohmic diffusion, hyperdiffusion, and the diffusion of $\nabla\cdot\mathbf{A}$. Whilst the use of Ohmic diffusion and hyperdiffusion are optional when running Hexa, the diffusion of $\nabla\cdot\mathbf{A}$ is always carried out.

B.6.1 Ohmic Diffusion

Ohmic diffusion ($\eta\mathbf{j}$) is the only physical diffusion term, with its origins in the resistive induction equation. The resistive coefficient, η is defined within Hexa as:

$$\eta = \eta' \frac{\Delta x^2}{\Delta t} \tag{26}$$

where η' is a user-defined dimensionless parameter. The diffusion term is calculated on the cell ribs, where $\mathbf{j}$ is already defined.

B.6.2 Hyperdiffusion

Hyperdiffusion is a form of artificial diffusion which is used to smooth out gradients in the force-free parameter, α, whilst conserving the magnetic helicity. Its form is:

$$\frac{\mathbf{B}}{B^2}\nabla \cdot \left(\eta_4 B^2 \nabla \alpha\right) \tag{27}$$

where η_4 is the coefficient of hyperdiffusion, defined within Hexa as:

$$\eta_4 = \eta_4' \frac{\Delta x^4}{\Delta t} \tag{28}$$

where η_4' is a user-defined dimensionless parameter. In calculating the hyperdiffusion, α must be calculated. This is achieved by calculating

$$\alpha = \frac{j_x B_x + j_y B_y + j_z B_z}{B^2} \tag{29}$$

at cell corners. Next the gradient of α is calculated on the ribs using a finite difference scheme. For example, the z derivative of α is calculated by

$$(\nabla\alpha)_z(i,j,k+\tfrac{1}{2}) = \frac{\alpha(i,j,k+1) - \alpha(i,j,k)}{\Delta z} \tag{30}$$

for $i = 1 : N_x + 1$; $j = 1 : N_y + 1$ and $k = 1 : N_z$. B^2 must then be averaged onto the cell ribs. This is carried out by the same process as that for averaging the advective term onto the ribs (e.g. see Figure 14). Now the divergence of $\eta_4 B^2 \nabla\alpha$ must be determined. In order to determine the divergence at the boundaries, ghost cells of $\eta_4 B^2 \nabla\alpha$ are required. These ghost cells are set so that the gradients of $\eta_4 B^2 \nabla\alpha$ normal to the boundaries are set to zero. The divergence of this quantity is then calculated on the cell corners. A schematic of the calculation of the divergence is displayed in Figure 15. Next $\frac{\mathbf{B}}{B^2}\nabla \cdot \left(\eta_4 B^2 \nabla\alpha\right)$ is calculated on the cell corners, and is finally averaged onto the cell ribs.

B.6.3 Diffusion of $\nabla \cdot \mathbf{A}$

The initial condition vector potentials providing the coronal fields used in Hexa use the Coloumb gauge: $\nabla \cdot \mathbf{A} = 0$. For consistency, it is desired that the Coloumb gauge is retained throughout the simulation. To this end a diffusive term is included in the induction equation which acts to maintain the Coloumb gauge by ensuring any deviations of $\nabla \cdot \mathbf{A}$ are diffused away. The diffusive term used is

$$\eta_0 \nabla(\nabla \cdot \mathbf{A}) \tag{31}$$

where η_0 is defined within Hexa as

$$\eta_0 = 0.05 \frac{\Delta x^2}{\Delta t} \tag{32}$$

In order to understand why this term diffuses $\nabla \cdot \mathbf{A}$, consider the induction equation with this diffusive term included:

$$\frac{\partial \mathbf{A}}{\partial t} = \ldots + \eta_0 \nabla(\nabla \cdot \mathbf{A}), \tag{33}$$

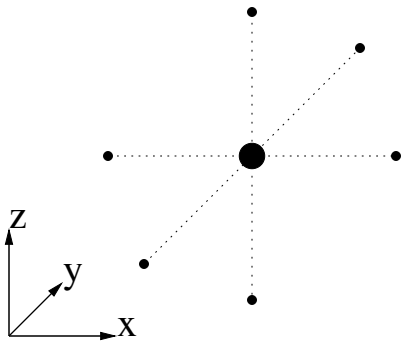

Figure 15: Calculating the divergence of a quantity. The divergence is calculated from rib values ($\cdot$) and is located at a cell corner ($\bullet$). Δx, Δy and Δz are denoted by the dotted lines in the x, y and z directions respectively.

where the ellipsis denotes the advective and any other non-ideal terms. If we take the divergence of the induction equation,

$$\frac{\partial}{\partial t}(\nabla \cdot \mathbf{A}) = \nabla \cdot (\ldots) + \eta_0 \nabla^2 (\nabla \cdot \mathbf{A}), \tag{34}$$

it is clear that this extra diffusive term included in the induction equation acts to diffuse $\nabla \cdot \mathbf{A}$.

In order to calculate this term, first the divergence of the vector potential is found at cell corners. $\nabla \cdot \mathbf{A}$ is set to zero along the top and side boundaries of the computational box, and the assumption

$$\frac{\partial A_z}{\partial z} = 0 \tag{35}$$

is made along the base of the computational box. Finally, the gradient of $\nabla \cdot \mathbf{A}$ is calculated on the cell ribs and is then multiplied by η_0.

B.7 Flux Imbalanced Lower Boundary Condition

If an open top boundary is used, the flux through the lower boundary need not be balanced. This imbalance is achieved by A_y containing a constant gradient in the x-direction. Let the difference of flux through the lower boundary between two consecutive frames be $\Delta\Phi$. During every timestep, a flux change of $\Delta\Phi\Delta t$ is introduced to the bottom of the computational box by the lower boundary evolution. Due to magnetofriction, information of this flux change takes a finite time to propagate to the upper boundary. Therefore the flux through the bottom boundary need not equal the flux through the top boundary, and hence $\nabla \cdot \mathbf{B} \neq 0$. In order to remedy this problem, the following extra steps are carried out in Hexa:

- The change in the flux imbalance per pixel between consecutive frames ($\Delta\Phi$) is determined.
- In each timestep, $\Delta x \Delta\Phi x \Delta t$ is added to A_y at all heights to ensure the flux through the lower boundary is equal to the flux through the top boundary, ensuring $\nabla \cdot \mathbf{B} = 0$

Acknowledgements

Gordon Gibb and Duncan Mackay would like to acknowledge the STFC 'Impact Acceleration Account' grant for financial support. Duncan Mackay would also like to acknowledge the Leverhulme Trust and the STFC consolidated grant for financial support.

References

Cheung, M. C. M. & DeRosa, M. L. (2012), 'A Method for Data-driven Simulations of Evolving Solar Active Regions', ApJ **757**, 147.

Finn, J. M. & Antonsen, T. M. (1985), 'Magnetic helicity: What is it and what it is good for?', *Comments Plasma Phys. Control. Fusion* **9(3)**, 111.

Finn, J. M., Guzdar, P. N. & Usikov, D. (1994), 'Three-dimensional force-free looplike magnetohydrodynamic equilibria', ApJ **427**, 475–482.

Gibb, G. P. S. (2015), The Formation and Eruption of Magnetic Flux Ropes in Solar and Stellar Coronae, PhD thesis, University of St Andrews.

Gibb, G. P. S., Jardine, M. M. & Mackay, D. H. (2014), 'Stellar differential rotation and coronal time-scales', MNRAS **443**, 3251–3259.

Gibb, G. P. S., Mackay, D. H., Green, L. M. & Meyer, K. A. (2014), 'Simulating the Formation of a Sigmoidal Flux Rope in AR10977 from SOHO/MDI Magnetograms', ApJ **782**, 71.

Longbottom, A. (1998), Force-Free Models of Filament Channels, *in* D. F. Webb, B. Schmieder & D. M. Rust, eds, 'IAU Colloq. 167: New Perspectives on Solar Prominences', Vol. 150 of *Astronomical Society of the Pacific Conference Series*, p. 274.

Mackay, D. H., Green, L. M. & van Ballegooijen, A. (2011), 'Modeling the Dispersal of an Active Region: Quantifying Energy Input into the Corona', ApJ **729**, 97.

Mackay, D. H. & van Ballegooijen, A. A. (2006), 'Models of the Large-Scale Corona. I. Formation, Evolution, and Liftoff of Magnetic Flux Ropes', ApJ **641**, 577–589.

Meyer, K. A., Sabol, J., Mackay, D. H. & van Ballegooijen, A. A. (2013), 'The Storage and Dissipation of Magnetic Energy in the Quiet Sun Corona Determined from SDO/HMI Magnetograms', ApJL **770**, L18.

Savcheva, A. S., Green, L. M., van Ballegooijen, A. A. & DeLuca, E. E. (2012), 'Photospheric Flux Cancellation and the Build-up of Sigmoidal Flux Ropes on the Sun', ApJ **759**, 105.

Valori, G., Kliem, B. & Keppens, R. (2005), 'Extrapolation of a nonlinear force-free field containing a highly twisted magnetic loop', AAP **433**, 335–347.

van Ballegooijen, A. A. (2004), 'Observations and Modeling of a Filament on the Sun', ApJ **612**, 519–529.

van Ballegooijen, A. A., Priest, E. R. & Mackay, D. H. (2000), 'Mean Field Model for the Formation of Filament Channels on the Sun', ApJ **539**, 983–994.

Yang, W. H., Sturrock, P. A. & Antiochos, S. K. (1986), 'Force-free magnetic fields - The magneto-frictional method', ApJ **309**, 383–391.

Yeates, A. R., Mackay, D. H. & van Ballegooijen, A. A. (2007), 'Modelling the Global Solar Corona: Filament Chirality Observations and Surface Simulations', Sol. Phys. **245**, 87–107.