CORONOS: A Synopsis

T. J. Dennis

September 13, 2017

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Contents

CORONOS: A Synopsis

T. J. Dennis

September 13, 2017

Chapter 1

Introduction

Wherein we introduce the reader to the gist of this document

"CORONOS" is the abbreviated name of a numerical code for use in space plasma physics called "CORONOS || SONOROC." The full name is adopted to accomodate the descriptive reverse acronym, "SONOROC" standing for "(S)ynthesized, (O)bject-based (N)umerical (O)bservatory for (R)educed plasma models with (O)ptional (C)UDA-accleration." [1]

A full history of CORONOS is beyond the scope of this document, which is intended to serve as a useful but very brief description of it, and which—it is hoped—will someday be replaced by a detailed manual for its use, adaptation, and expansion. For now it suffices to say that CORONOS is:

- "synthesized" because it descends from a suite (or rather "swarm") of older and more brittle codes which collectively were intended for similar applications.

- "object-based" because it is designed to be modular with individual classes defined for the separate and semi- independent specifications of the numerical domain of integration, the solving methodology, and the physics to be modeled respectively. Additional classes are (and/or—it is hoped—will be) defined to facilitate the management of Fast Fourier Transforms (FFT's), the organization and maintenance of parameters and their values during runs, matters pertaining to the in-

[1] The author, anxious to get started and needing *some* name for an initial source directory found the pretentious word "coronos" lurking in his mind. Once found it could not be lost, and so it was chosen with the expectation that some suitable acronym could be constructed from it some day. Regrettably, no suitable acronym ever suggested itself until, in a fit of frustration, the author considered reversing the letters.

put and output of data, the handling of errors and warnings, and extensions for the incremental implementation of GPU acceleration for certain numerically intensive operations. The classes themselves are also designed to be as modular as possible to allow for a high degree of adaptibility and expandibility while limiting and localizing the locations within the code where new bugs can arise during new development phases.

- a "Numerical Observatory" for what is hoped are obvious reasons.
- for "reduced plasma models" in the sense that it is focused on the solution to a class of coupled, non-linear time-dependent partial differential equations of a kind which is typified by—but not limited to—so-called "Reduced Magnetohydrodynamics," (**RMHD**) (Kadomtsev and Pogutse, 1974; Strauss, 1976, 1977).
- with "CUDA-acceleration" which is "optional" in the sense that the code is designed to be configurable to run on parallel high performance computing (HPC) clusters with or without the presence and utilization of graphics processing units (GPU's). This configurability is facilitated through the use of the GNU autotools build system, which also has the advantage of easing portability of the code across a variety of computing platforms.

In subsequent sections, I'll provide an overview of the general class of plasma models CORONOS is designed to integrate, and provide three specific examples that have already been either fully or partially implemented. Additionally, I will describe the method of solution currently implemented for all three of these examples, and the manner in which the numerical volume is specified. I'll also describe the three classes of initial conditions so far implemented, the two classes of boundary conditions so far implemented, and the extent to which these various conditions do and do not depend upon the choice of plasma model.

After these sections, I'll provide an illustration of CORONOS in action; demonstrating how to download, configure compile, and execute a simple test simulation, concluding with some samples of the kinds of diagnostic output that are available via the execution of post-processing script-sets that have been developed in parallel with the development of CORONOS itself. I will then outline in greater detail all diagnostic tools that either have

been developed, or are in development, to aid in the analysis of CORONOS' output. I also hope to include a section or sections describing suggested protocols for further expansion and adaptation of CORONOS for additional initial condition choices, boundary condition choices, and even plasma model choices of the same class as those now implemented. Throughout these sections I will offer, at least in general terms, the scientific motivation behind CORONOS, and I will conclude with some ideas of my own for its expansion and application.

Though far along in development, CORONOS remains a work in progress I will therefore conclude this document with a summary of its current state and a list of those development tasks that are still underway as of the distribution date of this document, and which the author regards as being essential before CORONOScan be said to be "finished."

1.1 Getting Started

1.1.1 Obtaining CORONOS

Obtaining a copy of CORONOS is easy. The source is publicly available at `https://github.com/tjdphd/coronos-unstable`. [2] I will illustrate by assuming that I have a terminal application open on my local host from where I have logged in via `ssh` to the login node of some suitable high-performance computing resource such as the San Diego Supercomputing Center's cluster "comet:" [3]

```
$ cd ~
$ mkdir repositories
$cd repositories
$
```

I also assume that the user's cluster has the `Git` version control tool installed. On comet `Git` is available upon logging in. Note however that it's

[2] Note the suffix "unstable" on the repository name. I'm intentionally directing the reader to this particular branch. There is a "stable" version but it is somewhat out-of-date and in also in the development stages anyway. It's best to remind the reader that CORONOS is a work in progress but directing her to the unstable branch which is currently the more portable of the two.

[3] comet has both **cpu** and **gpu** computing partitions and is a good choice for illustrating the two configuration options on one machine.

possible that on the reader's cluster, it may be that `Git` is installed but unavailable until "loaded" as a module using a command such as:

```
$ module load git
```

At this point one may click on the link above, or enter the address into the search field of a web browser to navigate to the public repository on `github`. Once there one may click on the green button labeled "Clone or download " which will open a dialog box with a field wherein the `https:` address for the repository may be copied and pasted on to the command line after the "`git clone`" command as follows:[4]

```
$ git clone https://github.com/tjdphd/coronos-unstable.git
Cloning into 'coronos-unstable'...
remote: Counting objects: 537, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 537 (delta 0), reused 1 (delta 0), pack-reused 534
Receiving objects: 100% (537/537), 805.77 KiB | 3.65 MiB/s, done.
Resolving deltas: 100% (320/320), done.
$
```

Upon pressing return a sub-directory of repositories called "`coronos-unstable`" should now be present and should contain a variety of files:

```
$ ls
coronos-unstable
$ ls coronos-unstable/
LICENSE  Makefile.am  README.md  TODO  configure.ac  coronos.in
crs-archive.s doc  idl  init_coronos.s  m4  revert.s src
tests
$
```

The items `src`, `doc`, `idl`, `tests`, and `m4` are all directories which the reader is encouraged to explore but to not (yet) otherwise manipulate. Their contents are as follows:

- `src` contains the `C++` source files for CORONOS.

[4] Alternatively, one may skip the web browsing completely and simply type this command oneself at the command prompt.

- `doc` contains the source for the latest version of this document as well as the document itself.
- `idl` contains a collection of post- processing IDL-based[5] "script-sets" that are useful for creating a variety of diagnostic graphics including animations. These are discussed in detail in chapter **??**.
- `tests` contains a number of files with names of the form `coronos.in_<suffix>`. These are pre-prepared parameter input files for test simulations of CORONOS. We will discuss these and the file `coronos.in` itself in chapter **??**.
- `m4` is a folder containing `m4`-language "macro" definitions. If you don't know what this means fear not. These files are required for the protocol to be described in the next section for configuring and building the CORONOSexecutable. If I have done my job you will not notice them being used at all.

The documents `LICENSE TODO` and `README.md` are self-explanatory. The files with the "`.s`" suffix are shell scripts, most of which are to be discussed in other sections, and the files `configure.ac` and `Makefile.am`, like the contents of the directory `m4` are required for the configuration and compilation of CORONOS which we now discuss.

1.1.2 Configuring and compiling CORONOS

[5]IDL is an acronym for "Interactive Data Language" and is a very powerful computing language of particular use for the analysis and graphical representation of large sets of numerical data. Though not free software it is usually available for use on high-performance computing clusters such as comet. If IDL is not available on your system please contact me I do hope to provide a python-based alternative using freely available software someday.

Chapter 2

Overview of "Reduced Plasma Models"

Wherein we introduce the reader to the gist of this document

The development of CORONOS has been motivated chiefly by a wish to advance previous studies relevant to the understanding of processes occurring in the solar corona. Studies of particular interest include high-Lundquist number simulations of magnetic reconnection (Ng and Ragunathan, 2011; Tassi et al., 2010), and the examination of heating in coronal loops, whether by the "nanoflare" model of Parker (Ng et al., 2012), or as a consequence of turbulent dissipation (Ng et al., 2017). In the course of development however, other possible applications have arisen as well, including studies of solar wind turbulence (Perez and Chandran, 2013), and power transmission and particle acceleration along the Io flux tube of Jupiter (Hess et al., 2010). Figure 2.1 provides an illustration of one context for which CORONOS is applicable; namely the modeling of coronal loops.

Figure One

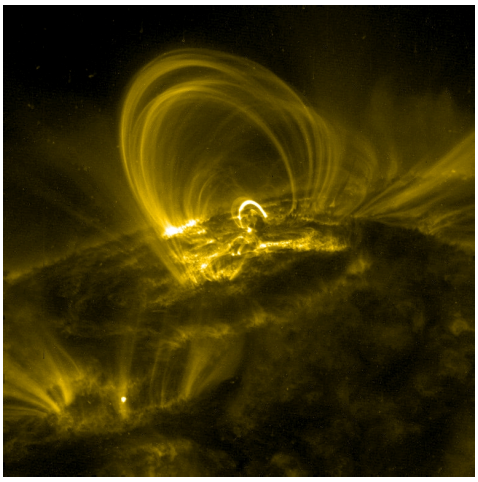
In the idealized model of a coronal loop shown in figure 2.1b, the local Cartesian coordinates (x, y, z) are to be identified with the dimensionless coordinates appearing in the reduced model examples given below, and are related to the global cylindrical coordinates (R, Φ, Z) according to,

$$x = (R - R_0)/a \tag{2.1a}$$

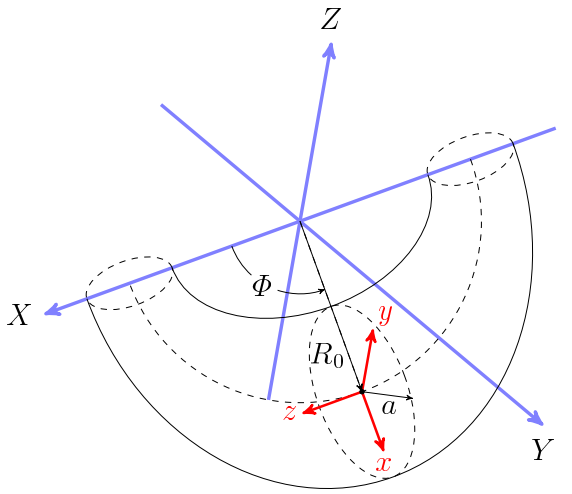
$$y = Z \tag{2.1b}$$

$$z = -\Phi, \tag{2.1c}$$

where R_0 serves as a characteristic toroidal radius for the loop center, and a as radius for the loop cross-section.



(a) Taken by the *Transition Region and Coronal Explorer*, (TRACE)—a mission of the Stanford-Lockheed Institute for Space Research, and part of the NASA small explorer program.



(b) Idealized model of a coronal loop.

Figure 2.1: One application for CORONOS: modeling coronal loops.

To emphasize CORONOS as a framework for the integration of an entire class of plasma models, I continue with a rather abstract and unfolding characterization of what is meant here by the phrase "Reduced Plasma Models." after which I will provide clarification by citing a number of examples at least some of which are likely to be familiar to the reader. In the section following this one, I will revert to the more abstract description of this class of model in the hope that the reader will quickly begin to see how the modularization sought in CORONOS begins at the level of model specification, and prior to the writing of any lines of code. In all sections where it is warranted—beginning with this one—a color-coding standard will be applied so that readers can follow the manner in which the representations of the various and distinct features of these models connect across the analytical and numerical representations of the models themselves. This approach, though requiring some patience on the part of the reader, is intended to help the reader as much as possible in grasping the

philosophy behind CORONOS' design, and to aid in understanding how the task of expanding CORONOS to include the implementation of additional reduced models is localized and minimized in the code.

Let $\mathscr{P}_\epsilon^{(z)}(F_i, G_i; N_f)$ represent a particular reduced plasma model consisting of N_f model equations which are nonlinear and coupled, and where the F_i refer to the N_f principle dependent fields variables to be solved for when integrating the model. Further let the G_i refer to N_f auxiliary fields whose definitions are related to the F_i by way of a Laplacian operator to be further specified below. Let it additionally be understood that all fields F_i and G_i are scalar fields depending on the three Cartesian spatial coordinates (x, y, z) and the time t. Let it be further understood that all field quantities, model parameters, and coordinates in space and time have been scaled to dimensionless units.

Finally, let it be understood that, intrinsic to the model $\mathscr{P}$, is a "preferred direction" (specifically z in all cases presented here), for which there exists a zeroth-order vector quantity (actually a magnetic "guide" field, $\mathbf{B}_G = B_0\hat{\mathbf{z}}$), and relative to which variations of higher order are understood to be the order of some power of an associated "small parameter" (ϵ). The guide field $\mathbf{B}_G$ can in, principle vary with x, y, and even z though in all cases examined here, it is simply the unit vector $\hat{\mathbf{z}}$. [1].

For any such model $\mathscr{P}$, of the kind defined above, the i'th equation of that model—giving the time-evolution of the field $F^{(i)}$—is expressible according to,

$$F_t^{(i)} = B^{(i)} + D_z^{(i)} + G^{(i)} + L_\perp^{(i)}, \tag{2.2}$$

[1] we note here that variation in z though, necessarily allowable for a full implementation of inhomogeneous reduced models would require enhancements to the specification of the CORONOS' numerical volume which have not yet been implemented

where,

$$F_t^{(i)} = \partial_t F^{(i)} \tag{2.3a}$$

$$B^{(i)} = \sum_{j \neq k} \epsilon_{ijk} \Big\{ b_{i,j,k}^{(F_j,F_k)} \left[F^{(j)}, F^{(k)}\right] + b_{i,j,k}^{(F_j,G_k)} \left[F^{(j)}, G^{(k)}\right] + b_{i,j,k}^{(G_j,G_k)} \left[G^{(j)}, G^{(k)}\right] \Big\} \tag{2.3b}$$

$$D_z^{(i)} = \sum_j \left\{ f_{ij}(z) \partial_z F^{(j)} + g_{ij}(z) \partial_z G^{(j)} \right\} \tag{2.3c}$$

$$L_\perp^{(i)} = c_{F^{(i)}} \boldsymbol{\nabla}_\perp^2 F^{(i)}. \tag{2.3d}$$

As a starting example, we consider homogeneous reduced magnetohydrodynamics (RMHD) (Kadomtsev and Pogutse, 1974; Strauss, 1976, 1977). Developed originally for the study of fusion plasmas, RMHD is a reduction of incompressible, resistive magnetohydrodynamics that assumes constant plasma density n and a strong, constant, zero-order guide field of strength B_0 (which we take to lie along $\hat{\mathbf{z}}$). The dynamics of interest occur at first-order with respect to the guide field, and are restricted to fluctuations of the magnetic and velocity fields perpendicular to the guide field. The dynamics thus occur on an Alfvén time scale $\tau_\mathrm{A} \sim L/v_\mathrm{A}$ where L is some characteristic length scale for the system and v_A is the Alfvén speed.

Choosing dimensionless units and the normalization $B_0 = v_\mathrm{A} = 1$, we may write the model equations for RMHD according to,

$$\partial_t \Omega = -[\phi, \Omega] + [A, J] + \partial_z J + \nu \boldsymbol{\nabla}_\perp^2 \Omega, \tag{2.4a}$$

$$\partial_t A = -[\phi, A] + \partial_z \phi + \eta \boldsymbol{\nabla}_\perp^2 A. \tag{2.4b}$$

The primary field variables in this pair of equations are the z-component of the vorticity Ω, and the poloidal magnetic flux A (to be referred to more simply in what follows as the "flux function.") These variables are related to the fluctuations in the velocity field $\mathbf{v}$ and the magnetic field $\mathbf{B}$ according to,

$$\mathbf{B} = \boldsymbol{\nabla}_\perp A \times \hat{\mathbf{z}} + B_0 \hat{\mathbf{z}} \tag{2.5a}$$

$$\mathbf{V} = \boldsymbol{\nabla}_\perp \Omega \times \hat{\mathbf{z}}. \tag{2.5b}$$

In all of equations (2.4a)−(2.5b), the operator $\boldsymbol{\nabla}_\perp$ for some field F is defined according to,

$$\boldsymbol{\nabla}_\perp F = \hat{\mathbf{x}} \partial_x F + \hat{\mathbf{y}} \partial_y F. \tag{2.6a}$$

Auxiliary fields consisting of a "Stream function" ϕ, and the parallel component of current density J (to be referred to henceforth simply as the "current density"), are then related via this operator to Ω and A respectively according to, [2]

$$\Omega = -\boldsymbol{\nabla}_\perp^2 \phi \tag{2.7a}$$

$$J = -\boldsymbol{\nabla}_\perp^2 A, \tag{2.7b}$$

where, for some field F,

$$\boldsymbol{\nabla}_\perp^2 F = \boldsymbol{\nabla}_\perp \cdot \boldsymbol{\nabla}_\perp F = \partial_{xx} F + \partial_{yy} F. \tag{2.8a}$$

This "perpendicular Laplacian operator" is seen to appear in the the dissipative terms of equations (2.4a) and (2.4b) where we also identify the dimensionless resistivity η, and dimensionless viscosity ν. Two other types of terms also appear in these equations. We see linear terms involving partial derivatives in z of the auxiliary fields J and ϕ, and we see nonlinear terms expressed as perpendicular Poisson brackets which we define here for any pair (f, g) of fields according to,

$$[f, g] = -\hat{\mathbf{z}} \cdot (\boldsymbol{\nabla}_\perp f \times \boldsymbol{\nabla}_\perp g) = \partial_y f \partial_x g - \partial_x f \partial_y g. \tag{2.9}$$

Comparing equations (2.4a) and (2.4b) to equation (2), we identify that for RMHD:

$$F^{(0)} = \Omega \tag{2.10a}$$

$$F^{(1)} = A, \tag{2.10b}$$

$$B^{(0)} = -[\phi, \Omega] + [A, J] \tag{2.11a}$$

$$B^{(1)} = -[\phi, A], \tag{2.11b}$$

$$D^{(0)} = \partial_z J \tag{2.12a}$$

$$D^{(1)} = \partial_z \phi, \tag{2.12b}$$

[2] In this document we use the sign conventions in Longcope (1993). In many other references, the convention chosen is such that the minus signs appearing here in equations (2.7a) and (2.7b) are absent.

$$L_{\perp}^{(0)} = \nu \boldsymbol{\nabla}_{\perp}^2 \Omega \tag{2.13a}$$

$$L_{\perp}^{(1)} = \eta \boldsymbol{\nabla}_{\perp}^2 A, \tag{2.13b}$$

and,

$$C_{F^{(0)}} = \nu \tag{2.14a}$$

$$C_{F^{(1)}} = \eta, \tag{2.14b}$$

while the "auxiliary terms" $G^{(i)}$ are absent.

The RMHD model has been extended to account for plasma media with inhomogeneities in plasma density $n = n(z)$ and zero-order magnetic field ($B_0 = B_0 z$), along the guide-field direction (Heinemann and Olbert, 1980; Velli et al., 1989). Typically, this inhomogeneous RMHD (**IRMHD**) model is expressed in terms of the Elsässer fields $\mathbf{z}^{\pm}$, (see for example Perez and Chandran (2013)). However, because CORONOS descends from an implementation of RMHD in terms of the fields ϕ, Ω, A, and J, we present the model here in terms of these.[3] In terms of these fields then, and using the same definitions and conventions already adopted above for RMHD, we may write the **IRMHD**model according to,

$$\partial_t \Omega = -[\phi, \Omega] + [A, J] - [U\partial_z \Omega - v_{\mathrm{A}} \partial_z J] + \left[k_p^- U\Omega - k_p^+ v_{\mathrm{A}} J\right] + \zeta \boldsymbol{\nabla}_{\perp}^2 \Omega \tag{2.15a}$$

$$\partial_t A = -[\phi, A] - [U\partial_z A - v_{\mathrm{A}} \partial_z \phi] + \left[k_m^- U A - k_m^+ v_{\mathrm{A}} \phi\right] + \zeta \boldsymbol{\nabla}_{\perp}^2 A. \tag{2.15b}$$

In addition to the quantities already defined for RMHD, some new quantities appear in these equations. These are the mean flow velocity along z $U = U(z)$, and a z-dependent Alfvén speed given in our dimensionless units by,

$$v_{\mathrm{A}}(z) = B_0(z)^2 n(z)^{-1/2}. \tag{2.16}$$

The quantities B_0, n, and v_{A} are related to one another by way of a steady-state condition given by, (Whang, 1980)

$$\frac{d}{dz}\left(\frac{n(z)U(z)}{B_0(z)}\right) = 0, \tag{2.17}$$

[3] This is ultimately because CORONOS currently implements **IRMHD**in terms of these fields, but the design of CORONOS allows in principle for alternatives, including an Elsässer form if desired.

and the four quantities $k_p^{\pm}$, $k_m^{\pm}$ are linear combinations of the inverse length scales associated with $n(z)$, $v_{\rm A}(z)$, and $B_0(z)$ as follows:

$$k_p^{\pm} = \frac{1}{2H_B} + \frac{1}{2H_A} \pm \frac{1}{4H_n} \tag{2.18a}$$

$$k_m^{\pm} = \frac{1}{2H_B} - \frac{1}{2H_A} \pm \frac{1}{4H_n}, \tag{2.18b}$$

where,

$$H_n^{-1} = \left| \frac{1}{n} \frac{\partial n}{\partial z} \right| \tag{2.19a}$$

$$H_B^{-1} = \left| \frac{1}{B_0} \frac{\partial B_0}{\partial z} \right| \tag{2.19b}$$

$$H_A^{-1} = \left| \frac{1}{v_{\rm A}} \frac{\partial v_{\rm A}}{\partial z} \right|. \tag{2.19c}$$

Finally, the single diffusivity ζ in this model appears in both equations, so that in the limits $U \to 0$, $v_{\rm A} \to 1$, and $k_p^{\pm}, k_m^{\pm} \to 0$, equations (2.15a) and (2.15b) reduce to equations (2.4a) and (2.4b) but with $\nu = \eta = \zeta$.

Specification of $n(z)$, $B_0(z)$, and $v_{\rm A}(z)$ is non-trivial and depends on the problem to be studied. We leave the matter to another section and instead focus on mathematical features of the model.[4] Referring again to equation (2) and comparing with equations (2.15a) and (2.15b) we identify the following for **IRMHD**:

$$F^{(0)} = \Omega \tag{2.20a}$$

$$F^{(1)} = A, \tag{2.20b}$$

$$B^{(0)} = -[\phi, \Omega] + [A, J] \tag{2.21a}$$

$$B^{(1)} = -[\phi, A], \tag{2.21b}$$

$$D^{(0)} = -[U\partial_z \Omega - v_{\rm A}\partial_z J] \tag{2.22a}$$

$$D^{(1)} = -[U\partial_z A - v_{\rm A}\partial_z \phi], \tag{2.22b}$$

[4] Variation in z of B_0 is particularly problematic as it requires addressing the issue of magnetic flux conservation along z. At present, the current implementation of **IRMHD**is limited to cases where n and $v_{\rm A}$ may both vary in z, but B_0 remains constant.

$$G^{(0)} = -[k_p^- U\Omega - k_p^+ v_\mathrm{A} J] \tag{2.23a}$$
$$G^{(1)} = -[k_m^- U A - k_m^+ v_\mathrm{A} \phi], \tag{2.23b}$$

$$L_\perp^{(0)} = \zeta \boldsymbol{\nabla}_\perp^2 \Omega \tag{2.24a}$$
$$L_\perp^{(1)} = \zeta \boldsymbol{\nabla}_\perp^2 A, \tag{2.24b}$$

and,

$$C_{F^{(0)}} = \zeta \tag{2.25a}$$
$$C_{F^{(1)}} = \zeta, . \tag{2.25b}$$

As a third example, and to illustrate that CORONOS anticipates reduced models with a larger number of fields, we write the equations for the cold-ion limit of the 3D four-field equations as presented by Tassi et al. (2010). The Four-Field model was developed to generalize the equations of RMHD to include finite Larmor radius effects, the adiabatic electron limit, and plasma compressibility. (Hazeltine et al., 1985, 1987). The 3D model of Tassi et al. (2010), as written using the notation and conventions adopted for this document, and with dissipative terms added, is as follows:

$$\partial_t \Omega = -[\phi, \Omega] - [J, A] + \partial_z J + \nu \boldsymbol{\nabla}_\perp^2 \Omega \tag{2.26a}$$
$$\partial_t \mathscr{H} = -[\phi, \mathscr{H}] - d_\beta [A, Z] + \partial_z \phi - d_\beta \partial_z Z + \eta \boldsymbol{\nabla}_\perp^2 A \tag{2.26b}$$
$$\partial_t Z = -[\phi, Z] - c_\beta [V, A] + d_\beta [J, A] + c_\beta \partial_z V - d_\beta \partial_z J + \kappa \boldsymbol{\nabla}_\perp^2 Z \tag{2.26c}$$
$$\partial_t V = -[\phi, V] - c_\beta [Z, A] + c_\beta \partial_z Z + \mu \boldsymbol{\nabla}_\perp^2 V. \tag{2.26d}$$

The fields, ϕ,, Ω, A, and J, are all as defined above, and we have introduced the additional field quantities,

$$\mathscr{H} = A + d_e^2 J \tag{2.27a}$$
$$Z = \hat{\mathbf{z}} \cdot \mathbf{B}/c_\beta \tag{2.27b}$$
$$V = \hat{\mathbf{z}} \cdot \mathbf{V}, \tag{2.27c}$$

where $\mathbf{V}$ and $\mathbf{B}$ are the plasma velocity and magnetic fields respectively, and the shift to an equation for the quantity $\mathscr{H}$ arises from the inclusion

of electron inertia in the model. Some new parameters also appear in equations (2.27a)-(2.27c) as well as in equations (2.26a)-(2.26d). d_e and d_i are the electron and ion inertial lengths respectfully, and are related to one another according to,

$$d_e = \sqrt{\frac{m_e}{m_i}} d_i. \tag{2.28}$$

The quantities c_β and d_β are defined by,

$$c_\beta = \sqrt{\beta/(1+\beta)} \tag{2.29a}$$

$$d_\beta = d_i c_\beta, \tag{2.29b}$$

where $\beta = (5/3)P_0/B_0^2$, is plasma β to lowest order. Finally, in addition to the diffusivities already defined for RMHD we have a parallel thermal conductivity κ and parallel viscosity μ in equations (2.26c) and (2.26d) respectively.

As before, let us now compare the model given by equations (2.26a)−(2.26d) to equation (2). Now we identify:

$$F^{(0)} = \Omega \tag{2.30a}$$

$$F^{(1)} = \mathscr{H} \tag{2.30b}$$

$$F^{(2)} = Z \tag{2.30c}$$

$$F^{(3)} = V \tag{2.30d}$$

$$B^{(0)} = -[\phi, \Omega] - [J, A] \tag{2.31a}$$

$$B^{(1)} = -[\phi, \mathscr{H}] - d_\beta [A, Z] \tag{2.31b}$$

$$B^{(2)} = -[\phi, Z] - c_\beta [V, A] + d_\beta [J, A] \tag{2.31c}$$

$$B^{(3)} = -[\phi, V] - c_\beta [Z, A] \tag{2.31d}$$

$$D^{(0)} = \partial_z J \tag{2.32a}$$

$$D^{(1)} = \partial_z \phi - d_\beta \partial_z Z \tag{2.32b}$$

$$D^{(2)} = c_\beta \partial_z V - d_\beta \partial_z J \tag{2.32c}$$

$$D^{(3)} = c_\beta \partial_z Z \tag{2.32d}$$

$$L_\perp^{(0)} = \nu \boldsymbol{\nabla}_\perp^2 \Omega \tag{2.33a}$$

$$L_\perp^{(1)} = \eta \boldsymbol{\nabla}_\perp^2 A \tag{2.33b}$$

$$L_\perp^{(2)} = \kappa \boldsymbol{\nabla}_\perp^2 Z \tag{2.33c}$$

$$L_\perp^{(3)} = \mu \boldsymbol{\nabla}_\perp^2 V. \tag{2.33d}$$

and,

$$C_{F^{(0)}} = \nu \tag{2.34a}$$

$$C_{F^{(1)}} = \eta \tag{2.34b}$$

$$C_{F^{(2)}} = \kappa \tag{2.34c}$$

$$C_{F^{(3)}} = \mu. \tag{2.34d}$$

And we note that—as in the case of RMHD—the auxiliary terms $G^{(i)}$ are absent.

Chapter 3

Description Numerical Volume and Integration Methodology

Wherein we introduce the reader to the gist of this document

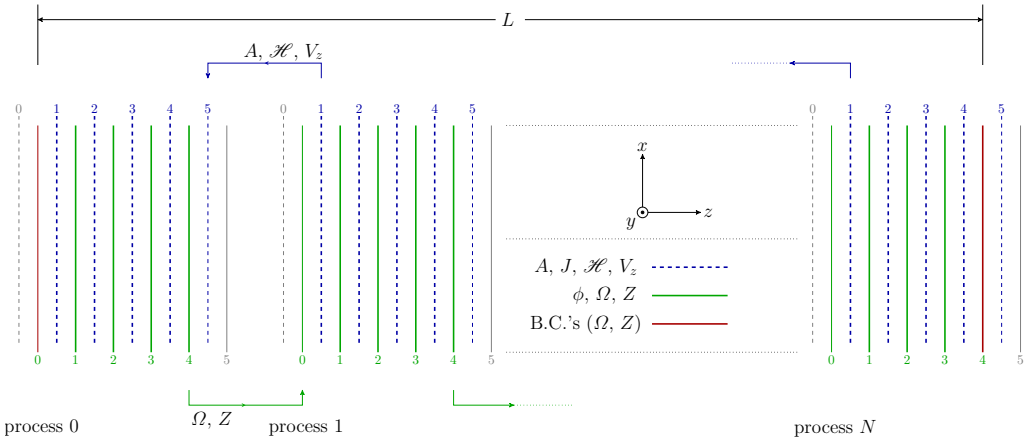


Figure 3.1: Domain decomposition for the case of a domain of length L along z, transverse side length of $L_\perp = 1$, with $N_z = 4$ principle layers per process, and N processes. Solid lines indicate layers of "mesh" M_1 while dashed lines indicate layers of mesh M_2 (See text). Currently, boundary conditions are specified on "ghost" layer 0 of process 0 and principle layer N_z of process $N - 1$ for fields defined for M_1. Field values on interior boundaries are shared across processes as indicated for finite differencing in z. Quantities specified on one grid but needed by the other are averaged across adjacent layers.

In this section, we discuss the specification of the numerical volume, its domain decomposition, and the discretization methodology used for all of the reduced models discussed above.

3.0.3 Numerical Volume

Figure 3.1 illustrates how the code divides the numerical volume among N processes, as well as how information is organized within each process. The volume is divided into a "stack of stacks." In what follows, I refer to a stack associated with a given process, as a "process-stack." Each process-stack consists of N_z principle layers, numbered 1 through N_z, as well as two additional "ghost" layers, numbered 0, and $N_z + 1$, at the "bottom" and "top" ends respectively of each process-stack. The figure illustrates this for the case $N_z = 4$. Note that each process-stack is actually a pair of interleaved stacks (M_1, M_2), (which I shall call "meshes") with (ϕ, Ω) (and perhaps Z) defined on M_1, and (A, J) [and perhaps $(\mathscr{H}, V)$] defined on M_2.

Taking the resolution of each layer to be $N_x \times N_y$, the discretization of any of the reduced models discussed above proceeds as follows for each process: We define the indices

$$n_x = 1, \ldots, N_x \tag{3.1a}$$

$$n_y = 1, \ldots, N_y \tag{3.1b}$$

$$n_z = 1, \ldots, N_z, \tag{3.1c}$$

$$\ell = N_x(n_x - 1) + n_y \tag{3.1d}$$

and consider points (x_ℓ, y_ℓ) located on any one of the layers of the process-stack, and with z-coordinate z_{n_z}. With these, we define the real-valued column-vector $\tilde{\mathscr{U}}_n$ to contain our discretized field values at time-step n in real Cartesian space. For example, for the field $F^{(i)}$ of some model, we discretize field values on the layers of a process-stack according to,

$$F^{(i)}(x, y, z, t) \to \tilde{\mathscr{U}}^{(i)}_{n,n_z,\ell} \equiv F^{(i)}\left(x_\ell, y_\ell, z_{n_z}, t_n\right) \tag{3.2}$$

Note that each such column vector contains N_R elements where,

$$N_R = N_x N_y (N_z + 2). \tag{3.3}$$

Next, we let the complex-valued column vector $\mathscr{U}^{(i)}_n$ contain the Fourier components for each of the N_z layers of $\tilde{\mathscr{U}}^{(i)}_n$ as determined by discrete,

transverse, Fast Fourier transform (FFT) which we symbolize as $\mathsf{F}_\perp\{\ldots\}$. Symbolically,

$$\mathscr{U}^{(i)}_{n,n_z,\lambda} = \mathscr{U}^{(i)}_{\lambda}(z_{n_z}, t_n) = \mathsf{F}_{\perp\lambda}\left\{\tilde{\mathscr{U}}^{(i)}_{n,n_z}\right\}. \tag{3.4}$$

I.e., at time $t = t_n$, the λ'th Fourier component for the values of the field $\tilde{\mathscr{U}}^{(i)}$ in layer n_z is given by $\mathscr{U}^{(i)}_{n,n_z,\lambda}$. The elements of $\mathscr{U}^{(i)}_{n,n_z}$ are specified with the index λ rather than ℓ because as a practical matter, the number of elements contained in the column vector $\mathscr{U}^{(i)}_n$ defined for a given process-stack is not equal to N_R. This is because $\tilde{\mathscr{U}}^{(i)}_n$ is real-valued, hence the FFT of each of its layers posesses the symmetry property,

$$\mathscr{U}^{(i)}_{n,n_z}\left(k_{x_p}, k_{y_q}\right) = \mathscr{U}^{(i)\,*}_{n,n_z}\left(-k_{x_p}, -k_{y_q}\right), \tag{3.5}$$

where "$*$" indicates complex conjugation, and where

$$k_{x_p} = \frac{2\pi p}{L_\perp} \qquad -\frac{N_x}{2} < p \leq \frac{N_x}{2} \tag{3.6a}$$

$$k_{y_q} = \frac{2\pi q}{L_\perp} \qquad -\frac{N_y}{2} < q \leq \frac{N_y}{2} \tag{3.6b}$$

are the discrete k_x, and k_y wave-number coordinates in transverse Fourier space. Because of this symmetry, it is standard memory-conserving practice when computing discrete real-to-complex FFT's to store only the positive wave-number portion of the result. For this reason, only $N_x(N_y/2+1)$ complex elements are needed per layer, so that $\mathscr{U}^{(i)}_n$ contains only N_C elements where,

$$N_C = N_x(N_y/2+1)(N_z+2). \tag{3.7}$$

Since, $\mathscr{U}^{(i)}_n$, analogously to $\tilde{\mathscr{U}}^{(i)}_n$, is understood to be a 1-dimensional column vector containing the discrete Fourier-space components of the field $F^{(i)}(t_n)$ specified for the N_z+2 layers of a process-stack, I define the index λ over the range,

$$1 < \lambda \leq N_x\left(N_y/2+1\right), \tag{3.8}$$

and refer to the coordinates (k_{x_p}, k_{y_q}) as simply $(k_{x_\lambda}, k_{y_\lambda})$.[1] Henceforth, except where otherwise noted, all complex-valued column vectors will be understood to be Fourier-space column vectors with N_c elements, with Fourier

[1] A technical matter not taken up here but relevant for implementation, is the ordering of the elements of $\mathscr{U}^{(i)}$ with respect to the wave-number coordinates $(k_{x_\lambda}, k_{y_\lambda})$. In what follows, whenever point-wise operations are indicated for elements of column vectors, it should be assumed that this ordering is accounted for.

components common to a layer ordered contiguously. Usually both the index λ and the index n_z may be suppressed without loss of clarity.

3.0.4 Model Discretization and Numerical Integration

Given the definitions and description provided in section 3.0.3, we may now outline the discretization of any of the reduced models discussed in section 2 and the method of integration used by CORONOS to evolve them in time.[2] This method is essentially that of (Longcope, 1993), as adapted to the formalism used here. When illustration is called for, we will use **IRMHD** as our model since for this model, every type of term appearing in equation (2) is represented.

We begin by considering the equation for the i'th field of some reduced model as given by,

$$\partial_t F^{(i)} = B^{(i)} + D_z^{(i)} + G^{(i)} + L_\perp^{(i)}. \tag{3.9}$$

We wish to adopt the spatial discretization of $F^{(i)}$ given by equation (3.2), as well as equivalent spatial discretizations of all the quantities on the right-hand-side, and apply a discrete transverse FFT over each layer of the result.

Before proceeding however, it is well to point out that, in general, equations of a model that evolve fields defined on M_1, require information about fields defined on M_2 and vice versa. This means that averaging must sometimes be done to obtain suitable values for these fields on the mesh corresponding to the quantity being evolved on that mesh. In what follows, we will indicate this with an overbar. However it's important to realize that averaging on M_1 proceeds differently from averaging on M_2. So to be clear let Q_1 be a quantity defined on M_1, and let Q_2 be a quantity defined on M_2. Then the meanings of $\bar{Q}_1$ and $\bar{Q}_2$ are as follows:

$$\bar{Q}_{1,i} = \frac{Q_{1,i} + Q_{1,i-1}}{2} \qquad \text{On } M_2 \tag{3.10}$$

$$\bar{Q}_{2,i} = \frac{Q_{1,i+1} + Q_{1,i}}{2} \qquad \text{On } M_1. \tag{3.11}$$

[2] We also remind the reader that the method described here is applicable in principle to any set of model equations that can be expressed in the form given by equation (2).

This subtlety must be considered again when discussing finite differences along z. For now though, let us proceed with the FFT.

We note that, in analogy to the continuous Fourier transform, the FFT has the property that for each layer n_z of some field column vector $\tilde{Q}_n$,

$$\mathsf{F}_\perp\left\{\partial_x \tilde{Q}_{n,n_z}\right\} = -i\mathbb{K}_x \odot Q_{n,n_z} \tag{3.12a}$$

$$\mathsf{F}_\perp\left\{\partial_y \tilde{Q}_{n,n_z}\right\} = -i\mathbb{K}_y \odot Q_{n,n_z}. \tag{3.12b}$$

In these expressions we have defined the column vectors $\mathbb{K}_x$ and $\mathbb{K}_y$ to contain the $N_x(N_y/2+1)$ Fourier wave-number coordinates k_{x_λ} and k_{y_λ} respectively, ordered appropriately according to storage conventions for the elements of Q_{n,n_z}. The operator $\odot$ is adopted to represent point-wise multiplication of the elements of these column vectors with the elements of Q_{n,n_z}, i.e.

$$\left[\mathbb{K}_x \odot Q_{n,n_z}\right]_\lambda = \mathbb{K}_{x,\lambda} Q_{n,n_z,\lambda} \tag{3.13a}$$

$$\left[\mathbb{K}_y \odot Q_{n,n_z}\right]_\lambda = \mathbb{K}_{y,\lambda} Q_{n,n_z,\lambda}, \tag{3.13b}$$

where there is no summation over the index λ implied. We note that equations (3.12a) and (3.12b) imply that,

$$\mathsf{F}_\perp\left\{\boldsymbol{\nabla}_\perp^2 \tilde{Q}_{n,n_z,\lambda}\right\} = -k_\lambda^2 Q_{n,n_z,\lambda}, \tag{3.14}$$

where,

$$k_\lambda^2 = k_{x_\lambda}^2 + k_{y_\lambda}^2. \tag{3.15}$$

If we were to now also adopt a first-order explicit finite difference scheme in time with time-step δt, we may write the result of the procedure above as,

$$\mathscr{U}_{n+1}^{(i,ex)} = \mathscr{U}_n^{(i)} + \delta t\left[\mathscr{B}_n^{(i)} + \mathscr{D}_n^{(i)} + \mathscr{G}_n^{(i)} - C_F^{(i)} k^2 \mathscr{U}_n^{(i)}\right]. \tag{3.16}$$

Strictly speaking, we have written this expression only for the (n_z,λ)'th element of $\mathscr{U}_{n+1}^{(i,ex)}$, however we may suppress these indices without loss of clarity in what follows. Later, we will write final expressions for the method in the more formal notation described above. The quantities $\mathscr{B}_n^{(i)}$, $\mathscr{D}_n^{(i)}$, and $\mathscr{G}_n^{(i)}$ are the Fourier-space equivalents of $B^{(i)}$, $D_z^{(i)}$, and $G^{(i)}$ respectively at

$t = t_n$, while the term $L_\perp^{(i)}$ has been handled using the result in equation (3.14).

As mentioned above, we use the case of **IRMHD** to illustrate these terms. First, for the Poisson bracket terms we have,

$$\mathscr{B}_n^{(0)} = -[\phi, \Omega]_n - [\bar{J}, \bar{A}]_n \tag{3.17a}$$

$$\mathscr{B}_n^{(1)} = -[\bar{\phi}, A]_n. \tag{3.17b}$$

Here, the bracketed quantities are understood to be the (n_z, λ) elements of the Fourier transforms of the results obtained from an evaluation of the bracket made using the "collocation" method.[3] In this method the required derivatives for the evaluation of the bracket are obtained in Fourier space making use of the property given by equations (3.12a) and (3.12b). These are then reverse transformed into real Cartesian space where they are combined as prescribed by the definition of the Poisson bracket given in equation (2.9). The result is then transformed back to Fourier space. For example, for the bracket $[X, Y]_{n,n_z}$, (now regarded as the section of a column vector corresponding to elements for layer n_z), we would evaluate the required real space derivatives as follows:

$$(\partial_x \tilde{X})_{n,n_z} = \mathsf{F}_\perp^{-1}\left\{-i\mathbb{K}_x \odot X_{n,n_z}\right\} \tag{3.18a}$$

$$(\partial_y \tilde{X})_{n,n_z} = \mathsf{F}_\perp^{-1}\left\{-i\mathbb{K}_y \odot X_{n,n_z}\right\} \tag{3.18b}$$

$$(\partial_x \tilde{Y})_{n,n_z} = \mathsf{F}_\perp^{-1}\left\{-i\mathbb{K}_x \odot Y_{n,n_z}\right\} \tag{3.18c}$$

$$(\partial_y \tilde{Y})_{n,n_z} = \mathsf{F}_\perp^{-1}\left\{-i\mathbb{K}_y \odot Y_{n,n_z}\right\}, \tag{3.18d}$$

where $\mathsf{F}_\perp^{-1}\{\ldots\}$ indicates reverse Fourier transform. Then,

$$[X, Y]_{n,n_z} = \mathsf{F}_\perp\left\{(\partial_y \tilde{X})_{n,nz} \odot (\partial_x \tilde{Y})_{n,n_z} - (\partial_x \tilde{X})_{n,nz} \odot (\partial_y \tilde{Y})_{n,n_z}\right\}. \tag{3.19}$$

The second term inside the brackets of equation (3.16) represents all terms containing partial derivatives in z. For the case of **IRMHD**, we would write,

$$\mathscr{D}_{n,n_z}^{(0)} = v_{\mathrm{A}}(z_{n_z})\left(\frac{\Delta J}{\Delta z}\right)_{n,n_z} - U(z_{n_z})\left(\frac{\Delta \Omega}{\Delta z}\right)_{n,n_z} \tag{3.20a}$$

$$\mathscr{D}_{n,n_z}^{(1)} = v_{\mathrm{A}}(z_{n_z})\left(\frac{\Delta \phi}{\Delta z}\right)_{n,n_z} - U(z_{n_z})\left(\frac{\Delta A}{\Delta z}\right)_{n,n_z}, \tag{3.20b}$$

[3] Also, where overbars are indicated, the averages are calculated prior to the evaluation of the bracket.

Where we have made the index n_z explicit in equations (3.24a) and (3.24b) because of the z-dependence of the quantities $v_{\rm A}$ and U.

We've previously pointed out that choosing some fields to be defined on M_1 and others to be defined on M_2 requires averaging when information on one of the meshes is needed on the other. This subtletly arises again when choosing our discretization for the finite difference. We may infer from figure 3.1 how second-order spatial finite differencing in z is accomplished. When, at some time-step n, a grid-layer located at, say z_{n_z}, requires the evaluation of a finite difference of some field quantity X_n, the finite difference is carried out according to one of the following,

$$\left(\frac{\Delta X_2}{\Delta z}\right)_n = \left(\frac{X_{1,n,n_z+1}-X_{1,n,n_z}}{z_{1,n_z+1}-z_{1,n_z}}\right) \tag{3.21a}$$

$$\left(\frac{\Delta X_1}{\Delta z}\right)_n = \left(\frac{X_{2,n,n_z}-X_{2,n,n_z-1}}{z_{2,n_z}-z_{2,n_z-1}}\right) \tag{3.21b}$$

Note that indices are provided for both X and z to indicate both the layer on a mesh (i.e., n_z) as well as the particular mesh upon which information about the quantity the quantity is needed. We can simplify this somewhat by noting that although the layers of M_1 are always *logically* staggered with respect to those of M_2, *they are not necessarily spatially staggered*. If the layers are distributed uniformly in z, it follows that,

$$\Delta z = (z_{n_z+1} - z_{n_z}) = (z_{n_z} - z_{n_z-1}), \tag{3.22}$$

so that the denominators above may henceforth be written simply as Δz.

From the point of view of second-order accuracy however, the distinction between grids is still important. When finite differences involving X *defined on* M_2 are needed *on layer* n_z *of* M_1 they are evaluated using equation (3.21a), and when finite differences *involving* X *defined on* M_1 are needed *on layer* n_z *of* M_2 they are evaluated using equation (3.21b). These are straightforward when the finite difference in question involves a field X which is defined on the "complementary" grid as just described. For example, the finite difference in J needed for the update, $\mathscr{U}^{(0)}_{n+1}$, is readily evaluated since $\mathscr{U}^{(0)}_{n+1}$ is defined on M_1 while J_{n,n_z+1} and J_{n,n_z} are both defined on M_2.

For **RMHD**, (and for **IRMHD** in the absence of a mean flow) this is the only finite difference required for the vorticity equation. However, when

there is a mean flow, the vorticity equation also requires the evaluation of a finite difference in Ω which is not defined on layers n_z and n_z+1 of M_2. To get around this problem, we calculate the average of Ω across the pair of layers (n_z-1, n_z) on grid M_1 to obtain suitable value of Ω_{n,n_z} on M_2 and across the pair of layers (n_z, n_z+1) on M_1 to obtain a suitable value for Ω_{n,n_z+1} on M_2. Then we can calculate the finite difference as above. A similar averaging must be done for the finite difference in A needed for the $\mathscr{U}_{i=1,n+1}$ update when there is a mean flow. The averaging is across the same set of layer-pairs (n_z-1, n_z) and (n_z, n_z+1), but these now result in G_1-defined values for A_{n,n_z-1} and A_{n,n_z} respectively. Whether on G_1 or G_2 these averagings lead to a finite difference of the form,

$$\left(\frac{\Delta \bar{X}}{\Delta z}\right)_n = \left(\frac{X_{n_z+1} - X_{n_z-1}}{2\Delta z}\right). \tag{3.23}$$

I.e. where an overbar appears above a field quantity in a finite difference, the finite difference must be evaluated using equation (3.23), and when the overbar is absent then we must use one of equations (3.21a) or (3.21b), with the choice depending on upon which mesh the field quantity being differenced is defined.

With this understood, it is best to now rewrite equations (3.24a) and (3.24b) as,

$$\mathscr{D}^{(0)}_{n,n_z} = v_{\rm A}(z_{n_z})\left(\frac{\Delta J}{\Delta z}\right)_{n,n_z} - U(z_{n_z})\left(\frac{\Delta \bar{\Omega}}{\Delta z}\right)_{n,n_z} \tag{3.24a}$$

$$\mathscr{D}^{(1)}_{n,n_z} = v_{\rm A}(z_{n_z})\left(\frac{\Delta \phi}{\Delta z}\right)_{n,n_z} - U(z_{n_z})\left(\frac{\Delta \bar{A}}{\Delta z}\right)_{n,n_z}, \tag{3.24b}$$

The third term inside the brackets of equation (3.16) collects terms with linear dependence on the fields themselves,

$$\mathscr{G}^{(0)}_{n,n_z} = U(z_{n_z})k_p^-(z_{n_z})\Omega_{n,n_z} - v_{\rm A}(z_{n_z})k_p^+(z_{n_z})J_{n,n_z} \tag{3.25a}$$

$$\mathscr{G}^{(1)}_{n,n_z} = U(z_{n_z})k_m^-(z_{n_z})A_{n,n_z} - v_{\rm A}(z_{n_z})k_m^+(z_{n_z})\phi_{n,n_z} \tag{3.25b}$$

Once again, we note that because the quantity-pairs mentioned above are defined for separate grids, the evaluation of the $\mathscr{G}^{(i)}_{n,n_z}$-terms must be considered carefully. We shall discuss this at greater length below as well.

Finally, the fourth term inside the brackets of equation (3.16) simply represents the Fourier transform of the dissipative terms. We take a moment

to note that in this term as well as in the term outside the brackets of equation (3.16), and in what follows; when one sees "$\mathscr{U}_n^{(i)}$" without any other superscripts, it refers simply to the values of the discretized fields, for all n_z and λ, at time-step n which are presumed known when commencing time-step $n+1$.

With these same definitions, we can also write the semi-implicit form,

$$\mathscr{U}_{n+1}^{(i,im)} = \mathscr{U}_n^{(i)} + \delta t \left[\mathscr{B}_n^{(i)} + \mathscr{D}_n^{(i)} + \mathscr{G}_n^{(i)} - C_F^{(i)} k^2 \mathscr{U}_{n+1}^{(i)} \right]. \tag{3.26}$$

For the time-step in δt, I now introduce the free, empirically-chosen, parameters $s_i \in (0,1)$—one for each component of $\mathscr{U}$—and use them to write a new expression for the $\mathscr{U}_{n+1}^{(i)}$ as a hybrid of the explicit and semi-implicit finite differences given above:

$$\mathscr{U}_{n+1}^{(i)} = s_i \mathscr{U}_{n+1}^{(i,im)} + (1-s_i)\mathscr{U}_{n+1}^{(i,ex)}. \tag{3.27}$$

using the expressions given in equations (3.16) and (3.26) for the $\mathscr{U}_i$'s, this becomes, after some manipulation,

$$\mathscr{U}_{n+1}^{(i)} = \left\{ \left[1 - (1-s_i)\, C_F^{(i)} \delta t k^2 \right] \mathscr{U}_n^{(i)} + \delta t \left[\mathscr{B}_n^{(i)} + \mathscr{D}_n^{(i)} + \mathscr{G}_n^{(i)} \right] \right\} \left[1 + s_i C_F^{(i)} \delta t k^2 \right]^{-1} \tag{3.28}$$

To ensure second-order accuracy in time, the scheme applies this differencing twice; once with time step $\delta t = f\Delta t \equiv \Delta t_p$ where $f \in (0,1)$, and using evaluations of the quantities $\mathscr{B}$, $\mathscr{D}$, and $\mathscr{G}$ based on the field values obtained at the end of time-step n; and once again for the full time step $\delta t = \Delta t$, this time using evaluations of the quantities $\mathscr{B}$, $\mathscr{D}$, and $\mathscr{G}$ based on field values obtained from the predictor step. We distinguish these latter evaluations from the former evaluations with a subscripted "(p)" and write,

$$\mathscr{U}_{(p),n+1}^{(i)} = \left\{ \left[1 - (1-s_i)\, C_F^{(i)} \Delta t_p k^2 \right] \mathscr{U}_n^{(i)} + \Delta t \left[\mathscr{B}_n^{(i)} + \mathscr{D}_n^{(i)} + \mathscr{G}_n^{(i)} \right] \right\} \left[1 + s_i C_F^{(i)} \Delta t_p k^2 \right]^{-1} \tag{3.29a}$$

$$\mathscr{U}_{(c),n+1}^{(i)} = \left\{ \left[1 - (1-s_i)\, C_F^{(i)} f \Delta t k^2 \right] \mathscr{U}_n^{(i)} + \Delta t \left[\mathscr{B}_{(p),n}^{(i)} + \mathscr{D}_{(p),n}^{(i)} + \mathscr{G}_{(p),n}^{(i)} \right] \right\} \left[1 + s_i C_F^{(i)} f \Delta t k^2 \right]^{-1} \tag{3.29b}$$

These expressions are valid in transverse Fourier space for some point $(k_{x_\lambda}, k_{y_\lambda})$ on a layer of the domain located at $z = z_{n_z}$. To write them in a way that accounts for all points on a layer, and for all layers, let us define vectors,

$$S^{(ex,i)}_{(p)} = [\mathbb{1} - g^{(ex,i)}_p \mathbb{K}^{(2)}] \tag{3.30a}$$

$$S^{(im,i)}_{(p)} = [\mathbb{1} + g^{(im,i)}_p \mathbb{K}^{(2)}]^{-1} \tag{3.30b}$$

$$S^{(ex;(i)}_{(c)} = [\mathbb{1} - g^{(ex,i)}_c \mathbb{K}^{(2)}] \tag{3.30c}$$

$$S^{(im;(i)}_{(c)} = [\mathbb{1} + g^{(im,i)}_c \mathbb{K}^{(2)}]^{-1}, \tag{3.30d}$$

with,

$$g^{(ex,i)}_p = (1 - s_i) C^{(i)}_F \Delta t_p \tag{3.31a}$$

$$g^{(im,i)}_p = s_i C^{(i)}_F \Delta t_p \tag{3.31b}$$

$$g^{(ex,i)}_c = (1 - s_i) C^{(i)}_F \Delta t \tag{3.31c}$$

$$g^{(im,i)}_c = s_i C^{(i)}_F \Delta t, \tag{3.31d}$$

and where,

$$\mathbb{K}^{(2)}_{(n_z-1)N_x(N_y/2+1)+\lambda} = k^2_\lambda; \qquad 0 \le n_z \le N_z + 2. \tag{3.32}$$

That is, $\mathbb{K}^{(2)}$ is a 1-dimensional column vector with N_C elements, containing $N_z + 2$ contiguous "copies" of the column vector containing the quantities k^2_λ. Also we have defined $\mathbb{1}$ to be a column vector of the same size as $\mathbb{K}^{(2)}$ with all elements set to unity. Note that these definitions suffice to specify dimensions of the column vectors defined in equations (3.30a)−(3.30d) as well.

Now, we may express the entire scheme for all layers of a process-stack and for the $(n + 1)$'th time-step very compactly using our notation for point-wise multiplication of column-vectors:

(3.33a)

$$U^{(i)}_{(p),n+1} = \left\{ S^{(ex,i)}_{(p)} \odot U^{(i)}_n + \Delta t_p \left[\mathscr{B}^{(i)}_n + \mathscr{D}^{(i)}_n + \mathscr{G}^{(i)}_n \right] \right\} \odot S^{(im,i)}_{(p)}$$

(3.33b)

$$U^{(i)}_{(c),n+1} = \left\{ S^{(ex,i)}_{(c)} \odot U^{(i)}_n + \Delta t \left[\mathscr{B}^{(i)}_{(p),n} + \mathscr{D}^{(i)}_{(p),n} + \mathscr{G}^{(i)}_{(p),n} \right] \right\} \odot S^{(im,i)}_{(c)},$$

As indicated by the figure, each of M_1, and M_2 consists of a total of $\texttt{p3}+2$ layers per process. The "principle" layers of each grid are those with indices ranging from 1 to `p3`, while layers 0 and $\texttt{p3}+1$ serve special purposes. These "ghost" layers are for the imposition of boundary conditions in the case of processes with exterior boundaries as well as for communication across processes when interior boundary layers of one grid require information from an adjacent layer of the other grid. This communication is as indicated in the figure. Note that each grid has a single extraneous layer. For G_1 this layer is at the top of the grid, while for G_2 it is at the bottom. In the figure they are colored gray to indicate that they are superfluous. These layers are present merely to indicate the manner in which the field-carrying vector $\mathscr{U}$ is implemented in the code. (Currently the code makes no use of this "wasted" memory, though, with some care taken, it could be made to.)

The relevant ghost layer on G_1 for any process occurs at the bottom. For the process of rank $r_p = 0$ this layer is an exterior boundary and is used for setting boundary conditions. For all other processes this layer receives its field values from layer `p3` of the G_1 grid instantiated on the process of rank $r_p - 1$ as indicated in the figure. In contrast, the relevant ghost layer on G_2 occurs at the top. Though in principle it could be used for setting an upper exterior boundary condition on the G_2 grid of process $N-1$, in its current form the code in fact sets an upper boundary condition on layer 4 of G_1—i.e. the exterior boundary conditions at both ends are conditions on the field Ω (or equivalently ϕ). In the figure, the layers upon which exterior boundary conditions are imposed are colored red.

Chapter 4

Initial Conditions and Boundary Conditions

Wherein we introduce the reader to the gist of this document

Chapter 5

An Illustration: Modeling Magnetic Reconnection in Homogeneous Reduced MHD

Wherein we introduce the reader to the gist of this document

Chapter 6

Reduced Model Conservation Laws and Post-Processing Diagnostics

Wherein we introduce the reader to the gist of this document

Chapter 7

Suggested Protocols for Adapting and Expanding CORONOS

Wherein we introduce the reader to the gist of this document

Chapter 8

Remarks On The Current State of CORONOS and Essential Tasks Remaining.

Wherein we introduce the reader to the gist of this document

Appendix A

Appendices

Wherein we introduce the reader to the gist of this document

A.1 A

A.1.1 A i

A.1.2 A ii

A.2 B

Bibliography

Hazeltine, R. D., Hsu, C. T., and Morrison, P. J. (1987). Hamiltonian four-field model for nonlinear tokamak dynamics. *Physics of Fluids*, 30:3204–3211.

Hazeltine, R. D., Kotschenreuther, M., and Morrison, P. J. (1985). A four-field model for tokamak plasma dynamics. *Physics of Fluids*, 28:2466–2477.

Heinemann, M. and Olbert, S. (1980). Non-WKB Alfven waves in the solar wind. *Journal of Geophysical Research*, 85:1311–1327.

Hess, S. L. G., Delamere, P., Dols, V., Bonfond, B., and Swift, D. (2010). Power transmission and particle acceleration along the Io flux tube. *Journal of Geophysical Research (Space Physics)*, 115:A06205.

Kadomtsev, B. B. and Pogutse, O. P. (1974). Nonlinear helical perturbations of a plasma in the tokamak. *Soviet Journal of Experimental and Theoretical Physics*, 38:575–589.

Longcope, D. W. (1993). *Theoretical studies of magnetohydrodynamic equilibria and dynamics of a solar coronal loop*. PhD thesis, Cornell Univ., Ithaca, NY.

Ng, C. S., Dennis, T. J., and Lin, L. (2017). From the Parker Model to Turbulent Heating of the Solar Corona. *In Preparation*.

Ng, C. S., Lin, L., and Bhattacharjee, A. (2012). High-Lundquist Number Scaling in Three-dimensional Simulations of Parker's Model of Coronal Heating. *The Astrophysical Journal*, 747:109.

Ng, C. S. and Ragunathan, S. (2011). High Lundquist Number Resistive MHD Simulations of Magnetic Reconnection: Searching for Secondary Island Formation. In Pogorelov, N. V., Audit, E., and Zank, G. P., editors, *5th International Conference of Numerical Modeling of Space Plasma Flows (ASTRONUM 2010)*, volume 444 of *Astronomical Society of the Pacific Conference Series*, page 124.

Perez, J. C. and Chandran, B. D. G. (2013). Direct Numerical Simulations of Reflection-driven, Reduced Magnetohydrodynamic Turbulence from the Sun to the Alfvén Critical Point. *The Astrophysical Journal*, 776:124.

Strauss, H. R. (1976). Nonlinear, three-dimensional magnetohydrodynamics of noncircular tokamaks. *Physics of Fluids*, 19:134–140.

Strauss, H. R. (1977). Dynamics of high beta Tokamaks. *Physics of Fluids*, 20:1354–1360.

Tassi, E., Morrison, P. J., Grasso, D., and Pegoraro, F. (2010). Hamiltonian four-field model for magnetic reconnection: nonlinear dynamics and extension to three dimensions with externally applied fields. *Nuclear Fusion*, 50(3):034007.

Velli, M., Grappin, R., and Mangeney, A. (1989). Turbulent cascade of incompressible unidirectional Alfven waves in the interplanetary medium. *Physical Review Letters*, 63:1807–1810.

Whang, Y. C. (1980). A magnetohydrodynamic model for corotating interplanetary structures. *Journal of Geophysical Research*, 85:2285–2295.