Introduction to FARGO3D

Pablo Benitez-Llambay / pbllambay@nbi.ku.dk

FARGO3D is a versatile HD/MHD code that runs on clusters of CPUs or GPUs, developed with special emphasis on protoplanetary disks. However, it can be used for general problems as well. The code is written in pure C, and the GPU routines ("kernels") are generated at compilation time, so usually you don't need to have knowledge about CUDA to extend its basic functionality.

The documentation is a very good starting point. It contains several examples and tutorials that may help when starting your own project with code. Today we will do the first run, and then we will use the code to measure the linear growth rate of the magnetorotational instability.

First run

In this exercise, you are going to learn the basis of FARGO3D by running the default setup, called `fargo`. It consists of a disk orbiting a solar-like star with a Jupiter-like planet embedded on it. The equations are solved on a 2D polar mesh ($r-\varphi$). The planet excites spiral density wakes on the disk and, because the planet is massive, will start to carve a gap around its orbit. The suggested tasks are the following:

1) Download the source files from fargo.in2p3.fr

2) Extract the file `fargo3d-1.2.tar.gz`:

```
$>tar -xvf fargo3d-1.2.tar.gz
```

3) Go to the main directory:

```
$>cd fargo3d-1.2
```

4) Build the code (compile the sources and link them to generate the executable, called `fargo3d`):

```
$>make SETUP=fargo
```

Note for Mac users: add the option: `FARGO_ARCH=MacIntel`

If you succeed, the following will be shown:

```
All objects are OK. Linking stage
FARGO3D SUMMARY:
===============
This built is SEQUENTIAL. Use "make para" to change that
SETUP:      'fargo'
(Use "make SETUP=[valid_setup_string]" to change set up)
(Use "make list" to see the list of setups implemented)
(Use "make info" to see the current sticky build options)
```

Optional: add the compilation options `PARALLEL=1` if you want to use `MPI` (an MPI distribution, such as OpenMPI, must be installed; see the manual for more info) or `GPU=1` if you want to run on a GPU (CUDA must be installed; see the manual for more info). If you have access to multiple GPUs, you can mix these two flags.

5) Run the simulation:

```
$>./fargo3d setups/fargo/fargo.par
```

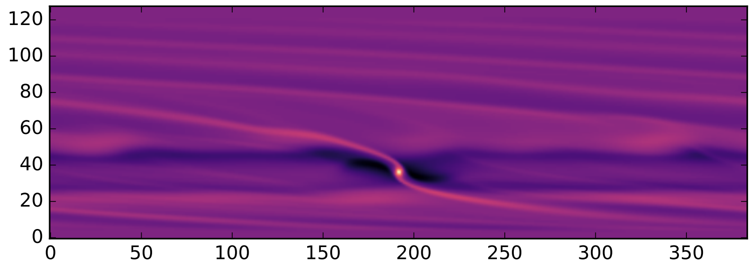
You will see the following:

```
Found 0 communicators
Standard version with no ghost zones in X
OUTPUTS 0 at date t = 0.000000 OK
..........
..........
..........
```

Wait for at least 10 orbits.

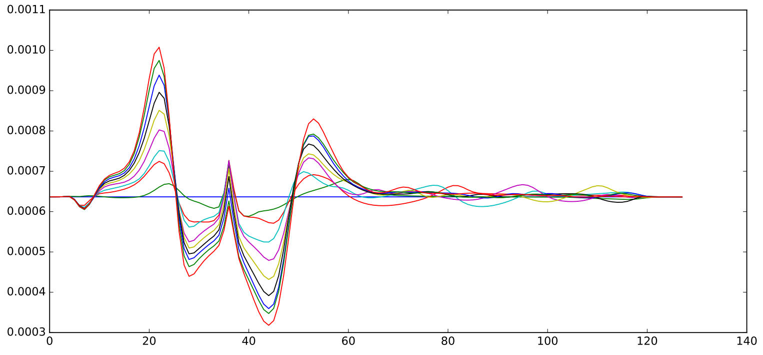
6) Go to the `outputs` directory and plot the density. It is stored in a file called `gasdens#.dat`:

```
$> cd outputs/fargo/
$> ipython
In [1]: from pylab import *
In [2]: sigma = fromfile("gasdens10.dat").reshape(384,128)
In [3]: imshow(log(sigma),origin=’lower’)
In [4]: show()
```



7) Now close the window. Next, plot the mean density:

```
for i in range(10):
  sigma = fromfile("gasdens{0:d}.dat".format(i)).reshape(128,384).mean(axis=1)
  plot(sigma)
  show()
```



8) Play with the parameters file `setups/fargo/fargo.par`). Try changing the number of cells, the aspect-ratio, the radial extension of the disk, etc...

I recommend that you have a look at the directories of the code and the output files. I also suggest that you visit the official documentation at fargo.in2p3.fr/Documentation. You can also have a look at the two ipython notebooks stored there.

MEASURING THE MRI GROWTH RATES

This exercise covers how to do something less trivial with the code. You will learn how to use FARGO3D to measure the growth rate of the magnetorotational instability (MRI) and compare to the values predicted by linear perturbation analysis. The methods and results of this exercise not only test the correctness of the solvers implemented in FARGO3D but also serve as a starting point for studying the nonlinear regime of the MRI.

Background

The MRI can be derived using the shearing-sheet approximation, which consists of a local expansion of the equations around a fiducial radius in a rotating frame. This treatment allows us to neglect the curvature terms and simplify the equations. After linearizing the Navier-Stokes equations in the incompressible limit assuming a constant magnetic field along the vertical direction z, neglecting vertical perturbations and writing all the perturbations in Fourier space ($\delta \propto \hat{\delta} e^{i k_n z}$ with n an integer number), we get the following set of equations (for the perturbations):

Fourier components of the linearized equations

$$\begin{aligned}
\frac{\partial \delta v_y}{\partial t} &= 2\Omega_0 \delta v_x + i\hat{k}_n \delta b_y,\\
\frac{\partial \delta v_x}{\partial t} &= -\frac{\Omega_0}{2}\delta v_y + i\hat{k}_n \delta b_x,\\
\frac{\partial \delta b_y}{\partial t} &= i\hat{k}_n \delta v_y,\\
\frac{\partial \delta b_x}{\delta t} &= -\frac{3}{2}\Omega_0 \delta b_y + i\hat{k}_n \delta v_\varphi,
\end{aligned}$$

where $\delta b_{x,y} = B_{x,y}/\sqrt{\mu_0 \rho}$ and $\hat{k}_n = k_n v_A$, with $v_A = B_z/\sqrt{\mu_0\rho}$ the Alfvén speed. For simplicity, we have omitted the "hat" symbols, but remember that the perturbations correspond to the Fourier components (which are in general complex quantities).

Defining $\vec{\delta} \equiv [\delta v_r, \delta v_\varphi, \delta b_r, \delta b_\varphi]$, the system can be written as:

$$\frac{1}{\Omega_0}\frac{\partial \vec{\delta}}{\partial t} = L\vec{\delta},$$

with L given by:

$$L = \begin{bmatrix} 0 & 2 & i\tilde{k}_n & 0 \\ -1/2 & 0 & 0 & i\tilde{k}_n \\ i\tilde{k}_n & 0 & 0 & 0 \\ 0 & i\tilde{k}_n & -3/2 & 0 \end{bmatrix},$$

and $\tilde{k} = \hat{k}/\Omega_0$.

The dispersion relation is found by computing the eigenvalues σ_i of L. The characteristic equation $(\det(L-\sigma I) = 0)$ is:

$$\sigma_n^4 + \left(2\tilde{k}_n^2 + 1\right)\sigma_n^2 + \tilde{k}_n^2\left(\tilde{k}_n^2 - 3\right) = 0.$$

This is a parabola in σ_n^2 and the roots are:

$$\sigma_{1,2,3,4} = \pm\left[-\frac{1+2\tilde{k}_n^2}{2} \pm \frac{1}{2}\sqrt{1+16\tilde{k}_n^2}\right]^{1/2}.$$

One of these eigenvalues is real and positive (under conditions that you will determine later), leading to the MRI instability:

Unstable eigenvalue

$$\sigma_n = \left[-\frac{1+2\tilde{k}_n^2}{2} + \frac{1}{2}\sqrt{1+16\tilde{k}_n^2} \right]^{1/2}$$

Once we have the eigenvalues of the operator L, the linear solution of the system can be easily obtained and written as:

Analytical solution for the linear growth

$$\begin{aligned}
\delta v_y(z,t) &= \frac{A}{N}\sigma_n \sin(k_n z) e^{\gamma_n t} \\
\delta v_x(z,t) &= \frac{A}{N}\frac{\sigma_n^2 + \tilde{k}_n^2}{2} \sin(k_n z) e^{\gamma_n t} \\
\delta b_y(z,t) &= \frac{A}{N}\tilde{k}_n \cos(k_n z) e^{\gamma_n t} \\
\delta b_x(z,t) &= -\frac{A}{N}\frac{2\tilde{k}_n \sigma_n}{\sigma_n^2 + \tilde{k_n}^2} \cos(k_n z) e^{\gamma_n t}
\end{aligned}$$

where $\gamma_n = \sigma\Omega_0$ is the growth rate of the instability and N is the normalization factor for the eigenvectors, given by $N = \sqrt{\sigma_n^2 + \left(\frac{\sigma_n^2+\tilde{k}_n^2}{2}\right)^2 + \tilde{k}_n^2 + \left(\frac{2\tilde{k}_n^2\sigma_n}{\sigma_n^2+\tilde{k}_n^2}\right)^2}$. The initial amplitude of the perturbations is given by A.

Exercise 1

Let's do some calculations:

1) Consider $\tilde{k}_n > 0$ and plot the dispersion relation for the unstable eigenvalue:

$$\sigma_n = \left[-\frac{1+2\tilde{k}_n^2}{2} + \frac{1}{2}\sqrt{1+16\tilde{k}_n^2} \right]^{1/2}$$

This plot shows the dimensionless growth rate $\sigma_n = \gamma_n/\Omega_0$ as a function of the dimensionless wave number $\tilde{k}_n = k_n v_a/\Omega_0$.

2) Which are the distinctive properties of this plot?

3) Show that the maximum growth rate occur at $\tilde{k}_{\rm max} = \sqrt{15}/4$ and prove that it is equal to $\sigma_{\rm max} = 3/4$.

4) Show that there exist a cut-off at $\tilde{k}_{\text{cut-off}} = \sqrt{3}$.

Exercise 2

Now that you understand the main features of this instability, we will recover them by solving the full set of nonlinear MHD equations. The goal is to use FARGO3D to build the dispersion relation, exciting the instability mode by mode and measuring the corresponding growth rate.

This is the exercise

Using FARGO3D, measure the linear growth rate of the MRI for different $\tilde{k}_n$ modes.

Reference guide:

To simplify the task, I'm providing you the setup called `mri-growth-sb`.

1) Extract the file `mri-growth-sb.tar.gz`

```
$>tar -xvf mri-growth-sb.tar.gz
```

Be sure that you are placed in right directory. Now we have to prepare and build the code.

2) Move the folder `mri-growth-sb` inside the `setups` directory:

```
$>mv Downloads/mri-growth-sb /home/user/fargo3d-1.2/setups/.
```

3) Go to the main directory and compile the setup:

```
$>make SETUP=mri-growth-sb
```

Remember that you can do it with `PARALLEL=1` or `GPU=1`.

Important note:

At this point, it could be nice to inspect some files included in the setup. For example, the file `condinit.c` contains the initial conditions of the problem. This is how any setup is normally initialized in FARGO3D.

The file `mri-growth-sb.par` is the parameter file, and allows you to modify the parameters of the problem. In this exercise, we will modify the vertical extension of the box `Zmax`, but if you have energy and time, you can also play with the amplitude of the perturbations, the radial (y) extension of the box and the number of cells `Ny` and `Nz`.

In the file `condinit.c`, the eigenvectors are initialized proportional to $\cos[2\pi z/\Delta Z]$, with $\Delta Z = Z_{\max} - Z_{\min}$. This is equivalent to taking $k_n = 2\pi/\Delta Z$. So, in order to change the k_n-modes, it is enough to change the vertical size of the box.

The size ΔZ_n of the box is related to the k_n-modes by the following relations:

$$k_n = \frac{2\pi}{\Delta Z_n} = \frac{\Omega_0 \tilde{k}_n}{v_A} \Longrightarrow \Delta Z_n = \frac{2\pi v_A}{\Omega_0 \tilde{k}_n}.$$

With the default values provided in the setup, $v_A \simeq 3.1 \times 10^{-3}$, so the maximum growth rate occur (at $\tilde{k}_{\max} = \sqrt{15}/4$) at:

$$\Delta Z_{\max} \simeq 0.0205,$$

which is the default `Zmax` value in the parameters file.

4) Run the simulation:

```
$>./fargo3d setups/mri-growth-sb/mri-growth-sb.par
```

you will see the following lines:

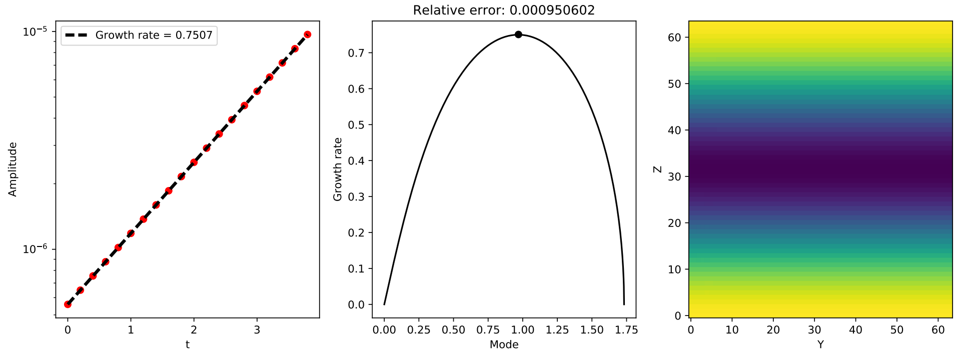
```
=============================================
You are computing the mode ktilde = 0.969228
=============================================

Found 2 communicators
Standard version with no ghost zones in X
OUTPUTS 0 at date t = 0.000000 OK
....................................
....................................
OUTPUTS 1 at date t = 0.200000 OK
....................................
```

Once the code is finished, it is time to measure the growth rate for the mode ($\sigma \simeq 3/4$ in this case). The idea consists in loading some scalar field, for example, the y component of the magnetic field, get the maximum value of it over the box and make a plot of the *maximum value* vs *time*. Since every field is supposed to have an exponential growth rate, you can take the logarithm of the maximum:

$$\log\left(\delta b_y{}^{\max}\right) \propto \gamma t + \text{constant}.$$

Plotting $\log\left(\delta b_y{}^{\max}\right)$ vs t and fitting a straight-line to it will allow you to get the growth rate. In particular, the slope of this line is the desired growth rate.

To measure it you can use the provided script, called `check_mode.py`, or you can use it as a reference to develop your own script. Here is an example:



Checking your understanding...

Why do we take the maximum value? Should the method work with any other arbitrary value?

5) Complete the curve with all the other modes. Modify the parameter file with the following suggested `Zmax` values:

`Zmax` ∈ [0.3974, 0.0911, 0.0514, 0.0358, 0.0274 , 0.0223, 0.0188, 0.0162, 0.0142 , 0.0127].

For each `Zmax` remember to change the `OutputDir` parameter to something like `outputs/mode1`, `outputs/mode2`, etc... so you can save the temporal evolution of each mode.

6) Plot everything together and compare with the analytical growth rate.

You may have a look at the script `run-modes.py`. If it does not plot anything, in the base FARGO3D directory (`fargo3d-1.2`) deactivate the existing matplotlib preferences file:

```
$>mv matplotlibrc matplotlibrc.backup
```

7) Investigate what happens when random noise is used to initialize the modes. "You can modify the default perturbations by replacing a few lines in `condinit.c` with:

```
vx[l] += AMPLITUDE*(-1.0+2.0*drand48());
vy[l]  = AMPLITUDE*(-1.0+2.0*drand48());
bx[l] += 0;
by[l]  = 0;
```

Note that we do not perturb the magnetic field to satisfy the “solenoidal condition”. Re-build the code and run the previous simulations again. Try increasing increasing the y-size of the box by changing the `Ymin` and `Ymax` parameters. What can you conclude from this experiment?

8) The previous analytical derivation relies on the *linear* equations. When are these equations valid?

Final note: when the modes are evolved for a long time there exists a saturation regime at which the fields do not increase anymore. At this point the dynamics becomes very complicated and interesting features are visible when you plot any scalar field (e.g. the y-component of the magnetic field). This is the non-linear regime of the MRI which breaks into what we call the MRI turbulence.