1 Pencil-code

In the folder 'pencil-code', find two run folders:

1. nonhel256a : simulation of small scale dynamo (SSD) or fluctuation dynamo
2. hel32a : simulation of large scale dynamo (LSD) or mean field dynamo

The equations being solved :

$$\begin{aligned}\frac{D\ ln\rho}{Dt} &= -\nabla\cdot\mathbf{u}\\ \frac{D\ \mathbf{u}}{Dt} &= -c_s^2\nabla\ ln\rho + \frac{\mathbf{J}\times\mathbf{B}}{\rho} + F_{visc} + f\\ \frac{\partial\mathbf{A}}{\partial t} &= \mathbf{u}\times\mathbf{B} + \eta\nabla^2\mathbf{A}\end{aligned}$$

$$F_{visc} = \nu\left[\nabla^2\mathbf{u} + \frac{1}{3}\nabla\cdot\nabla\mathbf{u} + 2S\cdot\nabla ln\rho\right]$$

$$f(\mathbf{x},t) = Re\{N f_{\mathbf{k}(t)}\ \ exp\,[i\mathbf{k}(t)\cdot\mathbf{x} + i\phi(t)]\}$$

For SSD, the forcing term $f(\mathbf{x},t)$ in the momentum equation is kept non-helical (by specifying $f_{\mathbf{k}(t)}$). But for LSD, its made helical to inject kinetic helicity.

A run folder contains:

- *.in files : for inputs like initial conditions, quantities to be printed periodically, parameter values, etc.
- *.f90 in src folder : source code (there are *.f90 programs for each component, like hydro.f90, magnetic.f90 etc).
- the setup configuration files (*.local) in src folder.

With the input files and configuration files in src, along with the source code, one can compile the code, which produces executable files. These executables are then run on the cluster. The run folder also contains a folder named "data" where all the output from the run is stored. Output from each processor is stored in folders named as proc0, proc1, proc2 and so on.

Now we will explore the setup in run folder in brief detail.

1.1 Run folder details

Inside the src folder find these two setup configuration files : Makefile.local and cparam.local.

Makefile.local :
Here each line specifies the module to be included (or not).
For eg:
MAGNETIC=magnetic - means the code for solving the induction equation is included.
GRAVITY=nogravity - means we are not including the term for gravity in the momentum equation.

cparam.local:
Here we specify the grid, computational resource required and the parallelization.
The first line has:

- ncpus - no. of cores on which the code is to be run
- nprocx, nprocy and nprocz : the values assigned specify the amount by which each coordinate axis is paralellized.
  For eg.: ncpus=64, nprocy=4,nprocz=ncpus/nprocy,nprocx=1 - each of the 64 cores will calculated quantities on entire x-axis, (1/4)th of the y-axis and (1/16)th of the z-axis. In this manner, the calculation of quantities on the N^3 grid is distributed amongst the 64 cpus.

The second line specifies the no. of grid points for each of the coordinate axes. In this case, there are 256 points for each of the axes and hence forms a grid cube of 256^3.

Now outside src folder, in the run folder there are following input files:
- start.in, run.in, print.in and video.in

start.in:
For each of the modules included one can specify the initial conditions.

1. Under "eos_init_pars" (eos: equation of state module), we have gamma=1. – We are considering isothermal condition.
2. Under "hydro_init_pars", we have not specified anything. Hence by default, the initial velocity of the fluid is taken to be 0.
3. Under "density_init_pars", we have not specified anything. Hence by default, density is constant everywhere.
4. Under "magnetic_init_pars", initaa='gaussian-noise' means that the initial magnetic field is a Gaussian random field whose amplitude is specified as amplaa=1e-4. The initial field is quite weak.

run.in:

1. run_pars
   - nt : total number of time steps for the run
   - it1: frequency at which diagnostic quantities in print.in are to be printed in 'data/time_series.dat'. For eg: it1=10, prints diagnostics after every 10 time steps (code evaluation time).
   - dsnap: the frequency at which the entire grid cube is to be saved/stored. For eg: dsnap=40. stores the cube after every 40 time-units.
   - dvid: the freq. at which slices from the cube are stored to be able to display and run them continuously as a movie.
   - dspec: the freq. at which spectra are stored.

2. forcing_pars: The forcing function is given as helical, but the helicity is specified to be 0 (relhel=0.). Hence its non-helical forcing.
   The value for "force" (here, force=0.02), specifies the strength of forcing term in the momentum equation and hence will control the amount of energy being input to the system. The rms value of velocity, urms in steady state in the run is sensitive to the value of "force" here. Hence this value has to be optimum, i.e. to increase the urms to get the maximum R_M (defined on the board), but to keep the flow subsonic (and avoid shocks) with Mach ~ 0.1.

3. magnetic and viscosity pars: Here eta and nu specify the value for resistivity and viscosity of the fluid. Here the value is again an optimum. We need to decrease it as much possible to maximize R_M, but need it to be large enough to dissipate the velocity field sufficiently to again avoid shocks and also to be able to resolve the small diffusive scales.

print.in:
This file contains a list of the diagnostic quantities that one wants to printed every 'it1' time-steps.

- it: time steps
- t: time-units
- dt: size of the time step
- rhom: mean of density over the volume of the domain
- urms: rms value of velocity field
- umax: the maximum value of velocity field in the domain
- brms: rms value of magnetic field

video.in:
List of all quantities whose slices are stored every 'dvid' time-units.

Time series:
In data folder, the "time_series.dat" contains the output for all quantities listed in print.in, every 'it1' time steps and shows how the simulation has progress. One can plot these quantities.

Come back to the run folder and type on command line : "idl" And in the prompt for idl, type : ".r ts" – This will show plots of urms and brms w.r.t time.

Hope you enjoyed the introduction to pencil-code simulations !

```
ssh -X account_name@perseus.ac.iucaa.ernet.in
```

Lets begin with the simulation nonhel256a, of small scale dynamo, which has already been run and kept.

Time series

```
> cd ~/pencil-code/nonhel256a
> ls
```

Now you are in the folder nonhel256a. We will look at the time series plots.

```
> idl
IDL> .r ts
```

This shows plots of urms and brms w.r.t time with yaxis in log scale.
This basically plots the data available in data/time_series.dat.

```
IDL> plot, ts.t, ts.urms
IDL> plot, ts.t, ts.brms
```

The above will plot the quantities in linear scale.

```
IDL> plot, ts.t, ts.urms, /ylog
IDL> plot, ts.t, ts.brms, /ylog
```

We can specify /ylog for log scale again.

```
IDL> plot, ts.t, ts.brms, /ylog, xrange=[10,100]
```

The command above zooms into the time between to 10 to 100 for example.

```
IDL> exit
```

Note : The commands in idl are stored. If you use the up arrow key, the previous
commands are retrieved.

Power Spectra

Exit the previous session and enter idl again.

```
> idl
IDL> power,'_kin','_mag',spec1=spec1, spec2=spec2, /all, i=n, tt=t, wait=0.2
```

You will see two spectra : kinetic and magnetic power spectra.

IDL> power,'_kin','_mag',spec1=spec1, spec2=spec2, /all, i=n, tt=t, wait=0.5
By increasing 'wait', we can slow down the speed of the running power spectra plots.

The kinetic spectrum initially builds up and then settles to a dynamically steady curve
once the turbulence has developed.
Then the magnetic power spectrum rises/evolves in a self-similar fashion.
Note that the peak of the magnetic spectrum is at large k, where the fields are growing the fastest,
in the kinematic stage (when fields are still growing).
But towards saturation, is there a shift in the peak of M(k) ?

IDL> exit

Movie

There are 2 ways to do this. One is two run the movie online in idl. This could be slow.
Otherwise collect the png images and stitch them together and watch the movie on your local machine.
Below we specify both ways:

1. Online:

```
>idl
IDL> device, decompose=0
IDL> loadct, 5
IDL> rvid_box, "bb3", tmin=400
```

To stop the display in between, press Ctrl+C

2. Make a movie offline:

```
>idl
IDL> loadct, 5
IDL> rvid_box, "bb3", tmin=400, tmax=1000, /png
IDL> exit
> mkdir movie
> mv *.png movie
> cd movie
> ffmpeg -framerate 15 -i img_%04d.png -c:v libx264 -r 30 -pix_fmt yuv420p ssd.mp4
```

Open another terminal and do the following:

```
> sftp  account_name@perseus.ac.iucaa.ernet.in
> cd pencil-code/nonhel256a/movie
```

```
> get ssd.mp4
```

Now you have it on your local system and can play it using your local movie player.!
We see the z-component of the magnetic field : bb3. Here the color scheme is
such that yellow and blue specifies around 3*brms fields of opposite signs.
The orange fields are close to 0.

We turn to the next simulation, a run of Large scale dynamo :

```
> cd ~/pencil-code/hel32a
```

This time we will run the simulation.
The code is already compiled the code. We can run the code on the cluster
The command below will setup the data directory with all the required folders and setup the initial conditions.

```
> bsub -q normal -n 16 -o test_%J.out ./start.csh
```

The command below will now start running the simulation.

```
> bsub -q normal -n 16 -o test_%J.out ./run.csh
```

To monitor the code, firstly one can check the data/time_series.dat file.

```
> tail data/time_series.dat
```

Then one can plot it as before by going into idl.

```
> idl
> .r ts
```

If you keep repeating the command .r ts, it takes the updated time_series.dat file and plots the updated curve.
In this way, one can monitor the progress graphically.

Once the dynamo has saturated, then we can look at the evolving spectra and the movie as before.

Note that here also initially the magnetic power spectrum is peaked at larger scales,
thats because the SSD grows fields faster than LSD. As the simulation proceeds, slowly the power
shifts to larger scales (or smaller k).
In the movie, the LSD is particularly dramatic as the field becomes more and more coherent
and produces fields on the scale of the system !