Pencil-code: quick reference guide.

Illa R. Losada, Michiel Lambrechts, Elizabeth Cole

June 24, 2013

Contents

1 Download the Pencil Code

The Pencil Code is an open source code written in the `fortran` language. General information can be found at

`http://www.nordita.org/pencil-code/`.

The latest version of the code can be downloaded with `svn`. In the directory where you want to put the code, type

```
svn checkout https://pencil-code.googlecode.com/svn/trunk/ pencil-code
--username NAME
```

where you replace NAME by your gmail name, and the password is the one generated by pencil-code.googlecode.com (→profile →settings).

An alternative, without using `svn`, for a non-up-to-date version of the pencil-code is

```
wget http://pencil-code.googlecode.com/files/pencil-code-r18525.tar.gz
```

The downloaded `pencil-code` directory contains several sub-directories

1. `doc`: a very important directory containing the pencil-code `manual.tex`. The pdf of the latest version of the manual is created simply by typing `make` in the `pencil-code/doc` directory. For example, the code structure can be further explored in Section 4 of the manual.
2. `samples`: contains many sample problems
3. `config`: has all the configuration files
4. ...

2 Configure the shell

2.1 tcsh shell

Under the tcsh shell, put in your $HOME/.cshrc file the following lines:

```
#
#  path for pencil code
#
if (! $?PENCIL_HOME) setenv PENCIL_HOME $HOME/pencil-code
if (-r $PENCIL_HOME/sourceme.csh) then
  set _sourceme_quiet; source $PENCIL_HOME/sourceme.csh; unset _sourceme_quiet
endif
```

Note: If you want to change your default $SHELL to tcsh, type

```
  chsh
  /bin/tcsh
```

The first command prompts you for your normal unix password.

Source your updated $HOME/.cshrc file by typing:

```
source .cshrc
```

2.2 bash shell

If you are under the bash shell, put in your $HOME/.bashrc file the following lines:

```
#
#  path for pencil code
#
if [ -z $PENCIL_HOME ]; then  export PENCIL_HOME=$HOME/pencil-code; fi
if [ -e $PENCIL_HOME/sourceme.sh ]; then
  set _sourceme_quiet; source $PENCIL_HOME/sourceme.sh; unset _sourceme_quiet
fi
```

Source your updated $HOME/.bashrc file by typing:

```
source .bashrc
```

2.3 Handy aliases

Also add the following useful alias:

With tcsh: `alias pc 'cd $PENCIL_HOME'` With bash: `alias pc='cd $PENCIL_HOME'`

such that you will directly move to the Pencil Code directory wherever you are by typing 'pc'.

3 Fortran

A fortran compiler is needed to compile the code.

3.1 Fortran on a mac machine

For Mac, you first need to install `Xcode` from the `AppleDeveloper` site `http://developer.apple.com/`. This requires you to first register as a member. An easy to install `gfortran` can be found at `http://gcc.gnu.org/wiki/GFortranBinaries`. Just download it and it comes with an installer. It installs in the directory `/usr/local/gfortran` with a symbolic link in `/usr/local/bin/gfortran`. It might be necessary to add the following line in the `.cshrc`-file in the home folder:

```
setenv PATH /usr/local/bin:\$PATH
```

4 Configure makefile.

In order to run the code a proper configuration file is needed. This file should contain all the information related with the compilers and their special options.

The configuration files should be located in the `config` directory and should have a special name related with the host-id. This id is easily obtained by:

```
pc_build --debug
```

And now create a new config file in: `config/hosts/YOUR-NAME/HOST-ID.conf`

The next step is customize this file, there are several examples in `$PENCIL_HOME/config/hosts`.

5 Configure the run

The structure of the Pencil code consists of different modules, containing different physics that you may or may not need for your problem.

5.1 Makefile

```
pc_setupsrc
```

This sets up the linking to the root `src` directory and makes a `data` directory that will contain the data produced by your simulation.

Basic configuration files for the make:

```
src/Makefile.local
src/cparam.local
```

The structure of the Pencil code consists of different modules, containing different physics that you may or may not need for your problem. These are set in `Makefile.local`.

Example of a file without mpi:

```
###                             -*-Makefile-*-
### Makefile for modular pencil code -- local part
### Included by 'Makefile'
###

MPICOMM=nompicomm
#MPICOMM=mpicomm
GRAVITY=gravity_simple
EOS=eos_idealgas
FORCING=noforcing
ENTROPY=noentropy
MAGNETIC=magnetic
MAGNETIC_MEANFIELD=magnetic/meanfield
DENSITY=density
HYDRO=hydro
```

The `src/cparam.local` file set the local settings concerning grid size and number of CPUs. My file (no mpi):

```
!  -*-f90-*-  (for Emacs)    vim:set filetype=fortran:  (for vim)
!
!  cparam.local
!
!  Local settings concerning grid size and number of CPUs.
!  This file is included by cparam.f90
!
integer, parameter :: ncpus=1,nprocx=1,nprocy=1,nprocz=ncpus/(nprocx*nprocy)
integer, parameter :: nxgrid=128,nygrid=1,nzgrid=128
!
```

5.2 Run configuration files

There are three basic configuration files where all the physics and outputs are specify: print.in, run.in and start.in.

- `print.in`: write here all the input magnitudes you need.
- `run.in`: parameters values, number of timesteps,...
- `start.in`: Initialisation parameters

6 Setting up python.

6.1 Python modules requirements.

The basic needed modules are: numpy and matplotlib.

- numpy: all array definitions and operations.
- matplotlib: plotting.

Other really useful modules are: ipython and scipy.

- ipython: enhanced python interpreter.
- scipy: science functions and utilities.

6.2 Installation

Untar the `tar.gz` file or go to the directory and simple type as root or sudoed:

```
python setup.py install
```

For a user installation (no root permission):

```
python setup.py install --user
```

6.3 Using the module.

Import the module:

```
import pencil as pc
```

Some useful functions:

pc.read_ts	Read "time_series.dat" file. Parameters are added as members of the class.
pc.read_slices	read 2D slice binary files and return two arrays: one of (nslices,vsize,hsize) and other of tim
pc.animate_interactive	Assemble a 2D animation from a 3D array.

7 Run a sample: an example

Construct a folder in which you would like to run the sample

```
> mkdir sample_test_jeans
```

Now we copy the sample from the pencil code directory

```
/sample_test_jeans> pc_newrun ~/pencil-code/samples/1d-tests/jeans-x jeans-x
```

The `pc_setupsrc` command sets up the linking to the root `src` directory and makes a `data` directory that will contain the data produced by your simulation.

```
/jeans-x> pc_setupsrc
```

You can inspect the included modules in

```
/jeans-x> vi src/Makefile.local
```

where I used the `vi` text editor. The grid size can be inspected in

```
/jeans-x> vi src/cparam.local
```

The code is compiled by

```
/jeans-x> pc_build
```

and this may take some time.

The initial conditions are set in

```
/jeans-x> vi start.in
```

and parameters necessary to run the code in time can be found in

```
/jeans-x> vi run.in
```

Time to run the code.

```
/jeans-x> pc_run
```

Visualizing the output can be either done with `idl` or `python`.

7.1 IDL visualization

Start `idl` from the command line. Several `idl` have been written (you can find them in `pencil-code/idl`) to facilitate inspecting the data (which can be found in raw format in `jeans-x/data` directory). For example, let us inspect the time series data

```
IDL> pc_read_ts, obj=ts
```

The structure `ts` contains several variables that can be inspected by

```
IDL> print, tag_names(ts)
IT T UMAX RHOMAX
```

The diagnostic `UMAX`, the maximal velocity, is available since it was set in `jeans-x/print.in` (more on diagnostic output can be found in section 5.4 in the manual). We can now plot the evolution of the maximal velocity in time

```
IDL> plot, ts.t, alog(ts.umax)
```

after the initial perturbation we inserted in `start.in`.

The complete state of the simulation, a snapshot, is saved (as `jeans-x/data/proc0/VAR$\dots@`, every `dsnap` time units (see `jeans-x/run.in`). These states can be inspected, for example

```
IDL> pc_read_var, obj=ff, ivar=1, /trimall
```

Similarly `tag_names` will provide us with the available variables,

```
IDL> print, tag_names(ff)
T X Y Z DX DY DZ UU LNRHO POTSELF
```

so the logarithm of the density in the simulated domain can be inspected by

```
IDL>plot, ff.lnrho
```

8 Another example: helically forced turbulence

Question: Simulate the saturation behaviour of a dynamo from helically forced turbulence with forcing wave-number $k_f = 3$ in units of the box wave-number $k_1 = 1$. Use the sample `samples/helical-MHDturb`.

1. Determine the critical value of the magnetic diffusivity above which there is a growth of the rms magnetic filed, `brms`, in the file `data/time\_series.data`
2. Determine the corresponding value of the magnetic Reynolds number, $Re_M = u_{rms}/\eta\kappa_f$
3. Determine the growth rate of the magnetic fiedl for a choseen value of the magnetic diffusivity about half the critical vaule.
4. Determine the estructure of the magnetic field. Consider the evolution of 3 different magnetic field averages: the `xy` average `bmz`, the `yz` average `bmx` and the `xz` average `bmy`. Run until saturation and determine which of the three averages dominates in the end.
5. Fit the resulting ($\bar{B^2}$) to the expression:

$$F(t;B_0,t_s) = B_0^2[1-\exp^{-2\eta\kappa_1^2(t-t_s)}] \tag{1}$$

8.1 Solution.

8.1.1 Critical value for the magnetic diffusivity.

Calculation of the critical value for the magnetic diffusivity implies running the code for different values of the magnetic diffusivity and checking up at which value the field starts growing.

In figure 1 the growth of the rms magnetic field versus time can be analyzed for different magnetic diffusivity values.

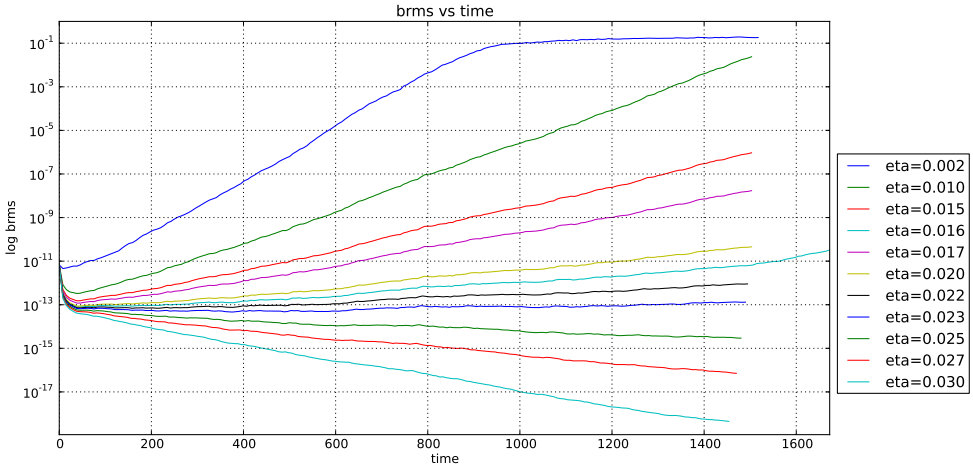

Figure 1: log(brms) vs time for different values of magnetic diffusivity.

Using these results, I set the critical value for the magnetic diffusivity in $\eta = 22e-3$.

8.1.2 Magnetic Reynolds number

The magnetic Reynolds number is defined as:

$$Re_M = \frac{u_{rms}}{\eta \kappa_f} \tag{2}$$

We have set $\kappa_f = 3$, the critical value for the magnetic diffusivity found is $\eta = 22e-3$ and the mean value for u_{rms} is $\bar{u_{rms}} = 0.14$, so the corresponding magnetic Reynolds number is $Re_M = 2.15$.
The magnetic Reynolds number is the ratio between convection and diffusion. When $Re_M \gg 1$, convection dominates, whereas for $Re_M \approx 1$, or less, diffusion becomes important. So, in this exercise, diffusion is about to become important. In fact, the table 8.1.2 shows that Re_M decreases as η increases

η	Re_M
2e-3	22.39
10e-3	4.73
15e-3	3.16
16e-3	2.97
17e-3	2.78
20e-3	2.37
22e-3	2.15
23e-3	2.06
25e-3	1.89
27e-3	1.75
30e-3	1.58

8.1.3 Growth rate of the magnetic field

In order to determine the growth rate of the magnetic field for a value of the magnetic diffusivity, one must fit the sloped part of the logarithm of *brms* vs time curve, as shown in the figures 2 and 3.

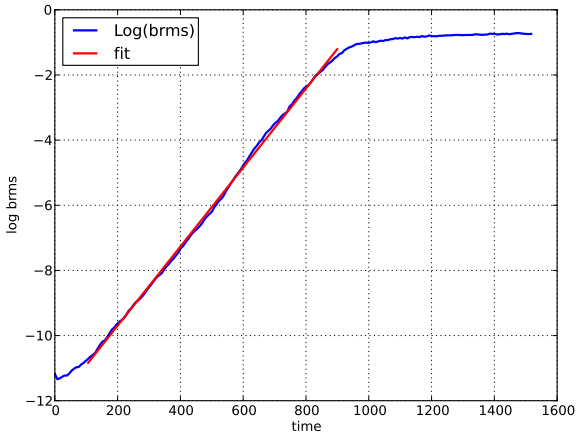

Figure 2: Growth rate of the magnetic field for a value of the magnetic diffusivity $\eta = 2e-3$.

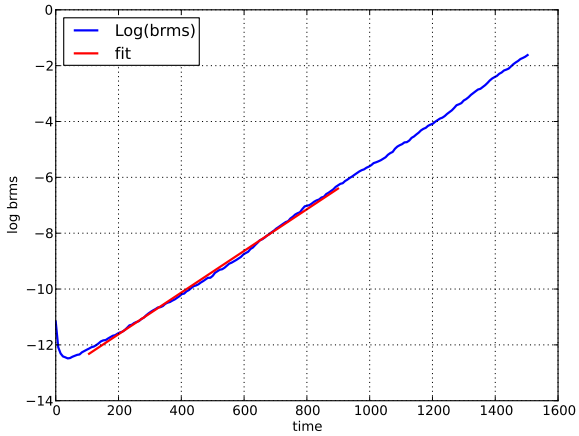

Figure 3: Growth rate of the magnetic field for a value of the magnetic diffusivity $\eta = 10e-3$.

The parameter a in a linear regression fit ($y = ax+b$) is a measure of the growth rate of the magnetic field. The growth depends on the value of the magnetic diffusivity, and it decreases as the magnetic diffusivity increases.

η	a	b
2e-3	0.0121	-12.12
10e-3	0.0075	-13.11

8.1.4 Structure of the magnetic field.

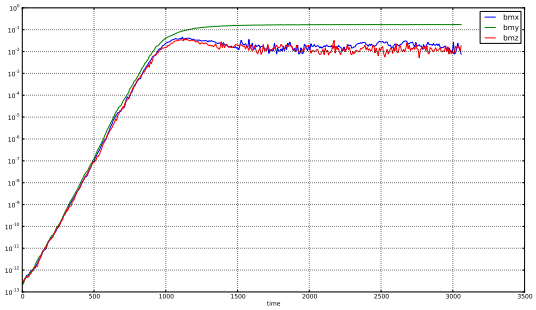

Figure 4: Averages of three different magnetic fields.

Figure 4 shows the evolution of three different magnetic field averages: the xy average bmz, the yz average bmx and the zx average bmy. From the figure one can see that the zx average dominates in the end.

8.1.5 Fitting B_{zx}

Fitting the strongest of the three field averages $< \bar{B^2} >$ to the expression:

$$F(t;B_0,t_s) = B_0^2[1-\exp^{-2\eta\kappa_1^2(t-t_s)}] \tag{3}$$

The strongest of the three field averages is the zx average represented in the figure 5 with different values of the expression 3. A rough fit for the field average takes the values $B_0 = 0.1712$ and $t_s = 930$. B_0 is the saturation field

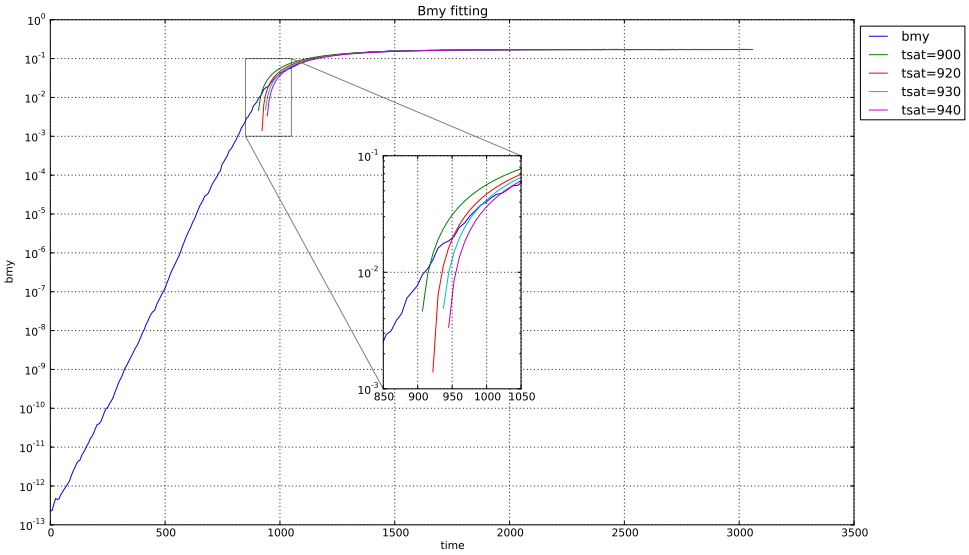

Figure 5: Bmy Fitting: $B_0 = 0.1712$ and $t_s = 930$.

value and t_s is the best fit curve just before saturation.