Helioseismology with MESA and GYRE

Laboratório de Astrofísica - 2023

Diogo Capelo *

October 9th 2023

Contents

1 Objective

The objective of this project is to create a model of the Sun and predict the p-mode frequencies of its adiabatic oscillations, and then compare this result against real observation data from the GOLF (Global Oscillations at Low Frequency) mission.

In order to accomplish this, the MESA (Modules for Experiments in Stellar Astrophysics) one-dimensional stellar evolution code will be used to solve the structure and material equations of stellar evolution and generate (simple) models of the Sun, and the GYRE code will be used to solve the eigenvalue problem consisting of the linearised adiabatic oscillation equations and produce the oscillation frequencies of the acoustic (or pressure) modes.

*diogo.capelo@tecnico.ulisboa.pt

This guide is divided into two main parts. Section 2 provides instructions in how to get, install, test, and use the MESA and GYRE codes for the purposes mentioned above. Section 3 deals with the treatment of observational data necessary to asteroseismic use, and the comparison of the theoretical (model-based) oscillations with the results from GOLF.

Shell commands assume the Bourne shell is being used, users with other shell environments should use the appropriate equivalent commands. If you aren't sure what shell environment your system uses, execute the command `echo $0` in your command line.

2 Modelling the Sun

In order to use the MESA and GYRE codes, you will have to install them first. The current (23.05.1) version of MESA already comes with GYRE pre-packaged, so it is strongly recommended that you start by getting the stellar evolutionary code first.

2.1 Note for Windows Users

If you are using a Linux distribution or MacOS as your operating system, you can skip this and go straight to Section 2.2.

MESA will run in any UNIX-based operating system, like Linux or MacOS, but not Windows. While some workarounds exist to run MESA in Windows, the complications are usually not worth the benefits. If your computer uses Windows, and you do not have any other operating system, the best way to get MESA working on it is to install a distribution of WSL (Windows Subsystem for Linux), which you can do easily by accessing the Microsoft store and searching for "Ubuntu". This will install a complete Ubuntu terminal environment in your computer. Alternatively, you can open the PowerShell or Windows Command Prompt in Administrator Mode and enter the command:

```
wsl --install
```

If you have a Virtual Machine environment running any Linux distribution, this will also work for using MESA , though you may experience some delays with in your modelling if insufficient resources were assigned to the partition – stellar evolution codes are efficient, but they are *not* light.

2.2 Installing MESA

Once you have a UNIX-based environment ready, you can access all MESA related information from the MESA documentation page. Specifically, you will want to check out the Installing MESA section. The version of MESA we will be using is the latest release, 23.05.1 .Make sure you have done all the following steps:

Setup The MESA code is written in Fortran, a compiled language (like C, and unlike Python) which means that in order to use it, your computer will have to be able to compile Fortran. The MESA SDK will set you up with this and all the other

dependencies that are required to use MESA . Install or update all the tools that are the pre-requisites to the single MESA pre-requisite (the MESA SDK). To do this, use a package management tool (like apt-get) followed by the name of the pre-requisite that can be found on the table on this page.

SDK Download the MESA Software Development Kit (SDK). Make sure you chose the correct operating system and (if you are a Mac user) hardware – there are different versions of the SDK for the Intel and ARM (M1 and M2) processors.

- Unpack the SDK in your computer using `tar xvfz sdk_file_name -C ~/` or click on it in the Finder if using MacOS
- Open the file .bashrc in your home directory and set the SDK environment variable by adding the line `export MESASDK_ROOT=~/`
- While in the .bashrc file, add another line with:
  `source $MESASDK_ROOT/bin/mesasdk_init.sh`
  Exit and save the file, and run `source .bashrc` to refresh it
- Run the command `gfortran --version` to verify that the Fortran compiler was correctly installed

Unpack Download and unpack the MESA directory from the Zenodo repository (the file is 2.1 GB). The unzipped and installed package will be around 20 GB, so make sure you have that much free on your disk.

Vars To use the software, you will have to set some system environment variables so everything in your computer will know where to find MESA resources (similar to what you did for the SDK). The best way to do this is to set these variables in your configurations file, .bashrc, which should be in your home directory. You will need to add the following lines to the end of this file, replacing the path where you extracted the MESA directory too in the first, and the number of threads you wish to assign to MESA calculations (the suggested number is $2 \times$ number of cores on your computer $- 2$), usually either 6 or 14:

```
export MESA_DIR=/full_path_to_MESA_root_dir
export OMP_NUM_THREADS=threads_to_use_with_MESA
```

Install Change into your mesa directory with `cd $MESA_DIR` and then compile MESA with the command `./install`

If MESA is successfully installed, it will output a message saying exactly that to your console. If you have obtained this message, you can proceed to Section 2.3.

2.3 Building a Sun

Now that MESA is operational, we have thew only tool we need to create a model of the Sun to as much (or as little) precision as we want. For the purpose of this project, we will create a model that is illustrative enough but, in order to keep running times short, is not good enough for scientific purposes.

2.3.1 Setting up a work directory

To begin using MESA , copy the template work directory `work` from inside the `$MESA_DIR/star` by using the command `cp -r $MESA_DIR/star/work ~/` . This directory contains all the essential files needed to create a simple model, and is a good starting point from which to build more complicated projects, so you will only need to directly change the files in this directory. Though you may need to consult files from the main MESA directory, it's not a good idea to change any of them unless you know exactly what you're doing – usually, an advanced development (not use) case.

Once you have copied the work directory (to any location) you can also rename it. Thanks to the environment variables we set up earlier, your computer will know how to reference calls to MESA , and so you can (and should) keep your work folders separate from your main MESA directory.

2.3.2 Understanding inlists

The MESA code can evolve a star (or a binary of stars) from the pre-Main Sequence (PMS) to its final stage of evolution. In order to do this, however, it is necessary to specify all the inputs that are relevant to the case in hand. Understanding what goes into making a star and, among that, what is important to our specific case is the first step of modelling any star. In MESA , the specification of these inputs is made in files called *inlists*. In general terms, an inlist is a list of variables and the values they are to assume. Importantly, MESA contains a very large number of variables that can be tuned to produce specific results in the physics or the numerics of the code, but the vast majority of these variables are only relevant for advanced users or for specific use cases. Therefore, for our purposes, it is not necessary to change (or even know) all the variables that can be changed. **In the absence of an explicit definition in an inlist, MESA will assume the default value for that variable.** An empty inlist will, therefore, still run and produce a model, according to the default values of all variables, which can be found in `MESA_DIR/star/defaults` . Generally, it's a good practice to always specify at least a few variables (mass and composition) in your inlist, so that other people that look at it will immediately know what type of star you are modelling without needing to look at the model itself.

A full inlist in MESA 23.05.1 usually includes variables divided into 5 sections (sometimes called namelists):

star_job Contains information about which MESA modules should be used, the atmosphere, the nuclear reactions network and rates, as well as some information about your

starting model (e.g. whether to start from the PMS or from another, already evolved model), the output, and the initial composition of your star.

eos Stands for Equation of State. This controls how MESA handles the connection between the structure equations and the thermodynamics, via the state variables.

kap Stands for opacity (κ). These options allow you to control radiative and conductive opacities for your models.

controls This is the main module to specify the physics relevant to your star, including its mass, convection and convective boundaries, mixing, and timestep and grid controls.

pgstar This module is responsible for providing on-the-fly plots of what's happening to your model as it is being run. This is a fantastic tool to understand the evolution of your model without having to make several plots once it has finished running.

In your `work` directory, you will find three files called `inlist` : `inlist_project`, `inlist_pgstar`, and `inlist` . The last of these you can find in many test cases, and if you open it, you can see that it just contains two lines in each section, including *another* inlist, called `inlist_project` . This is useful when you are trying to run several different versions of the same star, and therefore have to switch between different inlists, containing different controls. For now, you can leave it as it is.

The `inlist_pgstar` file contains some basic visualisation options. You can also leave this one as it is.

Finally, the `inlist_project` is the one we want to edit. When you open it, you can see it already contains some pre-filled values for several variables. Read through the inlist and try to get a sense for the variables that we are changing and why. Some of these need to be filled in – choose appropriate values and insert them in the lines that say `! fill here` .

2.3.3 Running your model

Once you are done looking at your inlist and filling in the missing standard solar values, compile the code with the command `./mk` and then run it from the start using `./rn` . MESA also allows you to rerun a model from a previously saved point (called photos; these are automatically saved during the execution) by using the command `./re photo_file` where `photo_file` stands for the name of a photo file saved during the execution.

Note the windows where the pgstar plots appear. What is being plotted and why is important? From the outputs of these windows, can you characterise the star you are modelling?

At the end of the execution, you will find the model outputs in a directory called 'LOGS' by default. You may have noticed that you can change this using the `log_directory` variable in the controls section in the inlist. There are several files there:

1. `history.data` Contains one line for each timestep of the model, characterising the star globally in that instant of time

2. `profile*.data` One file of this type is present for each timestep for which you asked MESA to print a profile output. Each of these files contains one line for each layer (dr) of your spherically-symmetric model, characterising the star at various depths at a single instant of time

3. `profiles.index` A list of profile numbers and what models they correspond to. Profiles may not be saved in chronological order.

4. `pgstar.dat` Contains information regarding the plots that pgstar draws during the evolution

2.3.4 Changing composition

Now that you have your first model of the Sun, it's time to try some changes. In particular, your previous model was made considering the composition determination of Grevesse and Sauval (1998). Asplund (and colleagues, including Grevesse), came up with a different composition determination in 2009, using more sophisticated techniques to analyse spectroscopic data. Open your inlist and think about what needs to be changed to reflect this new composition in your model. When you are done, run your model again (it might be a good idea to change the output directory, otherwise your previous model's files will be overwritten).

Do you notice any difference in the quantities being plotted by `pgstar`?

2.3.5 Select you solar model

You have now created two different models of the Sun, using two different compositions. Before we continue, you will have to choose one to proceed. Ideally, you would want to choose the one that best represents the current Sun. How can you accomplish this? How would you refine this process if you had more time?

If you feel the need to plot quantities from the outputs of your models, you can use Bill Wolf's MESA Explorer site or the program of your preference.

Hint 1: try to compare parameters of your model with solar observations.

Hint 2: compare the sound speed profiles for both compositions. The observed sound speed profiles for the Sun can be found in Basu et al. (2009) . The data files can be found in the project package in the class repository, in a directory called `SoundSpeed`. Calculate $\delta c/c$,

$$\frac{\delta c}{c} = \frac{c_{\rm obs} - c_{\rm model}}{c_{\rm model}} \tag{1}$$

where c is the speed of sound, and the subscripts obs and model stand for the values inferred from observation and predicted by your models, respectively. Note that you will likely have to interpolate either the sound speed profile for your model or the one from the inversion of helioseismic data (Basu's) to make sure you are comparing them at the same points.

2.4 Calculating adiabatic oscillation frequencies

Now, we will use GYRE to obtain the oscillation frequencies for the chosen solar model.

1. The package GYRE uses the output of mesa (solar.mesa) and computes the eigenmode frequencies for the corresponding model. To obtain files readable by GYRE our MESA inlist must contain specific controls. in our case, these controls are already in the inlist, and so you will already have the necessary GYRE file.

2. Run GYRE by executing the command `$GYRE_DIR/bin/gyre gyre.in` in your work directory.

3. A summary of the the computed frequencies can be found in the file `freqs_summary`. What do you expect to find?

4. Plot the frequencies in a frequency vs. radial order ℓ plane. Do your results match your expectations?

3 Spectrum Analysis

You can follow Part 2 of the instructions created by José Lopes that are on the course repository for this part of the project.