Solutions `MESA` Session 2
MESA output and Intermediate mass stars

2 Overshooting in an intermediate-mass star

We will now investigate the effect of overshooting. In order to do this, we evolve an intermediate-mass star for different levels of convective overshooting.

2. **Download and set up the work folder.** Instead of starting from the beginning and copying the model work directory like last time to edit, a `MESA` model has largely been constructed for you already.

   (a) Download the tar file from Brightspace and unpack it. This can be done using the archive manager by extracting the contents to your desired location on the **scratch** disc, or using the terminal by typing `tar -xvf session2.tar` in the directory where you downloaded the tar file and then moving the directory to your desired location on the **scratch** disc using a `cp -R` command. This file contains a `MESA` work folder in which we will work.

   (b) Once you open the `inlist_project` file, you will find a number of controls available for the model. First, choose a mass for the star between 2.5 M$_\odot$ and 10 M$_\odot$, and modify the inlist accordingly. Note that the stopping condition allows the star to evolve up to the end of core helium burning. You may further notice several lines commented out concerning Convective overshooting. We will run several models for different values of the overshooting parameter.

   (c) First, compile (`./mk`) and run (`./rn`) the model without any overshooting. Use the various `PGSTAR` windows to follow and understand the evolution of this star. One of the `PGSTAR` windows is a Kippenhahn diagram, which shows information about the structure of the star as it evolves. Also try to understand this plot.

   (d) After the model finishes, create a new copy of the work folder and rename the **old** folder to an appropriate name (e.g. identify it by the mass and overshoot value used, like `M3_0ov0` for mass 3 and overshoot 0). In the new work folder, uncomment the lines relating to convective overshooting in your inlist and change the overshooting parameter `overshoot_f(1)` to 0.25. Look for the meaning of this and other overshooting parameters in the file `$MESA_DIR/star/defaults/controls.defaults` or by checking the documentation online. [1] Then compile and run the code again. Repeat this process for an overshooting parameter of 0.5.

3. **Inspecting the results.** We can now analyse the data in the 3 history files. For this import `MESA Reader` into your python notebook, as explained above.

---

[1] Note that the documentation online is for the latest version of `MESA`, which may differ slightly from the version you are using. The bottom right of the documentation page shows which version of the docs you are viewing, but note this only dates back to version r15140, which is when MESA was migrated to GitHub. `$MESA_DIR/star/defaults/controls.defaults` will always show the information for the correct version for your installation.

To read in the data, you need to move the `history.data` files from your `scratch` directory to somewhere in your plotting folder, for example inside a folder called `data` and there inside a folder called after its run (e.g. `M3.0ov0`). For this example, you use the following line in your `Python` script:

```
1  f0_hist_data = mr.MesaData('data/M3.0ov0/LOGS/history.data')
```

To invoke the columns you want from the read-in data, in this case `f0_hist_data`, you use `f0_hist_data.X`, where X is the name of one of the columns of data inside `history.data`. The names of the columns can be found inside `history.data`, or you can print the available column names by typing:

```
1  print(f0_hist_data.bulk_names)
```

(a) Make an HR diagram containing the 3 models. What changes do you see in the main-sequence evolution? What changes appear in the evolution *after* the main sequence? Can you explain these changes?

(b) Make a plot of ρ_c vs T_c. How do the evolution tracks in this diagram change for different levels of overshooting?

(c) Construct a plot of the central helium abundance vs age for all 3 models and explain your findings.

(d) By what fraction is the main sequence lifetime increased for an overshooting parameter of 0.25 compared to the model without overshooting? By what fraction does the *helium burning* lifetime change? Compare your findings with your neighbours who, hopefully, have chosen a star of different mass.

3   Make your own Kippenhahn diagram

Kippenhahn diagrams (KHDs) show a star's internal structure over time and are useful for your final `MESA` project reports. Rather than taking screenshots from `PGSTAR`, you can use the Python script `mkipp`[2] to generate clearer, customizable plots. Here we will enable the necessary `MESA` output columns, help you set up `mkipp`, and get started with the scripts.

3.1   Setting `MESA` output

You can customize the `MESA` output through the `history_columns.list` and `profile_columns.list` files. In order to customize the output, just copy these files to your work directory, and uncomment the variables you want to include in the output. You can find default variations of these files in the `$MESA_DIR/star/defaults/` directory.

We will use the `mkipp.` tool from `https://github.com/orlox/mkipp`. As you can read on the github page, to use `mkipp` you need to have the following `MESA` output available: "IMPORTANT!! your `history_columns.list` needs to have `mixing_regions` and `mix_relr_regions` specified for MESA to output the neccesary data of the mixing regions in terms of mass and radius respectively. Also, newer versions of MESA include significantly less output in the profile files, so values such as `eps_nuc` need to be added to `profile_columns.list` as well."

1. Pick one star that you like, and make sure that all these variables are uncommented. (i.e., remove the '!' in front of the variable) for both `history_columns.list` and `profile_columns.list` files of that star's work folder.

---

[2] `https://github.com/orlox/mkipp`

2. When you uncomment `mix_relr_regions`, it will most likely say `<integer>` after it. You should change this to 10, so that it has the same value as for `mixing_regions`.

   For `MESA` to use the columns you have specified in these 2 files instead of the default set of columns, we have to make sure `MESA` knows that it needs to read these 2 files. To do this, we have to open `inlist_project` and include the following under `&star_job`:

```
! to specify which output columns we want in history.data and
   profile.data
   history_columns_file = 'history_columns.list'
   profile_columns_file = 'profile_columns.list'
```

   Now recompile and rerun your model.

3. Next, actually cloning the `mkipp` repository:

```
cd ~/codes  # or wherever you want to store the code
git clone git@github.com:orlox/mkipp.git
```

4. Unfortunately, `mkipp` is not available via `pip install`, so we have to make sure Python can find it. You can do this by adding the path to `mkipp` to your `PYTHONPATH` environment variable. To make your Jupyter notebook recognize and import a script or module from , you need to add that path to Python's search path at runtime.

```
import sys
import os
sys.path.append(os.path.expanduser("~/codes/mkipp"))
```

   Your notebook will now look for python functions in the /codes/mkipp directory as well, and so you can import the three python modules that live in there:

```
import mkipp
import kipp_data
import mesa_data
```

5. To run a Kippenhahn diagram, you need to define the `logs_dir` argument of the `KippArgs` function as a list containing the path to your MESA output directory. For example, you can create a Kippenhahn diagram using the following command:

```
mkipp.kipp_plot(mkipp.Kipp_Args(logs_dirs = ['data/M3.0ov0/LOG']))
```

   Do you understand what all the colors and hatching in the KHD mean?

6. Play around with the various options of `KippArgs` to customize your KHD. Have a look at the `mkipp/example.py` script for inspiration.