Minilab 1: Kelvin-Helmholtz Contraction

1 Starting from the Test Suite

The `MESA` test suite provides a common jumping-off point for getting started with projects without the need for building everything from scratch. For this lab, we will start with the `brown_dwarf` folder in the test suite. Start by copying the folder to a local directory where we can modify and work with it. Navigate to the directory where you want to work and then run

```
cp -r $MESA_DIR/star/test_suite/brown_dwarf .
cd brown_dwarf
```

Folders copied from the test suite require a bit of cleanup before they are ready to be used as local work directories. First, you need to delete the relative path references in a few files. In `make/makefile`, delete the line

```
  MESA_DIR = ../../../..
```

In `inlist`, delete the line

```
  mesa_dir = '../../..'
```

Now you're ready to run this test in your local directory, but you probably shouldn't, as it will take a while. If you do, you'll get some customized output at the end telling you that "all values are within tolerance." This is checked by a customized `run_star_extras.f` file that looks at specific properties of the final model and compares them to expectations. Of course, we want to modify things and do some exploration, and we are unlikely to "pass" the test once we do this, so for this test we should just replace the file with the default from `star/work/`:

```
cp -r $MESA_DIR/star/work/src/run_star_extras.f src/run_star_extras.f
```

Since this change involves modifying the source code for the `star` executable, you'll need to compile (`./mk`) for it to take effect. This is in contrast with `inlist` files, which get read in at run time, meaning there's no need to recompile after modifying an `inlist`. (N.B. Some folders in the test suite do more than just check the final output with `run_star_extras.f`. It's always a good idea to look through and try to understand what's being done here before replacing the file or leaving it to do whatever it wants.)

We're almost ready to begin modification and exploration, but this test directory has one more feature that may hinder us: very strict solver tolerances. To get rid of these, delete or comment out these lines in `inlist_brown_dwarf`:

```
! test with "Gold Standard" tolerances
      newton_iterations_limit = 20
      max_tries = 20
      iter_for_resid_tol2 = 30
      tol_residual_norm1 = 1d-9
      tol_max_residual1 = 1d-7
      tol_correction_norm = 1d-9
      tol_max_correction = 1d-7
```

2 Exploration of Pre Main Sequence Contraction

Instead of starting from a model, we want to specify the initial mass and start on the pre main sequence. That means we should delete these lines in `inlist_brown_dwarf`:

```
      load_saved_model = .true.
      saved_model_name = 'start.mod'
```

And uncomment these lines that already live in the inlist:

```
      ! create_pre_main_sequence_model = .true.
      ! pre_ms_T_c = 2d5
      ! pre_ms_relax_num_steps = 1
```

You may need to set `pre_ms_relax_num_steps = 50` to make it easier for the initial model relaxation to find a happy starting point on the pre main sequence. For this lab, we're also going to turn off all nuclear burning and just watch what we get from Kelvin-Helmholtz contraction. The easiest way to do that is to add this to `&controls`:

```
      ! Turn off burning
      max_abar_for_burning = 0
```

We're going to try to crowd-source a logarithmically distributed range of masses between $M = 0.03\,M_\odot$ and $M = 0.3\,M_\odot$. Below the lower bound of this mass range, it gets a little tricky to construct pre main sequence models with `MESA`, so we'll leave that for a later lab. In the `&controls` section of `inlist_brown_dwarf`, find the line

```
      initial_mass = 0.05
```

Generate a random number between -1.5 and -0.5, take 10 to the power of the result, and modify the `initial_mass` line accordingly:

```
      ! A random number between 0.03 and 0.3
      initial_mass = <your random mass between 0.03,0.3>
```

To get through the evolution in a reasonable amount of time, it may be necessary to loosen up the timestep controls by setting `varcontrol_target = 3d-4`, which allows for larger time steps. Finally, add these lines to your `&pgstar` section so you can watch the model as it evolves,

```
      Grid1_win_flag = .true.
      Grid1_win_width = 8 ! Or whatever value you prefer here.
```

and remember to uncomment this line in `&star_job` so that your plots will show up:

```
      ! pgstar_flag = .true.
```

We're now ready to run the model (`./rn`). Report the following information on the spreadsheet:

- initial mass
- core temperature when $R = 1.0\,R_\odot$
- core temperature when $R = 0.3\,R_\odot$
- core temperature when $R = 0.1\,R_\odot$

For hints on how to isolate this data, see the next page.

2.1 Hints

2.1.1 Using Stopping Conditions

Since we want to stop and get information for particular values of the radius, it may be useful to set a stopping condition. One way to do this would be to add this to the `&controls` section:

```
     photosphere_r_lower_limit = 1d0
```

This will stop evolution once the model contracts down to the specified radius (in $R_\odot$ units). Then you can use the terminal output to read off the core temperature, change the lower limit, and restart to move on to the next radius value.

2.1.2 Using pgstar

Here's an example of how one might construct a pgstar window to read off the needed information.

```
&pgstar

     History_Track1_win_flag = .true.
     History_Track1_yname = 'log_center_T'
     History_Track1_xname = 'log_R'
     History_Track1_yaxis_label = 'log T\dcenter'
     History_Track1_xaxis_label = 'log R/R\d\(2281)'
     History_Track1_reverse_yaxis = .false.
     History_Track1_reverse_xaxis = .false.

/ ! end of pgstar namelist
```

To keep the plots on the screen when the run ends, you could use the following option in `&star_job`:

```
     pause_before_terminate = .true.
```