Stars near the Eddington limit
MESA Summer School 2016

Norbert Langer, Ilka Petermann and Abel Schootemeijer

August 11, 2016

1 Minilab 1

In this minilab we investigate at what stellar masses the MESA models show envelope inflation during core hydrogen burning. To achieve this, we would like to track for different masses how the following stellar parameters behave as a function of time: $R/R_{\rm ZAMS}$ (ratio of the stellar radii at the end and beginning of core hydrogen burning), $\Gamma_{\rm max}$ (maximum value for the Eddington factor in the star) and $\Gamma_{\rm es,\,surf}$ (Eddington factor at the surface when considering only the electron scattering opacity). Later on, we investigate how the mixing length parameter influences the inflation of the envelope.

To get started: Download today's minilab at:
`http://mesastar.org/teaching-materials/`
Then, execute in your summer school directory
`tar -xvf <location>/minilabs_langer.tar.gz`
and enter the directory.

1.1 First run

To enhance your feeling for how a massive star evolves with MESA, we would like to start with a basic simulation.

- Randomly select one of the following initial stellar masses: $M \in \{10, 20, 40, 60, 80\}$ $\mathrm{M}_\odot$.
- Substitute this value in the `&controls` section in `inlist_project`, where it is set to the default value (`initial_mass = 1`).
- Stopping condition: we are investigating main sequence stars, so we want to stop at core hydrogen depletion. In this case we define the end of the main sequence as the point where the central hydrogen mass fraction drops below 10^{-3}. Take a look around in `mesa.sourceforge.net/controls_defaults.html` to find the right stopping condition. (**Hints:** `xa` can refer to not-yet-specified element; hydrogen is referred to as `h1`; the correct answer is in the hints section in case of emergency)

- Clean and compile the work folder:
  `./clean && ./mk`

- Run the model with:
  `./rn`

- Watch how the model evolves. What does happen to temperature, luminosity and radius?

1.2 Second run - set up inlist_pgstar

In the first run we did set up `inlist_pgstar` for you with some of `MESA`'s standard diagrams. We would now like to keep track of other variables while the models are evolving. Now it is your turn to configure `inlist_pgstar` to manage the plots.

A variable can only be displayed in the history panels of pgstar if it is also in the output of `LOGS/history.data`. To display an additional variable in the history file, you can either i) uncomment an extra variable in `history_columns.list` or ii) write a few lines of code in `src/run_star_extras.f` to create your very own variable. In this case, we have already created three variables for you in `src/run_star_extras.f`.

We want to track the following variables as a function of time: R/R_{ZAMS}, Γ_{max} and $\Gamma_{\mathrm{es,\,surf}}$.

- The variables R/R_{ZAMS}, Γ_{max} and $\Gamma_{\mathrm{es,\,surf}}$ are computed in `src/run_star_extras.f`. Look through that file, find the names of those variables and if you want, take some time to appreciate the variables.

Now you know the variable names, R/R_{ZAMS} et al. can be plotted on the y-axes in the history panels. We would like to plot them as a function of time. In `inlist_pgstar`, the empty plot has already been set up.

- To display the history plots, make the following adjustment:
  `History_Panels1_win_flag = .false.` $\longrightarrow$ `.true.`

- In `inlist_pgstar`, enter the variable names in
  `History_Panels1_yaxis_name(1) = ''`
  et cetera. Plot both Γ variables in the same panel.

- To use time on the x-axis, enter `star_age` as the variable of choice in
  `History_Panels1_xaxis_name = ''`

Now we are ready to run the model.

- (If you were dissatisfied with the time it took to run the last model, you can change the variable `varcontrol_target` in `inlist_project` to 10^{-4}. This will reduce the number of steps.)

- Run the model with:
  `./rn`

- Watch how $R/R_{\rm ZAMS}$, $\Gamma_{\rm max}$ and $\Gamma_{\rm es,\,surf}$ evolve.

Note: if the plot windows are too tiny or too large, you can change the width and aspect ratio in `inlist_pgstar`. It is possible by either cancelling the run with `ctrl + c`, changing and and restarting or by opening a new terminal and changing these variables on the fly.

Question: do you spot any correlations?

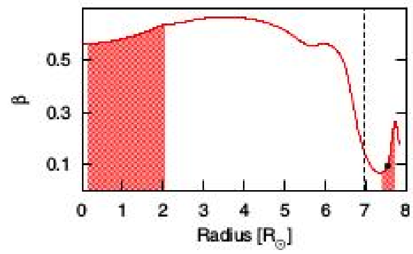

Figure 1: A profile plot of a mildly inflated star, where $\beta = P_{\rm gas}/P_{\rm tot}$. The dashed line represents the radius coordinate where $\beta < 0.15$ for the first time, when moving from the center to the surface. Figure is taken from Sanyal et al. (2015).

1.3 Third run - `mixing_length_alpha` and envelope inflaton

1.3.1 The inflated envelope

Before we study the inflated envelope in more detail, let's start with how exactly we define this inflated envelope. For this we consider the variable β, which is defined as $\beta = P_{\rm gas}/P_{\rm tot}$. As can be seen in Figure 1, the fractional gas pressure is relatively low at ∼6-7 $R_\odot$ and further outward. We define the *inflated envelope* to start at the point where β drops below 0.15 (when moving from the center to the surface). This point is represented in Figure 1 with a dashed line.
So to summarize our definitions:

i. The inflated envelope is the point where $\beta < 0.15$ and all layers further outward

ii. For this definition, the core is the rest of the star (it can be the whole star if $\beta > 0.15$ throughout the entire star)

1.3.2 The influence of $\alpha_{\rm MLT}$

The next thing that we want to test is how the mixing length parameter `mixing_length_alpha` influences the extent to which the envelope is inflated. Therefore, we ask you to run a model with a random value for the mixing length parameter and put the $R/R_{\rm ZAMS}$ of the last timestep in the Google spreadsheet document.

- Randomly select one of the following values for the mixing length parameter: `mixing_length_alpha` $\in \{1, 1.5, 3, 10, 100\}$.
- Enter this value in the `&controls` section in `inlist_project` and run the star (with the same mass) again with `./rn`.
- Obtain $R/R_{\rm ZAMS}$ from the last timestep from the data file `LOGS/history.data` (tip: the `LOGS/history.data` is much easier to read if it is displayed one line per row; this can be done with e.g. `less -S LOGS/history.data` or `:set nowrap` if you use `vim`).

- Enter this value and your initial paramters in the 'Mini1' tab in the Google spreadsheet document[1].

Question: Check `http//mesa.sourceforge.net/controls_defaults.html` for the default value of the mixing length parameter. How did the new value change your simulation?

Question: Try to compare your results with the results from students in your vicinity. Do you have an idea how the amount of envelope inflation (which increases the final `logR`) correlates with the chosen `mixing_length_alpha`?

Question: Can you think of a reason for that correlation?

[1] https://docs.google.com/spreadsheets/d/1Tf3mtR7UmgXFEWCtVrpybnKDPG5br2UqkDgh6hSlNDA/edit?ts=57a471c0#gid=1459085644

2 Minilab 2

In this minilab, we want to look into the models in a more detailed way. Therefore we study the profiles of the models, where we can look into each individual numerical zone for a given time.

Start in the same directory as the first minilab.

2.1 Profiles of the inflated envelope

2.1.1 Setting up `inlist_project` and `inlist_pgstar`

Apart from the history panels, we also would like to take a look at profile panels this run. This works similar to what we described in Sect. 1.2, but now use `r_in_rcore_units` (i.e. r / $R_{\rm core}$) on the x-axis and logρ, κ, Γ and β on the y-axis.

- Edit `inlist_project` to use a 60 $M_\odot$ star with the default value for the mixing length parameter.
- In `inlist_pgstar`, raise this flag to show profile panels:
  `Profile_Panels1_win_flag = .false.` $\longrightarrow$ `.true.`

We already filled in `r_in_rcore_units`, `logRho`, `opacity`, `Lrad_div_Ledd` and `pressure_beta` in the profile panel axis variables because we don't want to become repetitive.
(n.b. `r_in_rcore_units`, `Lrad_div_Ledd` and `pressure_beta` are not standard variables but are defined by us in the subroutine `data_for_extra_profile_columns` in `src/run_star_extras.f`.)

Next, we want to save the pgstar graphs every 10 timesteps so we can look back at the old profiles. They will be stored in the folder `png`.

- Save the history plots by adding the two lines:
  `History_Panels1_file_flag = .true.`
  `History_Panels1_file_cnt = 10`
- Save the profile plots by adding the two lines:
  `Profile_Panels1_file_flag = .true.`
  `Profile_Panels1_file_cnt = 10`
- Start the model with `./rn` and watch the profile functions.

After the run is finished, you can go to the `png` folder and look up snapshots of the simulation.

- Look in the `Profile_Panels` at two different timesteps: i) when the model is at half the total main sequence lifetime and ii) when it is at 90% of its main sequence lifetime. (Suggestions for commands to open a .png file: `display, gthumb, gimp, feh`)

Question: Do you see any correlating variables? If so, why do you think they are correlated?

Question: What is the difference between the models at $0.5\tau_{\rm MS}$ and $0.9\tau_{\rm MS}$?

Question: Why is the model at $0.9\tau_{\rm MS}$ more inflated?

2.2 The mass of the inflated envelope

We want to investigate how much mass is confined in all layers defined as the inflated envelope: $M_{\text{inflated env}}$. For this we need to define a new variable in `src/run_star_extras.f`, which we can call `inflated_env_mass` for example.

This might sound a bit time-consuming (especially if you are not familiar with `Fortran`) but fortunately there is a similar variable called `r_div_rcore` in `src/run_star_extras.f`. We use this variable as inspiration for the new `inflated_env_mass` variable.

2.2.1 Creating the new variable

To make our new envelope mass variable, we mimic the strategy that is used to create `r_div_rcore`. This variable is defined as the radius of the star divided by the radius of the core (i.e., everything but the inflated envelope).

- Open `src/run_star_extras.f`
- Tell the integer function `how_many_extra_history_columns` how many history columns we have now (5 instead of 4).
- In the subroutine `data_for_extra_history_columns`, add the line:
  `names(5) = "<your_variable_name>"`

Now we take a closer look at the `do` loop. The loop

```
do k=s% nz, 1, -1
```

loops the integer `k` from `nz`, which is the innermost numerical zone in the stellar model with the highest zone number, to zone `1`, which is the outermost zone. The loop takes steps of `-1`. Since we adopt the same strategy as for the `r_div_rcore` variable, we do not need to change this line and the next line

```
if ((s%Pgas(k)/s%P(k)) < 0.15) then
```

which allows for something to be done once we are in a numerical zone where $\beta < 0.15$ for the first time. However, what we do want to do is to add an extra line in the `do` loop: below this line

```
vals(4) = s% r(1) / s% r(k)
```

A small explanation: `s% r(1)` corresponds to the radius in the outer zone of the star; at the moment the β criterium is met, `s% r(k)` corresponds to R_{core}. (so $R_{\text{inflated env}}$ would be `s% r(1) - s% r(k)`).
(`s%` is necessary to point `MESA` to the file where the variables are stored)

- If there is no inflated envelope, $M_{\text{inflated env}} = 0$. Therefore, above the `do` loop, add the line
  `vals(5) = 0`
- The variable for the radius coordinate is `r`; can you guess the name of the variable of the mass coordinate?

- With the information given above, can you guess what would be the correct line for $M_{\text{inflated env}}$?
- `m` is in units of grams; divide by `msun` to obtain $M_{\text{inflated env}}$ in units of solar masses.
- Inside the `do` loop, add the line
  `vals(5) = <your_line_here>`
  that you thought of. You can also look up the correct line in the 'Hints' section at the last page of this file.

2.2.2 Setting up `inlist_project` and `inlist_pgstar`

In the next run you want to show the variable that you have created!

- In `inlist_pgstar`, add the line
  `History_Panels1_other_yaxis_name(1) = 'inflated_env_mass'`
  (or your own variable name if you thought another name was more suitable)

Since you have adjusted `src/run_star_extras.f`, you have to compile the work folder again before we can run the model.

- execute `./mk` and then `./rn`

Question: Describe the structure of your inflated models with simple terms.

Now we would like to see if there is a correlation between the inflated envelope mass and the value for `mixing_length_alpha`. Use the same value for `mixing_length_alpha` as you used in the last part of Minilab 1.

- Enter your value for the mixing length parameter in `inlist_project` and run the model
- Similar to what you did at the end of Minilab 1, obtain the final value for $M_{\text{inflated env}}$ and enter it in the Google spreadsheet document in the 'Mini2' tab, together with your `mixing_length_alpha`.

Question: Ask around again to learn what masses other students obtained. How do you interpret your result? What would you expect for adiabatic convection (`mixing_length_alpha` $\longrightarrow \infty$)?

3 Maxilab

We would like to test if the MESA models with an inflated envelope are unstable against pulsations when evolved at timesteps which are a fraction of the dynamical timescale.

To get started: Download today's maxilab at:
http://mesastar.org/teaching-materials/
Then, execute in your summer school directory
`tar -xvf <location>/maxilab_langer.tar.gz`
and enter the directory.

3.1 Prepare the model

3.1.1 Evolve the main sequence star

- Later on we want to know what the dynamical (free fall) timescale of the star is. MESA has a variable that estimates it as $\tau_{\rm dyn} = \sqrt{GM/R^3}$. Uncomment the variable `dynamical_timescale` in `history_columns.list`. If you don't want the column numbers of the other output variables to change, copy the new output variable to the bottom of the file and then uncomment it.

- Go to `inlist_project` and check that you evolve a 60 $M_\odot$ model. For the mixing length parameter, randomly take one of the following values: `mixing_length_alpha` $\in \{1, 3, 5\}$.

- We want to evolve a star on the main sequence until it has cooled to a specified effective temperature. You can *freely* choose between stopping at 40 kK or 35 kK. Look at `mesa.sourceforge.net/controls_defaults.html` again for inspiration for the correct stopping condition.

  (Small note for if your laptop is on one of the extreme sides in terms of speed: the 35 kK, $\alpha_{\rm MLT} = 5$ model takes the least steps to show pulsations and the 40 kK, $\alpha_{\rm MLT} = 3$ model takes the most.)

- Save the model by adding the following lines to the &`star_job` section:
  `save_model_when_terminate = .true.`
  `save_model_filename = '<your_alpha_mlt>_<your_model_temperature>kk.mod'`
  (or whatever you find appropriate)

- This is a new work folder, so clean and compile with `./clean && ./mk`

- Start the model with `./rn`

3.1.2 Creating a model with a short timestep

We set the timestep to a fraction of the dynamical timescale and check if we observe pulsations. The ∼minute timesteps we use to search for pulsations are almost ten orders of magnitude smaller than the timesteps that we saw during the main-sequence evolution. Therefore, we first run a model until the timesteps are at their low desired value.

- Go to the data file `LOGS/history.data` from your last simulation to find the dynamical timescale of the stellar model at the last timestep.
- In the &`controls` section in `inlist_project`, declare a value for the maximum timestep of $\tau_{\text{dyn}}/250$ (you know where to look now for the 'maximum timestep' variable!).
- Disable the lower limit on the effective temperature that you set earlier, otherwise the run will be terminated before the start.
- Let the simulation stop after ∼50 timesteps by changing
  `max_model_number = 10000`
  to the current model number + 50.
- Load the previously saved model by adding the following lines to the &`star_job` section:
  `load_saved_model = .true.`
  `saved_model_name = '<your_alpha_mlt>_<your_model_temperature>kk.mod'`
  (or whatever you found appropriate)
- Also save the model by changing the 'save file' lines to the following:
  `save_model_when_terminate = .true.`
  `save_model_filename = '<your_alpha_mlt>_<your_model_temperature>kk_stabledt.mod'`
  (or whatever you find appropriate)
- `./rn` the model.

3.1.3 Start the search for pulsations

Now we are ready to investigate the possible pulsations of the stellar models! This simulation might take a while so we would like to have it running before the intermezzo.
In the &`controls` section in `inlist_project`,

- Set the maximum model number to 10000 (or more).
- The amount of change of the chemical composition in the stellar model is negligible on the timescale on which we evolve it now. Therefore, we might as well add the following line:
  `dxdt_nuc_factor = 0`
  Which disables changes in composition (but not in nuclear energy generation rate). As a result, the simulation runs a bit faster.
- Load the previously saved model with the following lines in the &`star_job` section:
  `load_saved_model = .true.`

`saved_model_name = '<your_alpha_mlt>_<your_model_temperature>kk_stabledt.mod'` (or whatever you found appropriate)

- `save_model_when_terminate = .true.` can be set to `.false.`
- And `./rn`

Now the models are running, check how fast they are going (remember that the last run did not start at model number 1!). If they are slower than one model per second, it will probably take a bit long before the pulsations appear compared to the maxilab timescale. If this is the case, it is advisable to grab one of the models that have been evolved by us that are on the verge of starting the pulsations.

If that is the case:

- Discontinue the simulation with `ctrl + c`
- From `mesastar.org/teaching-materials`, download the `⋆_almost_pulsating.mod` file with the the same mixing length parameter and effective temperature as you used and copy it to your work folder.
- In `inlist_project`, point the model to load to this file.
- `./rn` the model.

Intermezzo

3.2 Study the pulsations

Once the model starts to show pulsations, it is possible to study some of the characteristics. From the $R(t)$ history plot, it is possible to estimate the e-folding time k and the pulsation period $P_{\rm puls}$.

- The e-folding time k is defined as the time that it takes for the pulsation amplitude to grow a factor e in size. Estimate the value for k.
- Try to estimate $P_{\rm puls}$ as accurately as you can by using the `xmin` and `xmax` variables for your History_Panels1; see `mesa.sourceforge.net/pgstar_defaults.html`. Use another terminal to adjust `inlist_pgstar` in your work folder.
- In the 'Maxi' tab in the Google doc, add the values that you obtained for k and $P_{\rm puls}$. Also mention the $T_{\rm eff}$ at which you interrupted your main sequence model and the mixing length parameter that you used.

- We are curious if P_{puls} changes with time. Could you check this for us? If it is not constant, how does it change?

Now let's look at some other model parameters.

- In `History_Panels1`, also plot the variable `max_L_div_Ledd`. How are the phases ϕ correlated of the stellar radius and Γ_{max}? (it could again be helpful the adjust the limits of the x-axis to see things more clearly)

  Question: In the equation $\phi_{\Gamma\text{max}} = \phi_{\text{radius}} + a$, what would be a? Or in other words, what is the phase lag of Γ_{max}?

  Question: How does the pulsation amplitude evolve in the long term? If `MESA` would describe the pulsations correctly, what would you expect to see in a real star? (You can check this by plotting other variables instead of/in addition to `max_L_div_Ledd`)

Hints

Section 1.1: Add the following lines to the &controls section in inlist_project:

```
xa_central_lower_limit_species(1) = 'h1'
xa_central_lower_limit(1) = 1d-3
```

Section 2.2: This is a correct version:

```
names(4) = "r_div_rcore"
names(5) = "inflated_env_mass"
vals(4) = 1
vals(5) = 0
do k=s% nz, 1, -1
   if ((s%Pgas(k)/s%P(k)) < 0.15) then
      vals(4) = s% r(1) / s% r(k)
      vals(5) = (s% m(1) - s% m(k)) / msun
      !the line below is for later use in data_for_extra_profile_columns
      r_div_rcore = s% r(1) / s% r(k)
      exit
   end if
end do
```