Stellar Mixing: Day 1

Pascale Garaud and Chris Mankovich

MESA Summer School, 8/12/2014

Minilab 1: Lithium Depletion in Hyades

- Download and untar `garaud_day1.tar.gz` from the Teaching Materials section of mesastar.org. Start in the `ms/` work directory. Here the master `inlist` just includes `inlist_project` and `inlist_pgstar_ms`.
- The default nuclear network `basic.net` only tracks 8 isotopes. Instead use an extended net by adding these lines to `&star_job` (copy+paste at your own risk):

  ```
  change_net = .true.
  new_net_name = 'pp_and_cno_extras.net'
  ```

  As the name suggests, this network adds some species and reactions that play a role in pp and cno chains, including lithium-7, carbon-13, and more.
- The included `history_columns.list` already includes columns for surfaces abundances, and the included `inlist_pgstar` has some plots set up for you to track relevant quantities during the evolution. There are a few potentially mysterious controls already included in your `inlist_project`; see the comments or ask Chris why those are there. log surface li7 mass fraction will also be output in the terminal.
- Tell `star` to start from a homogeneous pre-MS model by including the line

  ```
   create_pre_main_sequence_model = .true.
  ```

  in `&star_job`.
- Generate a random number between zero and one, and map linearly to the interval [0.5 $M_\odot$, 1.5 $M_\odot$] to determine your initial mass. Set a maximum age equal to the Hyades age (625 Myr). Search in `star/defaults/controls.defaults` to figure out what the exact control is called.
- **Can you explain your final surface lithium abundance relative to that found by your neighbors?** You can revisit your saved `pgstar` output under the `png/` directory—compare histories with your peers!
- **Convert your final surface lithium mass fraction to `N(Li)` and enter your final `log_Teff` and `N(Li)` on the google sheet.** Note the definition $\mathrm{N(Li)} = 12 + \log_{10}(\mathrm{X_{Li}})$, where X_{Li} is the mass fraction.

Minilab 2: Lithium Depletion in the Sun

- Keep your current inlist, but to save time just run from the included 1.0 $M_\odot$ ZAMS model `m1_z02_zams_extras.mod`. This model was evolved through the pre-MS phase with lithium and friends included. e.g. in `&star_job`, modify:

  ```
  create_pre_main_sequence_model = .false.
  load_saved_model = .true.
  saved_model_name = 'm1_z02_zams_extras.mod'
  ```

- Enable overshoot below the outer convection zone. Search in `star/defaults/controls.defaults` for the appropriate control. (Hint: the OCZ is a 'nonburning' convection zone.) For the overshoot parameter `f`, which represents the *e*-folding length for the overshoot diffusion coefficient in units of the local pressure scale height, use a (truly!) random value in the interval [0.05, 0.10].
- Set maximum age equal to the solar age (4.6 Gyr) and run. Watch the Mixing plot in particular. **Does your value of `f` succeed in reproducing the observed solar `N(Li)`? Enter your overshoot `f` and final surface `N(Li)` in the google sheet.**

Long Lab, part one: Canonical Mixing in RGB stars

- `cd` into the `garaud_day1/rgb` work directory.
- Again include overshoot below the OCZ, but this time with a more modest `f` value of 0.01. (If you need a reminder of what the control is called, consult your `ms/inlist_project`.)
- Run from the included ZAMS model `m1_z01_zams_extras.mod`, which is just like the model we used in the minilab except at half solar metallicity. This time include no age limit.
- Add an upper luminosity limit $\log L/L_\odot = 1.5$ as your stopping criterion, and run while we continue the lecture.
- This time the main `pgstar` window includes the wonderful Kippenhahn diagram, which shows mixing and burning throughout the star as a function of time. **Watch for the first dredge-up, when the inner boundary of the OCZ reaches a minimum in mass coordinate before it starts receding again. Note: you'll likely need to zoom in in mass coordinate, using `Kipp_mass_min` and `Kipp_mass_max`. You can make changes to `inlist_pgstar` in real time!**
- **The run should stop a little after the luminosity bump, which is when the H-burning shell reaches a mass coordinate which was previously part of the OCZ and stellar luminosity *decreases* a bit. We'll use a recent saved photo for the next exercise.**

Long Lab, part two: Fingering Convection in RGB Stars

- Modify your inlist to allow fingering ("thermohaline") mixing. Pick one of two prescriptions:
  1. Kippenhahn, Ruschenplatt, & Thomas 1980
  2. Brown, Garaud, & Stellmach 2013

For the efficiency parameter thermo_haline_coeff, choose a random value logarithmically distributed over the interval [10^1, 10^4] for Kippenhahn, or [10^0, 10^2] for Brown et al.

- MESA star saves restart photos every so often. Restart from a saved photo from just before the luminosity bump. (Hint: to decide which photo, open the most recent saved pgstar plot in your png/ directory and consult the Kippenhahn plot. Pick a model number just before stellar luminosity started decreasing. You would then restart from model 1700, say, by running ./re x700 in the command line. To avoid this photo being overwritten 1000 steps from now, it is wise to copy it now, e.g. via cp photos/x700 photos/before_bump and you'd restart accordingly with ./re before_bump.)
- Run until $\log L/L_\odot = 1.7$, and watch the Kippenhahn and Surface Abundance plots. **Does material from the H-burning shell get mixed up to the surface? How do surface abundances (Li) change?**
- Enter final surface li7 mass fraction, final log_L, and fingering prescription + efficiency, in the google sheet.

Long Lab, part three: Add-your-own-mixing

Disable any fingering convection option in your inlist. The goal is to use the other_D_mix hook to set up your own routine to implement chemical transport via the horizontal instrusion instability wherever both of the following criteria are met:

1. the presence of an inverse composition gradient, i.e. $\nabla_\mu < 0$, and
2. stability with respect to the Ledoux criterion, i.e. $\nabla < \nabla_L$, where $\nabla_L \equiv \nabla_{\rm ad} + \frac{\varphi}{\delta}\nabla_\mu$.

Look in star/other/other_D_mix.f for a skeleton subroutine. Useful comments are absent from that file in the present version, but refer to star/other/other_am_mixing.f for a perfectly analagous routine; this file does include helpful comments.

The other_D_mix hook only passes you id, which you can use to get the star pointer with a single call to the routine star_ptr from star/public. For more details on this and run_star_extras in general, consult Josiah's tutorial at http://mesa.sourceforge.net/run_star_extras.html or ask your nearest/favorite TA.

Your local copy of src/run_star_extras.f already includes a subroutine called get_diff_coeffs which uses data from the star_info structure to calculate:

- the thermal diffusivity κ_T,
- the molecular diffusivity κ_μ, and
- the viscosity ν.

Also included is a function numu(R0,r_th,Pr,tau) which returns the ratio Nu_μ of macroscopic transport diffusivity to the molecular diffusivity. The quantity R_0 is defined by

$$R_0 = \frac{\nabla - \nabla_{\rm ad}}{\frac{\varphi}{\delta}\nabla_\mu}, \tag{1}$$

where the numerator handily lives in the star_info pointer as s% gradT_sub_grada and the denominator is likewise available and is called s% gradL_composition_term. The quantities $r_{\rm th}$, Pr, and τ are defined by

$$r_{\rm th} = \frac{R_0 - 1}{1/\tau - 1}, \quad \mathrm{Pr} = \frac{\nu}{\kappa_T}, \quad \tau = \frac{\kappa_\mu}{\kappa_T}. \tag{2}$$

In both cases you'll need to calculate

$$\mathrm{Nu}_\mu^{\max} = \mathrm{Nu}_\mu(R_0 = 1,\, r_{\mathrm{th}} = 0, \mathrm{Pr}, \tau). \tag{3}$$

1. If you want to implement the simplest case, the diffusion coefficient for chemical transport will just be
$$D_{\mathrm{new}} = \mathrm{Nu}_\mu^{\max}\kappa_\mu. \tag{4}$$

2. More experienced users should try the harder, somewhat more realistic case in which the macroscopic diffusivity depends on layer height L relative to the finger size d, for example via
$$D_{\mathrm{new}} = \mathrm{Nu}_\mu^{\max}\kappa_\mu \tanh\left[\frac{L}{4d}\left(\mathrm{Nu}_\mu^{\max}\right)^{-1}\right]. \tag{5}$$
You can either treat the nondimensional number L/d as the free parameter, or you can calculate the (dimensional) finger size d from local quantities:
$$d = \left(\frac{\kappa_T \nu H_P}{\delta g(\nabla_{\mathrm{ad}} - \nabla)}\right)^{1/4} \quad \text{and} \quad \delta \approx 1 \text{ for an ideal gas.} \tag{6}$$
If you do this calculation using variables from the `star_info` pointer, you'll get out a d in centimeters—make sure you're thinking in consistent units when you choose a value for L.

   Whether you choose to treat the ratio L/d or the length L in cm as your parameter, you can add an inlist control for the parameter by setting e.g. `x_ctrl(1) = 1d2` in your `&controls` namelist, and grabbing that value in your routine using `s% x_ctrl(1)`. This way you can change the value for your free parameter at the inlist level and avoid recompiling many times.

When you have a D_{new} in hand for each zone in which the instability is active, you can finally make the change by adding the new diffusion coefficient to the existing `D_mix`. For instance, for an unstable zone with index `k`:

```
s% D_mix(k) = s% D_mix(k) + D_new
```

As a final tip, flag the zones as 'thermohaline' zones via

```
s% mixing_type(k) = 4
```

and they will appear in the Kippenhahn and Mixing plots with a pinkish-purplish color.

Remember to set `use_other_D_mix = .true.` in your inlist. Again restart from your photo saved before the luminosity bump. Again run to $\log L/L_\odot = 2$. **Does the new transport mechanism link the H-burning shell to convective envelope? Is the transport efficient enough to change surface abundances? How does your final surface Li abundance compare to the existing thermohaline implementation you tried before?**

Some more specific hints to help if you are stuck:

1. You need to loop over all zones and ask whether the current zone fulfills both criteria for instability. Simple `if` statements will do the trick.

2. Zones `k` with an inverse composition gradient will have `s% gradL_composition_term(k)` less than zero.

3. You can check for Ledoux stability by comparing the gradients `s% gradT` and `s% gradL` directly.

4. Look at the definition of the subroutine `get_diff_coeffs` to understand the call structure. All of the variables it takes as inputs can be found in the `star_info` pointer—ask a TA if you're stuck looking for one.

5. Setting `s% mixing_type(k)=4` for the zones in which your extra mixing is active will let you visualize the new mixing as purple regions on the Kippenhahn and Mixing plots.