A GUI FOR EVOLVE ZAMS

D. R. Schlegel
Computer Science Department

Here the early work on a new user interface for the Evolve ZAMS stellar evolution code is presented. The initial goal of this project is to provide an easy way to generate and visualize stellar evolution tracks for use in the classroom or lab. Eventually the code will allow for more complex analysis looking at star models as they evolve along an evolutionary track, becoming more useful for the researcher.

I. Introduction

A study of stellar evolution intends to discuss how a star changes throughout its lifetime. This can be a very difficult subject governed by so many equations that it would be nearly impossible for someone to accurately determine the evolution of a star without the use of a computer program. A computer program can often make the process much less painful but an experiment can still be difficult to set up requiring a lot of scripting and in depth knowledge of both stellar evolution and the computer code you intend to use.

The code which was used for this project – Evolve ZAMS, or often just called EZ – is no different than this. Like most research software it shows signs of being written for a specific purpose, and trying to do anything outside of that is very difficult. Therefore it is not very user friendly and the learning curve is very steep. In order to run a simulation you must of course know the parameters of the star to wish to simulate, but you must also understand how EZ works.

After one has figured out how to use EZ and they have produced output (in the form of a tab-delimited table printed to the console), the task then becomes analyzing the data. The tables generated are often 20 columns wide and 50 lines or so, intermixed with debug statements about the status of the star on its evolutionary track. This is great if perhaps you are analyzing the age of a star at which it leaves the main sequence which is only one point, but many times a graph is desired.

My goal with this project was to make this process as a whole something which is easy for the user to do. My aim was to provide an end-to-end solution to the user, beginning with a graphical interface for setting variables about the star, and ending with a tabular look at the two variables the user chose with the option of generating a GNUPlot script which can be easily loaded.

II. Evolve ZAMS

Evolve ZAMS itself is a stellar evolution code written by Bill Paxton from the University of California at Santa Barbara. The evolution code itself is derived from a one dimensional code written by Peter Eggleton. As the name of the code implies, the program evolves stars from the Zero Age Main Sequence (ZAMS) to a specified stopping point (Paxton, 2007).

The application uses simulation files which are compiled FORTRAN code calling each module required individually. This is more difficult to use than a scripting language since there are preconditions and postconditions to take into account with each module. Not limiting the user to a predefined set of commands provides a lot of flexibility for the user if they are willing to become acquainted with both FORTRAN and the modules available in the code.

The problem with this though is that the documentation available for EZ is slim to none. The main form of documentation is in the two demo simulations distributed with the code. These can be easily modified to change the mass of a star and some other basic variables but they do not do everything a user might be looking for, and they don't make it clear how they do much of anything. In order to produce results the user really wants they need to dig into the FORTRAN code, for which there is a browser supplied on the web site.

III. Basic Stellar Plots

Someone studying stellar evolution very often does not use just numerical values to do their analysis, graphs are very common. As a star evolves a great deal of variables need to be taken into account – from the temperature at the very center of a star, to the concentration of Helium in the star. There are a lot of very interesting plots possible, but perhaps the most common one is plotting a stars evolution on a HR – or Hertzsprung-Russell - diagram.

In its most basic form, an HR diagram plots the logarithm of the luminosity of a star (measured in terms of the luminosity of our sun) versus the log of the temperature of a star in a reversed axis. An example of this is provided in Fig. 1.

This figure shows this relationship for many stars during a single snapshot of time. From the image you can see where the main sequence lies, and where some stars which have evolved off of the main sequence now reside. While this plot can come largely from observational data, on these same axes a plot can be made of a single stars evolution from the main sequence through its different evolutionary phases.

Shown in Fig. 2 is an example of one such plot. A plot like this is impossible to arrive at observationally since stars exist for billions of years, but it can be arrived at mathematically and therefore, computationally. A plot like this shows much of how a stars appearance to the outside world will change as it evolves (since luminosity is related to brightness, and surface temperature is related to color).

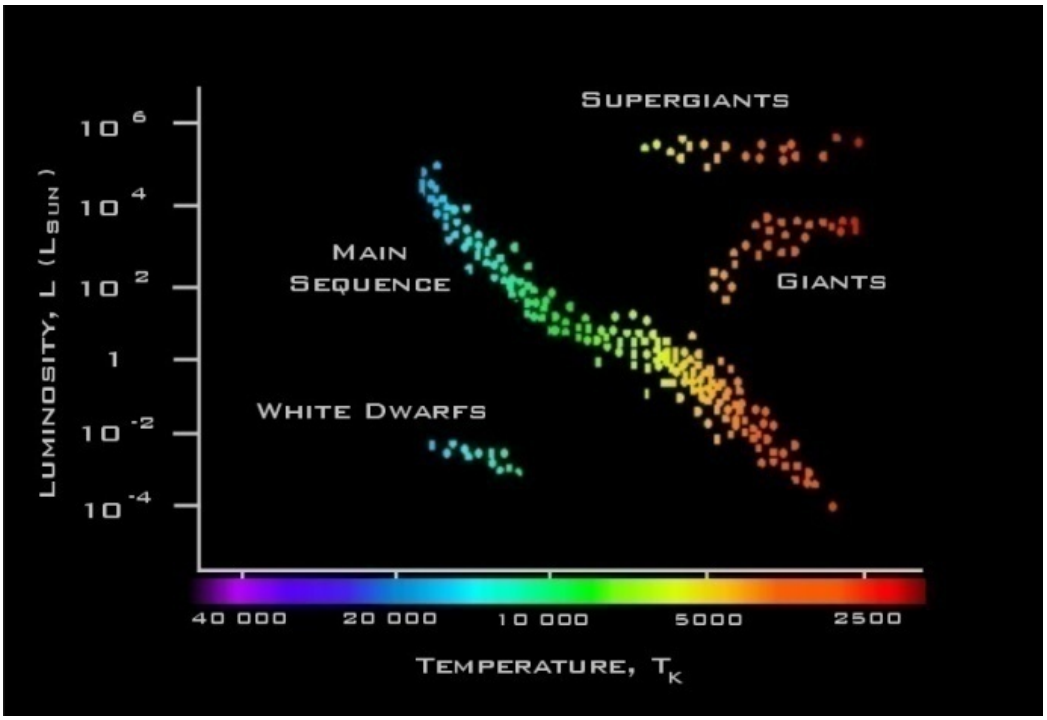

Fig. 1: HR Diagram (Brinkworth & Thomas, 2001)

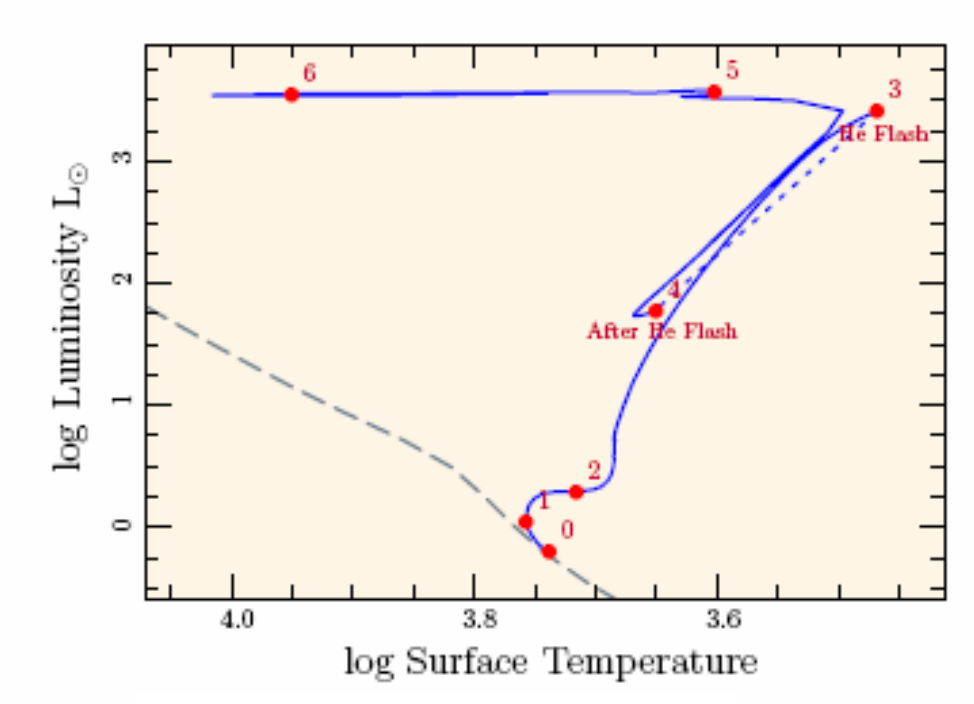

Fig. 2: Star Evolution (M=1.0, Z=0.3) (Paxton, 2007)

IV. The Project

The concepts behind stellar evolution can be very difficult to understand, but I feel that the software to simulate it doesn't have to be. The goal of this software is for it to be able to be used by students, professors, and researchers in their generation and analysis of stellar evolution graphs. The project design in general consists of three phases – generating/running the simulation (input), allowing the user to choose the data which is important to them, and the output phase in which a graph is created.

The first phase is that which abstracts the user from having to deal with the intricacies of the EZ stellar evolution package itself. The interface provides to the user options so that they can set the mass, metalicity, etc of the star they wish to evolve. Based on the user's selections the program automatically generates the FORTRAN code that EZ requires to run a simulation. This is compiled and run by the GUI, never requiring the user to touch a command prompt.

One the simulation has finished (after a couple of minutes usually), the tabular format of the data is broken down into its individual parts automatically. EZ tries to make itself more user friendly by dispersing throughout the output table comments about when the star is leaving the main sequence, when the helium flash occurs, etc. The program therefore has to clean up this data before it is presented to the user for use in graphing.

Most of these messages dispersed throughout the output can actually be turned off by editing the source code sufficiently, but this is not necessarily desirable. As the project evolves it is very likely that it will be desirable to plot these events on the graph, so keeping them around is in our best interest.

Once the data has been parsed it is presented to the user. The user chooses from drop down boxes which data they would like to plot (the column headers are preserved to make this easier). As this is done a preview of what the results look like graphically are shown to the user in a preview pane. This can help reduce errors choosing X and Y properly.

Finally once the user has selected what data they are interested in and selected any plotting preferences (such as reversing an axis or setting the ranges manually), the data can be output to a GNUPlot script. This includes both the data file and a script to load it and set the preferences the user selected.

V. Guaging Success

In order to call this project a success it was decided that it needed to plot some basic stellar evolution paths without forcing the user to touch any command line tools. While the project is in desperate need of usability testing, these goals appear to have been met. Fig. 3 is an evolution path created completely using this user interface for a star similar to our own sun.

Creating a graph such as this is very straight forward for anyone, even someone who knows nothing about how stellar evolution actually works. The goal has been to provide as many options to the user as possible while not asking the user to fill in many fields manually. This makes it so the user always knows what the possible values are and doesn't let the user "mess up." As time goes on and the amount of options increases one goal of the application should be maintaining this ease of use.

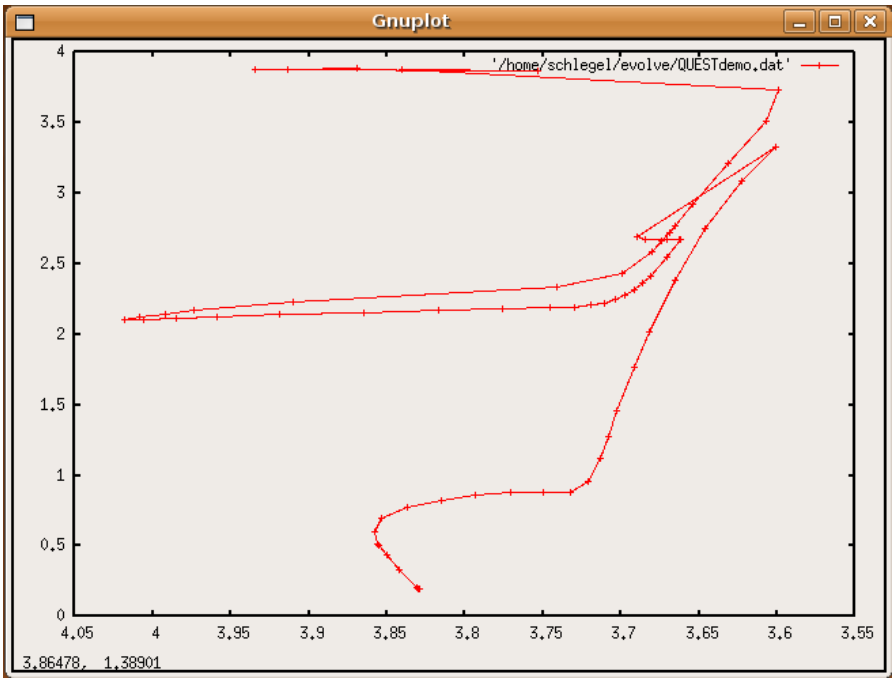

Fig. 3: Evolution Path (M=1.0, Z=0.2)

In addition to this, the application also has to provide active feedback to the user about what they are doing. To do this the program has a part of its display known as the "graph preview area." As the user is selecting what they want to plot this area actively updates showing the points to be plotted. Obviously it is not as sophisticated as a tool like GNUPlot, but it lets the user see what they are doing before they spend the time generating a GNUPlot script, which the program also automates.

VI. Future Work

The system presented here is still in its infancy but already could prove useful in perhaps an introductory astronomy or astrophysics course as an instructional tool. That said though, there are much more ambitious goals for this project as time goes on. These range from simple additions to the plotting abilities, to much more complex analysis of the models of stars as they evolve.

The first of these features I would like to see added is the ability to note important points along the evolutionary curve much like those in Fig. 2. This would increase the utility of the diagrams for physicists interested in a specific phase of evolution specifically.

The much larger and more significant long term goal for the project is examining the model of the star at points along the evolutionary path. EZ allows the user to specify in the

design of an experiment at what points they would like to save model data of the star, but this function is not currently used by this application.

This would obviously allow us to create much more advanced plots. An example of one such plot which one might generate is a plot of log ($[M - M_R]/M$) *vs T*. Specifically of interest in this plot is the photosphere (denoted by the black dots in Fig. 4) and determining its interaction with the hydrogen ionization front as the star evolves.

The real challenge here will not be the actual code generation itself I think, it will be providing a consistent and easy to use interface. The more functions and features that are added that don't necessarily fit to the same purpose, the more cluttered and complex the interface will be come.

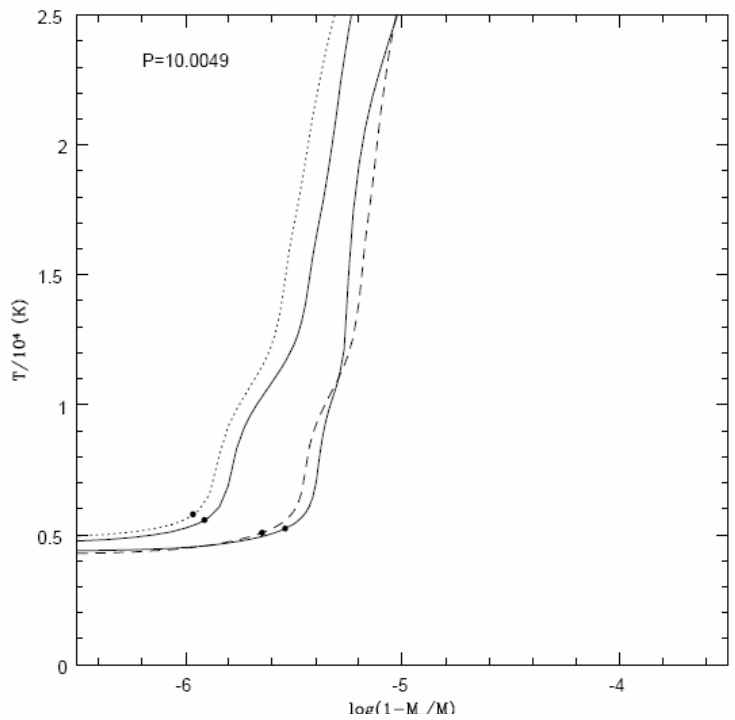

Fig. 4: Analysis of Mass at a Radius (Kanbur, Ngeow, & Buchler, (n.d.))

VII. Conclusion

Scientific research tools have a reputation for being extremely difficult to use. This learning curve makes it very difficult for an undergraduate student who perhaps does not have the UNIX background to use these tools. In turn, most research tasks are given to a graduate student with more experiance. Text based systems do have their appeal for power users, but for the casual user or for a student who is interested in only a certain function of the software a user interface is a much nicer presentation.

While still in its early form this project is already a useful tool for generating evolution paths of stars very quickly. I hope that this project has a long life ahead of it and that it will inspire others to create easy to use interfaces for research software. The amount of development time spent creating something like this will in the end save much more time for the scientific community as a whole and produce better researchers earlier in their college career.

VIII. References

Brinkworth, C., & Thomas, C. (2001). *Stars - An Overview*. Retrieved April 20, 2008, from http://www.le.ac.uk/ph/faulkes/web/stars/o_st_overview.html

Kanbur, S. M., Ngeow, C.-C., & Buchler, J. R. (n.d.). Period-colour and amplitude-colour relations in classical Cepheid variables II: the Galactic Cepheid models.

Paxton, B. (2007). *EZ Stellar Evolution*. Retrieved April 20, 2008, from http://www.kitp.ucsb.edu/~paxton/EZ-intro.html