Modeling Stars:
Two Method Comparison and Analysis

Najla Alahmadi, Michael Cock, Rebecca Halloran, Brady Metherall, Hector Robinson

March 16, 2017

1 Introduction

In this project, we modelled a spherically symmetric, static star by numerically integrating the four stellar equations. We started by doing this for a star that had the same core conditions as the sun, and then changed the conditions to determine final values for many other stars. We had to be careful with the initial conditions we used to ensure that the temperature and pressure went to zero at the surface. Using the results, we can plot the relation between luminosity and temperature of the stars to create an Hertzpring-Russell Diagram (HR diagram). We created code that would be able to take a stars composition, mass and some initial conditions to determine effective temperature, surface luminosity and radius so we can plot the surface luminosity versus the effective temperature or an HR diagram. Before we get there, lets talk about our program that will numerically integrate one star.

2 Method

To perform the calculations we started from the stellar core and integrated outward using the fourth order Runge-Kutta method. We specified a core pressure, temperature, density, and composition. From there, the outward integration calculates mass, luminosity, pressure, and temperature at every other radius. The integration stops when the boundary conditions are met, that is, the pressure or temperature goes to zero. Ideally they both go to zero at the same radius, if your initial conditions are valid. Once this point is reached it means we are at the surface, and have our final values. Figure 1 shows our main code that ran the simulations. Once we have this working for one star using the sun's conditions, we can extend the code to work for many stars. The way to do this is simply by changing the initial conditions. All of our stars had the same compositions; what we changed was the initial core temperature and pressure. We multiplied the sun's core temperature and pressure by different values to get different stars.

3 Results

Figure 2 shows our results using the initial conditions of the core of the sun. These plots have the general trend we would expect for a 1 solar mass star, but they do not get the exact final values of the sun. One thing that differs from our star to the sun is that it radiative transport only occurs for two radius steps, which is about 2×10^5 meters, and convective transport occurs for the rest of the star. This is in contrast to the sun, where the radiative zone is about 70% of the whole star. Because of this, the boundary conditions were not met perfectly; the surface temperature should go to zero but our pressure hit zero first, triggering the integration to stop every time. The mass of our star was about 0.78 $M_\odot$, it's radius was only around 0.38 $R_\odot$, which is considerably smaller, the luminosity was about 1.8 $L_\odot$ and there was a surface temperature of 8600 K. If the model was perfect, the final values for mass, luminosity, temperature, and pressure would be $1M_\odot$, $1L_\odot$, 5770 K, and 0 Pa, respectively.

```
#Define all our constants. This is where we pass X, Y, Z
initialize.init(0.7,0.25,0.05)
val = np.logspace(-5., 5., 100)
count = 1
r_end = 10**10 #in units of m

for i in val:

    print i, "solar initial conditions"
    print count, "out of 100"
    count += 1
    # Initial core conditions of the star
    P0 = i*2.477 *10**16. # units of pascal
    M0 = 10**-4 # units of kg
    L0 = 10**-4 # units of J/s or W
    T0 = i*1.571*10**7 # units of K
    rho0 = i*1.622*10**5 #units of kg/m^3

    data = open('stars.dat','a')

    P,M,L,T,r,kappa_array,rho_array = RK(10**-6, r_end, 10.**5., P0, M0, L0,
    T0, rho0, dP, dM, dL, dT, rho, kappa, stop, 0.000001)

    print >> data, r[-1], M[-1], T[-1], L[-1]
    data.close()
```

(a) First *main.py* calls *initialize.init* to define our constants, then the core conditions are specified. We then open our data file, run our Runge-Kutta method, and then print the results to the data file.

Figure 1: main method of the code.

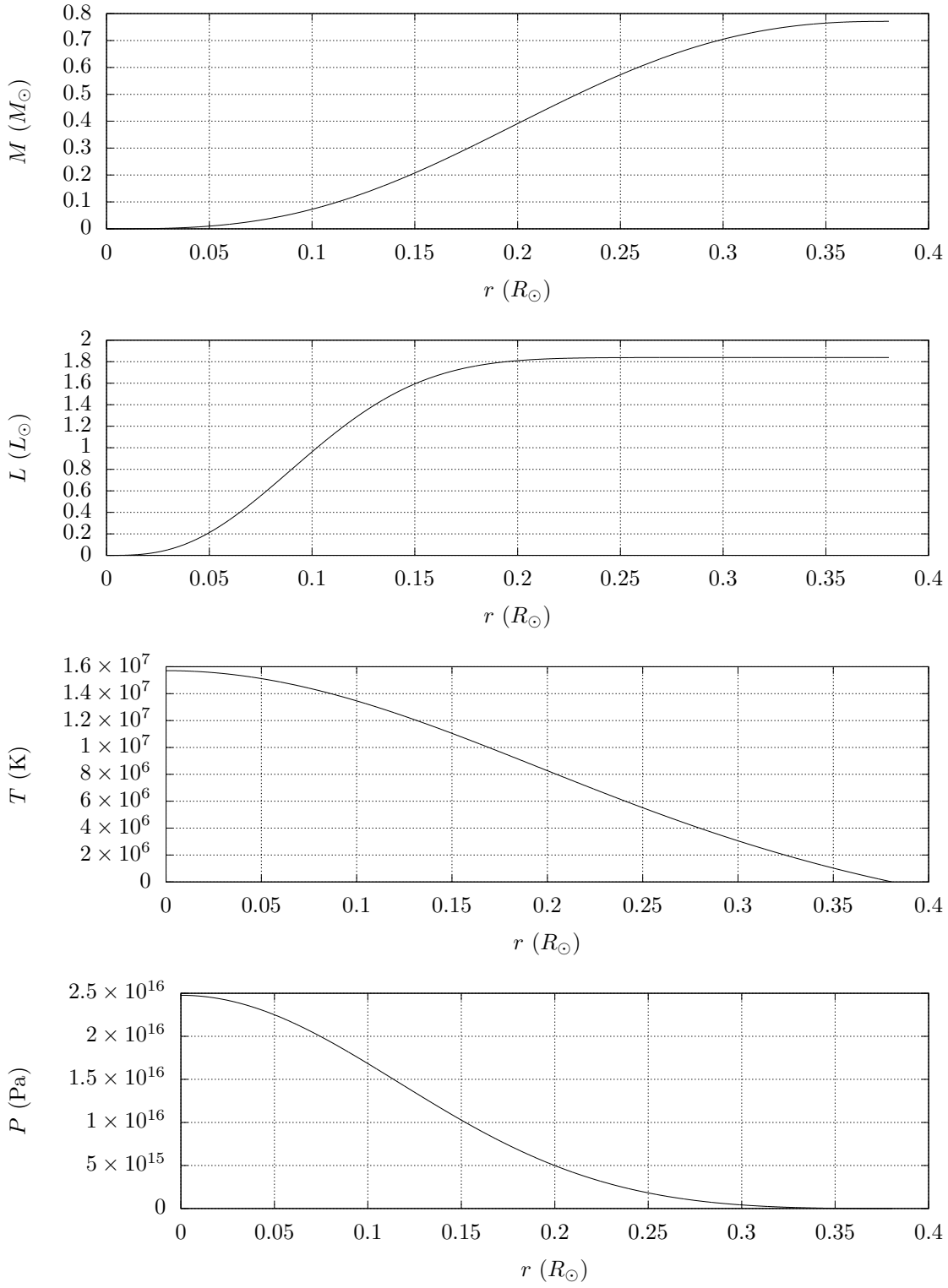

Figure 2: Simulation results of the Sun.

Knowing our model works generally, but not perfectly, we go ahead and use it on other initial conditions, to create other stars. Doing so, should produce a general form of the main sequence on an HR diagram. An HR diagram is a log plot of the luminosity vs. effective temperature. The effective temperature is observationally found by matching the color of a star with the temperature on a blackbody curve that emits that color the most. Due to this, the temperature measured observationally it is not the surface temperature, rather, the temperature from a depth related to the optical depth inside the star. Our attempts to calculate the optical depths of our stars resulted in wildly large values that were thousands of times larger than the star itself. However, during our integrations the pressure was going to zero, and the surface temperature did not. We then decided to use this temperature in place of the effective temperature. This is not perfect, but should still give us a good relationship between different stars temperatures. Our first attempt at an HR diagram used all of the raw data and is shown in Figure 3.

The first obvious thing that you'll notice is that the stars seem to have four distinct categories. A small group with high temperatures and luminosities, two groups with low temperatures and luminosities, and a large group somewhere in the middle. To help make some sense of what was going on, we have included the radii of the stars on a new plot, shown in Figure 4.

In this plot, we see that the two groups at the bottom have either large, or super small radii. The group with small radii are around $0.001R_{\odot}$ and the larger group are on average $10R_{\odot}$. Both of these groups also have very low luminosities. The group near the top has very high surface temperatures, they range between 10^8 and 10^9 K. This is about 100 times hotter than the core temperature of the sun. The one group in the middle however, shows promise. This luminosity of this group increases with temperature, and their radii increase gradually as well. Furthermore, this group exists in about the same region we would expect to find stars on an actual HR diagram. What we do is toss out all the data from the three other groups as they seem to be working past the limits of our model and are unphysical. The remaining region is focused in on and we reran the simulations to get more precise data here. We made two seperate plots of our final HR diagrams, one comparing it real star data as well as MESA (Modules for Experiments in Stellar Astrophysics) data (Figure 5), and the other showing the mass and radii of our simulated stars (Figure 6).

We can see that the shape resembles the main sequence part of the known HR diagram. We can explore a well known program to get a sense of what the HR diagram should look like and compare it to our own model. We will use MESA to do the comparison. MESA will give us a plot of the total life of a star. We can compare where our model shows the stars on the main sequence and we expect the MESA star to be on the main sequence at some point in its life. MESA considers different characteristics that we do not consider in our model. Our model is static, has no rotation and it is only for main sequence stars and MESA will allow for rotation and it does a full life of a star so it is not static and will not only be on the main sequence.

4 Discussion

The main goal of this project was to create an HR diagram by simulating stars and plotting the data. To do this we needed to complete many smaller tasks along the way. The first objective for our group was to get a profile on GitHub and ensure everyone was able to to push and pull properly. At first, there were of course a few troubles with ensuring our local versions remained up to date while making changes. Eventually, everyone was able to pull, add, commit then push a change with no problems. Splitting up work in a fair and convenient way was a small challenge, as it was difficult to work around each others schedules. Working in a large group made it easier to write the code because we were all able to bring our ideas together and come up with solutions faster. We learned to work as a group in an efficient way, even though our ideas would not work sometimes. We were able to listen to everyone and everyone was able to contribute. Different people have different styles of coding; when working in a group of five, each person had to learn to read and interpret each others' code. At the same time, we learnt to comment our code in a way which would allow the group members to understand what had been written and why. Overall, great project, had a blast working on it 10/10!

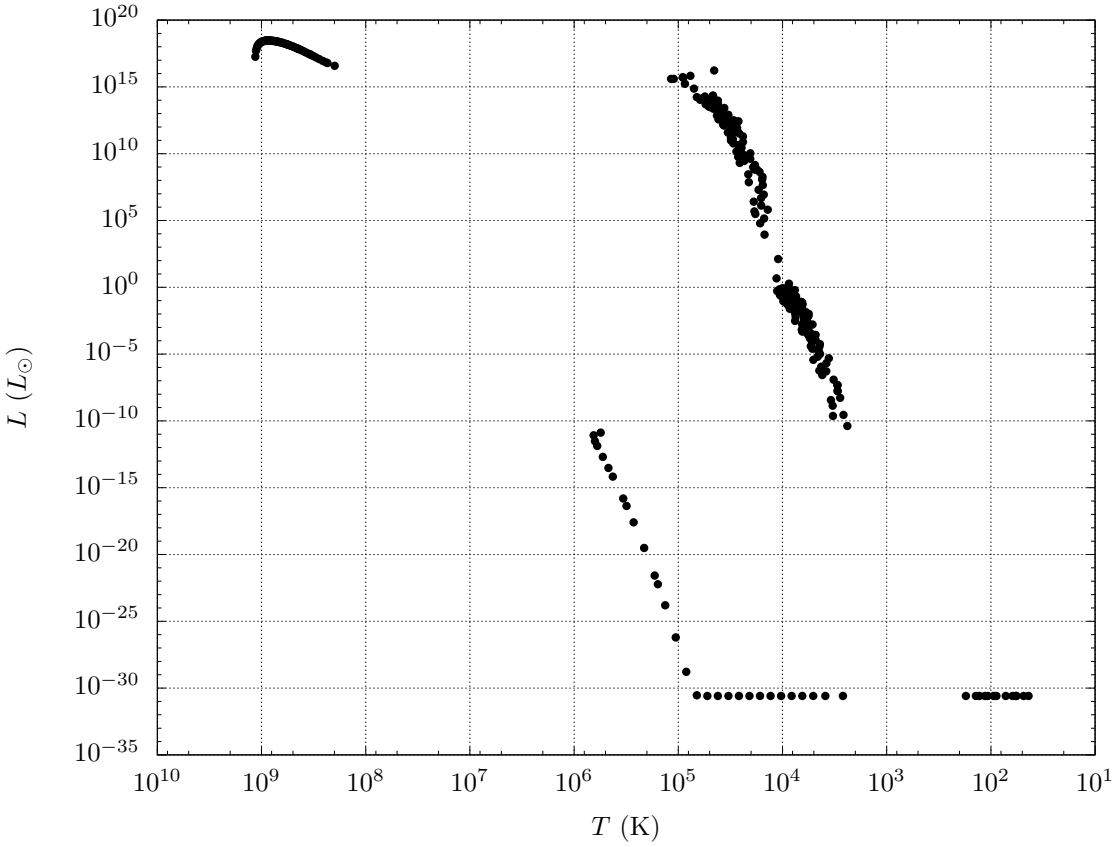

Figure 3: A crappy version of HR

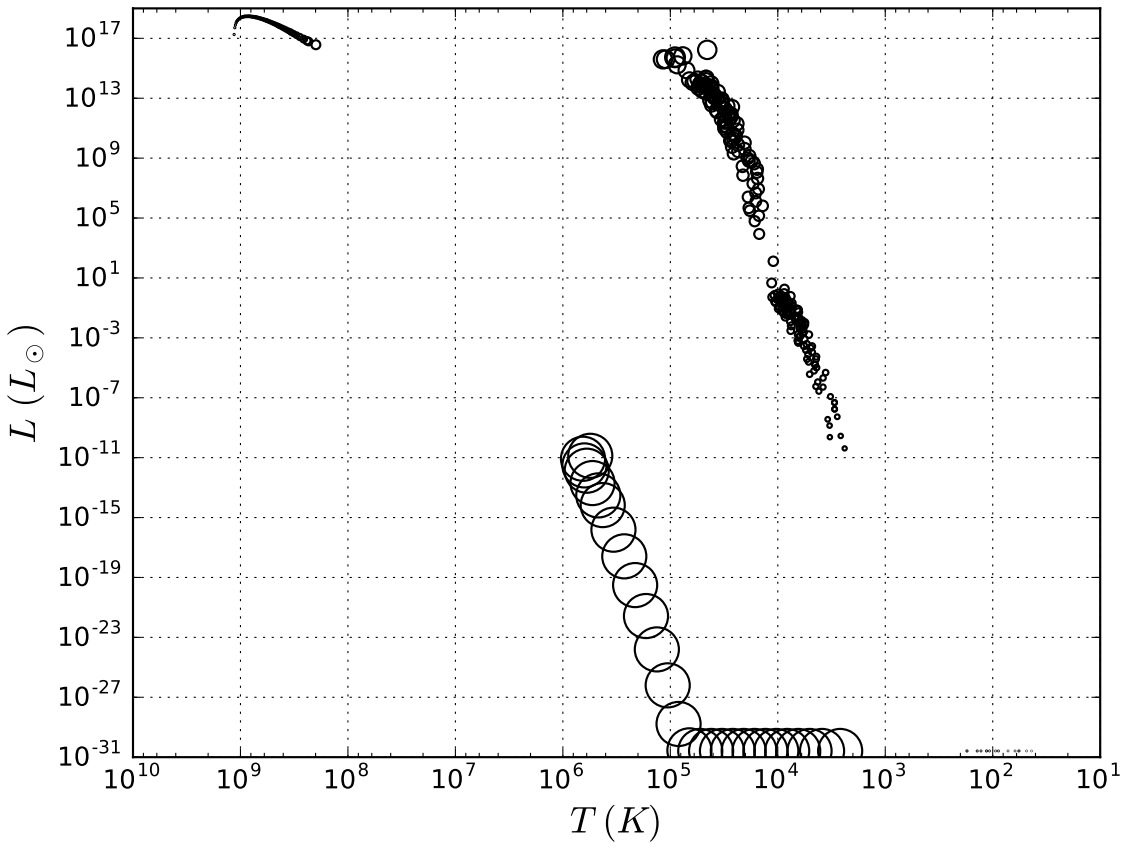

Figure 4: A crappy version of HR

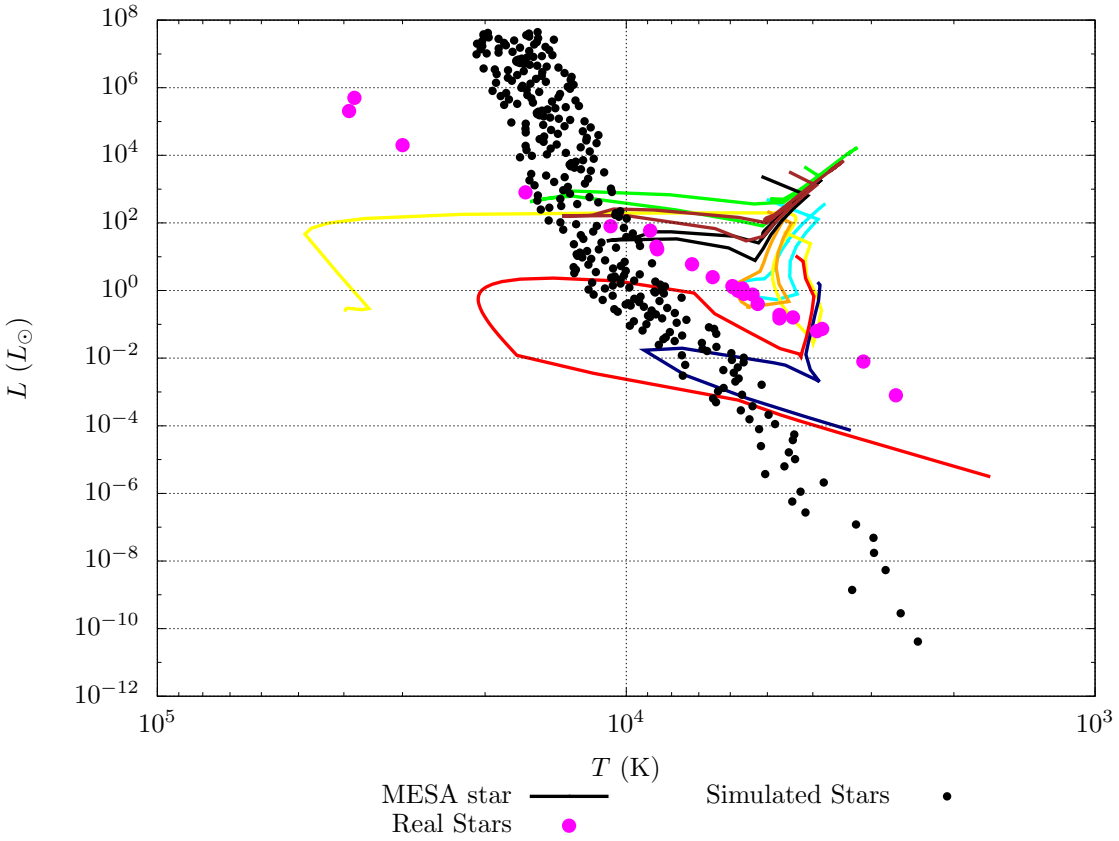

Figure 5: Our reproduction of the HR diagram.

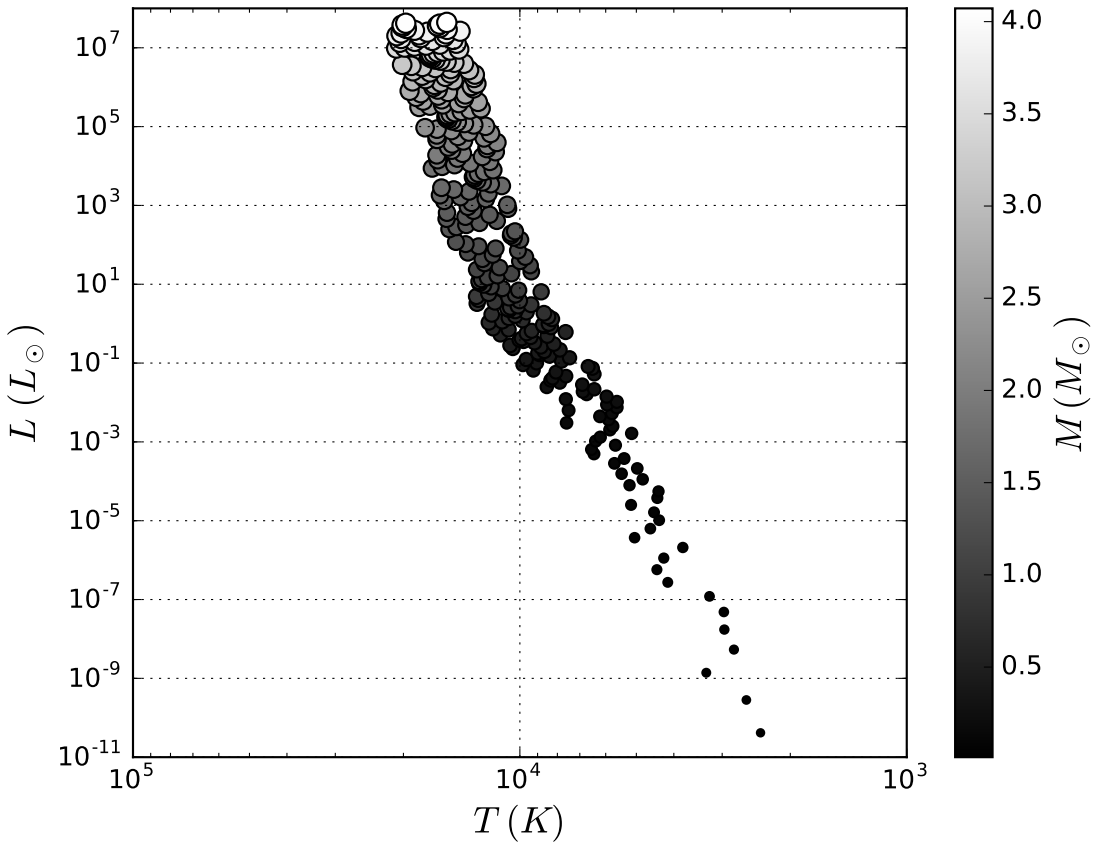

Figure 6: HR diagram from only simulated data