Homework 1: plasma lenses

Ryan Wills

ASTP720

February 1st, 2019

https://github.com/RyanWills16/ASTP720

The final homework materials are under the folder labelled hw1/a720. I had some trouble when I changed some file paths around on my computer, so the final materials were uploaded under a separate directory from the one I started in, but I was commiting frequently while writing the code up.

Problems 1, 2, 3

In order to find the FWHM, I used the equation for the number density for the pseudo-isothermal sphere. I used my root finders to get the FWHM by subtracting $N_0/2$ from the pseudo-isothermal density equation. I set up a list of thresholds and an option in my root finding functions to return the number of iterations. I generated evenly spaced points and then used exponents to get an good sample of thresholds from 10^{-14} to 10^{-3} and then plotted number of iterations veruses the natural log of the thresholds. The plot for each root finding method is shown in Figure 1. As you can see the bisect method is much slower than either Newton's method or secant method for very low threshold. Newton's and secant method stay pretty fast no matter the threshold. I found the FWHM to be about $3.4641 r_c$.

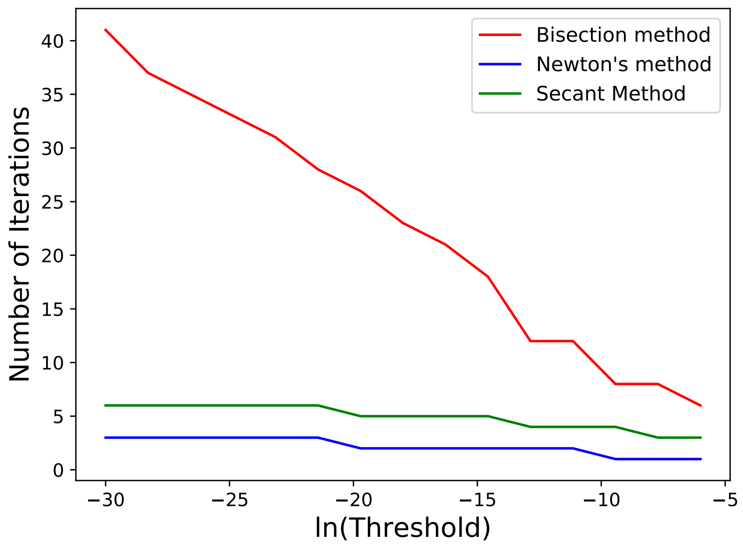

FIG. 1: Plot of iterations on the y-axis versus the natural log of the threshold on the x-axis. The bisect method is consistently takes longer than the other methods.

I also wanted to make a plot using my linear interpolation method for fun. This is seen in Figure 2. I used a function specified in the title of Figure 2 to generate some points and then used the linear interpolator to fill the gaps.

Problem 4

For creating the ray tracing diagrams, I used the equation for x' for the Gaussian lens. From there I created a function that subtracted x' from that equation and fed that function values of x' for which to find the roots and thus find x, the position on the lens. To do this, I used the secant method since it was the fastest and did require me to input the derivative of the function. Here something very strange happened. The secant method would get hung up on values of x' between 0.2 and 0.27, it would just iterate forever. I have no idea why this occurred or how to fix it. I used my verbose option to take a look at the calculation each iteration and it seems the function got stuck with negative values of x2 and just couldn't move on from there. To avoid this problem, I just avoided values in the range

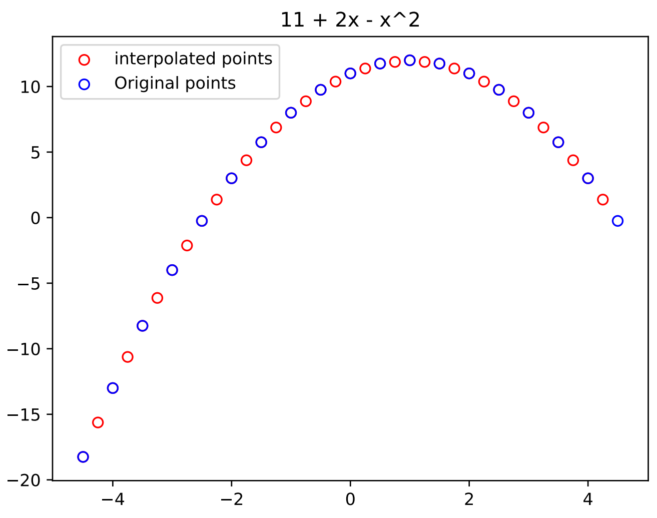

FIG. 2: The equation used is shown in the title, the red points represent the interpolated points while the blue points where the originals.

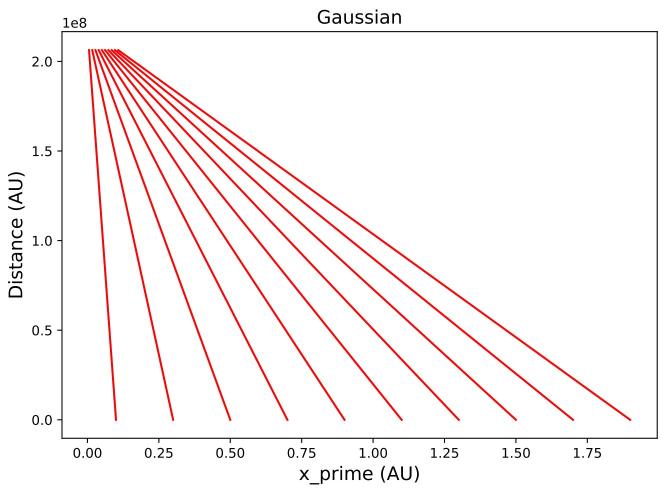

FIG. 3: Ray trace diagram for the spherical Gaussian lens. The light rays seem to come from a somewhat small region.

0.2 to 027. Otherwise the calculations were successful, I used normal matplotlib plotting to connect two points with a line, one at (x',0) and the other at (x, D) where D is the distance in AU to the plasma lens. I used iteration to add a line for each value of x' and its corresponding x value.

The ray trace diagram for the spherical Gaussian lens is shown in Figure 3. The rays seem to be converging to one point, but they don't seem to form any of the caustic regions seen in the Clegg, Fey, and Lazio figure from the homework 1 handout.

Problem 5

I used the same methods for the pseudo-isothermal plasma lens. The main difference is that the pseudo lens seems to be spread much further out in its x dimension than the spherical Gaussian was. I din't have as many issues with my root finding algorithm getting stuck on certain values of x'.

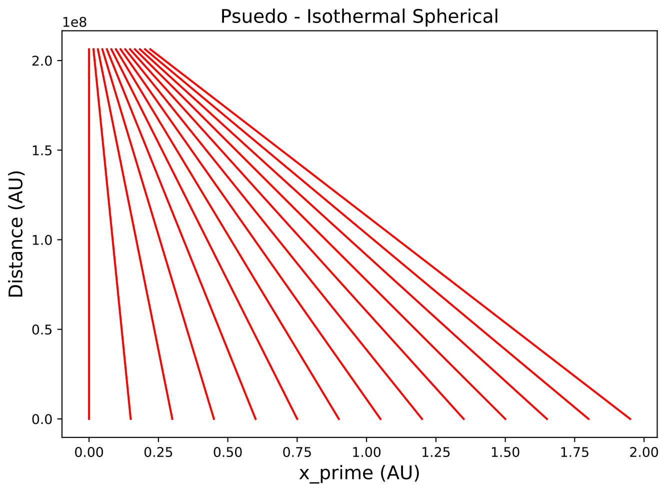

FIG. 4: Ray trace diagram for the pseudo-isothermal plasma lens. Much larger in its x dimension than the corresponding spherical Gaussian lens