PHY407 Lab Assignment 4
Vincent Fan 1002563380
Satchel Page 1001695686

Question 1. Equation of the orbit of a comet:

(Restating Question) An astronomer tracks the Cartesian coordinates of a comet circling the sun (origin) and makes 5 measurements summarized in the following table:

X	Y
-38.04	27.71
-35.28	17.27
-25.58	30.68
-28.80	31.50
-30.06	10.53

a) Since we want to fit the data points into an ellipse, we will need to find the equation to fit for. The first part of this question requires us to derive the parameters of the following ellipse equation in general form:

$$Ax^2 + 2Bxy + Cy^2 + 2Dx + 2Fy + G = 0,$$

and to convert them into astronomical parameters of interest.

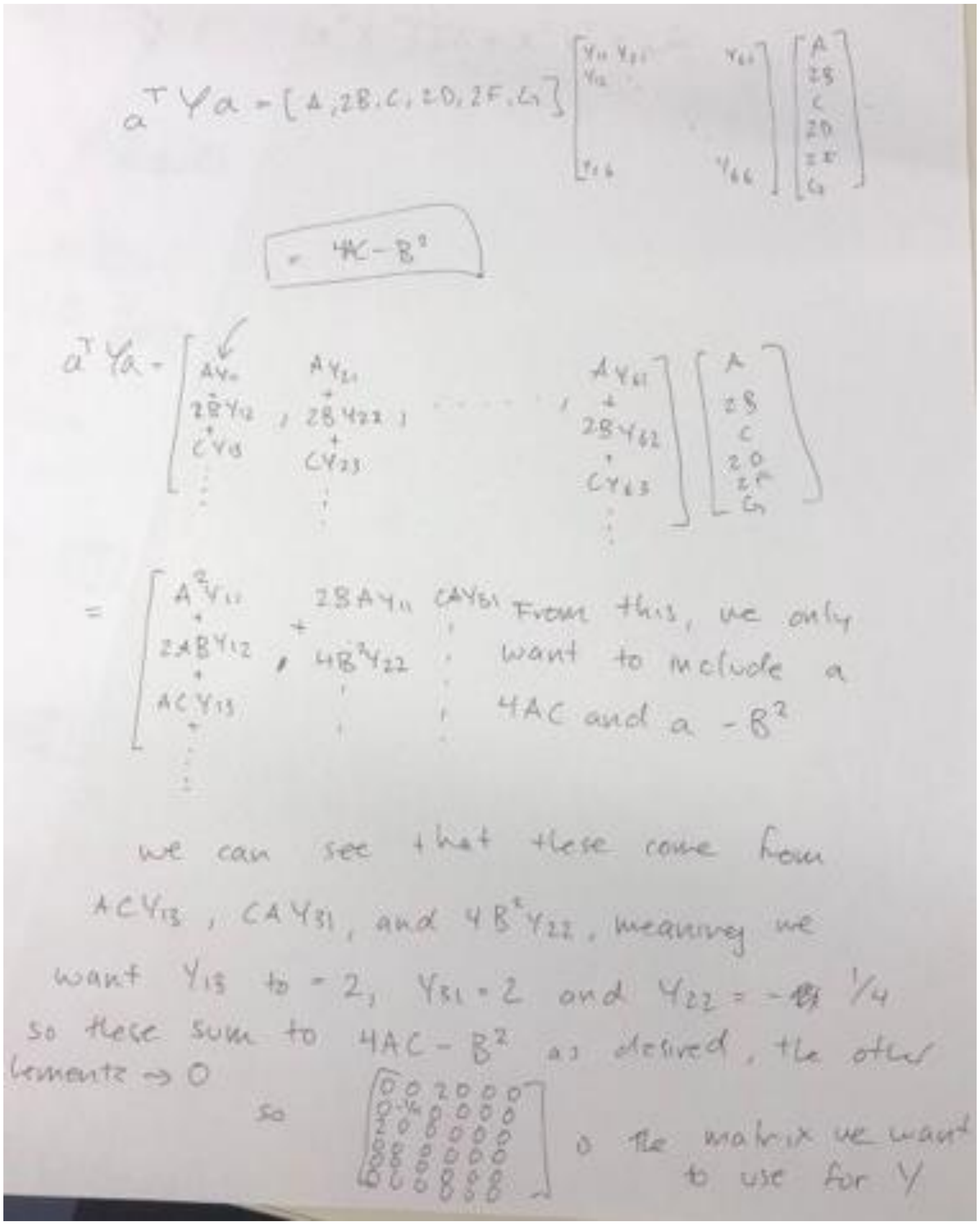
In the code, we took some equations from this website[1]

Following the process in the lab manual, we defined a coefficient matrix "a" and a variable matrix "X" for the above ellipse equation, where a = [A,2B,C,2D,2F,G] and X=[x^2, xy, y^2, x, y, 1]. The equation X · a = 0 would provide the values we need, but the functions equipped on python would give only the trivial solution that the coefficients are all zero. To get around this, we imposed the constraint that γ(a) = $B^2 - 4AC < 0$ and instead minimized the constrained function, such that:

$$\mathbf{a^T Y a = 4AC - B^2}$$

To get the elements of this Y matrix, we did the following derivation:

[1] http://mathworld.wolfram.com/Ellipse.html?fbclid=IwAR2Wr7xFpQQy4JTi8hQS4VEdC008owJuVItJBgiT9oj4QcBOywULCk8_0PI



In summary, we found that the matrix Y should be 6x6 with elements [1][3] = 2, [3][1] = 2 and [2][2] = -1/4.

Once we had this value, we could refer back to the lab manual and applied Y to the following equation:

$$\frac{1}{\lambda}a = S^{-1}Ya$$

Here, $X^TX = S$ and is solved for in our code, so we just needed to solve for the eigenvalues in the above equation. This was done with the numpy eig function as advised in the lab manual.

Finally, since B, D, and F are doubled in the "a" matrix, we divided their values by 2 to get the true parameters. These resulting astronomical parameters [A,2B,C,2D,2F,G] are the following:

A	float64	1	0.035703424536101036
B	float64	1	0.05422559771814994
C	float64	1	0.046477940755210084
D	float64	1	0.5316895183088881
E	float64	(6,)	[0.00000000e+00 0.(-2.2 ...
F	float64	1	-0.3317557898097743
G	float64	1	-0.7751574053699125

b) This next part asks for the code to find the equation of the orbit, the values for the perihelion, aphelion, period and eccentricity using the equation, and finally a plot of the orbit. To calculate the values of semi-major and semi-minor axis respectively, we used the following series of equations with the parameters we solved for previously:

$$a' = \sqrt{\frac{2\left(a f^2 + c d^2 + g b^2 - 2 b d f - a c g\right)}{\left(b^2 - a c\right)\left[\sqrt{(a-c)^2 + 4 b^2} - (a+c)\right]}}$$

$$b' = \sqrt{\frac{2\left(a f^2 + c d^2 + g b^2 - 2 b d f - a c g\right)}{\left(b^2 - a c\right)\left[-\sqrt{(a-c)^2 + 4 b^2} - (a+c)\right]}}.$$

Where a is semi-major axis, b is semi-minor. From a and b we got in these equations, we were able to fully derive the following system of equations for l_1 and l_2. This allowed us to calculate Period (T) and eccentricity (e) in the following given the minimum speed v_2 as 1350m/s which we converted into AU/yr:

$$a = \frac{1}{2}(l_1 + l_2)$$

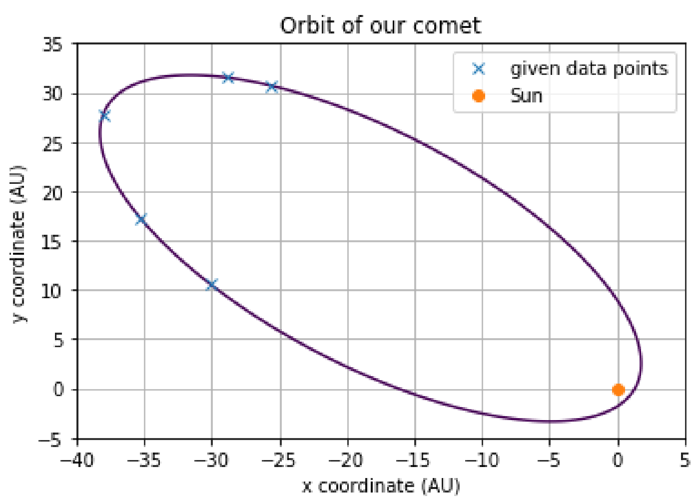
$$b = \sqrt{l_1 l_2}$$

$$T = \frac{2\pi ab}{l_1 v_1} \qquad e \equiv \sqrt{1 - \frac{b^2}{a^2}}, \qquad l_2 v_2 = l_1 v_1,$$

These calculation results can be seen below:

Perihelion distance= 2.512409049997533 AU
Aphelion distance= 46.20343403039125 AU
Period = 125.4046863744549 years
Eccentricity = 0.8968545388467706
Semi-major axis = 24.35792154019439 AU
Semi-minor axis = 10.774132252711532 AU

The outputted plot of orbit:



We can see that our data points match with the orbit (purple line), and we can observe that the distance between the sun and perihelion also matches with our calculated result. Therefore, we can say that the fitting was accurate to our data points.

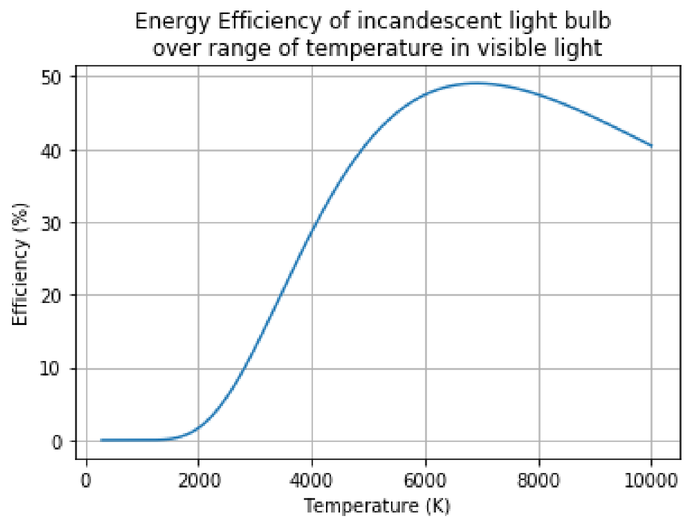
c) While no comets match our comet exactly, some are close, and of course there could be some error in our calculation. The most similar comet appears to be ATLAS

Comet name	Eccentricity	Semimajor axis (AU)	Period (years)	Perihelion distance (AU)
ATLAS	0.9034	25.616	129.65	2.475

Question 2. The Temperature of a Lightbulb

a) (Restating Question) This first question asks us to write a code which takes a temperature T and calculates the efficiency of a lightbulb for radiation between 380nm and 780 nm, then plot the function between 300K to 10,000K temperatures and identify an extremum in efficiency.

The efficiency curve we produced Is the following:



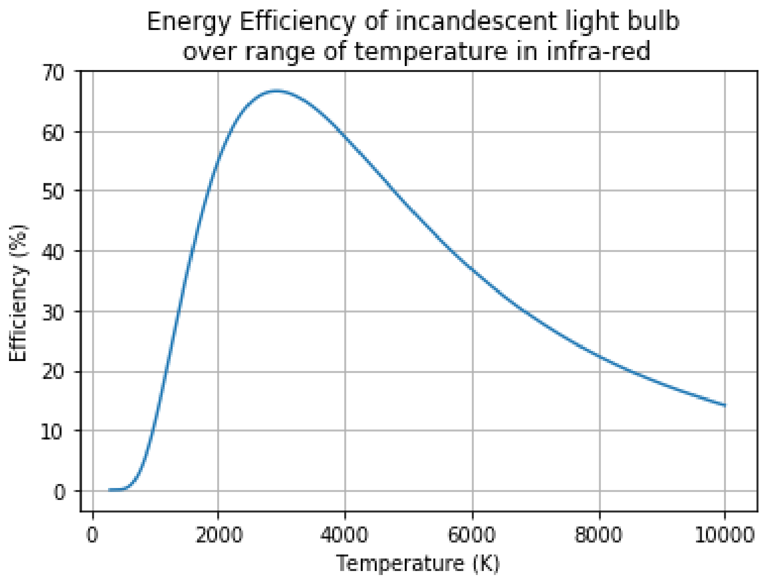
As was predicted in the question, we can see the extreme efficiency value from this graph to be somewhere around 7000K at just under 50% efficiency.

b) For this next section, we now want to find the value of maximum efficiency shown above as well as the corresponding temperature to an accuracy of **1K**. The code gives a detailed outline of our technique. The basic breakdown is that we created a range of temperatures with a step size of **1K** and we found the location of temperature range where the efficiency is the max, and the values from the output are:

maximum efficiency is: 49.0416073819 % with temperature 6913 K

c) We now want to investigate how the temperature corresponding with maximum efficiency would change if we used an infra-red bulb (780nm – 2250nm) instead of a visible spectrum light bulb. Secondly, we want to know whether or not it would be possible to run a tungsten bulb at maximum efficiency for either wavelength.

For the first part, using similar code to that used in part a), we plotted the Efficiency in percent against the Temperature to produce the following plot:



There are a few differences to note here. One being that the maximum efficiency is much higher at close to 70%, another being the corresponding temperature to maximum efficiency being substantially lower. Using similar code to that used in part b), we find these values accurate to 1K to be the following:

maximum efficiency is: 66.5502687849 % with temperature 2925 K

Finally, we want to know whether or not a tungsten bulb could run at either of these maximum efficiencies. The melting point for tungsten[2] is 3695K, meaning if tungsten bulb is emitting infra-red light then it can run at maximum efficiency at 2925K which falls below that melting point. However, the light bulb emitting visible spectrum has a maximum efficiency at 6913K, well beyond the melting point of tungsten, meaning tungsten bulb cannot operate at maximum efficiency if it emits visible light.

[2] https://en.wikipedia.org/wiki/Tungsten

Question 3. Solving non-linear functions:

a) Exercise 6.10 part from the textbook explores the relaxation method of solving a non-linear equation. For part a), given the equation $x = 1 - e^{-cx}$, where c is a known parameter (2) and x is an unknown, we are asked to calculate the solution to use the relaxation method to a magnitude of accuracy of at least 10^{-6}.

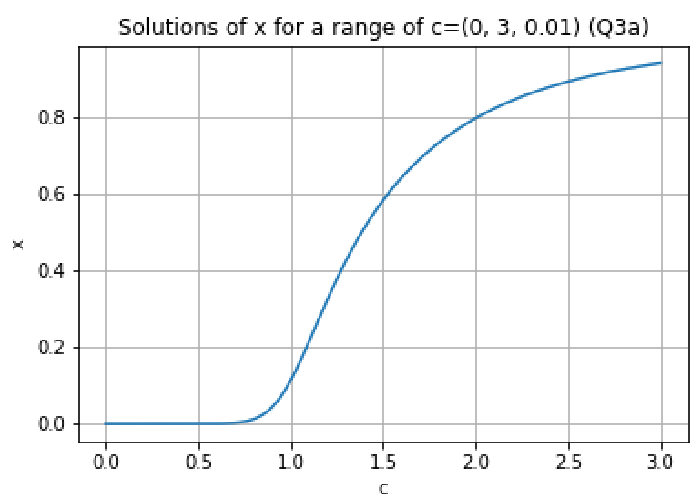
Using similar code to that used in the textbook page 251, where we can use the relaxation method while manually inputting iteration values, after many trials we found that k = 15 iterations provided the accuracy to the solution we were looking for, outputting a value of:

value of x with 15 iterations: 0.7968123336514794
value of 1 - exp(-2*x) with 15 iterations: 0.7968122127708882

We can see that these 2 values indeed have accuracy of up to 10^{-6}.

Next, part (b) asks us to modify the program to calculate the solution with the same accuracy if the values of c ranged from 0 to 3 with step size 0.01. Plotting these solutions should yield a phase transition

Doing so, we retrieve the following plot showing the regime around c = 0.8 that we were hoping to find:



Same as the textbook suggests, we can see a clear transition from a regime in which x=0 to a regime of nonzero x.

b) This next question asks us to tackle parts b, c, and d of exercise 6.11 from the textbook. Here, we are attempting a more efficient version of the relaxation method from the previous question, called the overrelaxation method. The difference here is that rather than having each iteration change by some value determined by f(x), it changes by a predictive value towards the true solution, allowing the equation to be solved in fewer iterations.

For part b), we are first asked to modify our code from the first part to allow it to print out the number of iterations it took to converge to a true solution. We modified the code, but already knew how many iterations it took from our code in part a), so when our program outputted **14** we knew we had the right set up. To change our code, we implemented an error function which was given in the textbook, and set the code up to add iterations until the value of the error, or degree of accuracy, was below 10**-6, and then outputted the number of iterations it took to get there:

The minimum number of iterations for an accuracy of 10-6 = 14**

For part c), we now want to compare the efficiency of the overrelaxation method to the relaxation method used previously, with efficiency just meaning the number of iterations it takes to achieve an answer of the desired accuracy. Here we used the same method as in the previous question to track the minimized number of iterations it took and simply replaced the values of omega until the iteration count was the lowest It could be. Starting at 0.5 as suggested in the text, we achieved an iteration count of 4, and replacing the number by slightly higher, or slightly lower values only increased this count, or kept it the same, so we decided to keep the omega value of 0.5 here, but anything within a range of about +/-0.25 seemed to work just as well.

Before we get to the output, the iteration count to get to our desired accuracy was cut down from 14 to 4, which is a multiple of 3.5 less iterations, demonstrating the substantial efficiency benefits from using the overrelaxation method over the relaxation method. Therefore, we conclude that, using the over-relaxation method here to find the derivative of f and increase the gradient, can achieve an increase in the rate of convergence to our desired accuracy.
The following was the complete output of our code:

The minimum number of iterations for an accuracy of 10-6 = 14**
0.796812631112
When we set omega to be 0.5 the minimum number of iterations for an accuracy of 10-6 is 4**
x ~ 0.796812372983

Part d) now asks us to consider whether a negative value of omega would ever be appropriate. The answer to this would be yes, because choosing an initial estimate to be higher than the true solution would cause additional iterations to lower the value which each successive calculation. This means that a negative value of omega would be the proper input in order to speed up the efficiency to get to towards the true solution.

c) This question asks us to apply the relaxation method to the biochemical process of Glycolysis.

For part a), we are asked to rearrange equations to derive the desired solutions shown in the textbook, the equations, derivations and solutions are as follows:

(1) $-x + ay + x^2y = 0$ and **(2)** $b - ay - x^2y = 0$

from **(2)** $b = ay + x^2y$

so $-x + ay + x^2y = -x + b = 0$

and $x = b$ **(3)**

Therefore: $b - ay - b^2y = 0$

So $y(a + b^2) = b$

And $y = b/(a+b^2)$ **(4)**

For part b), we first want to show that these equations can be rearranged further:

$y = b/(a+b^2) = x/(a+b^2)$ from **(3)**

and so, $x = y(a+b^2)$ **(5)**

Further: $y = b/(a+b^2) = b/(a+x^2)$ from **(3)**

So, $y = b/(a+x^2)$ **(6)**

Now, using values of a and b as 1 and 2 respectively, we wrote a program to apply to relaxation method to solve these equations. Running the program initially using the two-variable method outlined in the textbook exploded the iteration count preventing the solution. To adjust the equations, we determined that we needed both of the equations to be of one variable, so we rearranged them as follows:

For **(6)** $y = b/(a+x^2)$ we replaced the b with an x **(3)** to get $\mathbf{y = x/(a+x^2)}$
For **(5)** $ya + yx^2 = b$ so $yx^2 = b - ya$ and $x^2 = b/y - a$ so **x = sqrt (b/y – a)**

By rearranging these solutions this way, our code now outputted the following:
The number of iterations = 25
the solution for x ~ 1.99999801056
the solution for y ~ 0.40000000000000002

Approximating x to be 2 and y to be 0.4, we confirm that x = b = 2
As well as $y = b / (a + b^2) = 2 / (1 + 4) = 2 / 5 = 0.4$.