Question2a:

Find the efficiency of a light bulb from 300K to 10,000K for radiation between 380nm and 780nm.

The efficiency of the radiation for a range of wavelength λ on a certain temperature is calculate as:

$$\eta = \frac{E(\lambda_1, \lambda_2)}{E(0, \infty)}$$

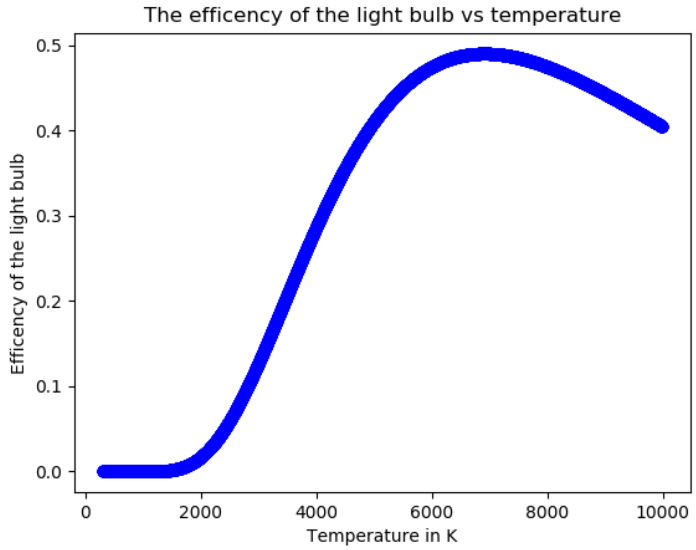
where Energy E is calculated as:

$$E = \int_{\lambda_1}^{\lambda_2} 2\pi A h c^2 \frac{\lambda^{-5}}{e^{\frac{K}{\lambda}} - 1} d\lambda\,, \quad K = \frac{hc}{k_b T}$$

To numerically integrate E on an infinite interval, we use change of variable to change the interval from infinite interval to a finite interval: First Let $\frac{K}{\lambda} = \frac{x}{-x-1}$, then by some algebraic manipulations, we have : $x = \frac{-k}{\lambda+k}$ and $\lim_{\lambda\to\infty} \frac{\lambda}{\lambda+k} = 1$, substitute the new equation into the energy formula, we get:

$$E = \int_0^1 2\pi A h c^2 \frac{\left(\frac{x}{x+1}\right)^{-5}}{e^{\frac{hc(x+1)}{x k_b T}} - 1} * \frac{1}{(x+1)^2} dx$$

In the code, we integrate this integration using gaussian quadrature, with 4000 intervals for the denominator between 0 and 0.0001 for the denominator of η instead of the full 0 to 1 interval and 400 for the numerator part of the equation The reason that we choose this 0 to 0.0001 is because if I integrate the full interval from 0 to 1, we need at least 1,000,000 intervals to have a adequate result, which is too much for our computers to handle when it tries to calculate the interval value using gaussxw(). Therefore, we choose the interval where most of the radiation intensity is concentrated. With this choice of interval, the total radiation energy will be slightly less than the real value. Which cause the whole efficiency graph to shift up by a small fraction (less than 0.1%). And the graph below is the result we calculated:



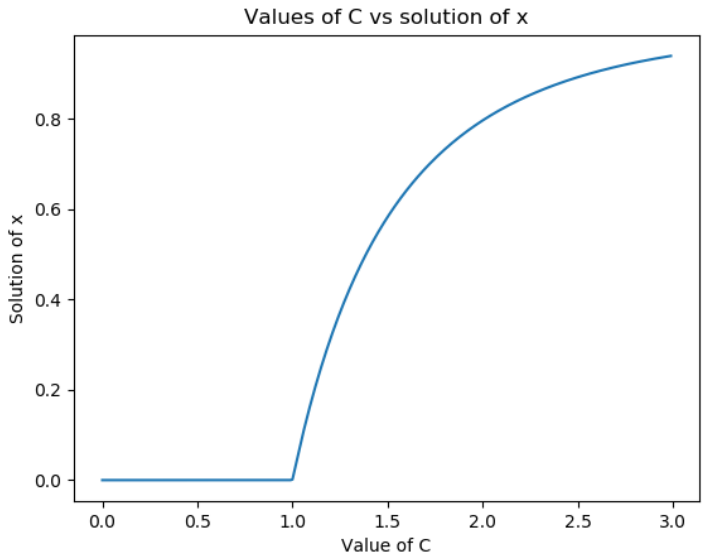
Question2b:

Calculate the temperature of maximum efficiency of the light bulb to within 1K:

To calculate the temperature of the maximum efficiency, we use the binary search method, we choose the interval from 6000K to 10000K, the result we get for maximum efficiency is 6912.5K, the implementation of the solution is the method in the python file as *method Question2_b_solution().*

Question3a:

The solution for exercise 6.10 on textbook is shown below and it is implemented as Question 3a in the Lab4_Q2_Q3 as method *Question3_a()*, the output of the graph is shown below:



Question3b:

For C = 2 and ω = 0.5, it took 8 iteration for overrelaxation method to find the solution of x while the conventional relaxation method took 17 iteration to find the solution to x. Overrelaxation method is indeed faster than relaxational method for C = 2 and ω = 0.5. The reason overrelaxation method is faster in this case is that in each iteration, it takes a bigger step than relaxational method, which decrease the number of iterations needed to approach the solution. The code implementation for this question is implemented as *Question3_b()* in the python file.

Part D:

Yes, there are cases that taking $\omega < 0$ let us find a solution that is faster. This case happens when our initial value is so close to the solution that the conventional method will overshoot the solution. Then in that case, if we take $-1 < \omega < 0$, it will decrease the step we took, hence prevent over shooting.

Question3c:

Part a:

Let $-x + ay + x^2y$ be equation 1, and $b - ay - x^2y$ be equation 2, since both equation are equal to zero, then equation 1 plus equation 2 is still 0, and the result equation is:

$$-x + b = 0$$

Hence, $x = b$. Now substitute the result back to equation 2, now we have:

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

$$ay + b^2y = b$$

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

Then we get $y = \frac{b}{a+b^2}$ as desire.

Part b:

For equation 1:

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

$$ay + x^2y = x$$

$$y(a + x^2) = x$$

Then we have $x = y(a + x^2)$ as desired, and for equation 2:

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

$$ay + x^2y = b$$

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

Then we arrive to $y = \frac{b}{a+x^2}$ as desired.

The code for this question is implemented as *Question3_c()*in the python file. Notice that since x become large so quick, I must set the iteration to be less than 5 times to prevent overflow.

Part c:

For this question, we manipulate equation 1 so that it becomes:

$$x = \sqrt{\frac{x}{y} + a}$$

and we keep y in its original form, the program we implemented gives the output of x = 2.85 and y = 0.21, which is different from what we expected, and we cannot find out where the calculation went wrong. The implementation of this question is in the python file as method *Question3_c_part_c()*.