PHY407 Lab 4

Pierino Zindel (1002429703) and Hayden Johnson (1002103537)

October 5, 2018

Distribution of work: Question 1 was completed by Pierino. Question 3 was completed by Hayden. Question 2 was a team effort.

1 Solving Linear Systems

1.1 Part b)

There are several different methods for solving a linear system of equations which perform differently when used in python. The three methods used in this question were Gaussian elimination, LU decomposition, and partial pivoting. The Gaussian elimination method was provided to us in the SolveLinear package and the LU decomposition used was the implementation provided by numpy's linalg package. The partial pivoting method was implemented following the steps outline in Newman's Computational Physics book to pivot the rows of the matrix and then using Gaussian elimination and back substitution to solve the system completely. The method was similar to the Gaussian method provided however the pivot points in the algorithm were determined by their absolution value and the entries of the matrix had to be rounded off to avoid precision errors within the computation that came up.

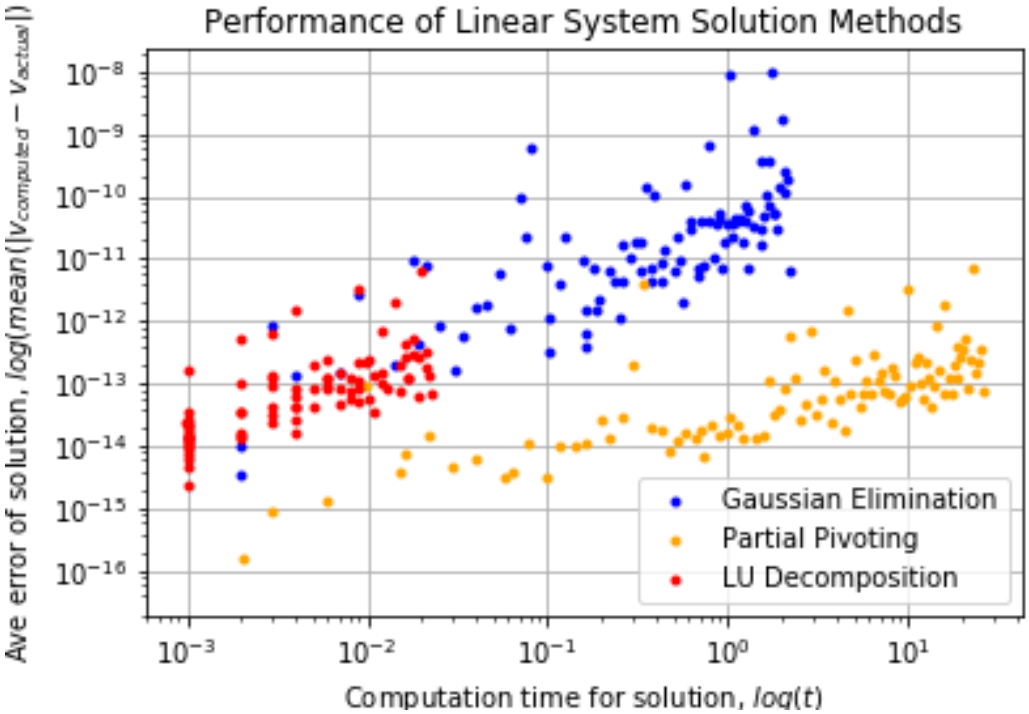

Figure 1: Log-Log plot of average absolute error vs the computation time taken for 3 linear system solution methods: Gaussian elimination, partial pivoting, and LU decomposition.

The performance of the methods is shown in figure 1. The format chosen for the figure is a log-log plot with the average absolute error of the solution, found by the each method for each matrix, on the y axis vs the time taken to solve each system, for each method for each matrix, on the x axis. This allows us to directly compare the overall performance of the methods. Comparing the performance of the three methods we see that the LU decomposition had the best performance overall. LU decomposition had relatively small error range and computation times, with larger matrices not taking extremely longer than small matrices. Partial pivoting performed as well as LU decomposition in terms of errors in computation, but took much longer to solve each system. From the figure we can also see that the linear trend of the partial pivoting method is slower, with respect to the error, than that of the other two methods which have similar trends. Additionally, we can see that the scattered pattern of LU decomposition is similar to that of partial pivoting by but compressed by the faster performance. Gaussian performed oppositely to partial pivoting, the matrices had larger errors in their solution but the time taken to solve the system was somewhere in between LU decomposition and partial pivoting. The time and error growth of Gaussian elimination is similar to that of LU decomposition but at a faster rate. Since the plot is log-log these linear trends are actually exponential, thus a minor increase between methods as depicted on the plot is amplified in the actual calculation.

1.2 Part c)

Linear systems arise in various circuit setups. We aim to use python to easily solve the linear systems corresponding to these circuits so that we may know what the voltages are for a given point and time.

Using the circuit equations and constants given in the lab manual, the linear systems was solved using the partial pivoting algorithm created fro part a) of the question. The returned components were then used to compute the voltages over time at the given points, using the equation provided $V_n = x_n e^{i\omega t}$. Their absolute values and their phases at $t = 0$ were outputted by the program and found to be:

- $|V_1| = 1.7014V$
- Phase of $V_1 = -0.0955rads$
- $|V_2| = 1.4806V$
- Phase of $V_2 = 0.2022rads$
- $|V_3| = 1.8608V$
- Phase of $V_3 = -0.0727rads$

The real component of the voltages were plotted for two cycles and are shown in figure 2. Resistor 6 was then replace with an inductor in the equations resulting in a change from an RC circuit to and RLC circuit. The system was solved again using the same method as before. The corresponding plot is shown in figure 3 and the outputted values were found to be:

- $|V_1| = 1.5621V$
- Phase of $V_1 = -0.0703rads$
- $|V_2| = 1.4994V$
- Phase of $V_2 = 0.3777rads$
- $|V_3| = 2.8113V$
- Phase of $V_3 = 0.2505rads$

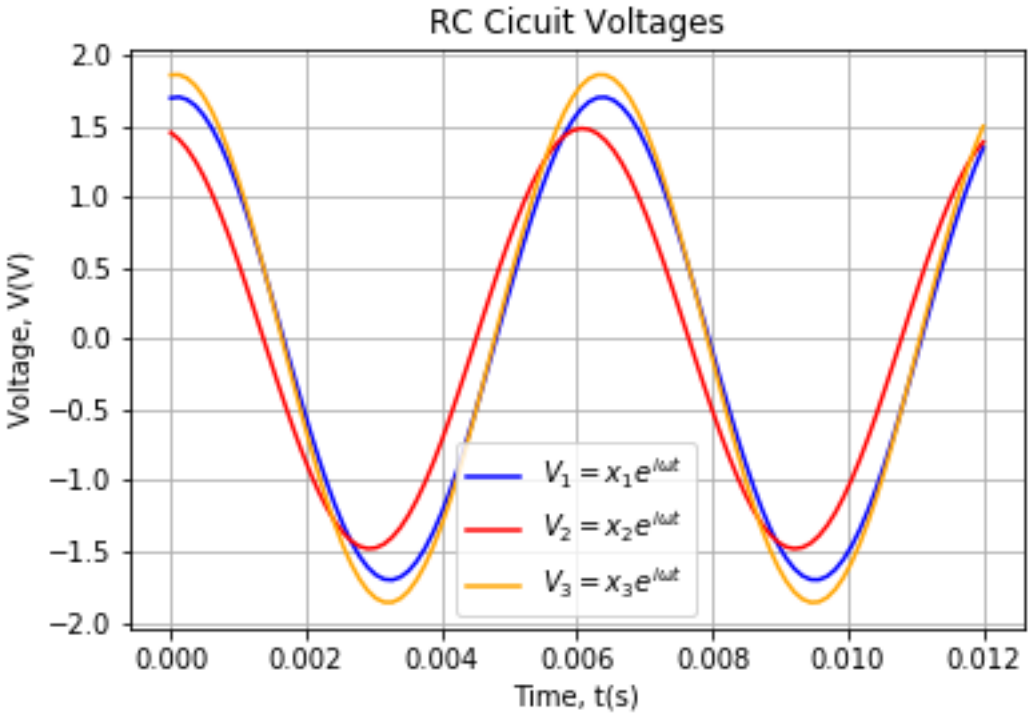

Figure 2: Real component of voltages for an RC circuit at three different reference points.

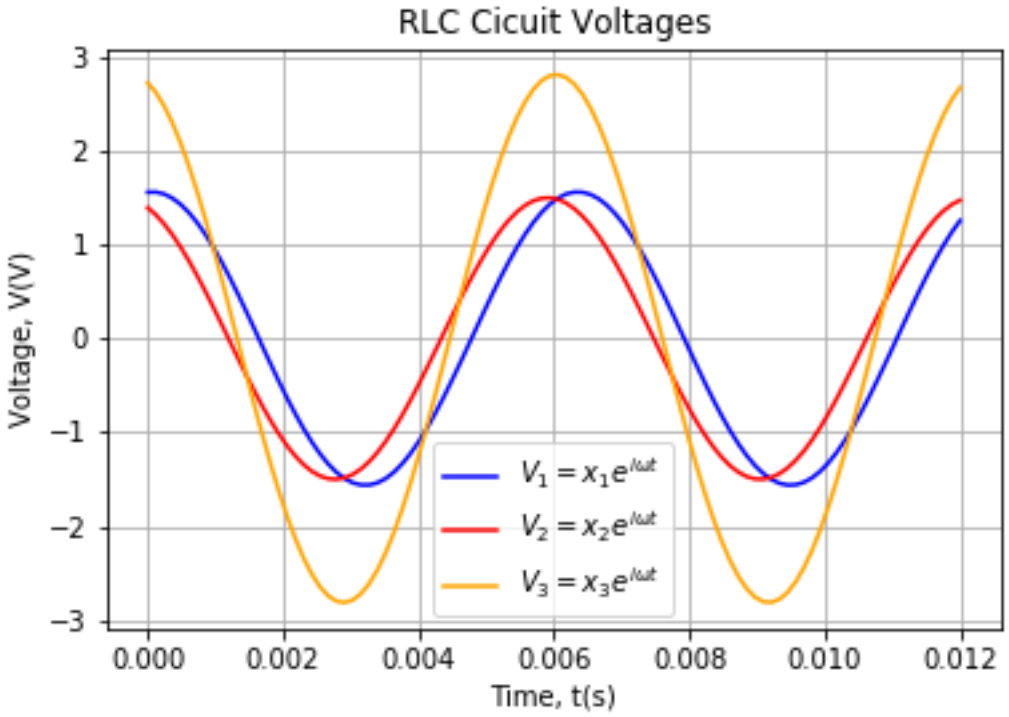

Figure 3: Real component of voltages for and RLC circuit at three different reference points.

As mentioned in the lab manual an inductor is less disipative and resistive than an equivalent resistor, thus ohms law tells us that by replacing a resistor by the equivalent inductor the resistance of the circuit decreases and thus the resulting voltage increases, which is exactly what our plots tell us about the change. Additionally our outputted values shown above largely remain the same from one circuit to the other, showing that the effect of imaginary impedance is fairly localized.

2 Asymmetric Quantum Well

For this question we seek to solve the asymmetric quantum well system to find the eigenvlaues and wavefunctions.

Table 1: First 10 eigenvalues computed for the matrix H using $N = 10$ and $N = 100$.

Eigenvalue	$N = 10$	$N = 100$
1	5.83637688305	5.83637648238
2	11.1810928503	11.1810915271
3	18.6628914519	18.6628895809
4	29.1441975297	29.14418873
5	42.6550744912	42.6550653711
6	59.1852573091	59.1852047261
7	78.7293594938	78.7293076667
8	101.285482927	101.284851996
9	126.851384597	126.850552271
10	155.555327436	155.425704967

Implementing the formula outline in the lab manual for part b of exercise 9.6 and then computing the eigenvalues for a 10x10 Hamiltonian matrix and 100x100 matrix we receive the output shown in table 1. Comparing the values for the 10x10 and the 100x100 matrix we see that the values are the same up to at least the first 5 or 6 decimal places, showing that the calculation of the eigenvalues is consistent from one matrix to the next, and increasing the size of the matrix has minimal effect on the accuracy of the calculation of the eigenvalues.

In order to plot the probability density of the wavefunctions of the ground and first two excited states, the value of the wavefunction $\psi(x)$ was calculated using the equation:

$$\psi(x) = \sum_{n+1}^{\infty} \psi_n \sin\left(\frac{\pi n x}{L}\right)$$

Where the values of ψ_n are the eigenvalues of the matrix H. The probability density is then given by $|\psi(x)|^2$. In order to normalize the probability density, we compute the normalization factor:

$$A = \int_0^L |\psi(x)|^2 dx$$

And then divide the wavefunction $\psi(x)$ by $\sqrt{A}$, so the normalized probability densities which are plotted in figure 4 are given by:

$$\frac{1}{A}|\psi(x)|^2$$

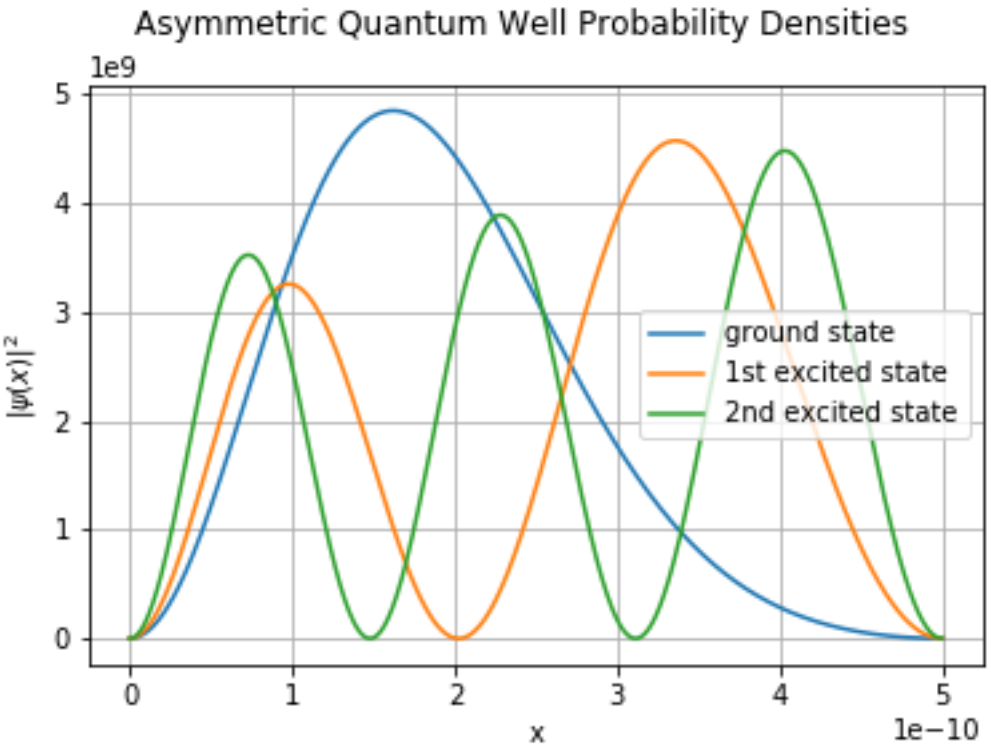

Figure 4: Plot of normalized probability densities for the ground and first two excited states of the electron.

3 Solving Non-linear Equations

3.1 Part a) 6.10

We want to use the relaxation method to solve the non-linear equation:

$$x = 1 - e^{-cx} \tag{1}$$

in the case $c = 2$, to a precision of at least 10^{-6}.

For each value in an array of different c values from 0 to 3 with a spacing of 0.01, an initial value of $x = 1$ was set, and $f(x)$ was calculated and taken as the new value of x, $f(x) = x'$. We know from equation 6.83 in the textbook that the error of the new value x' can be calculated as:

$$\varepsilon' \approx \left| \frac{x - x'}{1 - (1/f'(x))} \right| \tag{2}$$

Thus, the error was calculated for x', and if the error was greater than the desired accuracy of 10^{-6} then $f(x)$ was taken as the new x and the process was repeated until the accuracy criterion was met. Iterating over x like this independently for all values of c produced the graph shown in figure 5. Note the sharp transition at $c = 1.0$, from the regime where the solution $x = 0$ is arrived at to that where the relaxation method produces a non-trivial root. The point at which the transition occurs was experimentally observed to not depend on the initial value of x, as values of 0.1, 1, and 2 all produced identical graphs.

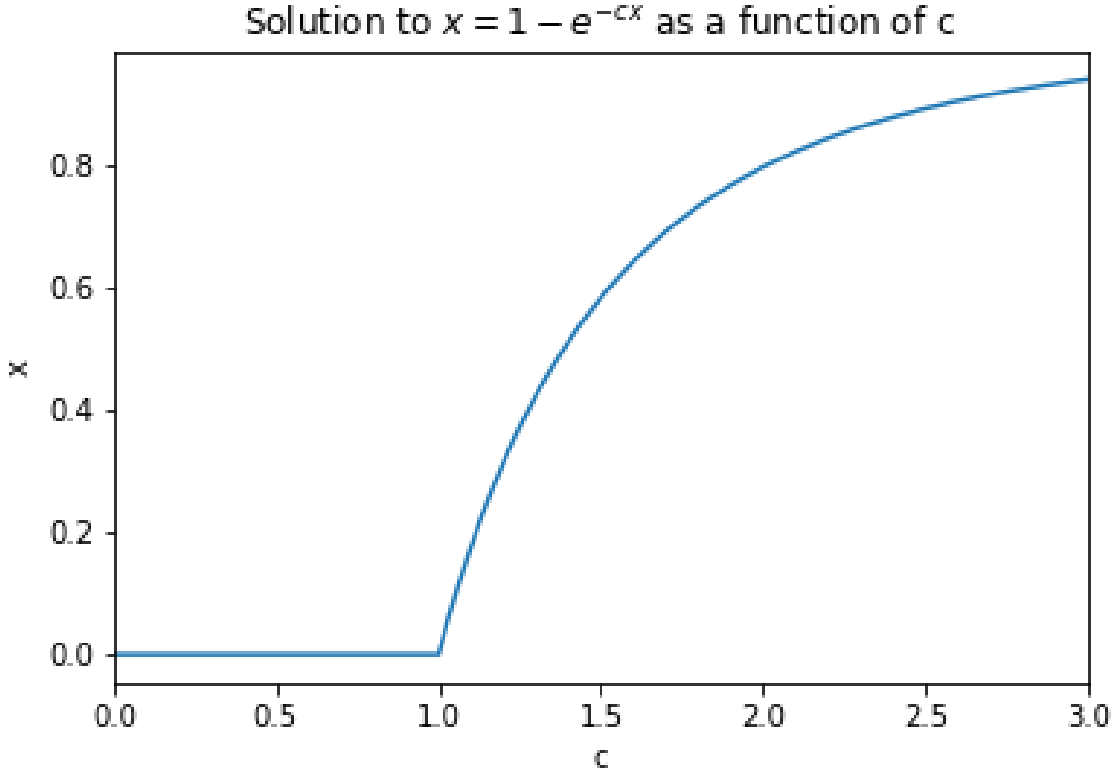

Figure 5: Graph of the value of the solution to the equation 1 as a function of the value of the parameter c.

3.2 Part b): 6.11

3.2.1 c)

We seek to solve equation 1 using the overrelaxation method as well as the regular relaxation method, and compare the number of iterations required for each method for the same value of c and initial x.

The code used in part a) was modified to also iterate over $f(x)$ using the overrelaxation method, where new values of x are computed as:

$$x' = (1 + w)f(x) - wx \tag{3}$$

and the error is computed as given in the textbook as:

$$\varepsilon' \approx \left| \frac{x - x'}{1 - (1/[(1 + w)f'(x) - w])} \right| \tag{4}$$

By trial and error, it was found that a value of $w = 0.7$ resulted in the minimal number of iterations required to reach the desired accuracy of 10^{-6}, from a starting point of $x = 1$ and in the case of $c = 2$. The overrelaxation method with $w = 0.7$ was able to calculate the solution of $x = 796812039515$ with only 4 iterations, as opposed to the 14 iterations required for the regular relaxation method to find a solution of $x = 0.796812333651$ for the same value of c and initial x.

3.2.2 d)

The question is whether there exist any circumstances in which using over-relaxation with a value of $w < 0$ would solve an equation faster than the regular relaxation method.

It is certainly true that there exist situations where this is true - for instance, consider the case where the solution lies at a local maximum or minimum of the function. If the function is fairly steep around this

local extremum, then the relaxation method will result in the value of x oscillating back and forth around the solution before eventually finding it (if the slope is not too steep). In these situations, since $w < 0$ means that the new value x' is taken closer to the old x value than merely $f(x)$, it is clear that the "underrelaxation" method will have oscillations about the solution that decrease in magnitude more quickly, and thus will produce a solution to given accuracy more quickly than regular relaxation.

3.3 Part c) 6.13

3.3.1 b)

We seek to solve the equation

$$5e^{-x} + x - 5 = 0 \tag{5}$$

using the binary search method, and also the relaxation method and Newton's method, in order to compute the Wien displacement constant based on the results given in 6.13 a).

The code used is attached in file Lab04_Q3_c.py. The values of the solution x computed by the three different methods, and the number of iterations required to reach the solution, are displayed in table 2. After looking at the function graphed on desmos, it was apparent that the solution lies near $x = 5$. Thus, for the binary search method, an initial interval of $[4, 6]$ was chosen, while an initial value of $x = 4$ was picked for overrelaxation and Newton's method. For overrelaxation, it was determined by trial and error that $w = 0.04$ was the optimal value to converge to the solution in the fewest number of iterations.

As can be seen from the table, overrelaxation and Newton's method are both significantly faster than the binary search method. Between overrelaxation and Newton's method, overrelaxation performed slightly better (by one iteratrion), but given that the number of iterations (3 and 4) is so small, this is not necessarily indicitive of any general relation between the two methods.

Table 2: Table of computed solutions and required number of iterations for three different methods.

Method	Value	Number of iterations
Binary search	4.965114116668701	22
Overrelaxation	4.96511421209	3
Newton	4.96511423174	4

Given our consensus solution to equation 5 of $x = 4.965114$, we can now compute the value of the displacement constant b:

$$\begin{aligned} b &= \frac{hc}{k_B x} \\ b &= \frac{(6.626 \times 10^{-34}\text{J} \cdot \text{s})(3.00 \times 10^{8}\text{m/s})}{(1.381 \times 10^{-23}\text{J} \cdot \text{s/K})(4.965114)} \\ b &= 2.899 \times 10^{-3}\text{m} \cdot \text{K} \end{aligned}$$

3.3.2 c)

Given that the sun emits light at a peak wavelength of $\lambda = 502\text{nm}$, we can compute its surface temperature as:

$$\begin{aligned} T &= \frac{b}{\lambda} \\ T &= \frac{2.899 \times 10^{-3}\text{m} \cdot \text{K}}{5.02 \times 10^{-7}\text{m}} \\ T &= 5775\text{K} \end{aligned}$$

Which agrees very well with the value that we expect for the surface temperature of the sun.