PHY407 Lab 4: Solving Equations

Work Distribution:
Emma Jarvis: Q1, Q3c
Lisa Nasu-Yu: Q2, Q3ab

October 8, 2021

1 Solving Linear Systems

1.a

We began by writing a function to solve linear equations that incorporates partial pivoting and compared this method to the Gaussian elimination method to ensure that they yield the same answer.

1.b

We then tested the accuracy and timing of the Gaussian elimination, partial pivoting and LU decomposition methods for solving x in linear equations of the form $Ax = v$. We began by choosing a range of 8 N values ranging from 5 to 600 where N is the size of the N by N matrix and the number of elements in the vector v. To compute the error of each of the three methods, we subtracted the the mean absolute value of the difference between the vector x and the product of the matrix A and the solution x. These error values for the 3 methods are depicted in Figure 1. This plot shows that the Gaussian elimination method results in the highest values of the errors that are 1 or 2 orders of magnitude greater than the errors for the other two methods. For all three methods, the error generally increases with N. Both axis of this figure are plotted with a log scale.

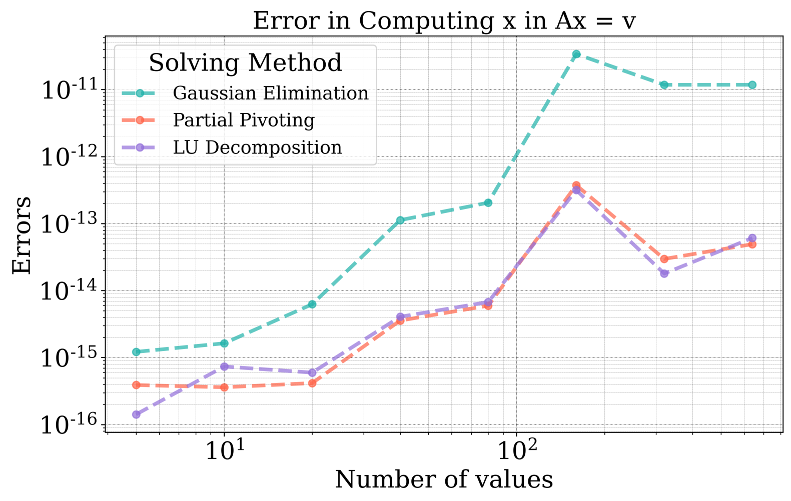


Figure 1: The error in solving for x in linear equations of the form $Ax = v$ for a range of matrix and vector sizes for three methods; Gaussian elimination, partial pivoting and LU decomposition. The Gaussian elimination method has higher errors than the other two methods. The error generally increases with N for each method.

We then computed the time that each method takes to generate a solution, x to linear equations of the form $Ax = v$. Figure 2 depict the time taken for the three methods. This figure shows that the Gaussian eliminate and partial pivoting methods take approximately the same time for all values of N. The LU decomposition method takes a shorter time than the other two methods for all values of N by 1 to 3 orders of magnitude. For all three methods, the timing increases with N. Both axis of the plot are plotted with a log scale.

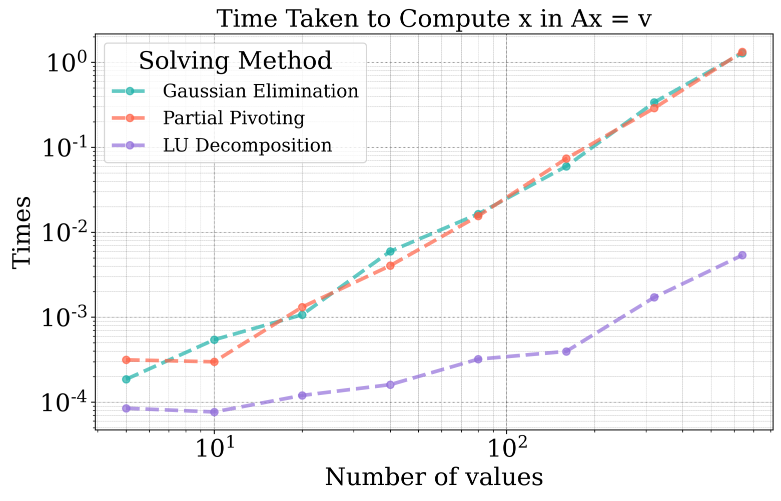


Figure 2: The time taken to solve for x in linear equations of the form $Ax = v$ for a range of matrix and vector sizes for three methods; Gaussian elimination, partial pivoting and LU decomposition. The LU decomposition method takes less time than the other two methods. The time generally increases with N for each method.

1.c

We then wrote a problem to solve for x_1, x_2 and x_3 in Eq. 1, 2 and 3. In these equations we set $R_1 = R_3 = R_5 = 1k\Omega = 1000\Omega$, $R_2 = R_4 = R_6 = 2k\Omega = 2000\Omega$, $C_1 = 1\mu F = 10^{-6}F$, $C_2 = 0.5\mu F = 0.5 \cdot 10^{-6}F, x_+ = 3V$, and $\omega = 1000s^{-1}$. We used the partial pivoting to solve this equation because this works with complex arguments. We then calculate the amplitudes of the 3 voltages corresponding to the these x values and the phases in degrees. These values are summarized in the printed output.

$$\left(\frac{1}{R_1} + \frac{1}{R_4} + i\omega C_1\right) x_1 - i\omega C_1 x_2 = \frac{x_+}{R_1} \tag{1}$$

$$- i\omega C_1 x_1 + \left(\frac{1}{R_2} + \frac{1}{R_5} + i\omega C_2\right) x_2 - i\omega C_2 x_3 = \frac{x_+}{R_2} \tag{2}$$

$$- i\omega C_2 x_2 + \left(\frac{1}{R_3} + \frac{1}{R_6} + i\omega C_2\right) x_3 = \frac{x_+}{R_3} \tag{3}$$

Printed Output:

```
The solution for x1, x2 and x3 in complex numbers is
        [1.69369369-0.16216216j 1.45045045+0.2972973j  1.85585586-0.13513514j]
The magnitudes of x1, x2 and x3 are  [1.70143907 1.48060535 1.86076932]
The phases of the coefficients are [-5.46909497 11.5834186  -4.16467265]
```

We then plotted the real component of these voltages as a function of time. Figure 3 depicts 2 and a half periods of the three voltages.

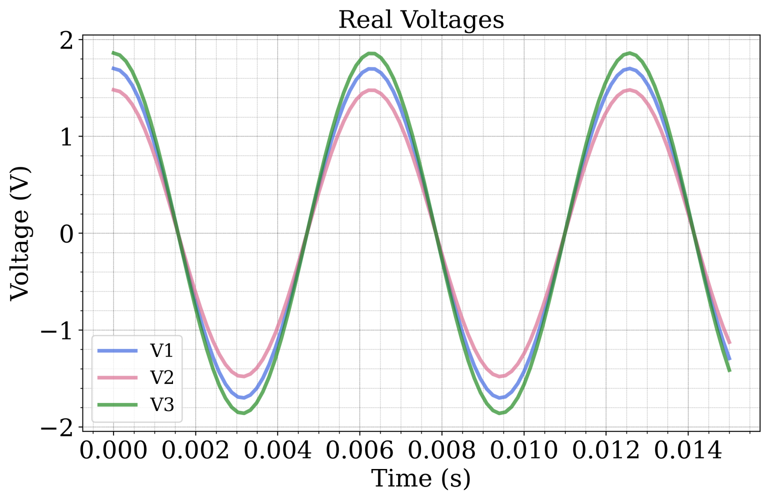
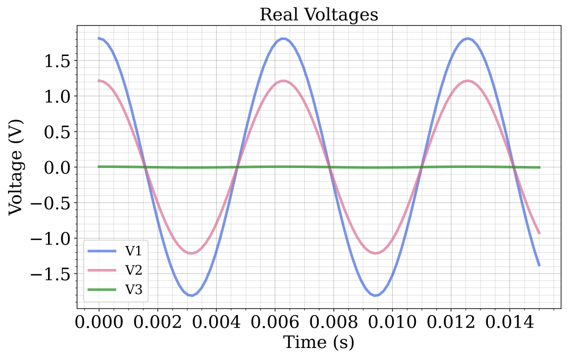


Figure 3: The real voltages as a function of time.

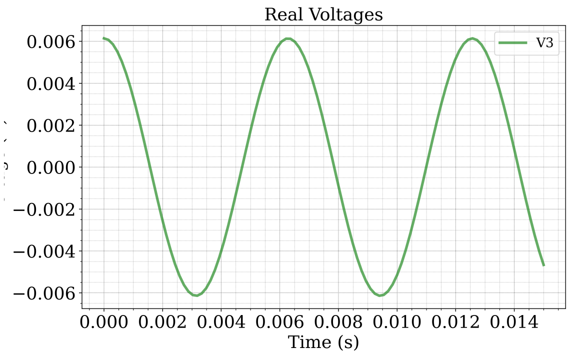
We then replaced the resistor R_6 with an inductor L with $L = R_6/\omega$ and repeated the same calculations. The printed output for the solutions to x_1, x_2 and x_3, the amplitudes and the phases are in the printed output below.

```
Printed Output

R6 = R6/omega

The solution in complex numbers is
        [1.77328281-0.37166526j 1.21578492-0.03158947j 0.00602076+0.00120735j]

The magnitudes of x1, x2 and x3 are  [1.81181317 1.21619524 0.00614062]

The phases of the coefficients are [-11.83736836  -1.48836846  11.33919606]
```

We then plotted the real component of these voltages as a function of time and this plot can be seen in Fig. 4a. This plot shows that by replacing a resistor with an inductor, the amplitude of the voltage decreases significantly.



(a)



(b)

Figure 4: Figure (a) depicts the three voltages when R_6 is replaced with an inductor $L = R_6/\omega$. Because this causes the amplitude for V_3 to decrease significantly, Figure (b) depicts just V_3 so that the oscillations can be seen.

We then replaced R_6 with an imaginary impedance of $R_6 = iR_6$. This effect on the voltage can be seen in Fig. 5. This causes the amplitude voltage V_3 to increase.

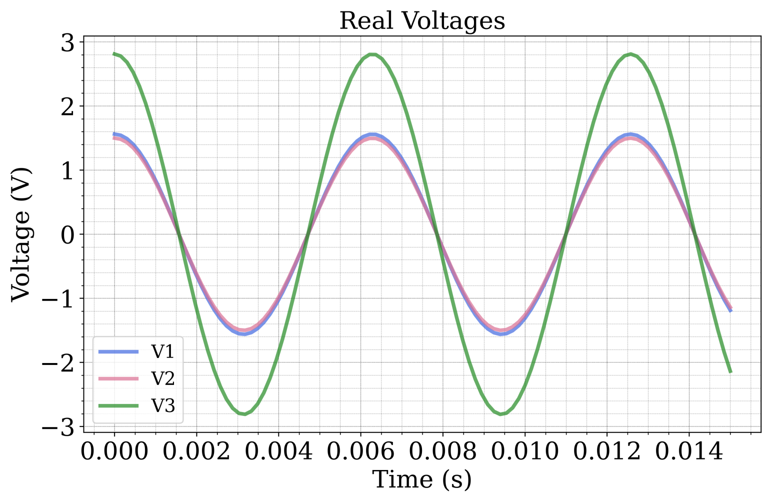


Figure 5: Voltages with $R_6 = iR_6$

2 Asymmetric Quantum Well

In Q6.9 of Newman, we computationally solved an asymmetric quantum well described by the following Hamiltonian.

$$\hat{H} = -\frac{\hbar^2}{2M}\frac{d^2}{dx^2} + V(x), \tag{4}$$

$$V(x) = a\pi/L$$

The components of H were provided in the lab manual as:

$$H_{m,n} = \begin{cases} 0 & m \neq n \text{ and both even/odd} \\ -\frac{8amn}{\pi^2(m^2-n^2)^2} & m \neq n \text{ one even, one odd} \\ \frac{1}{2}a + \frac{\pi^2 h^2}{2ML^2} & m = n \end{cases} \tag{5}$$

	10×10 array	100×100 array
1	4.904606993212836663	4.904556377971336367
2	7.343477283253767141	7.343306021793567062
3	9.242293166195555187	9.241843732411313184
4	10.90623882246644172	10.90488134936649978
5	12.47512303001076894	12.47091650309972266
6	14.01593946138781988	13.99977310560369048
7	15.57324641911305285e	15.51477851070706748
8	17.23565187114419928	17.02426127148189039
9	19.14784837911095394	18.53130779274701823
10	21.88206124734219316e	20.03715936450966950

Table 1: First 10 energy eigenvalues, in units of eV, computed using numpy.linalg.eigvsh for a 10×10 array and 100×100 array of H.

The ground state energy for the 10×10 array and 100×100 array of H were 4.904606993212836663 eV and 4.904556377971336367 eV, respectively. Both values were lower than the expected ground state energy of 5.84 eV, but the 10×10 array gave a more accurate result. Only the low energy eigenstates are significant for the ground state of the wavefunction. Increasing the size of the matrix with components that have little effect on the ground state introduces more computational error in the ground state eigenvalue.

The wave functions for the ground state and first 2 excited states were then solved for, using numpy.linalg.eigh and the 100×100 array of H to find the eigenstates for each. The wave function for each energy state was computed with Eq. 6.

$$\psi(x) = \sum_{n=1}^{\text{inf}} \psi sin \frac{\pi n x}{L} \tag{6}$$

The method numpy.linalg.eigh returns the eigenvectors in a 2 dimensional array. The first index corresponds to the energy level, and we summed over the second index. This resulted in a probability density that was skewed to towards the right, as shown in Figure 6. Swapping the indices reflected the probability density along $x = 2.5 \times 10^{-10}$m. The potential for this well, $V(x) = a\pi/L$, increases as x→0, so we expect the probability density to decrease as x→0.

Our wave function was not initially normalized. Integrating the probability density over all x gave a normalization factor of $A = 2.4999912855655775 \times 10^{-10}$. We divided the wave function by $\sqrt{A}$ to normalize it. The normalized probability density is shown in Figure 6.

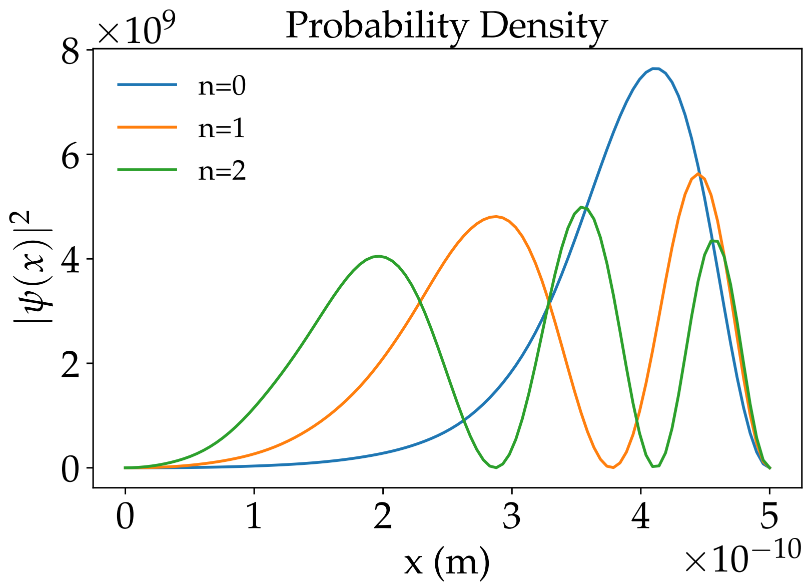


Figure 6: Normalized probability densities for the ground and first 2 excited states of the wave function solving $\hat{H} = \frac{\hbar^2}{2M}\frac{d^2}{dx^2} + V(x)$ for $V(x) = ax$Ł, $a = 10$eV, and L=5Å.

3 Solving Non-Linear Equation

3.a

In Exercise 6.10 of Newman, we compared solutions to the nonlinear equation, Eq. 7, for various c.

$$x = 1 - e^{-cx} \tag{7}$$

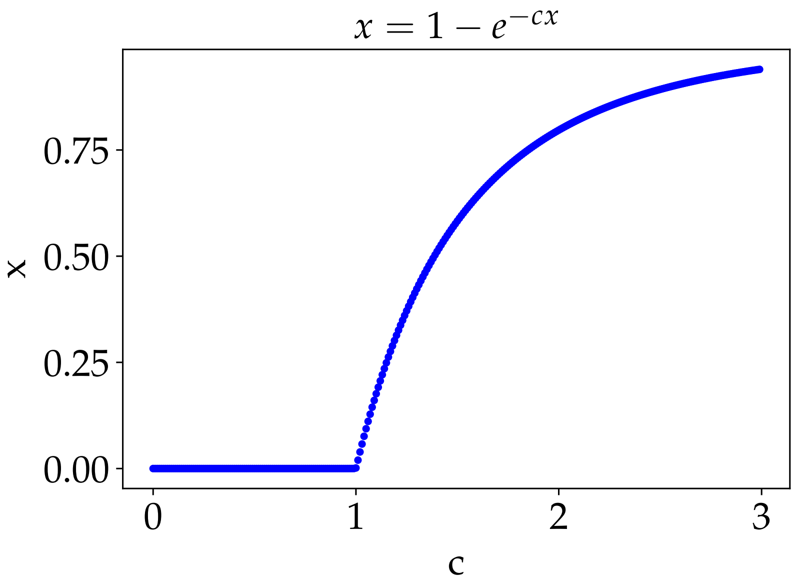


Figure 7: Solutions to the nonlinear equation, Eq. 7, for various c ranging from 0 to 3, with a step of 0.01. We observe the percolation transition at c=1.

3.b Overrelaxation

The solution to Eq. 7 with c=2 to an accuracy of 10^{-6} is 0.7968126311118457. It took 14 iterations to reach this accuracy using the relaxation method. We compared this to the number of iterations taken to obtain the same level of accuracy with the overrelaxation method, described by Eq. 8.

$$x' = (1 + w)f(x) - wx \tag{8}$$

w	x	Iterations
0	0.7968126311118457483	14
0.1	0.7968125600352318205	12
0.2	0.7968122737816650814	11
0.3	0.7968123153348108945	9
0.4	0.7968121809454787874	8
0.5	0.7968121566399141154	5
0.6	0.7968121249678459250	6
0.7	0.7968121302175436016	5
0.8	0.7968121645340635606	6
0.9	0.7968121504914545961	8
1.0	0.7968120570818892912	9

Table 2: Solutions to Eq. 7 to an accuracy of 10^{-6} and number of iterations taken using the overrelaxation method with various w ranging from 0 to 1.

The lowest number of iterations to converge using the overrelaxation method was 5 for $w = 0.5$ and $w = 0.7$. This is less than half the number of iterations it took to obtain the same accuracy with relaxation. For the values of w we computed, the number of iterations increased symmetrically on either side of $w = 0.6$.

Although the optimal w is normally positive, there are some cases in which a negative w will lead to a solution of some given accuracy with less iteration. One such example is $x = e^{1-x^2}$ (Equation 6.72 of Newman). In chapter 6.3.1 of Newman, it is discussed that with a starting value of $x = 0.5$ this equation does not converge using the relaxation method, but instead oscillates between 0.00167991112 and 2.71828415720. The solution is in between these 2 values, at $x = 1$. In this case, a positive w will overshoot and take us farther from the solution. A negative w will undershoot the next guess, allowing us to converge towards the solution at x=1.

3.c Wien's Displacement Law

We then wrote a binary search method to solve for x in Eq. 9. This x value is used to determine the wavelength of maximum radiation (Wien's law, Eq. 11) obtained from Planck's radiation law. The binary search method was used to compute the value within an error of $\epsilon = 10^{-6}$. For the binary search method, we chose an initial range for the answer to be between 4 and 6. We then performed this calculation using the relaxation method with a starting guess of 4. We then performed this calculation using Newton's method with a starting guess of 4. The number of iterations, value and errors for all methods are summarized in Table 3. This table shows that the Newton's method is the most efficient, only taking 4 iterations to get an accuracy below 10^{-6} whereas the relaxation method took 6 iterations and binary search took 21.

$$5e^{-x} + x - 5 = 0 \tag{9}$$

Method	Number of Iterations	x value	error
Binary Search	21	4.965113639831543	$9.5367431640625{\cdot}10^{-7}$
Relaxation Method	6	4.965114228727152	$8.346869151409919{\cdot}10^{-8}$
Newton's Method	4	4.965114231744276	$8.036238341446733{\cdot}10^{-12}$

Table 3: Comparison of different methods of solving Eq. 9. The binary search method took 4 and 6 as the end values of the range to search for the solution and the relaxation and Newton methods took 4 as the starting guess. Both methods were done to obtain a solution within an accuracy of $\epsilon = 10^{-6}$.

Table 4 depicts how many iterations all three methods take for a range of starting values. This Table shows that as the starting guess gets closer to the actual value, the number of iterations gets smaller for all three methods, and the Newton's method takes the least amount of iterations except for an initial guess of 1.

Starting Value/Initial Guess	1	2	3	4	5	6
Binary Search	23	22	22	21	20	21
Relaxation Method	7	7	6	6	5	5
Newton's Method	8	5	4	4	3	4

Table 4: Comparison of the number of iterations taken by three methods of solving nonlinear equations with a variety of initial guesses. For the binary search method, this initial value refers to just one of the first 2 initial values.

For the rest of this question, the result from the binary search method was used to perform all computations. Using this, we obtained the value of the displacement constant b in Eq. 10 when h is Planck's constant, c is the speed of light and k_B is the Boltzmann constant. This value was computed to be 0.0028977723006411364 m·K.

$$b = \frac{hc}{k_B x} \tag{10}$$

$$\lambda = \frac{b}{T} \tag{11}$$

We can rearrange Wien's law (Eq. 11) to solve for the temperature, $T = b/\lambda$. Given that the peak wavelength in the Sun's emitted radiation falls at $\lambda = 502$ nm, we can then calculate the surface temperature of the sun can be estimated as $T = \frac{0.0028977723006411364}{(502\cdot10^{-9})} = 5795.544601282273$ K. This gives a very good estimate of the temperature of the sun which is known to be 5778K.